Nachschlagewerk fuer die Programmiersprache MQL5

fuer Kundenterminal MetaTrader 5

Lernen Sie die Sprache MQL5 und loesen Sie jegliche Aufgaben:

- Erzeugung von eigenen Indikatoren fuer technische Analyse unterschiedlicher Kompliziertheit
- Autotrading - Automatisieren eines Handelssystems fuer Arbeit in verschiedenen Finanzmaerkten
- Ausarbeitung von analytischen Instrumenten auf Grundlage mathematischer Erkenntnisse und klassischer Methoden
- Schreiben von handelsinformationellen Systemen fuer ein grosses Komplex der Aufgaben (Trading, Monitoring, Signale usw)

© 2000-2025, MetaQuotes Ltd

Inhalt

Nachschlagewerk MQL5 72

1 Grundlagen der Sprache ... 75

Syntax ... 76
- Kommentare ... 77
- Identifikatoren ... 78
- Reservierte Wörter ... 79

Datentypen ... 81
- Ganzzahlige Typen ... 82
  - Typen char, short, int und long ... 84
  - Symbolkonstanten ... 87
  - Typ datetime ... 90
  - Typ color ... 91
  - Typ bool ... 92
  - Enumerationen ... 93
- Realtypen (double, float) ... 95
- Komplexen Zahlen (complex) ... 101
- Typ string ... 102
- Strukturen, Klassen und Schnittstellen ... 103
- Objekt des dynamischen Arrays ... 130
- Matrizen und Vektoren ... 132
- Typenreduzierung ... 140
- Typ void und Konstante NULL ... 145
- Benutzerdefinierte Typen ... 146
- Objektanzeiger ... 156
- Referenzen. Modifikator& und Schluesselwort this ... 160

Operationen und Ausdruecke ... 162
- Ausdruecke ... 163
- Arithmetische Operationen ... 164
- Zuordnungsoperationen ... 165
- Vergleichsoperationen ... 166
- Logische Operationen ... 167
- Bitorganisierte Operationen ... 169
- Andere Operationen ... 172
- Prioritaeten und Operationsordnung ... 176

Operatoren ... 178
- Zusammengesetzter Operator ... 180
- Operator-Ausdruck ... 181
- Rücksetzoperator return ... 182
- Konditionaloperator if-else ... 183
- Ternärer Operator ?: ... 184
- Umschalteroperator switch ... 186
- Loop-Operator while ... 188
- Loop-Operator for ... 189
- Loop-Operator do while ... 191
- Endoperator break ... 192
- Fortsetzungsoperator continue ... 193
- Matrixproduktoperator @ ... 194
- Operator der Objekterzegung new ... 196
- Operator der Objektenfernung delete ... 198

Funktionen ... 199
- Funktionsaufruf ... 201
- Parameteruebertragung ... 202
- Überladen von Funktionen ... 205

© 2000-2025, MetaQuotes Ltd.

Überladen von Operatoren ... 208
Beschreibung der Aussenfunktionen ... 221
Exportieren der Funktionen ... 223
Ereignisbearbeiter ... 224
Variablen ... 237
Lokale Variablen ... 241
Formale Parameter ... 243
Statische Parameter ... 245
Globale Variablen ... 247
Input Variablen ... 248
Extern Variablen ... 255
Initialisierung der Variablen ... 256
Sichtbereich und Lebensdauer der Variablen ... 258
Erzeugung und Entfernung der Objekte ... 260
Preprozessor ... 263
Makrosubstitution (#define) ... 265
Programmeigenschaften (#property) ... 268
Include Dateien (#include) ... 275
Import der Funktionen (#import) ... 276
Bedingte Kompilierung (#ifdef, #ifndef, #else, #endif) ... 279
Objektorientiertes Programmieren ... 281
Inkapsulation und Erweiterungsfaehigkeit der Typen ... 283
Vererbung ... 286
Polymorphismus ... 291
Überladen ... 295
Virtuelle Funktionen ... 296
Statische Klassenmitglieder ... 300
Funktionstemplates ... 304
Klassentemplates ... 308
Abstrakte Klassen ... 313
Namensraum ... 315
2 Konstanten, Enumerationen und Strukturen ... 319
Chartkonstanten ... 320
Typen der Chartereignisse ... 321
Chartperioden ... 329
Charteigenschaften ... 331
Positionieren des Charts ... 345
Darstellung des Charts ... 346
Beispiele von Arbeiten mit dem Chart ... 348
Objektkonstanten ... 407
Objekttypen ... 408
OBJ_VLINE ... 410
OBJ_HLINE ... 415
OBJ_TREND ... 420
OBJ_TRENDBYANGLE ... 427
OBJ_CYCLES ... 433
OBJ_ARROWED_LINE ... 439
OBJ_CHANNEL ... 445
OBJ_STDDEVCHANNEL ... 452
OBJ_REGRESSION ... 459
OBJ_PITCHFORK ... 465
OBJ_GANNLINE ... 473
OBJ_GANNFAN ... 480
OBJ_GANNGRID ... 487
OBJ_FIBO ... 494
OBJ_FIBOTIMES ... 501
OBJ_FIBOFAN ... 508
OBJ_FIBOARC ... 515

© 2000-2025, MetaQuotes Ltd.

OBJ_FIBOCHANNEL ... 522
OBJ_EXPANSION ... 530
OBJ_ELLIOTWAVE5 ... 538
OBJ_ELLIOTWAVE3 ... 546
OBJ_RECTANGLE ... 553
OBJ_TRIANGLE ... 559
OBJ_ELLIPSE ... 566
OBJ_ARROW_THUMB_UP ... 572
OBJ_ARROW_THUMB_DOWN ... 578
OBJ_ARROW_UP ... 584
OBJ_ARROW_DOWN ... 590
OBJ_ARROW_STOP ... 596
OBJ_ARROW_CHECK ... 602
OBJ_ARROW_LEFT_PRICE ... 608
OBJ_ARROW_RIGHT_PRICE ... 613
OBJ_ARROW_BUY ... 618
OBJ_ARROW_SELL ... 623
OBJ_ARROW ... 628
OBJ_TEXT ... 634
OBJ_LABEL ... 640
OBJ_BUTTON ... 648
OBJ_CHART ... 655
OBJ_BITMAP ... 662
OBJ_BITMAP_LABEL ... 669
OBJ_EDIT ... 676
OBJ_EVENT ... 683
OBJ_RECTANGLE_LABEL ... 688
Objekteigenschaften ... 694
Methode der Objektbindung ... 727
Bindungswinkel ... 732
Objektsichtbarkeit ... 735
Levels von Elliott Wellen ... 738
Gann Objekte ... 739
Web-Farben ... 741
Wingdings ... 743
Indikatorkonstanten ... 744
Preiskonstanten ... 745
Glaettungsmethoden ... 748
Linien der Indikatoren ... 749
Zeichnungsstile ... 751
Eigenschaften der Benutzerobjekte ... 757
Typen der Indikatoren ... 764
Identifikatoren der Datentypen ... 766
Medium Zustand ... 767
Zustand des Client-Terminals ... 768
Information über das ausgeführte MQL5-Programm ... 779
Information über das Symbol ... 784
Information über das Konto ... 920
Teststatistik ... 930
Handelskonstanten ... 935
Information über historische Daten des Instrumentes ... 936
Ordereigenschaften ... 937
Eigenschaften der Positionen ... 958
Eigenschaften der Deals ... 963
Typen der Handelsoperationen ... 968
Types of Trade Transactions ... 980
Handelsordern in DOM ... 983
Eingenschaften von Signalen ... 984

© 2000-2025, MetaQuotes Ltd.

Benannte Konstanten **986**
Vorbestimmte Makrosubstitutionen 987
Mathematische Konstanten 993
Konstanten der numerischenTypen 995
Deinitialisierungsgruende 998
Pruefung des Objektanzeigers 1000
Andere Konstanten 1001
Datenstrukturen **1005**
Datenstruktur 1006
Struktur der Eingabeparameter des Anzeigers 1007
Struktur der historischen Daten 1008
DOM Struktur 1009
Struktur der Handelsanforderung 1010
Struktur der Ergebnisse der Prüfung der Handelsanforderung 1023
Struktur des Ergebnisses der Handelsanfoderung 1024
Struktur der Handelstransaktion 1028
Struktur fuer Erfassung der laufenden Preise 1036
Struktur des Wirtschaftskalenders 1038
Kodes der Fehler und Warnungen **1047**
Rückgabecodes des Handelsservers 1048
Compiler Warnungen 1052
Kompilierungsfehler 1055
Ausführungsfehler 1066
Eingabe/Ausgabe Konstanten **1081**
Flaggen der Dateieroeffnung 1082
Dateieigenschaften 1084
Positionieren innerhalb der Dateien 1085
Kodeseite Verwenden 1086
MessageBox 1087

3 MQL5 Programme 1089

Programmausführung **1090**
Handelserlaubnis **1097**
Eregnisse des Client-Terminals **1101**
Ressourcen **1105**
Aufruf der Importfunktionen **1117**
Ausführungsfehler **1119**
Testen von Handelsstrategien **1120**

4 Zugeordnete Konstanten 1147

_AppliedTo **1148**
_Digits **1150**
_Point **1151**
_LastError **1152**
_Period **1153**
_RandomSeed **1154**
_StopFlag **1155**
_Symbol **1156**
_UninitReason **1157**
_IsX64 ..... **1158**

5 Allgemeine Funktionen 1159

Alert **1161**
CheckPointer **1166**
Comment **1168**
CryptEncode **1170**
CryptDecode **1172**
DebugBreak **1174**
ExpertRemove **1175**
GetPointer **1177**

© 2000-2025, MetaQuotes Ltd.

GetTickCount ... 1180
GetTickCount64 ... 1181
GetMicrosecondCount ... 1183
MessageBox ... 1185
PeriodSeconds ... 1190
PlaySound ... 1192
Print ... 1195
PrintFormat ... 1197
ResetLastError ... 1203
ResourceCreate ... 1204
ResourceFree ... 1208
ResourceReadImage ... 1211
ResourceSave ... 1215
SetReturnError ... 1220
SetUserError ... 1222
Sleep ... 1223
TerminalClose ... 1225
TesterHideIndicators ... 1227
TesterStatistics ... 1229
TesterStop ... 1231
TesterDeposit ... 1233
TesterWithdrawal ... 1236
TranslateKey ... 1239
ZeroMemory ... 1240

6 Operationen mit Arrays ... 1243

ArrayBsearch ... 1245
ArrayCopy ... 1249
ArrayCompare ... 1254
ArrayFree ... 1257
ArrayGetAsSeries ... 1266
ArrayInitialize ... 1269
ArrayFill ... 1271
ArrayIsDynamic ... 1273
ArrayIsSeries ... 1276
ArrayMaximum ... 1278
ArrayMinimum ... 1289
ArrayPrint ... 1300
ArrayRange ... 1303
ArrayResize ... 1304
ArrayInsert ... 1308
ArrayRemove ... 1311
ArrayReverse ... 1313
ArraySetAsSeries ... 1315
ArraySize ... 1317
ArraySort ... 1319
ArraySwap ... 1324
ArrayToFP16 ... 1326
ArrayToFP8 ... 1329
ArrayFromFP16 ... 1332
ArrayFromFP8 ... 1335

7 Matrizen und Vektoren ... 1338

Typen von Matrix und Vektor ... 1346
Enumerationen ... 1347
Initialisierung ... 1352
Assign ... 1355
CopyIndicatorBuffer ... 1357
CopyRates ... 1359
CopyTicks ... 1364

© 2000-2025, MetaQuotes Ltd.

CopyTicksRange ... 1367
Eye ... 1369
Identity ... 1371
Ones ... 1373
Zeros ... 1374
Full ... 1375
Tri ... 1376
Init ... 1378
Fill ... 1380
Random ... 1381
Manipulation ... 1383
HasNan ... 1385
Transpose ... 1386
TransposeConjugate ... 1388
TriL ... 1389
TriU ... 1390
Diag ... 1391
Row ... 1393
Col ... 1395
Copy ... 1397
Concat ... 1399
Compare ... 1401
CompareByDigits ... 1403
CompareEqual ... 1405
Flat ... 1408
Clip ... 1410
Reshape ... 1411
Resize ... 1413
Set ... 1415
SwapRows ... 1417
SwapCols ... 1418
Split ... 1419
Hsplit ... 1421
Vsplit ... 1423
ArgSort ... 1425
Sort ... 1426
Operationen ... 1428
Mathematische Operationen ... 1430
Mathematische Funktionen ... 1431
Produkte ... 1432
MatMul ... 1433
GeMM ... 1438
Power ... 1442
Dot ... 1445
Kron ... 1447
Inner ... 1449
Outer ... 1451
CorrCoef ... 1454
Cov ... 1458
Correlate ... 1461
Convolve ... 1464
Transformationen ... 1467
Cholesky ... 1468
Eig ... 1469
EigVals ... 1472
LU ... 1473
LUP ... 1475
QR ... 1477

© 2000-2025, MetaQuotes Ltd.

SVD ... 1479
Statistik ... 1482
ArgMax ... 1483
ArgMin ... 1484
Max ... 1485
Min ... 1486
Ptp ... 1487
Sum ... 1488
Prod ... 1489
CumSum ... 1491
CumProd ... 1493
Percentile ... 1495
Quantile ... 1497
Median ... 1499
Mean ... 1501
Average ... 1502
Std ... 1504
Var ... 1506
LinearRegression ... 1508
Eigenschaften ... 1511
Rows ... 1512
Cols ... 1513
Size ... 1514
Norm ... 1515
Cond ... 1518
Det ... 1521
SLogDet ... 1523
Rank ... 1524
Trace ... 1527
Spectrum ... 1528
Lösungen ... 1529
Solve ... 1530
LstSq ... 1531
Inv ... 1532
PInv ... 1534
Maschinelles Lernen ... 1535
Activation ... 1541
Derivative ... 1545
Loss ... 1547
LossGradient ... 1549
RegressionMetric ... 1551
ConfusionMatrix ... 1553
ConfusionMatrixMultilabel ... 1555
ClassificationMetric ... 1557
ClassificationScore ... 1560
PrecisionRecall ... 1564
ReceiverOperatingCharacteristic ... 1568
OpenBLAS ... 1572
Singular Value Decomposition ... 1575
SingularValueDecompositionDC ... 1576
SingularValueDecompositionQR ... 1578
SingularValueDecompositionQRPivot ... 1580
SingularValueDecompositionBisect ... 1584
SingularValueDecompositionJacobiHigh ... 1587
SingularValueDecompositionJacobiLow ... 1592
SingularValueDecompositionBidiagDC ... 1595
SingularValueDecompositionBidiagBisect ... 1598
SingularValueDecompositionBidiagQR ... 1601

© 2000-2025, MetaQuotes Ltd.

Eigen Values ... 1603
General Matrices ... 1605
EigenSolver ... 1606
EigenSolverX ... 1608
EigenSolverShur ... 1612
EigenSolver2 ... 1614
EigenSolver2X ... 1617
EigenSolver2Shur ... 1622
EigenSolver2Blocked ... 1625
EigenSolver2ShurBlocked ... 1628
Symmetric Matrices ... 1631
EigenSymmetricDC ... 1633
EigenSymmetricQR ... 1635
EigenSymmetricRobust ... 1637
EigenSymmetricBisect ... 1640
EigenSymmetricDC2s ... 1643
EigenSymmetricQR2s ... 1645
EigenSymmetricRobust2s ... 1647
EigenSymmetricBisect2s ... 1650
EigenSymmetric2DC ... 1653
EigenSymmetric2QR ... 1656
EigenSymmetric2Bisect ... 1659
Tridiagonal Matrices ... 1663
EigenTridiagonalDC ... 1665
EigenTridiagonalQR ... 1667
EigenTridiagonalRobust ... 1669
EigenTridiagonalBisect ... 1672
EigenTridiagonalQL ... 1675
EigenTridiagonalDCQ ... 1676
EigenTridiagonalQRQ ... 1678
EigenTridiagonalPosDefQ ... 1680
Orthogonal Factorizations ... 1682
FactorizationQR ... 1683
FactorizationQRNonNeg ... 1685
FactorizationQRPivot ... 1687
FactorizationLQ ... 1689
FactorizationQL ... 1691
FactorizationRQ ... 1693
Linear Equations ... 1695
LinearEquationsSolution ... 1697
LinearEquationsSolutionTriangular ... 1699
LinearEquationsSolutionSy ... 1701
LinearEquationsSolutionComplexSy ... 1703
LinearEquationsSolutionSyPD ... 1704
LinearEquationsSolutionGeTrid ... 1706
LinearEquationsSolutionSyTridPD ... 1708
Factorizations ... 1710
FactorizationPLU ... 1711
FactorizationPLUGeTrid ... 1713
FactorizationLDL ... 1715
FactorizationLDLSyTridPD ... 1717
FactorizationCholesky ... 1719
FactorizationCholeskySyPS ... 1720
Dynamic Mode Decomposition ... 1722
DynamicModeDecomposition ... 1723
DynamicModeDecompositionQR ... 1729
Singular Spectrum Analysis ... 1736
SingularSpectrumAnalysisSpectrum ... 1737

© 2000-2025, MetaQuotes Ltd.

SingularSpectrumAnalysisForecast ... 1738
SingularSpectrumAnalysisReconstructComponents ... 1739
SingularSpectrumAnalysisReconstructSeries ... 1740
Reductions ... 1741
ReduceToBidiagonal ... 1742
ReflectBidiagonalToQP ... 1745
ReduceSymmetricToTridiagonal ... 1747
ReflectTridiagonalToQ ... 1749
8 Datenverarbeitung ... 1751
CharToString ... 1753
CharArrayToString ... 1755
CharArrayToStruct ... 1757
StructToCharArray ... 1761
ColorToARGB ... 1764
ColorToString ... 1766
DoubleToString ... 1767
EnumToString ... 1768
IntegerToString ... 1770
ShortToString ... 1772
ShortArrayToString ... 1774
TimeToString ... 1776
NormalizeDouble ... 1779
StringToCharArray ... 1781
StringToColor ... 1783
StringToDouble ... 1784
StringToInteger ... 1785
StringToShortArray ... 1786
StringToTime ... 1788
StringFormat ... 1791
9 Mathematische Funktionen ... 1795
MathAbs ... 1797
MathArccos ... 1798
MathArcsin ... 1801
MathArctan ... 1804
MathArctan2 ... 1807
MathClassify ... 1811
MathCeil ... 1813
MathCos ... 1815
MathExp ... 1818
MathFloor ... 1821
MathLog ... 1823
MathLog10 ... 1827
MathMax ... 1831
MathMin ... 1833
MathMod ... 1835
MathPow ... 1836
MathRand ... 1840
MathRound ... 1841
MathSin ... 1843
MathSqrt ... 1846
MathSrand ... 1850
MathTan ... 1853
MathIsValidNumber ... 1856
MathExpm1 ... 1857
MathLog1p ... 1860
MathArccosh ... 1864
MathArcsinh ... 1867
MathArctanh ... 1870

© 2000-2025, MetaQuotes Ltd.

MathCosh 1873
MathSinh 1876
MathTanh 1879
MathSwap 1882
10 Stringfunktionen 1885
StringAdd 1886
StringBufferLen 1888
StringCompare 1889
StringConcatenate 1891
StringFill 1893
StringFind 1894
StringGetCharacter 1897
StringInit 1899
StringLen 1900
StringSetLength 1901
StringReplace 1902
StringReserve 1903
StringSetCharacter 1905
StringSplit 1907
StringSubstr 1908
StringToLower 1909
StringToUpper 1910
StringTrimLeft 1911
StringTrimRight 1912
11 Datum und Zeit 1913
TimeCurrent 1914
TimeTradeServer 1916
TimeLocal 1918
TimeGMT 1920
TimeDaylightSavings 1922
TimeGMTOffset 1923
TimeToStruct 1924
StructToTime 1925
12 Kontoinformation 1927
AccountInfoDouble 1928
AccountInfoInteger 1929
AccountInfoString 1931
13 Zustandspruefung 1932
GetLastError 1933
IsStopped 1934
UninitializeReason 1935
TerminalInfoInteger 1937
TerminalInfoDouble 1938
TerminalInfoString 1939
MQLInfoInteger 1940
MQLInfoString 1941
Symbol 1942
Period 1943
Digits 1944
Point 1945
14 Ereignisbehandlung 1946
OnStart 1948
OnInit 1951
OnDeinit 1954
OnTick 1957
OnCalculate 1963

© 2000-2025, MetaQuotes Ltd.

OnTimer 1967
OnTrade 1970
OnTradeTransaction 1975
OnBookEvent 1981
OnChartEvent 1984
OnTester 1991
OnTesterInit 1998
OnTesterDeinit 2005
OnTesterPass 2006

15 Marktinformation erhalten 2007

SymbolsTotal 2008
SymbolExist 2009
SymbolName 2011
SymbolSelect 2013
SymbolIsSynchronized 2015
SymbolInfoDouble 2016
SymbolInfoInteger 2018
SymbolInfoString 2020
SymbolInfoMarginRate 2022
SymbolInfoTick 2024
SymbolInfoSessionQuote 2025
SymbolInfoSessionTrade 2027
MarketBookAdd 2029
MarketBookRelease 2031
MarketBookGet 2033

16 Wirtschaftskalender 2034

CalendarCountryById 2035
CalendarEventById 2037
CalendarValueById 2040
CalendarCountries 2043
CalendarEventByCountry 2045
CalendarEventByCurrency 2047
CalendarValueHistoryByEvent 2049
CalendarValueHistory 2052
CalendarValueLastByEvent 2055
CalendarValueLast 2060

17 Zugang zu Zeitreihen und Indikatoren 2065

Richtung des Indizierens in Feldern, Puffern und Zeitreihen 2070
Datenzugriff organisieren 2074
SeriesInfoInteger 2084
Bars 2086
BarsCalculated 2088
IndicatorCreate 2090
IndicatorParameters 2092
IndicatorRelease 2094
CopyBuffer 2096
CopyRates 2101
CopySeries 2105
CopyTime 2109
CopyOpen 2112
CopyHigh 2115
CopyLow 2119
CopyClose 2122
CopyTickVolume 2125
CopyRealVolume 2129
CopySpread 2132
CopyTicks 2136

© 2000-2025, MetaQuotes Ltd.

CopyTicksRange 2142
iBars 2144
iBarShift 2145
iClose 2148
iHigh 2150
iHighest 2152
iLow 2153
iLowest 2155
iOpen 2156
iTime 2158
iTickVolume 2161
iRealVolume 2163
iVolume 2165
iSpread 2167

18 Benutzerdefinierte Symbole 2169

CustomSymbolCreate 2171
CustomSymbolDelete 2174
CustomSymbolSetInteger 2176
CustomSymbolSetDouble 2180
CustomSymbolSetString 2184
CustomSymbolSetMarginRate 2188
CustomSymbolSetSessionQuote 2193
CustomSymbolSetSessionTrade 2198
CustomRatesDelete 2203
CustomRatesReplace 2208
CustomRatesUpdate 2213
CustomTicksAdd 2218
CustomTicksDelete 2226
CustomTicksReplace 2234
CustomBookAdd 2243

19 Operationen mit Charts 2246

ChartApplyTemplate 2248
ChartSaveTemplate 2251
ChartWindowFind 2256
ChartTimePriceToXY 2258
ChartXYToTimePrice 2260
ChartOpen 2262
ChartFirst 2264
ChartNext 2265
ChartClose 2266
ChartSymbol 2268
ChartPeriod 2269
ChartRedraw 2270
ChartSetDouble 2273
ChartSetInteger 2275
ChartSetString 2277
ChartGetDouble 2279
ChartGetInteger 2281
ChartGetString 2283
ChartNavigate 2285
ChartID 2288
ChartIndicatorAdd 2290
ChartIndicatorDelete 2293
ChartIndicatorGet 2296
ChartIndicatorName 2298
ChartIndicatorsTotal 2300
ChartWindowOnDropped 2302
ChartPriceOnDropped 2303

© 2000-2025, MetaQuotes Ltd.

ChartTimeOnDropped 2304
ChartXOnDropped 2305
ChartYOnDropped 2306
ChartSetSymbolPeriod 2307
ChartScreenShot 2309
20 Handelsfunktionen 2312
OrderCalcMargin 2315
OrderCalcProfit 2319
OrderCheck 2322
OrderSend 2326
OrderSendAsync 2331
PositionsTotal 2342
PositionGetSymbol 2343
PositionSelect 2345
PositionSelectByTicket 2347
PositionGetDouble 2349
PositionGetInteger 2353
PositionGetString 2355
PositionGetTicket 2358
OrdersTotal 2360
OrderGetTicket 2361
OrderSelect 2363
OrderGetDouble 2367
OrderGetInteger 2371
OrderGetString 2377
HistorySelect 2380
HistorySelectByPosition 2382
HistoryOrderSelect 2386
HistoryOrdersTotal 2389
HistoryOrderGetTicket 2390
HistoryOrderGetDouble 2392
HistoryOrderGetInteger 2395
HistoryOrderGetString 2398
HistoryDealSelect 2401
HistoryDealsTotal 2404
HistoryDealGetTicket 2405
HistoryDealGetDouble 2408
HistoryDealGetInteger 2413
HistoryDealGetString 2416
21 Handelssignale 2420
SignalBaseGetDouble 2421
SignalBaseGetInteger 2422
SignalBaseGetString 2423
SignalBaseSelect 2424
SignalBaseTotal 2425
SignalInfoGetDouble 2426
SignalInfoGetInteger 2427
SignalInfoGetString 2428
SignalInfoSetDouble 2429
SignalInfoSetInteger 2430
SignalSubscribe 2431
SignalUnsubscribe 2432
22 Netzwerkfunktionen 2433
SocketCreate 2435
SocketClose 2438
SocketConnect 2441
SocketIsConnected 2445

© 2000-2025, MetaQuotes Ltd.

SocketIsReadable 2449
SocketIsWritable 2453
SocketTimeouts 2457
SocketRead 2459
SocketSend 2463
SocketTlsHandshake 2467
SocketTlsCertificate 2472
SocketTlsRead 2476
SocketTlsReadAvailable 2480
SocketTlsSend 2485
WebRequest 2489
SendFTP 2492
SendMail 2494
SendNotification 2496

23 Globalvariablen des Kundenterminals 2498

GlobalVariableCheck 2499
GlobalVariableTime 2500
GlobalVariableDel 2501
GlobalVariableGet 2502
GlobalVariableName 2503
GlobalVariableSet 2504
GlobalVariablesFlush 2505
GlobalVariableTemp 2506
GlobalVariableSetOnCondition 2507
GlobalVariablesDeleteAll 2508
GlobalVariablesTotal 2509

24 Dateioperationen 2510

FileSelectDialog 2513
FileFindFirst 2515
FileFindNext 2517
FileFindClose 2519
FileIsExist 2520
FileOpen 2523
FileClose 2526
FileCopy 2527
FileDelete 2530
FileMove 2533
FileFlush 2535
FileGetInteger 2537
FileIsEnding 2540
FileIsLineEnding 2541
FileReadArray 2547
FileReadBool 2549
FileReadDatetime 2553
FileReadDouble 2557
FileReadFloat 2560
FileReadInteger 2564
FileReadLong 2568
FileReadNumber 2571
FileReadString 2577
FileReadStruct 2579
FileSeek 2583
FileSize 2586
FileTell 2589
FileWrite 2592
FileWriteArray 2595
FileWriteDouble 2598
FileWriteFloat 2601

© 2000-2025, MetaQuotes Ltd.

FileWriteInteger 2604
FileWriteLong 2607
FileWriteString 2610
FileWriteStruct 2613
FileLoad 2616
FileSave 2618
FolderCreate 2620
FolderDelete 2623
FolderClean 2626
25 Benutzerindikatoren 2629
Stile der Indikator in den Beispielen 2633
DRAW_NONE 2645
DRAW_LINE 2648
DRAW_SECTION 2652
DRAW_HISTOGRAM 2656
DRAW_HISTOGRAM2 2660
DRAW_ARROW 2664
DRAW_ZIGZAG 2669
DRAW_FILLING 2674
DRAW_BARS 2679
DRAW_CANDLES 2685
DRAW_COLOR_LINE 2692
DRAW_COLOR_SECTION 2697
DRAW_COLOR_HISTOGRAM 2703
DRAW_COLOR_HISTOGRAM2 2708
DRAW_COLOR_ARROW 2713
DRAW_COLOR_ZIGZAG 2719
DRAW_COLOR_BARS 2724
DRAW_COLOR_CANDLES 2731
Zusammenhang zwischen Eigenschaften des Indikators und entsprechenden Funktionen 2738
SetIndexBuffer 2741
IndicatorSetDouble 2744
IndicatorSetInteger 2748
IndicatorSetString 2752
PlotIndexSetDouble 2755
PlotIndexSetInteger 2756
PlotIndexSetString 2760
PlotIndexGetInteger 2761
26 Graphische Objekte 2764
ObjectCreate 2766
ObjectName 2770
ObjectDelete 2771
ObjectsDeleteAll 2772
ObjectFind 2773
ObjectGetTimeByValue 2774
ObjectGetValueByTime 2775
ObjectMove 2776
ObjectsTotal 2777
ObjectSetDouble 2778
ObjectSetInteger 2782
ObjectSetString 2786
ObjectGetDouble 2788
ObjectGetInteger 2790
ObjectGetString 2792
TextSetFont 2794
TextOut 2797
TextGetSize 2801

© 2000-2025, MetaQuotes Ltd.

27 Technische Indikatoren ... 2802
iAC ... 2805
iAD ... 2810
iADX ... 2815
iADXWilder ... 2820
iAlligator ... 2825
iAMA ... 2832
iAO ... 2837
iATR ... 2842
iBearsPower ... 2847
iBands ... 2852
iBullsPower ... 2858
iCCI ... 2863
iChaikin ... 2868
iCustom ... 2873
iDEMA ... 2877
iDeMarker ... 2882
iEnvelopes ... 2887
iForce ... 2893
iFractals ... 2898
iFrAMA ... 2903
iGator ... 2908
Ilchimoku ... 2915
iBWMFI ... 2922
iMomentum ... 2927
iMFI ... 2932
iMA ... 2937
iOsMA ... 2942
iMACD ... 2947
iOBV ... 2953
iSAR ... 2958
iRSI ... 2963
iRVI ... 2968
iStdDev ... 2973
iStochastic ... 2978
iTEMA ... 2984
iTriX ... 2989
iWPR ... 2994
iVIDyA ... 2999
iVolumes ... 3004
28 Arbeit mit Ergebnisse der Optimierung ... 3009
FrameFirst ... 3011
FrameFilter ... 3012
FrameNext ... 3013
FrameInputs ... 3014
FrameAdd ... 3015
ParameterGetRange ... 3016
ParameterSetRange ... 3020
29 Arbeit mit Ereignissen ... 3022
EventSetMillisecondTimer ... 3023
EventSetTimer ... 3024
EventKillTimer ... 3025
EventChartCustom ... 3026
30 Arbeit mit OpenCL ... 3032
CLHandleType ... 3034
CLGetInfoInteger ... 3035
CLGetInfoString ... 3038

© 2000-2025, MetaQuotes Ltd.

CLContextCreate 3041
CLContextFree 3042
CLGetDeviceInfo 3043
CLProgramCreate 3048
CLProgramFree 3053
CLKernelCreate 3054
CLKernelFree 3055
CLSetKernelArg 3056
CLSetKernelArgMem 3057
CLSetKernelArgMemLocal 3058
CLBufferCreate 3059
CLBufferFree 3060
CLBufferWrite 3061
CLBufferRead 3066
CLExecute 3070
CLExecutionStatus 3072

31 Arbeiten mit der Datenbank 3073

DatabaseOpen 3077
DatabaseClose 3079
DatabaseImport 3080
DatabaseExport 3083
DatabasePrint 3089
DatabaseTableExists 3094
DatabaseExecute 3095
DatabasePrepare 3107
DatabaseReset 3116
DatabaseBind 3122
DatabaseBindArray 3127
DatabaseRead 3132
DatabaseReadBind 3133
DatabaseFinalize 3137
DatabaseTransactionBegin 3138
DatabaseTransactionCommit 3143
DatabaseTransactionRollback 3144
DatabaseColumnsCount 3145
DatabaseColumnName 3146
DatabaseColumnType 3147
DatabaseColumnSize 3148
DatabaseColumnText 3149
DatabaseColumnInteger 3150
DatabaseColumnLong 3151
DatabaseColumnDouble 3152
DatabaseColumnBlob 3153

32 Arbeiten mit DirectX 3154

DXContextCreate 3156
DXContextSetSize 3157
DXContextGetSize 3158
DXContextClearColors 3159
DXContextClearDepth 3160
DXContextGetColors 3161
DXContextGetDepth 3162
DXBufferCreate 3163
DXTextureCreate 3164
DXInputCreate 3170
DXInputSet 3171
DXShaderCreate 3172
DXShaderSetLayout 3173
DXShaderInputsSet 3174

© 2000-2025, MetaQuotes Ltd.

DXShaderTexturesSet 3175
DXDraw 3176
DXDrawIndexed 3177
DXPrimiveTopologySet 3178
DXBufferSet 3179
DXShaderSet 3180
DXHandleType 3181
DXRelease 3182
33 MetaTrader für Python 3183
initialize 3189
login 3191
shutdown 3194
version 3195
last_error 3197
account_info 3199
terminal_info 3202
symbols_total 3205
symbols_get 3206
symbol_info 3209
symbol_info_tick 3213
symbol_select 3215
market_book_add 3219
market_book_get 3220
market_book_release 3223
copy_rates_from 3224
copy_rates_frompos 3228
copy_rates_range 3231
copy_ticks_from 3234
copy_ticks_range 3237
orders_total 3240
orders_get 3241
order_calc_margin 3244
order_calc_profit 3247
order_check 3250
order_send 3254
positions_total 3259
positions_get 3260
history_orders_total 3263
history_orders_get 3265
history_deals_total 3268
history_deals_get 3270
34 ONNX Modelle 3274
ONNX Unterstützung 3275
Format-Konvertierung 3277
Automatische Konvertierung von Datentypen 3278
Erstellen eines Modells 3282
Ausführung eines Modells 3290
Ausführen im Strategy Tester 3296
OnnxCreate 3301
OnnxCreateFromBuffer 3302
OnnxRelease 3303
OnnxRun 3304
OnnxGetInputCount 3307
OnnxGetOutputCount 3308
OnnxGetInputName 3309
OnnxGetOutputName 3310
OnnxGetInputTypeInfo 3311
OnnxGetOutputTypeInfo 3312

© 2000-2025, MetaQuotes Ltd.

OnnxSetInputShape ... 3313
OnnxSetOutputShape ... 3314
Datenstrukturen ... 3315
35 Standardbibliothek ... 3319
Mathematik ... 3320
Statistik ... 3321
Statistische Eigenschaften ... 3324
MathMean ... 3325
MathVariance ... 3326
MathSkewness ... 3327
MathKurtosis ... 3328
MathMoments ... 3329
MathMedian ... 3330
MathStandardDeviation ... 3331
MathAverageDeviation ... 3332
Normalverteilung ... 3333
MathProbabilityDensityNormal ... 3337
MathCumulativeDistributionNormal ... 3339
MathQuantileNormal ... 3341
MathRandomNormal ... 3343
MathMomentsNormal ... 3344
Log-Normalverteilung ... 3345
MathProbabilityDensityLognormal ... 3349
MathCumulativeDistributionLognormal ... 3351
MathQuantileLognormal ... 3353
MathRandomLognormal ... 3355
MathMomentsLognormal ... 3356
Beta-Verteilung ... 3357
MathProbabilityDensityBeta ... 3361
MathCumulativeDistributionBeta ... 3363
MathQuantileBeta ... 3365
MathRandomBeta ... 3367
MathMomentsBeta ... 3368
Nichtzentrale Betaverteilung ... 3369
MathProbabilityDensityNoncentralBeta ... 3373
MathCumulativeDistributionNoncentralBeta ... 3375
MathQuantileNoncentralBeta ... 3377
MathRandomNoncentralBeta ... 3379
MathMomentsNoncentralBeta ... 3380
Gamma-Verteilung ... 3381
MathProbabilityDensityGamma ... 3385
MathCumulativeDistributionGamma ... 3387
MathQuantileGamma ... 3389
MathRandomGamma ... 3391
MathMomentsGamma ... 3392
Chi-Quadrat Verteilung ... 3393
MathProbabilityDensityChiSquare ... 3397
MathCumulativeDistributionChiSquare ... 3399
MathQuantileChiSquare ... 3401
MathRandomChiSquare ... 3403
MathMomentsChiSquare ... 3404
Nichtzentrale Chi-Quadrat Verteilung ... 3405
MathProbabilityDensityNoncentralChiSquare ... 3409
MathCumulativeDistributionNoncentralChiSquare ... 3411
MathQuantileNoncentralChiSquare ... 3413
MathRandomNoncentralChiSquare ... 3415
MathMomentsNoncentralChiSquare ... 3416
Exponentielle Verteilung ... 3417

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityExponential ... 3421
MathCumulativeDistributionExponential ... 3423
MathQuantileExponential ... 3425
MathRandomExponential ... 3427
MathMomentsExponential ... 3428
F-Verteilung ... 3429
MathProbabilityDensityF ... 3433
MathCumulativeDistributionF ... 3435
MathQuantileF ... 3437
MathRandomF ... 3439
MathMomentsF ... 3440
Nichtzentrale F-Verteilung ... 3441
MathProbabilityDensityNoncentralF ... 3445
MathCumulativeDistributionNoncentralF ... 3447
MathQuantileNoncentralF ... 3449
MathRandomNoncentralF ... 3451
MathMomentsNoncentralF ... 3452
t-Verteilung ... 3453
MathProbabilityDensityT ... 3457
MathCumulativeDistributionT ... 3459
MathQuantileT ... 3461
MathRandomT ... 3463
MathMomentsT ... 3464
Nichtzentrale t-Verteilung ... 3465
MathProbabilityDensityNoncentralT ... 3469
MathCumulativeDistributionNoncentralT ... 3471
MathQuantileNoncentralT ... 3473
MathRandomNoncentralT ... 3475
MathMomentsNoncentralT ... 3476
Logistische Verteilung ... 3477
MathProbabilityDensityLogistic ... 3481
MathCumulativeDistributionLogistic ... 3483
MathQuantileLogistic ... 3485
MathRandomLogistic ... 3487
MathMomentsLogistic ... 3488
Cauchy-Verteilung ... 3489
MathProbabilityDensityCauchy ... 3493
MathCumulativeDistributionCauchy ... 3495
MathQuantileCauchy ... 3497
MathRandomCauchy ... 3499
MathMomentsCauchy ... 3500
Gleichverteilung ... 3501
MathProbabilityDensityUniform ... 3505
MathCumulativeDistributionUniform ... 3507
MathQuantileUniform ... 3509
MathRandomUniform ... 3511
MathMomentsUniform ... 3512
Weibull-Verteilung ... 3513
MathProbabilityDensityWeibull ... 3517
MathCumulativeDistributionWeibull ... 3519
MathQuantileWeibull ... 3521
MathRandomWeibull ... 3523
MathMomentsWeibull ... 3524
Binomial-Verteilung ... 3525
MathProbabilityDensityBinomial ... 3528
MathCumulativeDistributionBinomial ... 3530
MathQuantileBinomial ... 3532
MathRandomBinomial ... 3534

© 2000-2025, MetaQuotes Ltd.

MathMomentsBinomial ... 3535
Negative Binomial-Verteilung ... 3536
MathProbabilityDensityNegativeBinomial ... 3539
MathCumulativeDistributionNegativeBinomial ... 3541
MathQuantileNegativeBinomial ... 3543
MathRandomNegativeBinomial ... 3545
MathMomentsNegativeBinomial ... 3546
Geometrische Verteilung ... 3547
MathProbabilityDensityGeometric ... 3551
MathCumulativeDistributionGeometric ... 3553
MathQuantileGeometric ... 3555
MathRandomGeometric ... 3557
MathMomentsGeometric ... 3558
Hypergeometrische Verteilung ... 3559
MathProbabilityDensityHypergeometric ... 3563
MathCumulativeDistributionHypergeometric ... 3565
MathQuantileHypergeometric ... 3567
MathRandomHypergeometric ... 3569
MathMomentsHypergeometric ... 3570
Poisson-Verteilung ... 3571
MathProbabilityDensityPoisson ... 3575
MathCumulativeDistributionPoisson ... 3577
MathQuantilePoisson ... 3579
MathRandomPoisson ... 3581
MathMomentsPoisson ... 3582
Hilfsfunktionen ... 3583
MathRandomNonZero ... 3588
MathMoments ... 3589
MathPowInt ... 3590
MathFactorial ... 3591
MathTrunc ... 3592
MathRound ... 3593
MathArctan2 ... 3595
MathGamma ... 3597
MathGammaLog ... 3598
MathBeta ... 3599
MathBetaLog ... 3600
MathBetaIncomplete ... 3601
MathGammaIncomplete ... 3602
MathBinomialCoefficient ... 3603
MathBinomialCoefficientLog ... 3604
MathHypergeometric2F2 ... 3605
MathSequence ... 3606
MathSequenceByCount ... 3607
MathReplicate ... 3608
MathReverse ... 3609
MathIdentical ... 3610
MathUnique ... 3611
MathQuickSortAscending ... 3612
MathQuickSortDescending ... 3613
MathQuickSort ... 3614
MathOrder ... 3615
MathBitwiseNot ... 3616
MathBitwiseAnd ... 3617
MathBitwiseOr ... 3618
MathBitwiseXor ... 3619
MathBitwiseShiftL ... 3620
MathBitwiseShiftR ... 3621

© 2000-2025, MetaQuotes Ltd.

MathCumulativeSum ... 3622
MathCumulativeProduct ... 3623
MathCumulativeMin ... 3624
MathCumulativeMax ... 3625
MathSin ... 3626
MathCos ... 3627
MathTan ... 3628
MathArcsin ... 3629
MathArccos ... 3630
MathArctan ... 3631
MathSinPi ... 3632
MathCosPi ... 3633
MathTanPi ... 3634
MathAbs ... 3635
MathCeil ... 3636
MathFloor ... 3637
MathSqrt ... 3638
MathExp ... 3639
MathPow ... 3640
MathLog ... 3641
MathLog2 ... 3643
MathLog10 ... 3644
MathLog1p ... 3645
MathDifference ... 3646
MathSample ... 3648
MathTukeySummary ... 3651
MathRange ... 3652
MathMin ... 3653
MathMax ... 3654
MathSum ... 3655
MathProduct ... 3656
MathStandardDeviation ... 3657
MathAverageDeviation ... 3658
MathMedian ... 3659
MathMean ... 3660
MathVariance ... 3661
MathSkewness ... 3662
MathKurtosis ... 3663
MathExpm1 ... 3664
MathSinh ... 3665
MathCosh ... 3666
MathTanh ... 3667
MathArcsinh ... 3668
MathArccosh ... 3669
MathArctanh ... 3670
MathSignif ... 3671
MathRank ... 3673
MathCorrelationPearson ... 3674
MathCorrelationSpearman ... 3675
MathCorrelationKendall ... 3676
MathQuantile ... 3677
MathProbabilityDensityEmpirical ... 3678
MathCumulativeDistributionEmpirical ... 3679
Fuzzy logic ... 3680
Membership functions ... 3682
CConstantMembershipFunction ... 3684
GetValue ... 3686
CCompositeMembershipFunction ... 3687

© 2000-2025, MetaQuotes Ltd.

CompositionType ... 3689
MembershipFunctions ... 3689
GetValue ... 3689
CDifferencTwoSigmoidalMembershipFunction ... 3691
A1 ... 3693
A2 ... 3693
C1 ... 3694
C2 ... 3694
GetValue ... 3695
CGeneralizedBellShapedMembershipFunction ... 3696
A ... 3698
B ... 3698
C ... 3699
GetValue ... 3699
CNormalCombinationMembershipFunction ... 3700
B1 ... 3702
B2 ... 3702
Sigma1 ... 3703
Sigma2 ... 3703
GetValue ... 3704
CNormalMembershipFunction ... 3705
B ... 3707
Sigma ... 3707
GetValue ... 3708
CP_ShapedMembershipFunction ... 3709
A ... 3711
B ... 3711
C ... 3712
D ... 3712
GetValue ... 3713
CProductTwoSigmoidalMembershipFunctions ... 3714
A1 ... 3716
A2 ... 3716
C1 ... 3717
C2 ... 3717
GetValue ... 3718
CS_ShapedMembershipFunction ... 3719
A ... 3721
B ... 3721
GetValue ... 3722
CSigmoidalMembershipFunction ... 3723
A ... 3725
C ... 3725
GetValue ... 3726
CTrapezoidMembershipFunction ... 3727
X1 ... 3729
X2 ... 3729
X3 ... 3730
X4 ... 3730
GetValue ... 3731
CTriangularMembershipFunction ... 3732
X1 ... 3734
X2 ... 3734
X3 ... 3735
ToNormalMF ... 3735
GetValue ... 3735
CZ_ShapedMembershipFunction ... 3737
A ... 3739

© 2000-2025, MetaQuotes Ltd.

B ... 3739
GetValue ... 3740
IMembershipFunction ... 3741
GetValue ... 3741
Fuzzy systems rules ... 3742
CMamdaniFuzzyRule ... 3743
Conclusion ... 3744
Weight ... 3744
CSugenoFuzzyRule ... 3745
Conclusion ... 3746
CSingleCondition ... 3747
Not ... 3747
Term ... 3748
Var ... 3748
CConditions ... 3750
ConditionsList ... 3750
Not ... 3751
Op ... 3751
CGenericFuzzyRule ... 3752
Conclusion ... 3752
Condition ... 3753
CreateCondition ... 3753
Fuzzy systems variables ... 3755
CFuzzyVariable ... 3756
AddTerm ... 3757
GetTermByName ... 3757
Max ... 3757
Min ... 3758
Terms ... 3758
Values ... 3759
CSugenoVariable ... 3760
Functions ... 3760
GetFuncByName ... 3761
Values ... 3761
Fuzzy terms ... 3762
MembershipFunction ... 3763
Fuzzy systems ... 3764
Mamdani system ... 3765
AggregationMethod ... 3765
Calculate ... 3766
DefuzzificationMethod ... 3766
EmptyRule ... 3766
ImplicationMethod ... 3766
Output ... 3767
OutputByName ... 3767
ParseRule ... 3767
Rules ... 3768
Sugeno system ... 3769
Calculate ... 3769
CreateSugenoFunction ... 3770
EmptyRule ... 3771
Output ... 3771
OutputByName ... 3771
ParseRule ... 3771
Rules ... 3772
OpenCL ... 3773
BufferCreate ... 3775
BufferFree ... 3776

© 2000-2025, MetaQuotes Ltd.

BufferFromArray ... 3777
BufferRead ... 3778
BufferWrite ... 3779
Execute ... 3780
GetContext ... 3781
GetKernel ... 3782
GetKernelName ... 3783
GetProgram ... 3784
Initialize ... 3785
KernelCreate ... 3786
KernelFree ... 3787
SetArgument ... 3788
SetArgumentBuffer ... 3789
SetArgumentLocalMemory ... 3790
SetBuffersCount ... 3791
SetKernelsCount ... 3792
Shutdown ... 3793
SupportDouble ... 3794
Basisklasse CObject ... 3795
Prev ... 3796
Prev ... 3797
Next ... 3798
Next ... 3799
Compare ... 3800
Save ... 3802
Load ... 3804
Type ... 3806
Datensammlungen ... 3807
CArray ... 3809
Step ... 3811
Step ... 3812
Total ... 3813
Available ... 3814
Max ... 3815
IsSorted ... 3816
SortMode ... 3817
Clear ... 3818
Sort ... 3819
Save ... 3820
Load ... 3822
CArrayChar ... 3824
Reserve ... 3827
Resize ... 3828
Shutdown ... 3829
Add ... 3830
AddArray ... 3831
AddArray ... 3832
Insert ... 3834
InsertArray ... 3835
InsertArray ... 3837
AssignArray ... 3839
AssignArray ... 3840
Update ... 3842
Shift ... 3843
Delete ... 3844
DeleteRange ... 3845
At ... 3846
CompareArray ... 3848

© 2000-2025, MetaQuotes Ltd.

CompareArray ... 3849
InsertSort ... 3850
Search ... 3851
SearchGreat ... 3852
SearchLess ... 3853
SearchGreatOrEqual ... 3854
SearchLessOrEqual ... 3855
SearchFirst ... 3856
SearchLast ... 3857
SearchLinear ... 3858
Save ... 3859
Load ... 3861
Type ... 3863
CArrayShort ... 3864
Reserve ... 3867
Resize ... 3868
Shutdown ... 3869
Add ... 3870
AddArray ... 3871
AddArray ... 3872
Insert ... 3874
InsertArray ... 3875
InsertArray ... 3876
AssignArray ... 3878
AssignArray ... 3879
Update ... 3881
Shift ... 3882
Delete ... 3883
DeleteRange ... 3884
At ... 3885
CompareArray ... 3887
CompareArray ... 3888
InsertSort ... 3889
Search ... 3890
SearchGreat ... 3891
SearchLess ... 3892
SearchGreatOrEqual ... 3893
SearchLessOrEqual ... 3894
SearchFirst ... 3895
SearchLast ... 3896
SearchLinear ... 3897
Save ... 3898
Load ... 3900
Type ... 3902
CArrayInt ... 3903
Reserve ... 3906
Resize ... 3907
Shutdown ... 3908
Add ... 3909
AddArray ... 3910
AddArray ... 3911
Insert ... 3913
InsertArray ... 3914
InsertArray ... 3915
AssignArray ... 3917
AssignArray ... 3918
Update ... 3920
Shift ... 3921

© 2000-2025, MetaQuotes Ltd.

Delete ... 3922
DeleteRange ... 3923
At ... 3924
CompareArray ... 3926
CompareArray ... 3927
InsertSort ... 3928
Search ... 3929
SearchGreat ... 3930
SearchLess ... 3931
SearchGreatOrEqual ... 3932
SearchLessOrEqual ... 3933
SearchFirst ... 3934
SearchLast ... 3935
SearchLinear ... 3936
Save ... 3937
Load ... 3939
Type ... 3941
CArrayLong ... 3942
Reserve ... 3945
Resize ... 3946
Shutdown ... 3947
Add ... 3948
AddArray ... 3949
AddArray ... 3950
Insert ... 3952
InsertArray ... 3953
InsertArray ... 3954
AssignArray ... 3956
AssignArray ... 3957
Update ... 3959
Shift ... 3960
Delete ... 3961
DeleteRange ... 3962
At ... 3963
CompareArray ... 3965
CompareArray ... 3966
InsertSort ... 3967
Search ... 3968
SearchGreat ... 3969
SearchLess ... 3970
SearchGreatOrEqual ... 3971
SearchLessOrEqual ... 3972
SearchFirst ... 3973
SearchLast ... 3974
SearchLinear ... 3975
Save ... 3976
Load ... 3978
Type ... 3980
CArrayFloat ... 3981
Delta ... 3984
Reserve ... 3985
Resize ... 3986
Shutdown ... 3987
Add ... 3988
AddArray ... 3989
AddArray ... 3990
Insert ... 3992
InsertArray ... 3993

© 2000-2025, MetaQuotes Ltd.

InsertArray 3994
AssignArray 3996
AssignArray 3997
Update 3999
Shift 4000
Delete 4001
DeleteRange 4002
At 4003
CompareArray 4005
CompareArray 4006
InsertSort 4007
Search 4008
SearchGreat 4009
SearchLess 4010
SearchGreatOrEqual 4011
SearchLessOrEqual 4012
SearchFirst 4013
SearchLast 4014
SearchLinear 4015
Save 4016
Load 4018
Type 4020
CArrayDouble 4021
Delta 4024
Reserve 4025
Resize 4026
Shutdown 4027
Add 4028
AddArray 4029
AddArray 4030
Insert 4032
InsertArray 4033
InsertArray 4034
AssignArray 4036
AssignArray 4037
Update 4039
Shift 4040
Delete 4041
DeleteRange 4042
At 4043
CompareArray 4045
CompareArray 4046
Minimum 4047
Maximum 4048
InsertSort 4049
Search 4050
SearchGreat 4051
SearchLess 4052
SearchGreatOrEqual 4053
SearchLessOrEqual 4054
SearchFirst 4055
SearchLast 4056
SearchLinear 4057
Save 4058
Load 4060
Type 4062
CArrayString 4063
Reserve 4066

© 2000-2025, MetaQuotes Ltd.

Resize ... 4067
Shutdown ... 4068
Add ... 4069
AddArray ... 4070
AddArray ... 4071
Insert ... 4073
InsertArray ... 4074
InsertArray ... 4075
AssignArray ... 4077
AssignArray ... 4078
Update ... 4080
Shift ... 4081
Delete ... 4082
DeleteRange ... 4083
At ... 4084
CompareArray ... 4086
CompareArray ... 4087
InsertSort ... 4088
Search ... 4089
SearchGreat ... 4090
SearchLess ... 4091
SearchGreatOrEqual ... 4092
SearchLessOrEqual ... 4093
SearchFirst ... 4094
SearchLast ... 4095
SearchLinear ... 4096
Save ... 4097
Load ... 4099
Type ... 4101
CArrayObj ... 4102
FreeMode ... 4107
FreeMode ... 4108
Reserve ... 4110
Resize ... 4111
Clear ... 4113
Shutdown ... 4114
CreateElement ... 4115
Add ... 4117
AddArray ... 4119
Insert ... 4122
InsertArray ... 4124
AssignArray ... 4126
Update ... 4128
Shift ... 4130
Detach ... 4131
Delete ... 4133
DeleteRange ... 4134
At ... 4136
CompareArray ... 4138
InsertSort ... 4139
Search ... 4140
SearchGreat ... 4142
SearchLess ... 4144
SearchGreatOrEqual ... 4146
SearchLessOrEqual ... 4148
SearchFirst ... 4150
SearchLast ... 4152
Save ... 4154

© 2000-2025, MetaQuotes Ltd.

Load ... 4156
Type ... 4158
CList ... 4159
FreeMode ... 4162
FreeMode ... 4163
Total ... 4165
IsSorted ... 4166
SortMode ... 4167
CreateElement ... 4168
Add ... 4169
Insert ... 4171
DetachCurrent ... 4173
DeleteCurrent ... 4174
Delete ... 4175
Clear ... 4176
IndexOf ... 4177
GetNodeAtIndex ... 4178
GetFirstNode ... 4179
GetPrevNode ... 4180
GetCurrentNode ... 4181
GetNextNode ... 4182
GetLastNode ... 4183
Sort ... 4184
MoveToIndex ... 4185
Exchange ... 4186
CompareList ... 4187
Search ... 4188
Save ... 4190
Load ... 4192
Type ... 4194
CTreeNode ... 4195
Owner ... 4200
Left ... 4201
Right ... 4202
Balance ... 4203
BalanceL ... 4204
BalanceR ... 4205
CreateSample ... 4206
RefreshBalance ... 4207
GetNext ... 4208
SaveNode ... 4209
LoadNode ... 4210
Type ... 4211
CTree ... 4212
Root ... 4218
CreateElement ... 4219
Insert ... 4220
Detach ... 4221
Delete ... 4222
Clear ... 4223
Find ... 4224
Save ... 4225
Load ... 4226
Type ... 4227
Template-Sammlungen von Daten ... 4228
ICollection<T> ... 4231
Add ... 4232
Count ... 4233

© 2000-2025, MetaQuotes Ltd.

Contains ... 4234
CopyTo ... 4235
Clear ... 4236
Remove ... 4237
IEqualityComparable<T> ... 4238
Equals ... 4239
HashCode ... 4240
IComparable<T> ... 4241
Compare ... 4242
IComparer<T> ... 4243
Compare ... 4244
IEqualityComparer<T> ... 4245
Equals ... 4246
HashCode ... 4247
IList<T> ... 4248
TryGetValue ... 4249
TrySetValue ... 4250
Insert ... 4251
IndexOf ... 4252
LastIndexOf ... 4253
RemoveAt ... 4254
IMap<TKey,TValue> ... 4255
Add ... 4256
Contains ... 4257
Remove ... 4258
TryGetValue ... 4259
TrySetValue ... 4260
CopyTo ... 4261
ISet<T> ... 4262
ExceptWith ... 4264
IntersectWith ... 4265
SymmetricExceptWith ... 4266
UnionWith ... 4267
IsProperSubsetOf ... 4268
IsProperSupersetOf ... 4269
IsSubsetOf ... 4270
IsSupersetOf ... 4271
Overlaps ... 4272
SetEquals ... 4273
CDefaultComparer<T> ... 4274
Compare ... 4275
CDefaultEqualityComparer<T> ... 4276
Equals ... 4277
HashCode ... 4278
CRedBlackTreeNode<T> ... 4279
Value ... 4280
Parent ... 4281
Left ... 4282
Right ... 4283
Color ... 4284
IsLeaf ... 4285
CreateEmptyNode ... 4286
CLinkedListNode<T> ... 4287
List ... 4288
Next ... 4289
Previous ... 4290
Value ... 4291
CKeyValuePair<TKey,TValue> ... 4292

© 2000-2025, MetaQuotes Ltd.

Key ... 4293
Value ... 4294
Clone ... 4295
Compare ... 4296
Equals ... 4297
HashCode ... 4298
CArrayList<T> ... 4299
Capacity ... 4301
Count ... 4302
Contains ... 4303
TrimExcess ... 4304
TryGetValue ... 4305
TrySetValue ... 4306
Add ... 4307
AddRange ... 4308
Insert ... 4309
InsertRange ... 4310
CopyTo ... 4311
BinarySearch ... 4312
IndexOf ... 4313
LastIndexOf ... 4314
Clear ... 4315
Remove ... 4316
RemoveAt ... 4317
RemoveRange ... 4318
Reverse ... 4319
Sort ... 4320
CHashMap<TKey,TValue> ... 4321
Add ... 4323
Count ... 4324
Comparer ... 4325
Contains ... 4326
ContainsKey ... 4327
ContainsValue ... 4328
CopyTo ... 4329
Clear ... 4330
Remove ... 4331
TryGetValue ... 4332
TrySetValue ... 4333
CHashSet<T> ... 4334
Add ... 4336
Count ... 4337
Contains ... 4338
Comparer ... 4339
TrimExcess ... 4340
CopyTo ... 4341
Clear ... 4342
Remove ... 4343
ExceptWith ... 4344
IntersectWith ... 4345
SymmetricExceptWith ... 4346
UnionWith ... 4347
IsProperSubsetOf ... 4348
IsProperSupersetOf ... 4349
IsSubsetOf ... 4350
IsSupersetOf ... 4351
Overlaps ... 4352
SetEquals ... 4353

© 2000-2025, MetaQuotes Ltd.

CLinkedList<T> 4354
Add 4356
AddAfter 4357
AddBefore 4358
AddFirst 4359
AddLast 4360
Count 4361
Head 4362
First 4363
Last 4364
Contains 4365
CopyTo 4366
Clear 4367
Remove 4368
RemoveFirst 4369
RemoveLast 4370
Find 4371
FindLast 4372
CQueue<T> 4373
Add 4374
Enqueue 4375
Count 4376
Contains 4377
TrimExcess 4378
CopyTo 4379
Clear 4380
Remove 4381
Dequeue 4382
Peek 4383
CRedBlackTree<T> 4384
Add 4386
Count 4387
Root 4388
Contains 4389
Comparer 4390
TryGetMin 4391
TryGetMax 4392
CopyTo 4393
Clear 4394
Remove 4395
RemoveMin 4396
RemoveMax 4397
Find 4398
FindMin 4399
FindMax 4400
CSortedMap<TKey, TValue> 4401
Add 4403
Count 4404
Comparer 4405
Contains 4406
ContainsKey 4407
ContainsValue 4408
CopyTo 4409
Clear 4410
Remove 4411
TryGetValue 4412
TrySetValue 4413
CSortedSet<T> 4414

© 2000-2025, MetaQuotes Ltd.

Add ... 4416
Count ... 4417
Contains ... 4418
Comparer ... 4419
TryGetMin ... 4420
TryGetMax ... 4421
CopyTo ... 4422
Clear ... 4423
Remove ... 4424
ExceptWith ... 4425
IntersectWith ... 4426
SymmetricExceptWith ... 4427
UnionWith ... 4428
IsProperSubsetOf ... 4429
IsProperSupersetOf ... 4430
IsSubsetOf ... 4431
IsSupersetOf ... 4432
Overlaps ... 4433
SetEquals ... 4434
GetViewBetween ... 4435
GetReverse ... 4436
CStack<T> ... 4437
Add ... 4438
Count ... 4439
Contains ... 4440
TrimExcess ... 4441
CopyTo ... 4442
Clear ... 4443
Remove ... 4444
Push ... 4445
Peek ... 4446
Pop ... 4447
ArrayBinarySearch<T> ... 4448
ArrayIndexOf<T> ... 4449
ArrayLastIndexOf<T> ... 4450
ArrayReverse<T> ... 4451
Compare ... 4452
Equals<T> ... 4455
GetHashCode ... 4456
Dateien ... 4459
CFile ... 4460
Handle ... 4462
Filename ... 4463
Flags ... 4464
SetUnicode ... 4465
SetCommon ... 4466
Open ... 4467
Close ... 4468
Delete ... 4469
IsExist ... 4470
Copy ... 4471
Move ... 4472
Size ... 4473
Tell ... 4474
Seek ... 4475
Flush ... 4476
IsEnding ... 4477
IsLineEnding ... 4478

© 2000-2025, MetaQuotes Ltd.

FolderCreate ... 4479
FolderDelete ... 4480
FolderClean ... 4481
FileFindFirst ... 4482
FileFindNext ... 4483
FileFindClose ... 4484
CFileBin ... 4485
Open ... 4487
WriteChar ... 4488
WriteShort ... 4489
WriteInteger ... 4490
WriteLong ... 4491
WriteFloat ... 4492
WriteDouble ... 4493
WriteString ... 4494
WriteCharArray ... 4495
WriteShortArray ... 4496
WriteIntegerArray ... 4497
WriteLongArray ... 4498
WriteFloatArray ... 4499
WriteDoubleArray ... 4500
WriteObject ... 4501
ReadChar ... 4502
ReadShort ... 4503
ReadInteger ... 4504
ReadLong ... 4505
ReadFloat ... 4506
ReadDouble ... 4507
ReadString ... 4508
ReadCharArray ... 4509
ReadShortArray ... 4510
ReadIntegerArray ... 4511
ReadLongArray ... 4512
ReadFloatArray ... 4513
ReadDoubleArray ... 4514
ReadObject ... 4515
CFileTxt ... 4516
Open ... 4517
WriteString ... 4518
ReadString ... 4519
Strings ... 4520
CString ... 4521
Str ... 4523
Len ... 4524
Copy ... 4525
Fill ... 4526
Assign ... 4527
Append ... 4528
Insert ... 4529
Compare ... 4530
CompareNoCase ... 4531
Left ... 4532
Right ... 4533
Mid ... 4534
Trim ... 4535
TrimLeft ... 4536
TrimRight ... 4537
Clear ... 4538

© 2000-2025, MetaQuotes Ltd.

ToUpper ... 4539
ToLower ... 4540
Reverse ... 4541
Find ... 4542
FindRev ... 4543
Remove ... 4544
Replace ... 4545
Grafische Objekte ... 4546
CChartObject ... 4547
ChartId ... 4550
Window ... 4551
Name ... 4552
NumPoints ... 4553
Attach ... 4554
SetPoint ... 4555
Delete ... 4556
Detach ... 4557
ShiftObject ... 4558
ShiftPoint ... 4559
Time ... 4560
Price ... 4562
Color ... 4564
Style ... 4565
Width ... 4566
Background ... 4567
Selected ... 4568
Selectable ... 4569
Description ... 4570
Tooltip ... 4571
Timeframes ... 4572
Z_Order ... 4573
CreateTime ... 4574
LevelsCount ... 4575
LevelColor ... 4576
LevelStyle ... 4578
LevelWidth ... 4580
LevelValue ... 4582
LevelDescription ... 4584
GetInteger ... 4586
SetInteger ... 4588
GetDouble ... 4590
SetDouble ... 4592
GetString ... 4594
SetString ... 4596
Save ... 4598
Load ... 4599
Type ... 4600
Objekte "Linien" ... 4601
CChartObjectVLine ... 4602
Create ... 4603
Type ... 4604
CChartObjectHLine ... 4605
Create ... 4606
Type ... 4607
CChartObjectTrend ... 4608
Create ... 4610
RayLeft ... 4611
RayRight ... 4612

© 2000-2025, MetaQuotes Ltd.

Save ... 4613
Load ... 4614
Type ... 4615
CChartObjectTrendByAngle ... 4616
Create ... 4618
Angle ... 4619
Type ... 4620
CChartObjectCycles ... 4621
Create ... 4622
Type ... 4623
Objekte "Kanäle" ... 4624
CChartObjectChannel ... 4625
Create ... 4627
Type ... 4628
CChartObjectRegression ... 4629
Create ... 4631
Type ... 4632
CChartObjectStdDevChannel ... 4633
Create ... 4635
Deviations ... 4636
Save ... 4637
Load ... 4638
Type ... 4639
CChartObjectPitchfork ... 4640
Create ... 4642
Type ... 4643
Gann-Werkzeuge ... 4644
CChartObjectGannLine ... 4645
Create ... 4647
PipsPerBar ... 4648
Save ... 4649
Load ... 4650
Type ... 4651
CChartObjectGannFan ... 4652
Create ... 4654
PipsPerBar ... 4655
Downtrend ... 4656
Save ... 4657
Load ... 4658
Type ... 4659
CChartObjectGannGrid ... 4660
Create ... 4662
PipsPerBar ... 4663
Downtrend ... 4664
Save ... 4665
Load ... 4666
Type ... 4667
Fibonacci-Werkzeuge ... 4668
CChartObjectFibo ... 4669
Create ... 4671
Type ... 4672
CChartObjectFiboTimes ... 4673
Create ... 4674
Type ... 4675
CChartObjectFiboFan ... 4676
Create ... 4677
Type ... 4678
CChartObjectFiboArc ... 4679

© 2000-2025, MetaQuotes Ltd.

Create ... 4681
Scale ... 4682
Ellipse ... 4683
Save ... 4684
Load ... 4685
Type ... 4686
CChartObjectFiboChannel ... 4687
Create ... 4689
Type ... 4690
CChartObjectFiboExpansion ... 4691
Create ... 4693
Type ... 4694
Elliott-Werkzeuge ... 4695
CChartObjectElliottWave3 ... 4696
Create ... 4698
Degree ... 4699
Lines ... 4700
Save ... 4701
Load ... 4702
Type ... 4703
CChartObjectElliottWave5 ... 4704
Create ... 4706
Type ... 4708
Objekte "Figuren" ... 4709
CChartObjectRectangle ... 4710
Create ... 4711
Type ... 4712
CChartObjectTriangle ... 4713
Create ... 4714
Type ... 4715
CChartObjectEllipse ... 4716
Create ... 4717
Type ... 4718
Objekte "Pfeile" ... 4719
CChartObjectArrow ... 4720
Create ... 4722
ArrowCode ... 4724
Anchor ... 4726
Save ... 4728
Load ... 4729
Type ... 4730
Arrows with fixed code ... 4731
Create ... 4733
ArrowCode ... 4735
Type ... 4737
Steuerelemente ... 4738
CChartObjectText ... 4739
Create ... 4741
Angle ... 4742
Font ... 4743
FontSize ... 4744
Anchor ... 4745
Save ... 4746
Load ... 4747
Type ... 4748
CChartObjectLabel ... 4749
Create ... 4751
X_Distance ... 4752

© 2000-2025, MetaQuotes Ltd.

Y_Distance ... 4753
X_Size ... 4754
Y_Size ... 4755
Corner ... 4756
Time ... 4757
Price ... 4758
Save ... 4759
Load ... 4760
Type ... 4761
CChartObjectEdit ... 4762
Create ... 4764
TextAlign ... 4765
X_Size ... 4766
Y_Size ... 4767
BackColor ... 4768
BorderColor ... 4769
ReadOnly ... 4770
Angle ... 4771
Save ... 4772
Load ... 4773
Type ... 4774
CChartObjectButton ... 4775
State ... 4777
Save ... 4778
Load ... 4779
Type ... 4780
CChartObjectSubChart ... 4781
Create ... 4783
X_Distance ... 4784
Y_Distance ... 4785
Corner ... 4786
X_Size ... 4787
Y_Size ... 4788
Symbol ... 4789
Period ... 4790
Scale ... 4791
DateScale ... 4792
PriceScale ... 4793
Time ... 4794
Price ... 4795
Save ... 4796
Load ... 4797
Type ... 4798
CChartObjectBitmap ... 4799
Create ... 4801
BmpFile ... 4802
X_Offset ... 4803
Y_Offset ... 4804
Save ... 4805
Load ... 4806
Type ... 4807
CChartObjectBmpLabel ... 4808
Create ... 4810
X_Distance ... 4811
Y_Distance ... 4812
X_Offset ... 4813
Y_Offset ... 4814
Corner ... 4815

© 2000-2025, MetaQuotes Ltd.

X_Size ... 4816
Y_Size ... 4817
BmpFileOn ... 4818
BmpFileOff ... 4819
State ... 4820
Time ... 4821
Price ... 4822
Save ... 4823
Load ... 4824
Type ... 4825
CChartObjectRectLabel ... 4826
Create ... 4828
X_Size ... 4829
Y_Size ... 4830
BackColor ... 4831
Angle ... 4832
BorderType ... 4833
Save ... 4834
Load ... 4835
Type ... 4836
Benutzerdefinierte Grafiken ... 4837
CCanvas ... 4838
Attach ... 4842
Arc ... 4843
Pie ... 4847
FillPolygon ... 4851
FillEllipse ... 4852
GetDefaultColor ... 4853
ChartObjectName ... 4854
Circle ... 4855
CircleAA ... 4856
CircleWu ... 4857
Create ... 4858
CreateBitmap ... 4859
CreateBitmapLabel ... 4861
Destroy ... 4863
Ellipse ... 4864
EllipseAA ... 4865
EllipseWu ... 4866
Erase ... 4867
Fill ... 4868
FillCircle ... 4869
FillRectangle ... 4870
FillTriangle ... 4871
FontAngleGet ... 4872
FontAngleSet ... 4873
FontFlagsGet ... 4874
FontFlagsSet ... 4875
FontGet ... 4876
FontNameGet ... 4877
FontNameSet ... 4878
FontSet ... 4879
FontSizeGet ... 4880
FontSizeSet ... 4881
Height ... 4882
Line ... 4883
LineAA ... 4884
LineWu ... 4885

© 2000-2025, MetaQuotes Ltd.

LineHorizontal ... 4886
LineVertical ... 4887
LineStyleSet ... 4888
LineThick ... 4889
LineThickVertical ... 4891
LineThickHorizontal ... 4892
LoadFromFile ... 4893
PixelGet ... 4894
PixelSet ... 4895
PixelSetAA ... 4896
Polygon ... 4897
PolygonAA ... 4898
PolygonWu ... 4899
PolygonThick ... 4900
PolygonSmooth ... 4901
Polyline ... 4902
PolylineSmooth ... 4903
PolylineThick ... 4904
PolylineWu ... 4905
PolylineAA ... 4906
Rectangle ... 4907
Resize ... 4908
ResourceName ... 4909
TextHeight ... 4910
TextOut ... 4911
TextSize ... 4912
TextWidth ... 4913
TransparentLevelSet ... 4914
Triangle ... 4915
TriangleAA ... 4916
TriangleWu ... 4917
Update ... 4918
Width ... 4919
CChartCanvas ... 4920
ColorBackground ... 4924
ColorBorder ... 4925
ColorText ... 4926
ColorGrid ... 4927
MaxData ... 4928
MaxDescrLen ... 4929
ShowFlags ... 4930
IsShowLegend ... 4931
IsShowScaleLeft ... 4932
IsShowScaleRight ... 4933
IsShowScaleTop ... 4934
IsShowScaleBottom ... 4935
IsShowGrid ... 4936
IsShowDescriptors ... 4937
IsShowPercent ... 4938
VScaleMin ... 4939
VScaleMax ... 4940
NumGrid ... 4941
DataOffset ... 4942
DataTotal ... 4943
DrawDescriptors ... 4944
DrawData ... 4945
Create ... 4946
AllowedShowFlags ... 4947

© 2000-2025, MetaQuotes Ltd.

ShowLegend 4948
ShowScaleLeft 4949
ShowScaleRight 4950
ShowScaleTop 4951
ShowScaleBottom 4952
ShowGrid 4953
ShowDescriptors 4954
ShowValue 4955
ShowPercent 4956
LegendAlignment 4957
Accumulative 4958
VScaleParams 4959
DescriptorUpdate 4960
ColorUpdate 4961
ValuesCheck 4962
Redraw 4963
DrawBackground 4964
DrawLegend 4965
DrawLegendVertical 4966
DrawLegendHorizontal 4967
CalcScales 4968
DrawScales 4969
DrawScaleLeft 4970
DrawScaleRight 4971
DrawScaleTop 4972
DrawScaleBottom 4973
DrawGrid 4974
DrawChart 4975
CHistogramChart 4976
Gradient 4981
BarGap 4982
BarMinSize 4983
BarBorder 4984
Create 4985
SeriesAdd 4986
SeriesInsert 4987
SeriesUpdate 4988
SeriesDelete 4989
ValueUpdate 4990
DrawData 4991
DrawBar 4992
GradientBrush 4993
CLineChart 4994
Filled 4998
Create 4999
SeriesAdd 5000
SeriesInsert 5001
SeriesUpdate 5002
SeriesDelete 5003
ValueUpdate 5004
DrawChart 5005
DrawData 5006
CalcArea 5007
CPieChart 5008
Create 5012
SeriesSet 5013
ValueAdd 5014
ValueInsert 5015

© 2000-2025, MetaQuotes Ltd.

ValueUpdate 5016
ValueDelete 5017
DrawChart 5018
DrawPie 5019
LabelMake 5020
3D Grafik 5021
CCanvas3D 5022
AmbientColorGet 5024
AmbientColorSet 5025
Attach 5026
Create 5027
Destroy 5028
DXContext 5029
DXDispatcher 5030
InputScene 5031
LightColorGet 5032
LightColorSet 5033
LightDirectionGet 5034
LightDirectionSet 5035
ObjectAdd 5036
ProjectionMatrixGet 5037
ProjectionMatrixSet 5038
Render 5039
RenderBegin 5040
RenderEnd 5041
ViewMatrixGet 5042
ViewMatrixSet 5043
ViewPositionSet 5044
ViewRotationSet 5045
ViewTargetSet 5046
ViewUpDirectionSet 5047
Preischarts 5048
ChartID 5053
Mode 5054
Foreground 5055
Shift 5056
ShiftSize 5057
AutoScroll 5058
Scale 5059
ScaleFix 5060
ScaleFix_11 5061
FixedMax 5062
FixedMin 5063
PointsPerBar 5064
ScalePPB 5065
ShowOHLC 5066
ShowLineBid 5067
ShowLineAsk 5068
ShowLastLine 5069
ShowPeriodSep 5070
ShowGrid 5071
ShowVolumes 5072
ShowObjectDescr 5073
ShowDateScale 5074
ShowPriceScale 5075
ColorBackground 5076
ColorForeground 5077
ColorGrid 5078

© 2000-2025, MetaQuotes Ltd.

ColorBarUp ... 5079
ColorBarDown ... 5080
ColorCandleBull ... 5081
ColorCandleBear ... 5082
ColorChartLine ... 5083
ColorVolumes ... 5084
ColorLineBid ... 5085
ColorLineAsk ... 5086
ColorLineLast ... 5087
ColorStopLevels ... 5088
VisibleBars ... 5089
WindowsTotal ... 5090
WindowIsVisible ... 5091
WindowHandle ... 5092
FirstVisibleBar ... 5093
WidthInBars ... 5094
WidthInPixels ... 5095
HeightInPixels ... 5096
PriceMin ... 5097
PriceMax ... 5098
Attach ... 5099
FirstChart ... 5100
NextChart ... 5101
Open ... 5102
Detach ... 5103
Close ... 5104
BringToTop ... 5105
EventObjectCreate ... 5106
EventObjectDelete ... 5107
IndicatorAdd ... 5108
IndicatorDelete ... 5109
IndicatorsTotal ... 5110
IndicatorName ... 5111
Navigate ... 5112
Symbol ... 5113
Period ... 5114
Redraw ... 5115
GetInteger ... 5116
SetInteger ... 5117
GetDouble ... 5118
SetDouble ... 5119
GetString ... 5120
SetString ... 5121
SetSymbolPeriod ... 5122
ApplyTemplate ... 5123
ScreenShot ... 5124
WindowOnDropped ... 5125
PriceOnDropped ... 5126
TimeOnDropped ... 5127
XOnDropped ... 5128
YOnDropped ... 5129
Save ... 5130
Load ... 5131
Type ... 5132
Wissenschaftliche Grafiken ... 5133
GraphPlot ... 5134
CAxis ... 5138
AutoScale ... 5140

© 2000-2025, MetaQuotes Ltd.

Min ... 5141
Max ... 5142
Step ... 5143
Name ... 5144
Color ... 5145
ValuesSize ... 5146
ValuesWidth ... 5147
ValuesFormat ... 5148
ValuesDateTimeMode ... 5149
ValuesFunctionFormat ... 5150
ValuesFunctionFormatCBData ... 5152
NameSize ... 5153
ZeroLever ... 5154
DefaultStep ... 5155
MaxLabels ... 5156
MinGrace ... 5157
MaxGrace ... 5158
SelectAxisScale ... 5159
CColorGenerator ... 5160
Next ... 5161
Reset ... 5162
CCurve ... 5163
Type ... 5166
Name ... 5167
Color ... 5168
XMax ... 5169
XMin ... 5170
YMax ... 5171
YMin ... 5172
Size ... 5173
PointsSize ... 5174
PointsFill ... 5175
PointsColor ... 5176
GetX ... 5177
GetY ... 5178
LinesStyle ... 5179
LinesIsSmooth ... 5180
LinesSmoothTension ... 5181
LinesSmoothStep ... 5182
LinesWidth ... 5183
LinesEndStyle ... 5185
HistogramWidth ... 5186
CustomPlotCBData ... 5187
CustomPlotFunction ... 5188
PointsType ... 5192
StepsDimension ... 5193
TrendLineCoefficients ... 5194
TrendLineColor ... 5195
TrendLineVisible ... 5196
Update ... 5198
Visible ... 5200
CGraphic ... 5201
Create ... 5204
Destroy ... 5205
Update ... 5206
ChartObjectName ... 5207
ResourceName ... 5208
XAxis ... 5209

© 2000-2025, MetaQuotes Ltd.

YAxis ... 5210
GapSize ... 5211
BackgroundColor ... 5212
BackgroundMain ... 5213
BackgroundMainSize ... 5214
BackgroundMainColor ... 5215
BackgroundSub ... 5216
BackgroundSubSize ... 5217
BackgroundSubColor ... 5218
GridLineColor ... 5219
GridBackgroundColor ... 5220
GridCircleRadius ... 5221
GridCircleColor ... 5222
GridHasCircle ... 5223
GridAxisLineColor ... 5224
HistoryNameWidth ... 5225
HistoryNameSize ... 5226
HistorySymbolSize ... 5227
TextAdd ... 5228
LineAdd ... 5229
CurveAdd ... 5230
CurvePlot ... 5233
CurvePlotAll ... 5234
CurveGetByIndex ... 5235
CurveGetByName ... 5236
CurveRemoveByIndex ... 5237
CurveRemoveByName ... 5238
CurvesTotal ... 5239
MarksToAxisAdd ... 5240
MajorMarkSize ... 5241
FontSet ... 5242
FontGet ... 5243
Attach ... 5244
CalculateMaxMinValues ... 5245
Height ... 5246
IndentDown ... 5247
IndentLeft ... 5248
IndentRight ... 5249
IndentUp ... 5250
Redraw ... 5251
ResetParameters ... 5252
ScaleX ... 5253
ScaleY ... 5254
SetDefaultParameters ... 5255
Width ... 5256
Indikatoren ... 5257
Basisklassen ... 5258
CSpreadBuffer ... 5259
Size ... 5261
SetSymbolPeriod ... 5262
At ... 5263
Refresh ... 5264
RefreshCurrent ... 5265
CTimeBuffer ... 5266
Size ... 5268
SetSymbolPeriod ... 5269
At ... 5270
Refresh ... 5271

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent 5272
CTickVolumeBuffer 5273
Size 5275
SetSymbolPeriod 5276
At 5277
Refresh 5278
RefreshCurrent 5279
CRealVolumeBuffer 5280
Size 5282
SetSymbolPeriod 5283
At 5284
Refresh 5285
RefreshCurrent 5286
CDoubleBuffer 5287
Size 5289
SetSymbolPeriod 5290
At 5291
Refresh 5292
RefreshCurrent 5293
COpenBuffer 5294
Refresh 5295
RefreshCurrent 5296
CHighBuffer 5297
Refresh 5298
RefreshCurrent 5299
CLowBuffer 5300
Refresh 5301
RefreshCurrent 5302
CCloseBuffer 5303
Refresh 5304
RefreshCurrent 5305
CIndicatorBuffer 5306
Offset 5308
Name 5309
At 5310
Refresh 5311
RefreshCurrent 5312
CSeries 5313
Name 5315
BuffersTotal 5316
Timeframe 5317
Symbol 5318
Period 5319
RefreshCurrent 5320
BufferSize 5321
BufferResize 5322
Refresh 5323
PeriodDescription 5324
CPriceSeries 5325
BufferResize 5327
GetData 5328
Refresh 5329
MinIndex 5330
MinValue 5331
MaxIndex 5332
MaxValue 5333
CIndicator 5334
Handle 5336

© 2000-2025, MetaQuotes Ltd.

Status ... 5337
FullRelease ... 5338
Create ... 5339
BufferResize ... 5340
BarsCalculated ... 5341
GetData ... 5342
Refresh ... 5345
Minimum ... 5346
MinValue ... 5347
Maximum ... 5348
MaxValue ... 5349
MethodDescription ... 5350
PriceDescription ... 5351
VolumeDescription ... 5352
AddToChart ... 5353
DeleteFromChart ... 5354
CIndicators ... 5355
Create ... 5356
Refresh ... 5357
Klassen für Arbeit mit Zeitreihen ... 5358
CiSpread ... 5359
Create ... 5361
BufferResize ... 5362
GetData ... 5363
Refresh ... 5365
CiTime ... 5366
Create ... 5368
BufferResize ... 5369
GetData ... 5370
Refresh ... 5372
CiTickVolume ... 5373
Create ... 5375
BufferResize ... 5376
GetData ... 5377
Refresh ... 5379
CiRealVolume ... 5380
Create ... 5382
BufferResize ... 5383
GetData ... 5384
Refresh ... 5386
CiOpen ... 5387
Create ... 5389
GetData ... 5390
CiHigh ... 5392
Create ... 5394
GetData ... 5395
CiLow ... 5397
Create ... 5399
GetData ... 5400
CiClose ... 5402
Create ... 5404
GetData ... 5405
Trendindikatoren ... 5407
CiADX ... 5408
MaPeriod ... 5410
Create ... 5411
Main ... 5412
Plus ... 5413

© 2000-2025, MetaQuotes Ltd.

Minus ... 5414
Type ... 5415
CiADXWilder ... 5416
MaPeriod ... 5418
Create ... 5419
Main ... 5420
Plus ... 5421
Minus ... 5422
Type ... 5423
CiBands ... 5424
MaPeriod ... 5426
MaShift ... 5427
Deviation ... 5428
Applied ... 5429
Create ... 5430
Base ... 5431
Upper ... 5432
Lower ... 5433
Type ... 5434
CiEnvelopes ... 5435
MaPeriod ... 5437
MaShift ... 5438
MaMethod ... 5439
Deviation ... 5440
Applied ... 5441
Create ... 5442
Upper ... 5443
Lower ... 5444
Type ... 5445
CiIchimoku ... 5446
TenkanSenPeriod ... 5448
KijunSenPeriod ... 5449
SenkouSpanBPeriod ... 5450
Create ... 5451
TenkanSen ... 5452
KijunSen ... 5453
SenkouSpanA ... 5454
SenkouSpanB ... 5455
ChinkouSpan ... 5456
Type ... 5457
CiMA ... 5458
MaPeriod ... 5460
MaShift ... 5461
MaMethod ... 5462
Applied ... 5463
Create ... 5464
Main ... 5465
Type ... 5466
CiSAR ... 5467
SarStep ... 5469
Maximum ... 5470
Create ... 5471
Main ... 5472
Type ... 5473
CiStdDev ... 5474
MaPeriod ... 5476
MaShift ... 5477
MaMethod ... 5478

© 2000-2025, MetaQuotes Ltd.

Applied ... 5479
Create ... 5480
Main ... 5481
Type ... 5482
CiDEMA ... 5483
MaPeriod ... 5485
IndShift ... 5486
Applied ... 5487
Create ... 5488
Main ... 5489
Type ... 5490
CiTEMA ... 5491
MaPeriod ... 5493
IndShift ... 5494
Applied ... 5495
Create ... 5496
Main ... 5497
Type ... 5498
CiFrAMA ... 5499
MaPeriod ... 5501
IndShift ... 5502
Applied ... 5503
Create ... 5504
Main ... 5505
Type ... 5506
CiAMA ... 5507
MaPeriod ... 5509
FastEmaPeriod ... 5510
SlowEmaPeriod ... 5511
IndShift ... 5512
Applied ... 5513
Create ... 5514
Main ... 5515
Type ... 5516
CiVIDyA ... 5517
CmoPeriod ... 5519
EmaPeriod ... 5520
IndShift ... 5521
Applied ... 5522
Create ... 5523
Main ... 5524
Type ... 5525
Oszillatoren ... 5526
CiATR ... 5527
MaPeriod ... 5529
Create ... 5530
Main ... 5531
Type ... 5532
CiBearsPower ... 5533
MaPeriod ... 5535
Create ... 5536
Main ... 5537
Type ... 5538
CiBullsPower ... 5539
MaPeriod ... 5541
Create ... 5542
Main ... 5543
Type ... 5544

© 2000-2025, MetaQuotes Ltd.

CiCCI ... 5545
MaPeriod ... 5547
Applied ... 5548
Create ... 5549
Main ... 5550
Type ... 5551
CiChaikin ... 5552
FastMaPeriod ... 5554
SlowMaPeriod ... 5555
MaMethod ... 5556
Applied ... 5557
Create ... 5558
Main ... 5559
Type ... 5560
CiDeMarker ... 5561
MaPeriod ... 5563
Create ... 5564
Main ... 5565
Type ... 5566
CiForce ... 5567
MaPeriod ... 5569
MaMethod ... 5570
Applied ... 5571
Create ... 5572
Main ... 5573
Type ... 5574
CiMACD ... 5575
FastEmaPeriod ... 5577
SlowEmaPeriod ... 5578
SignalPeriod ... 5579
Applied ... 5580
Create ... 5581
Main ... 5582
Signal ... 5583
Type ... 5584
CiMomentum ... 5585
MaPeriod ... 5587
Applied ... 5588
Create ... 5589
Main ... 5590
Type ... 5591
CiOsMA ... 5592
FastEmaPeriod ... 5594
SlowEmaPeriod ... 5595
SignalPeriod ... 5596
Applied ... 5597
Create ... 5598
Main ... 5599
Type ... 5600
CiRSI ... 5601
MaPeriod ... 5603
Applied ... 5604
Create ... 5605
Main ... 5606
Type ... 5607
CiRVI ... 5608
MaPeriod ... 5610
Create ... 5611

© 2000-2025, MetaQuotes Ltd.

Main ... 5612
Signal ... 5613
Type ... 5614
CiStochastic ... 5615
Kperiod ... 5617
Dperiod ... 5618
Slowing ... 5619
MaMethod ... 5620
PriceField ... 5621
Create ... 5622
Main ... 5623
Signal ... 5624
Type ... 5625
CiTriX ... 5626
MaPeriod ... 5628
Applied ... 5629
Create ... 5630
Main ... 5631
Type ... 5632
CiWPR ... 5633
CalcPeriod ... 5635
Create ... 5636
Main ... 5637
Type ... 5638
Volumenindikatoren ... 5639
CiAD ... 5640
Applied ... 5642
Create ... 5643
Main ... 5644
Type ... 5645
CiMFI ... 5646
MaPeriod ... 5648
Applied ... 5649
Create ... 5650
Main ... 5651
Type ... 5652
CiOBV ... 5653
Applied ... 5655
Create ... 5656
Main ... 5657
Type ... 5658
CiVolumes ... 5659
Applied ... 5661
Create ... 5662
Main ... 5663
Type ... 5664
Bill Williams Indikatoren ... 5665
CiAC ... 5666
Create ... 5668
Main ... 5669
Type ... 5670
CiAlligator ... 5671
JawPeriod ... 5673
JawShift ... 5674
TeethPeriod ... 5675
TeethShift ... 5676
LipsPeriod ... 5677
LipsShift ... 5678

© 2000-2025, MetaQuotes Ltd.

MaMethod ... 5679
Applied ... 5680
Create ... 5681
Jaw ... 5682
Teeth ... 5683
Lips ... 5684
Type ... 5685
CiAO ... 5686
Create ... 5688
Main ... 5689
Type ... 5690
CiFractals ... 5691
Create ... 5693
Upper ... 5694
Lower ... 5695
Type ... 5696
CiGator ... 5697
JawPeriod ... 5699
JawShift ... 5700
TeethPeriod ... 5701
TeethShift ... 5702
LipsPeriod ... 5703
LipsShift ... 5704
MaMethod ... 5705
Applied ... 5706
Create ... 5707
Upper ... 5709
Lower ... 5710
Type ... 5711
CiBWMFI ... 5712
Applied ... 5714
Create ... 5715
Main ... 5716
Type ... 5717
Benutzerdefinierter Indikator ... 5718
NumBuffers ... 5719
NumParams ... 5720
ParamType ... 5721
ParamLong ... 5722
ParamDouble ... 5723
ParamString ... 5724
Type ... 5725
Handelsklassen ... 5726
CAccountInfo ... 5727
Login ... 5729
TradeMode ... 5730
TradeModeDescription ... 5731
Leverage ... 5732
StopoutMode ... 5733
StopoutModeDescription ... 5734
MarginMode ... 5735
MarginModeDescription ... 5736
TradeAllowed ... 5737
TradeExpert ... 5738
LimitOrders ... 5739
Balance ... 5740
Credit ... 5741
Profit ... 5742

© 2000-2025, MetaQuotes Ltd.

Equity ... 5743
Margin ... 5744
FreeMargin ... 5745
MarginLevel ... 5746
MarginCall ... 5747
MarginStopOut ... 5748
Name ... 5749
Server ... 5750
Currency ... 5751
Company ... 5752
InfoInteger ... 5753
InfoDouble ... 5754
InfoString ... 5755
OrderProfitCheck ... 5756
MarginCheck ... 5757
FreeMarginCheck ... 5758
MaxLotCheck ... 5759
CSymbolInfo ... 5760
Refresh ... 5764
RefreshRates ... 5765
Name ... 5766
Select ... 5767
IsSynchronized ... 5768
Volume ... 5769
VolumeHigh ... 5770
VolumeLow ... 5771
Time ... 5772
Spread ... 5773
SpreadFloat ... 5774
TicksBookDepth ... 5775
StopsLevel ... 5776
FreezeLevel ... 5777
Bid ... 5778
BidHigh ... 5779
BidLow ... 5780
Ask ... 5781
AskHigh ... 5782
AskLow ... 5783
Last ... 5784
LastHigh ... 5785
LastLow ... 5786
TradeCalcMode ... 5787
TradeCalcModeDescription ... 5788
TradeMode ... 5789
TradeModeDescription ... 5790
TradeExecution ... 5791
TradeExecutionDescription ... 5792
SwapMode ... 5793
SwapModeDescription ... 5794
SwapRollover3days ... 5795
SwapRollover3daysDescription ... 5796
MarginInitial ... 5797
MarginMaintenance ... 5798
MarginLong ... 5799
MarginShort ... 5800
MarginLimit ... 5801
MarginStop ... 5802
MarginStopLimit ... 5803

© 2000-2025, MetaQuotes Ltd.

TradeTimeFlags ... 5804
TradeFillFlags ... 5805
Digits ... 5806
Point ... 5807
TickValue ... 5808
TickValueProfit ... 5809
TickValueLoss ... 5810
TickSize ... 5811
ContractSize ... 5812
LotsMin ... 5813
LotsMax ... 5814
LotsStep ... 5815
LotsLimit ... 5816
SwapLong ... 5817
SwapShort ... 5818
CurrencyBase ... 5819
CurrencyProfit ... 5820
CurrencyMargin ... 5821
Bank ... 5822
Description ... 5823
Path ... 5824
SessionDeals ... 5825
SessionBuyOrders ... 5826
SessionSellOrders ... 5827
SessionTurnover ... 5828
SessionInterest ... 5829
SessionBuyOrdersVolume ... 5830
SessionSellOrdersVolume ... 5831
SessionOpen ... 5832
SessionClose ... 5833
SessionAW ... 5834
SessionPriceSettlement ... 5835
SessionPriceLimitMin ... 5836
SessionPriceLimitMax ... 5837
InfoInteger ... 5838
InfoDouble ... 5839
InfoString ... 5840
NormalizePrice ... 5841
COrderInfo ... 5842
Ticket ... 5844
TimeSetup ... 5845
TimeSetupMsc ... 5846
OrderType ... 5847
TypeDescription ... 5848
State ... 5849
StateDescription ... 5850
TimeExpiration ... 5851
TimeDone ... 5852
TimeDoneMsc ... 5853
TypeFilling ... 5854
TypeFillingDescription ... 5855
TypeTime ... 5856
TypeTimeDescription ... 5857
Magic ... 5858
PositionId ... 5859
VolumeInitial ... 5860
VolumeCurrent ... 5861
PriceOpen ... 5862

© 2000-2025, MetaQuotes Ltd.

StopLoss ... 5863
TakeProfit ... 5864
PriceCurrent ... 5865
PriceStopLimit ... 5866
Symbol ... 5867
Comment ... 5868
InfoInteger ... 5869
InfoDouble ... 5870
InfoString ... 5871
StoreState ... 5872
CheckState ... 5873
Select ... 5874
SelectByIndex ... 5875
CHistoryOrderInfo ... 5876
TimeSetup ... 5878
TimeSetupMsc ... 5879
OrderType ... 5880
TypeDescription ... 5881
State ... 5882
StateDescription ... 5883
TimeExpiration ... 5884
TimeDone ... 5885
TimeDoneMsc ... 5886
TypeFilling ... 5887
TypeFillingDescription ... 5888
TypeTime ... 5889
TypeTimeDescription ... 5890
Magic ... 5891
PositionId ... 5892
VolumeInitial ... 5893
VolumeCurrent ... 5894
PriceOpen ... 5895
StopLoss ... 5896
TakeProfit ... 5897
PriceCurrent ... 5898
PriceStopLimit ... 5899
Symbol ... 5900
Comment ... 5901
InfoInteger ... 5902
InfoDouble ... 5903
InfoString ... 5904
Ticket ... 5905
SelectByIndex ... 5906
CPositionInfo ... 5907
Time ... 5909
TimeMsc ... 5910
TimeUpdate ... 5911
TimeUpdateMsc ... 5912
PositionType ... 5913
TypeDescription ... 5914
Magic ... 5915
Identifier ... 5916
Volume ... 5917
PriceOpen ... 5918
StopLoss ... 5919
TakeProfit ... 5920
PriceCurrent ... 5921
Commission ... 5922

© 2000-2025, MetaQuotes Ltd.

Swap ... 5923
Profit ... 5924
Symbol ... 5925
Comment ... 5926
InfoInteger ... 5927
InfoDouble ... 5928
InfoString ... 5929
Select ... 5930
SelectByIndex ... 5931
SelectByMagic ... 5932
SelectByTicket ... 5933
StoreState ... 5934
CheckState ... 5935
CDealInfo ... 5936
Order ... 5938
Time ... 5939
TimeMsc ... 5940
DealType ... 5941
TypeDescription ... 5942
Entry ... 5943
EntryDescription ... 5944
Magic ... 5945
PositionId ... 5946
Volume ... 5947
Price ... 5948
Commision ... 5949
Swap ... 5950
Profit ... 5951
Symbol ... 5952
Comment ... 5953
InfoInteger ... 5954
InfoDouble ... 5955
InfoString ... 5956
Ticket ... 5957
SelectByIndex ... 5958
CTrade ... 5959
LogLevel ... 5963
SetExpertMagicNumber ... 5964
SetDeviationInPoints ... 5965
SetTypeFilling ... 5966
SetTypeFillingBySymbol ... 5967
SetAsyncMode ... 5968
SetMarginMode ... 5969
OrderOpen ... 5970
OrderModify ... 5972
OrderDelete ... 5973
PositionOpen ... 5974
PositionModify ... 5975
PositionClose ... 5976
PositionClosePartial ... 5977
PositionCloseBy ... 5979
Buy ... 5980
Sell ... 5981
BuyLimit ... 5982
BuyStop ... 5983
SellLimit ... 5985
SellStop ... 5987
Request ... 5989

© 2000-2025, MetaQuotes Ltd.

RequestAction ... 5990
RequestActionDescription ... 5991
RequestMagic ... 5992
RequestOrder ... 5993
RequestSymbol ... 5994
RequestVolume ... 5995
RequestPrice ... 5996
RequestStopLimit ... 5997
RequestSL ... 5998
RequestTP ... 5999
RequestDeviation ... 6000
RequestType ... 6001
RequestTypeDescription ... 6002
RequestTypeFilling ... 6003
RequestTypeFillingDescription ... 6004
RequestTypeTime ... 6005
RequestTypeTimeDescription ... 6006
RequestExpiration ... 6007
RequestComment ... 6008
RequestPosition ... 6009
RequestPositionBy ... 6010
Result ... 6011
ResultRetcode ... 6012
ResultRetcodeDescription ... 6013
ResultDeal ... 6014
ResultOrder ... 6015
ResultVolume ... 6016
ResultPrice ... 6017
ResultBid ... 6018
ResultAsk ... 6019
ResultComment ... 6020
CheckResult ... 6021
CheckResultRetcode ... 6022
CheckResultRetcodeDescription ... 6023
CheckResultBalance ... 6024
CheckResultEquity ... 6025
CheckResultProfit ... 6026
CheckResultMargin ... 6027
CheckResultMarginFree ... 6028
CheckResultMarginLevel ... 6029
CheckResultComment ... 6030
PrintRequest ... 6031
PrintResult ... 6032
FormatRequest ... 6033
FormatRequestResult ... 6034
CTerminalInfo ... 6035
Build ... 6037
IsConnected ... 6038
IsDLLsAllowed ... 6039
IsTradeAllowed ... 6040
IsEmailEnabled ... 6041
IsFtpEnabled ... 6042
MaxBars ... 6043
CodePage ... 6044
CPUCores ... 6045
MemoryPhysical ... 6046
MemoryTotal ... 6047
MemoryAvailable ... 6048

© 2000-2025, MetaQuotes Ltd.

MemoryUsed ... 6049
IsX64 ... 6050
OpenCLSupport ... 6051
DiskSpace ... 6052
Language ... 6053
Name ... 6054
Company ... 6055
Path ... 6056
DataPath ... 6057
CommonDataPath ... 6058
InfoInteger ... 6059
InfoString ... 6060
Module von Strategien ... 6061
Basisklassen von Expert Advisors ... 6064
CExpertBase ... 6065
InitPhase ... 6068
TrendType ... 6069
UsedSeries ... 6070
EveryTick ... 6071
Open ... 6072
High ... 6073
Low ... 6074
Close ... 6075
Spread ... 6076
Time ... 6077
TickVolume ... 6078
RealVolume ... 6079
Init ... 6080
Symbol ... 6081
Period ... 6082
Magic ... 6083
ValidationSettings ... 6084
SetPriceSeries ... 6085
SetOtherSeries ... 6086
InitIndicators ... 6087
InitOpen ... 6088
InitHigh ... 6089
InitLow ... 6090
InitClose ... 6091
InitSpread ... 6092
InitTime ... 6093
InitTickVolume ... 6094
InitRealVolume ... 6095
PriceLevelUnit ... 6096
StartIndex ... 6097
CompareMagic ... 6098
CExpert ... 6099
Init ... 6104
Magic ... 6105
InitSignal ... 6106
InitTrailing ... 6107
InitMoney ... 6108
InitTrade ... 6109
Deinit ... 6110
OnTickProcess ... 6111
OnTradeProcess ... 6112
OnTimerProcess ... 6113
OnChartEventProcess ... 6114

© 2000-2025, MetaQuotes Ltd.

OnBookEventProcess ... 6115
MaxOrders ... 6116
Signal ... 6117
ValidationSettings ... 6118
InitIndicators ... 6119
OnTick ... 6120
OnTrade ... 6121
OnTimer ... 6122
OnChartEvent ... 6123
OnBookEvent ... 6124
InitParameters ... 6125
DeinitTrade ... 6126
DeinitSignal ... 6127
DeinitTrailing ... 6128
DeinitMoney ... 6129
DeinitIndicators ... 6130
Refresh ... 6131
Processing ... 6132
SelectPosition ... 6134
CheckOpen ... 6135
CheckOpenLong ... 6136
CheckOpenShort ... 6137
OpenLong ... 6138
OpenShort ... 6139
CheckReverse ... 6140
CheckReverseLong ... 6141
CheckReverseShort ... 6142
ReverseLong ... 6143
ReverseShort ... 6144
CheckClose ... 6145
CheckCloseLong ... 6146
CheckCloseShort ... 6147
CloseAll ... 6148
Close ... 6149
CloseLong ... 6150
CloseShort ... 6151
CheckTrailingStop ... 6152
CheckTrailingStopLong ... 6153
CheckTrailingStopShort ... 6154
TrailingStopLong ... 6155
TrailingStopShort ... 6156
CheckTrailingOrderLong ... 6157
CheckTrailingOrderShort ... 6158
TrailingOrderLong ... 6159
TrailingOrderShort ... 6160
CheckDeleteOrderLong ... 6161
CheckDeleteOrderShort ... 6162
DeleteOrders ... 6163
DeleteOrder ... 6164
DeleteOrderLong ... 6165
DeleteOrderShort ... 6166
LotOpenLong ... 6167
LotOpenShort ... 6168
LotReverse ... 6169
PrepareHistoryDate ... 6170
HistoryPoint ... 6171
CheckTradeState ... 6172
WaitEvent ... 6173

© 2000-2025, MetaQuotes Ltd.

NoWaitEvent ... 6174
TradeEventPositionStopTake ... 6175
TradeEventOrderTriggered ... 6176
TradeEventPositionOpened ... 6177
TradeEventPositionVolumeChanged ... 6178
TradeEventPositionModified ... 6179
TradeEventPositionClosed ... 6180
TradeEventOrderPlaced ... 6181
TradeEventOrderModified ... 6182
TradeEventOrderDeleted ... 6183
TradeEventNotIdentified ... 6184
TimeframeAdd ... 6185
TimeframesFlags ... 6186
CExpertSignal ... 6187
BasePrice ... 6190
UsedSeries ... 6191
Weight ... 6192
PatternsUsage ... 6193
General ... 6194
Ignore ... 6195
Invert ... 6196
ThresholdOpen ... 6197
ThresholdClose ... 6198
PriceLevel ... 6199
StopLevel ... 6200
TakeLevel ... 6201
Expiration ... 6202
Magic ... 6203
ValidationSettings ... 6204
InitIndicators ... 6205
AddFilter ... 6206
CheckOpenLong ... 6207
CheckOpenShort ... 6208
OpenLongParams ... 6209
OpenShortParams ... 6210
CheckCloseLong ... 6211
CheckCloseShort ... 6212
CloseLongParams ... 6213
CloseShortParams ... 6214
CheckReverseLong ... 6215
CheckReverseShort ... 6216
CheckTrailingOrderLong ... 6217
CheckTrailingOrderShort ... 6218
LongCondition ... 6219
ShortCondition ... 6220
Direction ... 6221
CExpertTrailing ... 6222
CheckTrailingStopLong ... 6224
CheckTrailingStopShort ... 6225
CExpertMoney ... 6226
Percent ... 6228
ValidationSettings ... 6229
CheckOpenLong ... 6230
CheckOpenShort ... 6231
CheckReverse ... 6232
CheckClose ... 6233
Module der Handelssignale ... 6234
Signale von Accelerator Oscillator ... 6237

© 2000-2025, MetaQuotes Ltd.

Signale von Adaptive Moving Average ... 6240
Signale von Awesome Oscillator ... 6244
Signale von Oszillator Bears Power ... 6248
Signale von Oszillator Bulls Power ... 6250
Signale von Oszillator Commodity Channel Index ... 6252
Signale von DeMarker-Oszillator ... 6256
Signale von Double Exponential Moving Average ... 6260
Signale von Indikator Envelopes ... 6264
Signale von Fractal Adaptive Moving Average ... 6267
Signale von Intraday-Zeitfilter ... 6271
Signale von MACD ... 6273
Signale von Moving Average ... 6279
Signale von Parabolic SAR ... 6283
Signale von Relative Strength Index ... 6285
Signale von Relative Vigor Index ... 6291
Signale von Stochastic Oszillator ... 6293
Signale von Triple Exponential Average ... 6298
Signale von Triple Exponential Moving Average ... 6302
Signale von Williams Percent Range ... 6306
Klassen von Trailing ... 6309
CTrailingFixedPips ... 6310
StopLevel ... 6312
ProfitLevel ... 6313
ValidationSettings ... 6314
CheckTrailingStopLong ... 6315
CheckTrailingStopShort ... 6316
CTrailingMA ... 6317
Period ... 6319
Shift ... 6320
Method ... 6321
Applied ... 6322
InitIndicators ... 6323
ValidationSettings ... 6324
CheckTrailingStopLong ... 6325
CheckTrailingStopShort ... 6326
CTrailingNone ... 6327
CheckTrailingStopLong ... 6328
CheckTrailingStopShort ... 6329
CTrailingPSAR ... 6330
Step ... 6332
Maximum ... 6333
InitIndicators ... 6334
CheckTrailingStopLong ... 6335
CheckTrailingStopShort ... 6336
Klassen von Kapitalmanagement ... 6337
CMoneyFixedLot ... 6338
Lots ... 6340
ValidationSettings ... 6341
CheckOpenLong ... 6342
CheckOpenShort ... 6343
CMoneyFixedMargin ... 6344
CheckOpenLong ... 6345
CheckOpenShort ... 6346
CMoneyFixedRisk ... 6347
CheckOpenLong ... 6348
CheckOpenShort ... 6349
CMoneyNone ... 6350
ValidationSettings ... 6351

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong ... 6352
CheckOpenShort ... 6353
CMoneySizeOptimized ... 6354
DecreaseFactor ... 6356
ValidationSettings ... 6357
CheckOpenLong ... 6358
CheckOpenShort ... 6359
Panels und Dialoge ... 6360
CRect ... 6362
Left ... 6363
Top ... 6364
Right ... 6365
Bottom ... 6366
Width ... 6367
Height ... 6368
SetBound ... 6369
Move ... 6370
Shift ... 6371
Contains ... 6372
Format ... 6373
CDateTime ... 6374
MonthName ... 6376
ShortMonthName ... 6377
DayName ... 6378
ShortDayName ... 6379
DaysInMonth ... 6380
DateTime ... 6381
Date ... 6382
Time ... 6383
Sec ... 6384
Min ... 6385
Hour ... 6386
Day ... 6387
Mon ... 6388
Year ... 6389
SecDec ... 6390
SecInc ... 6391
MinDec ... 6392
MinInc ... 6393
HourDec ... 6394
HourInc ... 6395
DayDec ... 6396
DayInc ... 6397
MonDec ... 6398
MonInc ... 6399
YearDec ... 6400
YearInc ... 6401
CWnd ... 6402
Create ... 6406
Destroy ... 6407
OnEvent ... 6408
OnMouseEvent ... 6409
Name ... 6410
ControlsTotal ... 6411
Control ... 6412
ControlFind ... 6413
Rect ... 6414
Left ... 6415

© 2000-2025, MetaQuotes Ltd.

Top ... 6416
Right ... 6417
Bottom ... 6418
Width ... 6419
Height ... 6420
Move ... 6421
Shift ... 6422
Resize ... 6423
Contains ... 6424
Alignment ... 6425
Align ... 6426
Id ... 6427
IsEnabled ... 6428
Enable ... 6429
Disable ... 6430
IsVisible ... 6431
Visible ... 6432
Show ... 6433
Hide ... 6434
IsActive ... 6435
Activate ... 6436
Deactivate ... 6437
StateFlags ... 6438
StateFlagsSet ... 6439
StateFlagsReset ... 6440
PropFlags ... 6441
PropFlagsSet ... 6442
PropFlagsReset ... 6443
MouseX ... 6444
MouseY ... 6445
MouseFlags ... 6446
MouseFocusKill ... 6447
OnCreate ... 6448
OnDestroy ... 6449
OnMove ... 6450
OnResize ... 6451
OnEnable ... 6452
OnDisable ... 6453
OnShow ... 6454
OnHide ... 6455
OnActivate ... 6456
OnDeactivate ... 6457
OnClick ... 6458
OnChange ... 6459
OnMouseDown ... 6460
OnMouseUp ... 6461
OnDragStart ... 6462
OnDragProcess ... 6463
OnDragEnd ... 6464
DragObjectCreate ... 6465
DragObjectDestroy ... 6466
CWndContainer ... 6467
Destroy ... 6469
OnEvent ... 6470
OnMouseEvent ... 6471
ControlsTotal ... 6472
Control ... 6473
ControlFind ... 6474

© 2000-2025, MetaQuotes Ltd.

Add ... 6475
Delete ... 6476
Move ... 6477
Shift ... 6478
Id ... 6479
Enable ... 6480
Disable ... 6481
Show ... 6482
Hide ... 6483
MouseFocusKill ... 6484
Save ... 6485
Load ... 6486
OnResize ... 6487
OnActivate ... 6488
OnDeactivate ... 6489
CWndObj ... 6490
OnEvent ... 6492
Text ... 6493
Color ... 6494
ColorBackground ... 6495
ColorBorder ... 6496
Font ... 6497
FontSize ... 6498
ZOrder ... 6499
OnObjectCreate ... 6500
OnObjectChange ... 6501
OnObjectDelete ... 6502
OnObjectDrag ... 6503
OnSetText ... 6504
OnSetColor ... 6505
OnSetColorBackground ... 6506
OnSetFont ... 6507
OnSetFontSize ... 6508
OnSetZOrder ... 6509
OnDestroy ... 6510
OnChange ... 6511
CLabel ... 6512
Create ... 6517
OnSetText ... 6518
OnSetColor ... 6519
OnSetFont ... 6520
OnSetFontSize ... 6521
OnCreate ... 6522
OnShow ... 6523
OnHide ... 6524
OnMove ... 6525
CBmpButton ... 6526
Create ... 6533
Border ... 6534
BmpNames ... 6535
BmpOffName ... 6536
BmpOnName ... 6537
BmpPassiveName ... 6538
BmpActiveName ... 6539
Pressed ... 6540
Locking ... 6541
OnSetZOrder ... 6542
OnCreate ... 6543

© 2000-2025, MetaQuotes Ltd.

OnShow ... 6544
OnHide ... 6545
OnMove ... 6546
OnChange ... 6547
OnActivate ... 6548
OnDeactivate ... 6549
OnMouseDown ... 6550
OnMouseUp ... 6551
CButton ... 6552
Create ... 6559
Pressed ... 6560
Locking ... 6561
OnSetText ... 6562
OnSetColor ... 6563
OnSetColorBackground ... 6564
OnSetColorBorder ... 6565
OnSetFont ... 6566
OnSetFontSize ... 6567
OnCreate ... 6568
OnShow ... 6569
OnHide ... 6570
OnMove ... 6571
OnResize ... 6572
OnMouseDown ... 6573
OnMouseUp ... 6574
CEdit ... 6575
Create ... 6581
ReadOnly ... 6582
TextAlign ... 6583
OnObjectEndEdit ... 6584
OnSetText ... 6585
OnSetColor ... 6586
OnSetColorBackground ... 6587
OnSetColorBorder ... 6588
OnSetFont ... 6589
OnSetFontSize ... 6590
OnSetZOrder ... 6591
OnCreate ... 6592
OnShow ... 6593
OnHide ... 6594
OnMove ... 6595
OnResize ... 6596
OnChange ... 6597
OnClick ... 6598
CPanel ... 6599
Create ... 6604
BorderType ... 6605
OnSetText ... 6606
OnSetColorBackground ... 6607
OnSetColorBorder ... 6608
OnCreate ... 6609
OnShow ... 6610
OnHide ... 6611
OnMove ... 6612
OnResize ... 6613
OnChange ... 6614
CPicture ... 6615
Create ... 6620

© 2000-2025, MetaQuotes Ltd.

Border ... 6621
BmpName ... 6622
OnCreate ... 6623
OnShow ... 6624
OnHide ... 6625
OnMove ... 6626
OnChange ... 6627
CScroll ... 6628
Create ... 6631
OnEvent ... 6632
MinPos ... 6633
MaxPos ... 6634
CurrPos ... 6635
CreateBack ... 6636
CreateInc ... 6637
CreateDec ... 6638
CreateThumb ... 6639
OnClickInc ... 6640
OnClickDec ... 6641
OnShow ... 6642
OnHide ... 6643
OnChangePos ... 6644
OnThumbDragStart ... 6645
OnThumbDragProcess ... 6646
OnThumbDragEnd ... 6647
CalcPos ... 6648
CScrollV ... 6649
CreateInc ... 6655
CreateDec ... 6656
CreateThumb ... 6657
OnResize ... 6658
OnChangePos ... 6659
OnThumbDragStart ... 6660
OnThumbDragProcess ... 6661
OnThumbDragEnd ... 6662
CalcPos ... 6663
CScrollH ... 6664
CreateInc ... 6670
CreateDec ... 6671
CreateThumb ... 6672
OnResize ... 6673
OnChangePos ... 6674
OnThumbDragStart ... 6675
OnThumbDragProcess ... 6676
OnThumbDragEnd ... 6677
CalcPos ... 6678
CWndClient ... 6679
Create ... 6682
OnEvent ... 6683
ColorBackground ... 6684
ColorBorder ... 6685
BorderType ... 6686
VScrolled ... 6687
HScrolled ... 6688
CreateBack ... 6689
CreateScrollV ... 6690
CreateScrollH ... 6691
OnResize ... 6692

© 2000-2025, MetaQuotes Ltd.

OnVScrollShow ... 6693
OnVScrollHide ... 6694
OnHScrollShow ... 6695
OnHScrollHide ... 6696
OnScrollLineDown ... 6697
OnScrollLineUp ... 6698
OnScrollLineLeft ... 6699
OnScrollLineRight ... 6700
Rebound ... 6701
CListView ... 6702
Create ... 6708
OnEvent ... 6709
TotalView ... 6710
AddItem ... 6711
Select ... 6712
SelectByText ... 6713
SelectByValue ... 6714
Value ... 6715
CreateRow ... 6716
OnResize ... 6717
OnVScrollShow ... 6718
OnVScrollHide ... 6719
OnScrollLineDown ... 6720
OnScrollLineUp ... 6721
OnItemClick ... 6722
Redraw ... 6723
RowState ... 6724
CheckView ... 6725
CComboBox ... 6726
Create ... 6732
OnEvent ... 6733
AddItem ... 6734
ListViewItems ... 6735
Select ... 6736
SelectByText ... 6737
SelectByValue ... 6738
Value ... 6739
CreateEdit ... 6740
CreateButton ... 6741
CreateList ... 6742
OnClickEdit ... 6743
OnClickButton ... 6744
OnChangeList ... 6745
ListShow ... 6746
ListHide ... 6747
CCheckBox ... 6748
Create ... 6754
OnEvent ... 6755
Text ... 6756
Color ... 6757
Checked ... 6758
Value ... 6759
CreateButton ... 6760
CreateLabel ... 6761
OnClickButton ... 6762
OnClickLabel ... 6763
CCheckGroup ... 6764
Create ... 6770

© 2000-2025, MetaQuotes Ltd.

OnEvent ... 6771
AddItem ... 6772
Value ... 6773
CreateButton ... 6774
OnVScrollShow ... 6775
OnVScrollHide ... 6776
OnScrollLineDown ... 6777
OnScrollLineUp ... 6778
OnChangeItem ... 6779
Redraw ... 6780
RowState ... 6781
CRadioButton ... 6782
Create ... 6784
OnEvent ... 6785
Text ... 6786
Color ... 6787
State ... 6788
CreateButton ... 6789
CreateLabel ... 6790
OnClickButton ... 6791
OnClickLabel ... 6792
CRadioGroup ... 6793
Create ... 6799
OnEvent ... 6800
AddItem ... 6801
Value ... 6802
CreateButton ... 6803
OnVScrollShow ... 6804
OnVScrollHide ... 6805
OnScrollLineDown ... 6806
OnScrollLineUp ... 6807
OnChangeItem ... 6808
Redraw ... 6809
RowState ... 6810
Select ... 6811
CSpinEdit ... 6812
Create ... 6818
OnEvent ... 6819
MinValue ... 6820
MaxValue ... 6821
Value ... 6822
CreateEdit ... 6823
CreateInc ... 6824
CreateDec ... 6825
OnClickInc ... 6826
OnClickDec ... 6827
OnChangeValue ... 6828
CDialog ... 6829
Create ... 6832
OnEvent ... 6833
Caption ... 6834
Add ... 6835
CreateWhiteBorder ... 6836
CreateBackground ... 6837
CreateCaption ... 6838
CreateButtonClose ... 6839
CreateClientArea ... 6840
OnClickCaption ... 6841

© 2000-2025, MetaQuotes Ltd.

OnClickButtonClose ... 6842
ClientAreaVisible ... 6843
ClientAreaLeft ... 6844
ClientAreaTop ... 6845
ClientAreaRight ... 6846
ClientAreaBottom ... 6847
ClientAreaWidth ... 6848
ClientAreaHeight ... 6849
OnDialogDragStart ... 6850
OnDialogDragProcess ... 6851
OnDialogDragEnd ... 6852
CAppDialog ... 6853
Create ... 6856
Destroy ... 6857
OnEvent ... 6858
Run ... 6859
ChartEvent ... 6860
Minimized ... 6861
IniFileSave ... 6862
IniFileLoad ... 6863
IniFileName ... 6864
IniFileExt ... 6865
CreateCommon ... 6866
CreateExpert ... 6867
CreateIndicator ... 6868
CreateButtonMinMax ... 6869
OnClickButtonClose ... 6870
OnClickButtonMinMax ... 6871
OnAnotherApplicationClose ... 6872
Rebound ... 6873
Minimize ... 6874
Maximize ... 6875
CreateInstanceId ... 6876
ProgramName ... 6877
SubwinOff ... 6878
36 Übergang von MQL4 ... 6879
37 MQL5 Funktionenliste ... 6883
38 MQL5 Konstantenliste ... 6921

© 2000-2025, MetaQuotes Ltd.

MQL5 Referenz

MetaQuotes Language 5 (MQL5) ist eine Hochsprache für die Entwicklung von technischen Indikatoren, Handelsrobotern und Hilfsprogrammen, die den Finanzhandel automatisieren. MQL5 wurde von MetaQuotes für ihre Handelsplattform entwickelt. Die Sprachsyntax ist sehr nah an C++ und ermöglicht es Programmierern, Anwendungen im Stil der objektorientierten Programmierung (OOP) zu entwickeln.

Zusätzlich zur MQL5-Sprache enthält das Handelsplattform-Paket auch die MetaEditor IDE mit hoch entwickelten Code-Schreibwerkzeugen, wie Templates, Snippets, Debugging, Profiling und Autovervollständigung, sowie die integrierte Dateiversionierung MQL5 Storage.

Die Sprachunterstützung ist auf der Website MQL5 Algotrading Community verfügbar, die eine riesige freie CodeBase und eine Fülle von Artikeln enthält. Diese Artikel decken alle Aspekte des modernen Handels ab, einschließlich neuronaler Netze, Statistik und Analyse, Hochfrequenzhandel, Arbitrage, Test und Optimierung von Handelsstrategien, dem Einsatz von automatisierten Handelsrobotern und mehr.

Händler und MQL5-Programmentwickler können im Forum miteinander kommunizieren, Anwendungen in Auftrag geben und selber entwickeln, indem sie den Freelance Service nutzen, oder geschützte Programme im Market der automatisierten Handelsanwendungen kaufen und verkaufen.

Die MQL5-Sprache bietet spezialisierte Handelsfunktionen und vordefinierte Ereignisbehandlungen zur Unterstützung von Programmierern bei der Entwicklung von Expert Advisors (EAs), die automatisch Handelsprozesse nach bestimmten Handelsregeln ausführen. Zusätzlich zu den EAs erlaubt MQL5 die Entwicklung von benutzerdefinierten technischen Indikatoren, Skripten und Bibliotheken.

Diese MQL5-Sprachreferenz enthält Funktionen, Operationen, reservierte Wörter und andere Sprachkonstruktionen, die in Kategorien unterteilt sind. Die Referenz enthält auch Beschreibungen der Klassen in der Standardbibliothek, die für die Entwicklung von Handelsstrategien, Bedienfeldern, benutzerdefinierten Grafiken und für den Dateizugriff verwendet werden.

Zusätzlich gibt es in der CodeBase die Bibliothek ALGLIB für die numerische Analyse, die zur Lösung verschiedener mathematischer Probleme verwendet werden kann.

Algo Trading Bücher

Etwas Neues zu lernen ist immer eine Herausforderung. Um Anfängern zu helfen, haben wir zwei umfassende Bücher über die MQL5-Programmierung veröffentlicht, die sich an alle richten, die das Erstellen von Handelsrobotern und Anwendungen für den algorithmischen Handel beherrschen wollen.

Diese Bücher bieten eine systematische und strukturierte Darstellung des Materials, um den Lernprozess erheblich zu erleichtern. Detaillierte Code-Beispiele, die Schritt für Schritt das Erstellen von Handelsrobotern und -anwendungen erklären, ermöglichen ein tieferes Verständnis der Varianten des algorithmischen Handels. Die Bücher enthalten zahlreiche praktische Übungen, die helfen, das erworbene Wissen zu festigen und die Programmierfähigkeiten in realen Handelsumgebungen zu entwickeln.

„MQL5-Programmieren für Händler", ist das vollständigste und detaillierteste Lehrbuch zu MQL5, das für Programmierer aller Stufen geeignet ist. Anfänger lernen die Grundlagen: Das Buch stellt Entwicklungswerkzeuge und grundlegende Programmierkonzepte vor. Auf der Grundlage dieses

© 2000-2025, MetaQuotes Ltd.

Materials werden Sie Ihre erste Anwendung auf der Handelsplattform MetaTrader 5 erstellen, kompilieren und ausführen. Nutzer mit Erfahrung in anderen Programmiersprachen können sofort mit dem Anwendungsteil fortfahren: Erstellen von Handelsrobotern und analytischen Anwendungen in MQL5.

„Neuronale Netze für den algorithmischen Handel mit MQL5", ist ein Leitfaden für die Verwendung von Methoden des maschinellen Lernens in Handelsrobotern für die Plattform MetaTrader 5. Sie werden schrittweise in die Grundlagen der neuronalen Netze und deren Anwendung im algorithmischen Handel eingeführt. Im Laufe des Kurses werden Sie Ihre eigene KI-Lösung aufbauen und trainieren und nach und nach neue Funktionen hinzufügen. Zusätzlich zum Erlernen von MQL5 werden Sie Python- und OpenCL-Programmierkenntnisse erwerben und integrierte Matrix- und Vektormethoden erforschen, die die Lösung komplexer mathematischer Probleme mit prägnantem und effizientem Code ermöglichen.

Artikel über die Entwicklung von Handelsanwendungen

MQL5 Artikel sind eine ausgezeichnete Quelle, um das volle Potenzial der Sprache zu erforschen und decken eine breite Palette praktischer algorithmischer Handelsaufgaben ab. Zur einfachen Navigieren sind alle Artikel in Abschnitte wie Beispiel, Expert Advisors, Machine Learning und weitere gegliedert. Jeden Monat werden Dutzende von neuen Artikeln auf der Website der MQL5 Algotrading Community veröffentlicht, geschrieben von Händlern für Händler. Lesen und diskutieren Sie diese Artikel, um den modernen algorithmischen Handel zu beherrschen. Für Anfänger haben wir eine Liste von 16 empfohlenen Artikeln für ein schnelles Eintauchen in MQL5 zusammengestellt.

Arten von MQL5-Anwendungen

MQL5-Programme sind in fünf spezialisierte Typen unterteilt, die auf den von ihnen implementierten automatisierten Handelsaufgaben basieren:

- Der **Expert Advisor** ist ein automatisiertes Handelssystem, das mit einem Chart verbunden ist. Ein Expert Advisor enthält Ereignisbehandlungen, um auf vordefinierte Ereignisse zu reagieren, die die entsprechenden Elemente der Handelsstrategie durchführen. Zum Beispiel, das Ereignis der Programminitialisierung und Deinitialisierung, neue Ticks, Timer-Ereignisse, Änderungen der Tiefe des Marktes, auf dem Chart und benutzerdefinierte Ereignisse.
  Neben der Berechnung der Handelssignale auf der Grundlage der implementierten Regeln, können Expert Advisors auch den Handel automatisieren und die Aufträge direkt an den Handelsserver senden. Expert Advisor werden in <*Terminal_Directory*>*MQL5**Experts*. gespeichert.
- Der **benutzerdefinierte Indikator** ist ein technischer Indikator, der von einem Benutzer zusätzlich zu den in die Handelsplattform integrierten Standardindikatoren entwickelt wurde. Benutzerdefinierte Indikatoren, wie auch Standardindikatoren, können nicht automatisch handeln, sondern nur analytische Funktionen umsetzen und bereitstellen. Benutzerdefinierte Indikatoren können Werte anderer Indikatoren für Berechnungen verwenden und können von einem Expert Advisor aufgerufen werden.
  Sie werden im Verzeichnis <*Terminal_Directory*>*MQL5**Indicators* gespeichert.
- Das **Script** ist ein Programm, das nur einmal ausgeführt wird. Im Gegensatz zu einem Expert Advisor behandeln Skripte kein Ereignis außer Trigger, Initialisierung und Deinitialisierung. Der Code eines Skripts muss die Funktion OnStart() enthalten.
  Skripte werden in <*Terminal_DIrectory*>*MQL5**Scripts gespeichert*.

© 2000-2025, MetaQuotes Ltd.

- **Dienst** (Service) ist ein Programm, das im Gegensatz zu Indikatoren, Expert Advisors und Skripten nicht an ein Chart gebunden sein muss, damit es läuft. So wie Skripte reagieren Dienste nicht auf Ereignisse außer ihrem Start. Um einen Dienst zu starten, sollte dessen Code in der Funktion OnStart stehen. Die Dienste akzeptieren keine anderen Ereignisse außer den Start, aber sie sind in der Lage, nutzerdefinierte Ereignisse mit EventChartCustom an einen Chart zu senden. Dienste werden im <*Terminal_Directory*>*MQL5**Services*. gespeichert.
- Die **Bibliothek** ist eine Ansammlung von benutzerdefinierten Funktionen. Bibliotheken sind dazu gedacht, häufig verwendete Algorithmen benutzerdefinierter Programme aufzunehmen und für eine vielfältige Verwendung zur Verfügung zu stellen.
  Bibliotheken befinden sich in <*Terminal_Directory*>*MQL5**Libraries*.
- Die **Include-Datei** ist ein Quelltext der am häufigsten verwendeten Blöcke von benutzerdefinierten Programmen. Solche Dateien werden in die Quelltexte von Expert Advisors, Scripts, benutzerdefinierten Indikatoren und Bibliotheken während der Kompilierung eingebunden. Die Verwendung von eingebundenen Dateien ist wegen der zusätzlichen Belastung beim Aufruf von Bibliotheksfunktionen besser als die Verwendung von Bibliotheken.
  Include-Dateien können im gleichen Verzeichnis gespeichert werden, in dem sich die Originaldatei befindet. In diesem Fall wird die Anweisung #include mit doppelten Anführungszeichen verwendet. Eine andere Möglichkeit ist, die Include-Dateien im Verzeichnis <*Terminal_Directory*>*MQL5**Include* zu speichern. In diesem Fall muss #include mit spitzen Klammern verwendet werden.

© 2000-2025, MetaQuotes Ltd

Grundlagen der Sprache

MetaQuotes Language 5 (MQL5) ist eine objektorientierte Programmiersprache für Entwicklung automatischer Handelsstrategien, technische Benutzerindikatoren für die Analyse verschiedenartiger Finanzmärkte. Sie erlaubt nicht nur verschiedene Expertsysteme zu schreiben, sondern auch eigene graphische Werkzeuge zu erzeugen, die Handelsentscheidungen zu treffen helfen.

MQL5 zugrunde liegt das Konzept der weit verbreiteten Programmiersprache C++, im Vergleich zu MQL4 sind darin Enumerationen, Strukturen, Klassen und Verarbeitung von Ereignissen. Wegen der Erweiterung der Anzahl eingebauter Grundtypen, ist die Zusammenwirkung ausgeführter Programme auf MQL5 mit anderen Anwendungen mittels dll maximal erleichtert. Syntax der Sprache MQL5 ist der Syntax C++ ähnlich, das erlaubt Programme aus modernen Programmiersprache in diese Sprache leicht zu übertragen.

für Lernen der Sprache MQL5 sind alle Themen nach folgenden Abschnitten gruppiert:

- Syntax
- Datentypen
- Operationen und Ausdrücke
- Anweisungen
- Funktionen
- Variablen
- Preprozessor
- Objektorientiertes Programmieren
- Namespaces

© 2000-2025, MetaQuotes Ltd.

Syntax

Syntaktisch gesehen ist Programmiersprache der Handelsstrategien MQL5 Programmiersprache C++ sehr ähnlich, außer einiger Möglichkeiten:

- es gibt keine Adressenarithmetik;
- es gibt keinen Identifikator goto;
- eine anonyme Enumeration kann nicht vereinbart werden;
- es gibt keine Mehrfachvererbung.

Sehen Sie auch

Enumerationen, Strukturen und Klassen, Vererbung

© 2000-2025, MetaQuotes Ltd.

Kommentare

Mehrzeilige Kommentare fangen mit zwei Symbolen /* an und schließen mit zwei Symbolen */ ab. Diese Kommentare können nicht eingebettet sein.Einzeilige Kommentare fangen mit zwei Symbolen // an, schließen mit dem Symbol der neuen Zeile und können in mehrzeilige Kommentare eingebettet sein. Kommentare sind sind überall erlaubt, wo Spaces sein können, und jede Anzahl der Spaces zulässt.

Beispiele:

```
//--- Einzeiliger Kommentar
/*  Mehrzeiliges
                // Eingebetteter einzeiliger Kommentar
    Kommentar
*/
```

© 2000-2025, MetaQuotes Ltd.

Identifikatoren

Identifikatoren werden als Namen für Variablen und Funktionen verwendet. Die Länge des Identifikatoren kann nicht mehr als 63 Zeichen sein.

Zulässige Symbole eines Identifikatoren: Zahlen 0-9, lateinische Gross- und Kleinbuchstaben a-z und A-Z, die als verschiedene Symbole erkennt werden, Unterstreichungszeichen (_). Das erste Symbol kann keine Zahl sein.

Identifikator muss nicht mit dem reservierten Wort zusammenfallen.

Beispiele:

```
NAME1 name1 Total_5 Paper
```

Sehen Sie auch

Variablen, Funktionen

© 2000-2025, MetaQuotes Ltd.

Reservierte Wörter

Die unten aufgezählten Identifikatoren werden als reservierte Wörter fixiert, jedem entspricht die bestimmte Handlung, anders können sie nicht verwendet werden:

Datentypen

bool	float	uint
char	int	ulong
class	long	union
color	short	ushort
datetime	string	void
double	struct	
enum	uchar	

Spezifikatoren des Zuganges

const	private	virtual
delete	protected	
override	public	

Speicherklassen

extern	input	static

Anweisungen

break	dynamic_cast	operator
case	else	pack
continue	for	return
default	if	sizeof
delete	new	switch
do	offsetof	while

Die übrigen

this	#define	#import
true	#ifdef	#include

© 2000-2025, MetaQuotes Ltd.

this	#define	#import
false	#ifndef	#property
template	#else	group
typename	#endif	namespace

© 2000-2025, MetaQuotes Ltd.

Datentypen

Jedes Programm operiert mit Daten. Daten können verschiedener Arten sein abhängig vom Zweck. ZB für Zugang zu Feldelementen werden Felder ganzzahligen Typs verwendet. Preisdaten sind Fliesspunktdaten mit doppelter Genauigkeit. Das ist damit verbunden, dass es in der Sprache MQL5 kein Sondertyp für Preisdaten vorausgesehen ist.

Daten verschiedener Typen werden mit verschiedener Geschwindigkeit verarbeitet. Ganzzahlige Daten werden am schnellsten verarbeitet. Für die Verarbeitung der Daten mit doppelter Genauigkeit wird Coprozessor verwendet. Aber wegen der komplizierten inneren Darstellung von Fliesspunktdaten werden sie länger als ganzzahlige Daten verarbeitet.

Am laengsten werden Zeilendaten verarbeitet. Das ist mit dynamischer Verteilung-Neuverteilung des Operativspeichers verbunden.

Hauptdatentypen:

- ganzzahlige (char, short, int, long, uchar, ushort, uint, ulong)
- logische (bool)
- Literalwerte (char, uchar)
- Zeilen (string)
- Fliesspunktdaten (double, float)
- Farbe (color)
- Datum und Zeit (datetime)
- Enumerationen (enum)

Zusammengesetzte Datentypen:

- Strukturen;
- Klassen.

In OOP Termini werden komplizierte Datentypen als abstrakte Datentypen genannt.

Typen color und datetime sind erst für die Darstellung und Parametereingabe sinnvoll, die von aussen festgelegt werden - aus der Tabelle der Experteigenschaften oder aus dem Benutzerindikator (Zubehör "Inputs"). Datentypen color und datetime werden als Ganzzahle dargestellt. Ganzzahlige Typen mit den Fliesspunktdaten werden arithmetische Typen (Zahlentypen) genannt.

In Ausdrücken wird implizite Typenreduzierung verwendet, wenn explizite Typenreduzierung nicht angegeben wird.

Sehen Sie auch

Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

Ganzzahlige Typen

Ganzzahlige Typen werden in der Sprache MQL5 von elf Arten vertreten. Einiger von ihnen können mit den anderen verwendet werden, wenn die Programmlogik es fordert, dabei muss man aber die Regeln der Typenreduzierung in Aussicht nehmen.

In der Tabelle werden charakteristische Daten jedes Typs angegeben. Außerdem wird in der senkrechten Spalte für jeden Typ der entsprechende Typ in Programmiersprache C++ angegeben.

Typ	Größe in Bytes	minimale Größe	Maximale Größe	Analog in der Sprache C++
char	1	-128	127	char
uchar	1	0	255	unsigned char, BYTE
bool	1	0(false)	1(true)	bool
short	2	-32 768	32 767	short, wchar_t
ushort	2	0	65 535	unsigned short, WORD
int	4	- 2 147 483 648	2 147 483 647	int
uint	4	0	4 294 967 295	unsigned int, DWORD
color	4	-1	16 777 215	int, COLORREF
long	8	-9 223 372 036 854 775 808	9 223 372 036 854 775 807	__int64
ulong	8	0	18 446 744 073 709 551 615	unsigned __int64
datetime	8	0 (1970.01.01 0:00:00)	32 535 244 799 (3000.12.31 23:59:59)	__time64_t

Werte der ganzzahligen Typen können als numerische Konstanten, Farbenliterals, Literals Datum-Zeit, Symbolkonstanten und Enumerationen.

Zusätzliche Aliasnamen für Integer-Typen (für C/C++-Kompatibilität)

Um die Übertragung von Code aus anderen Sprachen, wie z.B. C und C++, zu vereinfachen, wurden Aliase für Standard-Integer-Typen in die Sprache MQL5 aufgenommen. Diese Aliasnamen führen keine neuen Typen ein, sondern sind alternative Namen für bestehende Typen in MQL5. Sie können in allen Kontexten verwendet werden, in denen die Basistypen anwendbar sind.

© 2000-2025, MetaQuotes Ltd.

Typ	Entspricht dem Typ in MQL5	Größe in Bytes	Vorzeichen	Kleinster Wert	Größter Wert
int8_t	char	8	mit Vorzeichen	-128	127
uint8_t	uchar	8	kein Vorzeichen	0	255
int16_t	short	16	mit Vorzeichen	-32 768	32 767
uint16_t	ushort	16	kein Vorzeichen	0	65 535
int32_t	int	32	mit Vorzeichen	- 2 147 483 648	2 147 483 647
uint32_t	uint	32	kein Vorzeichen	0	4 294 967 295
int64_t	long	64	mit Vorzeichen	-9 223 372 036 854 775 808	9 223 372 036 854 775 807
uint64_t	ulong	64	kein Vorzeichen	0	18 446 744 073 709 551 615

Diese Aliase sind nützlich bei der Portierung bestehender Bibliotheken und Algorithmen von C/C++ nach MQL5, insbesondere in Projekten mit einem hohen Maß an Plattformkompatibilität. Alle neuen Namen sind in Skripten, EAs und Indikatoren verfügbar.

Sehen Sie auch

Datenverarbeitung, Konstanten der Zahltypen

© 2000-2025, MetaQuotes Ltd.

Typen char, short, int und long

char

Ganzzahliger Typ char nimmt ein Byte (8 Bit) des Speicherplatzes ein und hilft im Dualsystem 2^8 Werte=256 auszudrücken. Ganzzahliger Typ char kann sowohl positive, als auch negative Werte. Änderungsbereich ist von -128 bis 127.

uchar

Ganzzahliger Typ uchar nimmt 1 Byte des Speicherplatzes ein, wie der Typ char, aber zum Unterschied vom Typ char ist uchar ausschliesslich für positive Werte bestimmt. Die minimale Größe betraegt 0, maximale Größe betraegt 255. Der erste Buchstabe im Namen des Types uchar ist die Abkürzung des Wortes unsigned (zeichenlos).

short

Der ganzzahliger Typ short nimmt 2 Byte (16 Bit) und erlaubt eine Menge der Größen auszudrücken, die 2 erhebt in der sechzehnten Potenz ist: 2^16=65 536. Da der Typ short ein Zeichensymbol ist und sowohl positive, als auch negative Größen enthält, ist sein Änderungsbereich von -32 768 bis 32 767.

ushort

Die zeichenlose Art vom Typ short ist der Typ ushort, dessen Größe auch 2 Byte ist. Minimale Größe ist 0, maximale Größe 65 535.

int

Ganzzahliger Typ int hat die Größe 4 Byte (32 Bit). Minimale Größe -2 147 483 648, maximale Größe 2 147 483 647.

uint

Zeichenloser ganzzahliger Typ nimmt 4 Byte des Speicherplatzes ein und erlaubt ganzzahlige Größen von 0 bis 4 294 967 295 auszudrücken.

long

Ganzzahliger Typ long hat die Größe 8 Byte (64 Bit). Minimale Größe -9 223 372 036 854 775 808, maximale Größe 9 223 372 036 854 775 807.

ulong

Ganzzahliger Typ ulong nimmt auch 8 Byte ein und erlaubt die Größen von 0 bis 18 446 744 073 709 551 615 zu speichern.

Beispiele:

```
char  ch=12;
```

© 2000-2025, MetaQuotes Ltd.

```
short sh=-5000;
int   in=2445777;
```

Da zeichenlose ganzzahlige Typen für Speicherung negativer Werte nicht bestimmt sind, kann der Versuch, negative Größe zu erfahren zu unerwarteten Folgen führen. Solch ein Script führt zur Endlosschleife:

```
//--- Endlosschleife
void OnStart()
  {
   uchar  u_ch;

   for(char ch=-128;ch<128;ch++)
     {
      u_ch=ch;
      Print("ch = ",ch," u_ch = ",u_ch);
     }
  }
```

Richtig wird auf dieser Weise:

```
//--- richtige Variante
void OnStart()
  {
   uchar  u_ch;

   for(char ch=-128;ch<=127;ch++)
     {
      u_ch=ch;
      Print("ch = ",ch," u_ch = ",u_ch);
      if(ch==127) break;
     }
  }
```

Ergebnis:

```
ch= -128  u_ch= 128
ch= -127  u_ch= 129
ch= -126  u_ch= 130
ch= -125  u_ch= 131
ch= -124  u_ch= 132
ch= -123  u_ch= 133
ch= -122  u_ch= 134
ch= -121  u_ch= 135
```

© 2000-2025, MetaQuotes Ltd.

```
ch= -120  u_ch= 136
ch= -119  u_ch= 137
ch= -118  u_ch= 138
ch= -117  u_ch= 139
ch= -116  u_ch= 140
ch= -115  u_ch= 141
ch= -114  u_ch= 142
ch= -113  u_ch= 143
ch= -112  u_ch= 144
ch= -111  u_ch= 145
 ...
```

Beispiele:

```
//--- negative Größen können nicht in zeichenlosen Typen gespeichert werden
uchar  u_ch=-120;
ushort u_sh=-5000;
uint   u_in=-401280;
```

Hexadezimal:Zahlen 0-9, Buchstaben a-f oder A-F für Größen 10-15; fangen mit Ox oder OX an.

Beispiele:

```
0x0A, 0x12, 0X12, 0x2f, 0xA3, 0Xa3, 0X7C7
```

Sehen Sie auch

Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

Symbolkonstanten

Symbole als Zeilenelemente in MQL5 - sind Indexe in Symbolvorrat Unicode. Sie sind Hexadezimalwerte, die in ganzzahlige Typen umgewandelt werden können und mit denen mit ganzzahligen Operationen manipuliert werden kann, solche wie Addition und Substrahieren.

Jedes einzelne Symbol in Anführungszeichen oder hexadezimaler ASCII-Kode des Symbols der Art '\x10' ist eine Symbolkanstante und hat den Typ ushort. ZB Schreibung der Art '0' ist ein numerischer Wert 30, der dem Index von Null in der Tabelle der Symbole entspricht.

Beispiel:

```
void OnStart()
  {
//--- bestimmen wir Symbolkanstanten
   int symbol_0='0';
   int symbol_9=symbol_0+9; // erhalten wir das Symbol '9'
//--- geben wir Werte von Konstanten aus
   printf("In dezimaler Form: symbol_0 = %d,  symbol_9 = %d",symbol_0,symbol_9);
   printf("In hexadezimaler Form: symbol_0 = 0x%x,  symbol_9 = 0x%x",symbol_0,symbol_
//--- fuegen wir Konstanten in eine Zeile hinzu
   string test="";
   StringSetCharacter(test,0,symbol_0);
   StringSetCharacter(test,1,symbol_9);
//--- so sehen sie in einer Zeile aus
   Print(test);
  }
```

Backslash ist ein Kontrollzeichen für Compiler beim Umgang mit Konstantzeilen und Symbolonstanten im Ausgangstext eines Programms. Einige Symbole, ZB einzelner Anführungsstrich ('), Doppelte Anführungszeichen ("), backslash (\) und Kontrollsymbole können als Symbolkombination dargestellt werden, die mit backslash (\) anfaengt, entsprechend der unten angegebenewn Tabelle:

Symbolname	Mnemokode oder Darstellung	Aufzeichnung in MQL5	Numerischer Wert
neue Zeile (Zeilenvorschub)	LF	'\n'	10
horizontales Tabulieren	HT	'\t'	9
Vorschub rückgeben	CR	'\r'	13
backslash	\	'\\'	92
einzelner Anführungsstrich	'	'\''	39
Doppelanführungszeichen	"	'\"'	34
hexadezimaler Kode	hhhh	'\xhhhh'	von 1 bis 4 hexadezimale Zeichen
dezimaler Kode	d	'\d'	dezimale Zahl von 0 bis 65535

© 2000-2025, MetaQuotes Ltd.

Wenn das nach slash folgende Symbol sich von de aufgelisteten unterscheidet, ist das Ergebnis unbestimmt.

Beispiel

```
void OnStart()
  {
//--- erklären wir Symbolkanstanten
   int a='A';
   int b='$';
   int c='©';       // Kode 0xA9
   int d='\xAE';    // Symbolkode ®
//--- geben wir Konstante für Abdrucken aus
   Print(a,b,c,d);
//--- fuegen wir ein Symbol in die Zeile zu
   string test="";
   StringSetCharacter(test,0,a);
   Print(test);
//--- Ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,0,b);
   Print(test);
//--- Ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,0,c);
   Print(test);
//--- Ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,0,d);
   Print(test);
//--- stellen wir Symbole als eine Zahl dar
   int a1=65;
   int b1=36;
   int c1=169;
   int d1=174;
//--- setzen wir Symbol in die Zeile zu
   StringSetCharacter(test,1,a1);
   Print(test);
//--- ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,1,b1);
   Print(test);
//--- ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,1,c1);
   Print(test);
//--- ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,1,d1);
   Print(test);
  }
```

Wir es schon oben angegeben wurde, ist der Wert von einer Symbolkonstante (oder Variable) Index in der Symboltabelle darstellt, da ein Index eine Ganzzahl ist, ist es zulaessig es verschieden zu schreiben.

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//---
   int a=0xAE;      // Symbolkode ® entspricht dem Literal '\xAE'
   int b=0x24;      // Symbolkode $ entspricht dem Literal '\x24'
   int c=0xA9;      // Symbolkode © entspricht dem Literal '\xA9'
   int d=0x263A;    // Symbolkode O entspricht dem Literal '\x263A'
//--- geben wir Werte aus
   Print(a,b,c,d);
//--- fuegen wir Symbol in die Zeile hinzu
   string test="";
   StringSetCharacter(test,0,a);
   Print(test);
//--- ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,0,b);
   Print(test);
//--- ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,0,c);
   Print(test);
//--- ersetzen wir Symbol in einer Zeile
   StringSetCharacter(test,0,d);
   Print(test);
//--- Kodes der Farben
   int a1=0x2660;
   int b1=0x2661;
   int c1=0x2662;
   int d1=0x2663;
//---fuegen wir Pikssymbol hinzu
   StringSetCharacter(test,1,a1);
   Print(test);
//--- fuegen wir Herzsymbol hinzu
   StringSetCharacter(test,2,b1);
   Print(test);
//--- fuegen wir Karosymbol hinzu
   StringSetCharacter(test,3,c1);
   Print(test);
//--- fuegen wir Treffsymbol hinzu
   StringSetCharacter(test,4,d1);
   Print(test);
//--- Beispiel der Symbolliterale in der Zeile
   test="Dame\x2660As\x2662";
   printf("%s",test);
  }
```

Interne Darstellung des Symbolliterals ist Typ ushort. Symbolkonstanten können die Werte von 0 bis 65535 annehmen.

Sehen Sie auch

StringSetCharacter(), StringGetCharacter(), ShortToString(), ShortArrayToString(), StringToShortArray()

© 2000-2025, MetaQuotes Ltd.

Typ datetime

Typ datetime ist für Datum- und Zeitspeicherung als Sekundenanzahl bestimmt, die seit 01 Januar 1970 vegangen ist. Es nimmt 8 Byte des Speicherplatzes ein.

Daten- und Zeitkonstanten können als Literalzeile dargestellt werden, die aus 6 Teilen besteht. Diese Teile repraesentieren die Zahlgroesse des Jahres, Monats, der Zahl (oder des Monatstages, des Monats, des Jahres), der Stunde, der Minute und der Sekunde. Konstante wird in einzelnen Anführungsstrich gesetzt und faengt mit Symbol D an.

Änderungsbereich vom 1 Januar 1970bis 31 Dezember 3000. Fallen kann entweder das Datum (Jahr, Monat, Monatstag) oder die Zeit (Stunden, Minuten, Sekunden), oder alles zusammengenommen.

In der wörtlichen Einstellung des Datums ist wünschenswert, den Jahr, Monat und Tag anzugeben, sonst zeigt der Compiler Warnungen über unvollständigen wörtlichen Eintrag.

Bespiele:

```
datetime NY=D'2015.01.01 00:00';      // Zeit der Anfang des Jahres 2015
datetime d1=D'1980.07.19 12:30:27';   // Jahr Monat Tag Stunden Minuten Sekunden
datetime d2=D'19.07.1980 12:30:27';   // äquivalent zu D'1980.07.19 12:30:27';
datetime d3=D'19.07.1980 12';         // äquivalent zu D'1980.07.19 12:00:00'
datetime d4=D'01.01.2004';            // äquivalent zu D'01.01.2004 00:00:00'
datetime compilation_date=__DATE__;              // Datum der Kompilierung
datetime compilation_date_time=__DATETIME__;     // Datum und Zeit der Kompilierung
datetime compilation_time=__DATETIME__-__DATE__;// Zeit der Kompilierung
//--- Beispiele, in denen Compiler/Warnungen zurückgegeben werden
datetime warning1=D'12:30:27';        // äquivalent zu D'[Datum der Kompilierung] 12:3
datetime warning2=D'';                // äquivalent zu __DATETIME__
```

Sehen Sie auch

Datenstruktur, Datum und Zeit, TimeToString, StringToTime

© 2000-2025, MetaQuotes Ltd.

Typ color

Typ color ist für die Speicherung farbenbezogener Information bestimmt und nimmt 4 Bytes des Speicherplatzes ein. Das erste Byte wird nicht berücksichtigt, die anderen 3 Bytes enthalten RGB-Komponenten.

Farbkonstanten können durch drei verschiedene Arten vertreten werden: literal, ganzzahlig oder mit Hilfe des Namens (nur für die genannten Web-Farben).

Literaldarstellung besteht aus drei Teilen. Sie stellen die Zahlgroessen der Intensitaet von drei Hauptfarbarten: rot (red), gruen (green), blau(blue). Konstante beginnt mit Symbol c und wird in einzelne Anführungszeichen gesetzt. Zahlgroessen der Intensitaet von Farbkomponenten sind von 0 bis 255.

Ganzzahlige Darstellung wird als Hexadezimalzahl oder als Dezimalzahl eingetragen. Hexadezimalzahl ist der Art 0x00BBGGRR, wo RR -Intensitaetsgroesse der roten Farbkomponente, GG - der gruenen, BB - der blauen. Dezimalkonstanten haben keine direkte Darstellung in RGB. Sie bilden Dezimalgroesse der ganzzahligen Darstellung.

Benannte Farben spiegeln den sogenannten Bestand von Web-Farben wieder.

Beispiele:

```
//--- Literalwerte
C'128,128,128'    // grau
C'0x00,0x00,0xFF' // blau
//Farbbezeichnungen
clrRed               // rot
clrYellow            // gelb
clrBlack             // schwarz
//--- ganzzahlige Darstellungen
0xFFFFFF          // weiss
16777215          // weiss
0x008000          // gruen
32768             // gruen
```

Sehen Sie auch

Web-Farben, ColorToString, StringToColor, Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

Typ bool

Typ bool ist für die Speicherung logischer Werte true (richtig) oder false (falsch) bestimmt, deren Zahldarstellung 1 oder 0 ist.

Beispiele:

```
bool a = true;
bool b = false;
bool c = 1;
```

Interne Darstellung ist die Ganzzahl mit der Größe 1 Byte. Es muss hervorgehoben werden, dass es in logischen Ausdrücken zulaessig ist, anstatt des Typs bool andere ganzzahlige Typen oder Realtypen oder Ausdrücke dieser Typen zu verwenden, Compiler wird kein Fehler zeigen. In diesem wird der Wert 0 als false interpretiert, alle andere Werte als true.

Beispiele:

```
   int i=5;
   double d=-2.5;
   if(i) Print("i = ",i," und Wert true hat");
   else Print("i = ",i," und Wert false hat");

   if(d) Print("d = ",d," und Wert true hat");
   else Print("d = ",d," und Wert false hat");

   i=0;
   if(i) Print("i = ",i," und Wert true hat");
   else Print("i = ",i," und Wert false hat");

   d=0.0;
   if(d) Print("d=",d,"und Wert true hat");
   else Print("d=",d,"und Wert false hat");

//--- Ergebnisse der Ausführung
//   i= 5 und Wert true hat
//   d= -2.5 und Wert true hat
//   i= 0 und Wert false hat
//   d= 0 und Wert false hat
```

Sehen Sie auch

Logische Operationen, Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Enumerationen

Daten des Enumeration-Typs enum gehören zur einer begrenzten Datenmenge. Definition des aufzählbaren Typs:

```
enum Name des aufzählbaren Typs
  {
   Liste von Bedeutungen
  };
```

Liste von Bedeutungen stellt Konstantenliste dar, die durch Kommas voneinander getrennt sind.

Beispiel:

```
enum months  // Enumeration der benannten Konstanten
   {
    January,
    February,
    March,
    April,
    May,
    June,
    July,
    August,
    September,
    October,
    November,
    December
   };
```

Nach der Vereinbarung von Enumeration erscheint ein neuer ganzzahliger 4-Byte Datentyp. Vereinbarung des neuen Datentyps ermöglicht dem Compiler Typen der übertragenen Parameter strikt zu kontrollieren, weil die Enumeration neue benannten Konstanten einführt. In dem angeführten Beispiel hat die benannte Konstante die Größe 0, February hat die Größe 1, December hat die Größe 11.

Regel: Wenn einer benannten Konstante - dem Enumeration-Glied keinen konkreten Wert zugeordnet wird, wird ihr Wert automatisch erzeugt. Wenn es das Erste Enumeration-Glied ist, wird der Wert 0 zugeordnet. Für alle folgenden Glieder werden die Werter aufgrund des Wertes vom vorangehenden Gliedes aufgezählt und eine Eins hinzugefügt.

Beispiel:

```
enum intervals  // Enumeration der benannten Konstanten
   {
    month=1,     // Interval aus einem Monat
    two_months,  // zwei Monate
    quarter,     // drei Monate - Quartal
    halfyear=6,  // Halbjahr
    year=12,     // Jahr - 12 Monate
   };
```

© 2000-2025, MetaQuotes Ltd.

Bemerkungen

- Zum Unterschied von C++, ist die Größe der inneren Darstellung des aufzählbaren Typs in MQL5 immer 4 Byte. D.h. sizeof(months) gibt den Wert 4 zurück.
- Zum Unterschied von C++, kann in MQL5 keine anonyme Enumeration erklärt werden. D.h. nach Schlüsselwort enum kann immer ein eindeutiger Name angegeben werden.

Sehen Sie auch

Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

Realtypen (double, float)

Realtypen (Fliesspunkttypen) stellen die Werte mit Bruchanteil dar. In der Sprache MQL5 gibt es zwei Typen für Fliesspunktzahlen. Darstellungsart der Realzahlen im Computerspeicher ist vom Standard IEEE 754 bestimmt und hängt nicht von Plattformen, Operationssystemen und Programmiersprachen ab.

Typ	Größe in Bytes	minimale positive Größe	Maximale Größe	Analog in C++
float	4	1.175494351e-38	3.402823466e+38	float
double	8	2.2250738585072014e-308	1.7976931348623158e+308	double

double

Der reelle Zahlentyp double belegt 64 Bits (1 Bit für das Vorzeichen, 11 für den Exponenten und 52 für die Mantisse) .

float

Der reelle Zahlentyp float belegt 32 Bits (1 Bit für das Vorzeichen, 8 für den Exponenten und 23 für die Mantisse) .

vector

Ein eindimensionales Array mit Zahlen des Typs double. Der Speicher für die Daten wird dynamisch zugewiesen. Die Vektoreigenschaften können mit den Methoden ermittelt werden, mit denen die Vektorgröße geändert werden. Der Ausdruck vector<double> kann für Template-Funktionen verwendet werden.

vectorf

Eindimensionales Array mit Zahlen des Typs float kann anstelle von vector verwendet werden, wenn die geringere Genauigkeit keine Rolle spielt. Der Ausdruck vector<float> kann für Template-Funktionen verwendet werden.

vectorc

Ein eindimensionales Array mit Zahlen des Typs complex für komplexe Zahlen. Der Ausdruck vector<complex> kann für Template-Funktionen verwendet werden. Operationen mit Vektoren vom Typ vectorc sind noch nicht implementiert.

matrix

© 2000-2025, MetaQuotes Ltd.

Eine Matrix ist ein zweidimensionales Array mit Zahlen des Typs double. Der Speicher für die Matrixelemente wird dynamisch verwaltet. Die Matrixeigenschaften können mit den Methoden ermittelt werden, mit denen die Matrixform geändert werden können. Der Ausdruck matrix<double> kann für Template-Funktionen verwendet werden.

matrixf

Ein zweidimensionales Array mit Zahlen des Typs float kann anstelle von matrix verwendet werden, wenn die geringere Genauigkeit keine Rolle spielt. Der Ausdruck matrix<float> kann für Template-Funktionen verwendet werden.

matrixc

Ein zweidimensionales Array mit Zahlen des Typs complex ist für die Verarbeitung komplexer Zahlen gedacht. Der Ausdruck matrix<complex> kann für Template-Funktionen verwendet werden. Operationen mit Matrizen vom Typ matrixc sind noch nicht implementiert.

Name double bedeutet, dass die Genauigkeit dieser Zahlen um das Doppelte die Genaugkeit der Zahlen der Art float übertrifft. In meistens Faellen ist die Art double am bequemsten. Die begrenzte Genauigkeit der Zahle float ist oft nicht ausreichend. Der Grund, warum float noch verwendet wird, - Speicheroekonomie bei der Bewahrung (das ist sehr wichtig für grosse Felder der Realzahlen).

Konstanten mit Fliesspunkt bestehen aus dem ganzzahligen Teil, Punkt (.) und Bruchanteil. Ganzzahliger Teil und Bruchanteil stellen Folgen von Dezimalzahlen dar.

Beispiele:

```
double a=12.111;
double b=-956.1007;
float  c =0.0001;
float  d =16;
```

Es gibt ein wissenschaftliches Schreibverfahren der Realkonstanten. Oft ist dieses Schreibverfahren kampakter, als das traditionelle.

Beispiel:

```
double c1=1.12123515e-25;
double c2=0.00000000000000000000000112123515; // 24 Nulle nach Rechendezimalpunkt

Print("1. c1 =",DoubleToString(c1,16));
// Ergebnis: 1. c1 = 0.0000000000000000

Print("2. c1 =",DoubleToString(c1,-16));
// Ergebnis: 2. c1 = 1.1212351499999999e-025

Print("3. c2 =",DoubleToString(c2,-16));
// Ergebnis: 3. c2 = 1.1212351499999999e-025
```

© 2000-2025, MetaQuotes Ltd.

Man muss im Kopf behalten, dass Realzahlen im Bionaersystem mit begrenzter Genauigkeit gespeichert werden, während das allgemeinguelgige System Dezimalsystem ist. Darum können viele Zahlen, die genau im Dezimalsystem geschrieben werden, können im Dualsystem nur als Infinitesimalbruch geschrieben werden.

ZB. werden die Zahlen 0.3 und 0.7 im Computer als Infinitesimalbrueche vertreten, während die Zahl 0, 25 wird genau gespeichert, da es die Potenz der zwei ist.

In diesem Zusammenhang ist es nicht empfehlenswert, zwei Realzahlen miteinander zu vergleichen, weil dieser Vergleich nicht korrekt ist.

Beispiel:

```
void OnStart()
  {
//---
   double three=3.0;
   double x,y,z;
   x=1/three;
   y=4/three;
   z=5/three;
   if(x+y==z)
      Print("1/3 + 4/3 == 5/3");
   else
      Print("1/3 + 4/3 != 5/3");
// Ergebnis: 1/3 + 4/3 != 5/3
  }
```

Wenn es aber notwendog ist, zwei Realzahlen miteinander zu vergleichen, kann es mit zwei Verfahren gemacht werden. Das erste Verfahren besteht im Vergleich des Differenzbetrages der zwei Zahlen mit irgendwelcher kleinen Größe, die die Genauigkeit des Vergleiches festlegt.

Beispiel:

```
bool EqualDoubles(double d1,double d2,double epsilon)
  {
   if(epsilon<0)
      epsilon=-epsilon;
//---
   if(d1-d2>epsilon)
      return false;
   if(d1-d2<-epsilon)
      return false;
//---
   return true;
   }
void OnStart()
  {
   double d_val=0.7;
   float  f_val=0.7;
   if(EqualDoubles(d_val,f_val,0.000000000000001))
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(d_val,"equals",f_val);
   else
      Print("Different: d_val=",DoubleToString(d_val,16),"  f_val=",DoubleToString(f_v
// Ergebnis: Different: d_val= 0.7000000000000000   f_val= 0.6999999880790710
  }
```

Es muss bemerkt werden, dass der Wert von epsilon im angeführten Beispiel nicht kleiner als der der internen Konstante DBL_EPSILON sein kann. Der Wert dieser Konstante ist 2.2204460492503131e-016. Für den Typ float ist die entsprechende Konstante FLT_EPSILON = 1.192092896e-07. Der Sinn dieser Werter besteht darin, dass es der minimale Wert ist, der der Bedingung entspricht 1.0+DBL_EPSILON != 1.0 (für die Zahlen float 1.0+FLT_EPSILON != 1.0).

Das zweite Verfahren setzt den Vergleich normalisierter Differenz von zwei Realzahlen mit dem Nullwert. Es ist sinnlos, Differenz normalisierter Zahlen mit Null zu vergleichen, denn im Ergebnis jeder mathemati schen Operation mit normalisierter Zahlen wird das Ergebnis nicht normalisiert.

Beispiel:

```
bool CompareDoubles(double number1,double number2)
  {
   if(NormalizeDouble(number1-number2,8)==0)
      return(true);
   else
      return(false);
  }
void OnStart()
  {
   double d_val=0.3;
   float  f_val=0.3;
   if(CompareDoubles(d_val,f_val))
      Print(d_val,"equals",f_val);
   else
      Print("Different: d_val=",DoubleToString(d_val,16),"  f_val=",DoubleToString(f_v
// Ergebnis: Different: d_val= 0.3000000000000000   f_val= 0.3000000119209290
  }
```

Im Ergebnis einiger Operationen des mathematischen Coprozessors kann eine ungueltige Realzahl entstehen, die nicht in mathematischen Operationen und Vergleichsoperationen verwendet werden kann, denn das Ergebnis der Operationen mit ungueltigen Realzahlen ist nicht bestimmt. ZB beim Versuch arc sin 2 auszurechnen, bekommt man das Ergebnis minus Infinitaet.

Beispiel:

```
   double abnormal = MathArcsin(2.0);
   Print("MathArcsin(2.0) =",abnormal);
// Ergebnis:  MathArcsin(2.0) = -1.#IND
```

Ausser minus Infinitaet gibt es auch plus Infinitaet und NaN (keine Zahl). Um festzustellen, das diese Zahl ungueltg ist, kann man die funktion MathIsValidNumber() benutzen Nach Standard IEEE haben Sie spezielle Maschienendarstellung. ZB plus Infinitaet für den Typ double hat Bitdarstellung 0x7FF0 0000 0000 0000.

© 2000-2025, MetaQuotes Ltd.

Beispiele:

```
struct str1
  {
   double d;
  };
struct str2
  {
   long l;
  };

//--- fangen wir an
   str1 s1;
   str2 s2;
//---
   s1.d=MathArcsin(2.0);         // wir bekommen ungueltige Zahl -1.#IND
   s2=s1;
   printf("1.  %f %I64X",s1.d,s2.l);
//---
   s2.l=0xFFFF000000000000;      // ungueltige Zahl -1.#QNAN
   s1=s2;
   printf("2.  %f %I64X",s1.d,s2.l);
//---
   s2.l=0x7FF7000000000000;      // maximale Nichtzahl SNaN
   s1=s2;
   printf("3.   %f %I64X",s1.d,s2.l);
//---
   s2.l=0x7FF8000000000000;      // minimale Nic QNhtzahlaN
   s1=s2;
   printf("4.   %f %I64X",s1.d,s2.l);
//---
   s2.l=0x7FFF000000000000;      // maximale Nichtzahl QNaN
   s1=s2;
   printf("5.   %f %I64X",s1.d,s2.l);
//---
   s2.l=0x7FF0000000000000;      // plus Infinitum 1.#INF und minimale Nichtzahl SNaN
   s1=s2;
   printf("6.   %f %I64X",s1.d,s2.l);
//---
   s2.l=0xFFF0000000000000;      // minus Infinitum -1.#INF
   s1=s2;
   printf("7.  %f %I64X",s1.d,s2.l);
//---
   s2.l=0x8000000000000000;      // negative Null -0.0
   s1=s2;
   printf("8.  %f %I64X",s1.d,s2.l);
//---
   s2.l=0x3FE0000000000000;      // 0.5
   s1=s2;
```

© 2000-2025, MetaQuotes Ltd.

```
   printf("9.   %f %I64X",s1.d,s2.l);
//---
   s2.l=0x3FF0000000000000;     // 1.0
   s1=s2;
   printf("10.  %f %I64X",s1.d,s2.l);
//---
   s2.l=0x7FEFFFFFFFFFFFFF;     // minimale normalisierte Zahl(MAX_DBL)
   s1=s2;
   printf("11.  %.16e %I64X",s1.d,s2.l);
//---
   s2.l=0x0010000000000000;     // minimale positive normalisierte (MIN_DBL)
   s1=s2;
   printf("12.  %.16e %.16I64X",s1.d,s2.l);
//---
   s1.d=0.7;                    // zeigen wir, dass die Zahl 0.7 - Infinitesimalbruch
   s2=s1;
   printf("13.  %.16e %.16I64X",s1.d,s2.l);
/*
1.  -1.#IND00 FFF8000000000000
2.  -1.#QNAN0 FFFF000000000000
3.   1.#SNAN0 7FF7000000000000
4.   1.#QNAN0 7FF8000000000000
5.   1.#QNAN0 7FFF000000000000
6.   1.#INF00 7FF0000000000000
7.  -1.#INF00 FFF0000000000000
8.  -0.000000 8000000000000000
9.   0.500000 3FE0000000000000
10.  1.000000 3FF0000000000000
11.  1.7976931348623157e+308 7FEFFFFFFFFFFFFF
12.  2.2250738585072014e-308 0010000000000000
13.  6.9999999999999996e-001 3FE6666666666666
*/
```

Sehen Sie auch

DoubleToString, NormalizeDouble, Konstanten der numerischen Datentypen

© 2000-2025, MetaQuotes Ltd.

Komplexen Zahlen (complex)

Der eingebaute complex Typ ist eine Struktur mit zwei double:

```
struct complex
  {
   double            real;    // reeler Teile
   double            imag;    // imaginärer Teil
  };
```

Der Typ "complex" kann als Parameter für MQL5-Funktionen per Wert übergeben werden (im Gegensatz zu normalen Strukturen, die nur per Referenz übergeben werden). Bei Funktionen, die aus DLLs importiert werden, muss der Typ "complex" ausschließlich als Referenz übergeben werden.

Das Suffix 'i' wird verwendet, um komplexe Konstanten zu kennzeichnen:

```
complex square(complex c)
  {
   return(c*c);
  }
void OnStart()
  {
   Print(square(1+2i));  // Die Übergaben einer Konstanten als Parameter
  }
// ausgegeben wird "(-3,4)", die komplexe Zahl in Textform
```

Für komplexe Zahlen sind derzeit nur einfache Operationen verfügbar: =, +, -, *, /, +=, -=, *=, /=, ==,!=.

Später wird die Unterstützung weiterer mathematischer Funktionen hinzukommen, die die Berechnung von Absolutwert, Sinus, Cosinus und anderen ermöglichen.

vectorc

Ein eindimensionales Array mit Zahlen des Typs complex für komplexe Zahlen. Der Ausdruck vector<complex> kann für Template-Funktionen verwendet werden. Operationen mit Vektoren vom Typ vectorc sind noch nicht implementiert.

matrixc

Ein zweidimensionales Array mit Zahlen des Typs complex ist für die Verarbeitung komplexer Zahlen gedacht. Der Ausdruck matrix<complex> kann für Template-Funktionen verwendet werden. Operationen mit Matrizen vom Typ matrixc sind noch nicht implementiert.

© 2000-2025, MetaQuotes Ltd.

Typ string

Typ string ist für die Speicherung von Textzeilen bestimmt. Textzeile ist eine Symbolenfolge in Unicode Format mit Null am Ende. Zeilenkonstante ist eine Symbolefolge in Unicode mit doppelten Anführungszeichen: "Das ist eine Zeilenkonstante"

Wenn es notwendig ist, in die Zeile ein Doppelanführungszeichen einzugeben, muss man vor diesem Zeichen einen nach links weisender Schrägstrich (\) stellen. In die Zeile können verschiedene Sondersymbolkonstanten eingegeben werden, vor denen ein nach links weisender Schrägstrich steht (\).

Beispiele:

```
string svar="This is a character string";
string svar2=StringSubstr(svar,0,4);
Print("Copyright Symbol\t\x00A9");
FileWrite(handle,"Diese Zeile hat das CR-Zeichen \n");
string T5path="C:\\Program Files\\MetaTrader 5";
```

Um den Quellcode lesbar zu machen, können Sie lange konstante Strings in Teile ohne Zusatz Operationen aufteilen. Beim Kompilierung werden diese Teile automatisch in einem langen String kombiniert werden:

```
//--- Ein langer konstanter Sting deklarieren
   string HTML_head="<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.0 Transitional//EN\"'
                    " \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd\">\n"
                    "<html xmlns=\"http://www.w3.org/1999/xhtml\">\n"
                    "<head>\n"
                    "<meta http-equiv=\"Content-Type\" content=\"text/html; charset=ut
                    "<title>Trade Operations Report</title>\n"
                    "</head>";
//--- Den konstanten String im Journal anzeigen
   Print(HTML_head);
```

Sehen Sie auch

Datenverarbeitung, Zeilenfunktionen, FileOpen, FileReadString, FileWriteString

© 2000-2025, MetaQuotes Ltd.

Strukturen, Klassen und Schnittstellen

Strukturen

Eine Struktur umfasst eine Reihe von Elementen beliebigen Typs (außer dem Typ void). Auf diese Weise gruppiert eine Struktur logisch verbundene Daten unterschiedlicher Typen.

Strukturdeklaration

Der Strukturdatentyp wird wie folgt bestimmt:

```
struct Strukturname
  {
   Elementenbeschreibung
  };
```

Der Strukturname kann nicht als Identifikator verwendet werden (Name der Variable oder Funktion). Man bedenke, dass die Strukturelemente in MQL5 ohne Ausrichtung direkt aufeinander folgen. In der Sprache C++ wird diese Festlegung an den Kompiler mit der Anweisung gerichtet.

```
#pragma pack(1)
```

Wenn es notwendig ist, eine andere Ausrichtung in der Struktur durchzuführen, muss man "Platzhalter" der gewünschten Größen verwenden.

Beispiel:

```
struct trade_settings
  {
   uchar  slippage;     // Wert der zulässigen Slippage - Größe 1 Byte
   char   reserved1;    // 1 Byte Platzhalter
   short  reserved2;    // 2 Byte Platzhalter
   int    reserved4;    // noch 4 Byte Platzhalter. Bereitgestellte Ausrichtung von 8
   double take;         // Preis eines fixierten Take-Profits
   double stop;         // Preis eines schützenden Stop-Loss
  };
```

Eine solche Beschreibung ausgerichteter Strukturen ist nur für die Übergabe an importierte dll-Funktionen notwendig.

Achtung: dieses Beispiel zeigt falsch entworfene Daten. Es ist besser, zuerst die Daten von Take-Profit und Stop-Loss mit dem Typ double und dann das Slippage-Member des Uchartyps zu deklarieren. In diesem Fall ist die interne Darstellung der Daten immer gleich, unabhängig von dem im *#pragma pack()* angegebenen Wert.

Wenn die Struktur Variablen der Art string und/oder Objekt des dynamischen Feldes enthält, weist der Kompiler für eine solche Struktur einen impliziten Konstruktor zu, in dem alle Mitglieder vom Typ string zurückgesetzt werden und die dynamischen Array-Objekt werden korrekt initialisiert.

Einfache Strukturen

© 2000-2025, MetaQuotes Ltd.

Strukturen, die keine Zeichenketten, Klassenobjekte, Pointer und Objekte von dynamischen Arrays enthalten, werden als einfache Strukturen bezeichnet. Variablen einfacher Strukturen sowie deren Arrays können als Parameter an importierte Funktionen aus DLL. übergeben werden.

Das Kopieren von einfachen Strukturen ist nur in zwei Fällen erlaubt:

- Wenn die Objekte zum gleichen Strukturtyp gehören.
- Wenn die Objekte linear miteinander verbunden sind, was bedeutet, dass eine Struktur ein Nachkomme einer anderen ist.

Um ein Beispiel zu geben, entwickeln wir die nutzerdefinierte Struktur CustomMqlTick mit ihrem Inhalt, der identisch ist mit der integrierten Struktur MqlTick. Der Kompiler erlaubt es nicht, den Wert des MqlTick-Objekts in das Objekt vom Typ CustomMqlTick zu kopieren. Auch die direkte Typenumwandlung zum benötigten Typ verursacht einen Kompilerfehler:

```
      //--- Kopieren einfacher Strukturen verschiedener Typen ist verboten
      my_tick1=last_tick;               // der Kompiler wirft hier den Fehler aus

      //--- Eine Typenumwandlung von unterschiedlichen Strukturen ist auch verboten
      my_tick1=(CustomMqlTick)last_tick;// der Kompiler wirft hier den Fehler aus
```

Daher bleibt nur eine Option - das Kopieren der Werte der Strukturelemente, eines nach dem anderen. Es ist weiterhin erlaubt, die Werte des gleichen Typs von CustomMqlTick zu kopieren.

```
      CustomMqlTick my_tick1,my_tick2;
      //--- Es ist erlaubt, die Objekte des gleichen Typs CustomMqlTick wie folgt zu k
      my_tick2=my_tick1;

      //--- Erstellen Sie ein Array aus den Objekten der einfachen CustomMqlTick-Stru
      CustomMqlTick arr[2];
      arr[0]=my_tick1;
      arr[1]=my_tick2;
```

Die Funktion ArrayPrint() wird aufgerufen, um die Werte des Arrays *arr[] im Journal auszudrucken*.

```
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Entwickeln einer Struktur ähnlich der integrierten MqlTick
   struct CustomMqlTick
     {
      datetime          time;          // Aktualisierungszeit des letzten Preises
      double            bid;           // Aktueller Bid-Preis
      double            ask;           // Aktueller Ask-Preis
      double            last;          // Aktueller Preis des letzten Handels (Last)
      ulong             volume;        // Volumen des aktuellen, letzten Preises
      long              time_msc;      // Aktualisierungszeit des letzten Preises in M
      uint              flags;         // Tick-Flags
     };
   //--- Abrufen des letzten Tickwertes
```

© 2000-2025, MetaQuotes Ltd.

```
   MqlTick last_tick;
   CustomMqlTick my_tick1,my_tick2;
//--- Versuch die Daten von MqlTick nach CustomMqlTick zu kopieren
   if(SymbolInfoTick(Symbol(),last_tick))
     {
      //--- Kopieren von unverbundenen Strukturen ist verboten
      //1. my_tick1=last_tick;               // der Kompiler wirft hier den Fehler aus

      //--- Auch eine Typenumwandlung von unverbundenen Strukturen untereinander ist v
      //2. my_tick1=(CustomMqlTick)last_tick;// Der Kompiler wirft hier den Fehler aus

      //--- Daher werden die Strukturmitglieder einzeln kopiert
      my_tick1.time=last_tick.time;
      my_tick1.bid=last_tick.bid;
      my_tick1.ask=last_tick.ask;
      my_tick1.volume=last_tick.volume;
      my_tick1.time_msc=last_tick.time_msc;
      my_tick1.flags=last_tick.flags;

      //--- Es ist erlaubt, die Objekte des gleichen Typs CustomMqlTick wie folgt zu k
      my_tick2=my_tick1;

      //--- Erstellen Sie ein Array aus den Objekten der einfachen CustomMqlTick-Struk
      CustomMqlTick arr[2];
      arr[0]=my_tick1;
      arr[1]=my_tick2;
      ArrayPrint(arr);
//--- Beispiel einer Anzeige der Werte des Arrays mit den Objekten vom Typ CustomMqlTi
      /*
                         [time]    [bid]    [ask]    [last] [volume]     [time_msc] [flags]
      [0] 2017.05.29 15:04:37 1.11854 1.11863 +0.00000  1450000 1496070277157       2
      [1] 2017.05.29 15:04:37 1.11854 1.11863 +0.00000  1450000 1496070277157       2
      */
     }
   else
      Print("SymbolInfoTick() failed, error = ",GetLastError());
  }
```

Das zweite Beispiel zeigt die Eigenschaften des linearen Kopierens einfacher Strukturen. Angenommen, wir haben die Grundstruktur Tier, von der die Strukturen für Katze und Hund abgeleitet werden. Wir können die Objekte Tier und Katze sowie Tier und Hund ineinander kopieren, aber wir können Katze und Hund nicht ineinander kopieren, obwohl beide Nachkommen der Struktur Tier sind.

```
//--- Struktur zur Beschreibung von Hund
struct Hund: Tier
  {
   bool              hunting;       // Beutejäger
  };
//--- Struktur der Beschreibung der Katzen
```

© 2000-2025, MetaQuotes Ltd.

```
struct Katze: Tier
  {
   bool              home;          // Eigenzüchtung
  };
//--- Erstellen der abgeleiteten Strukturen
  Hund hund;
  Katze katze;
//--- Kann vom Vor- auf den Nachfahren kopiert werden (Tier ==> Hund)
  hund=ein_Tier;
   hund.swim=true;    // Hunde können schwimmen
//--- Objekte der Nachfahren können nicht aufeinander kopiert werden (Hund != Katze)
   katze=hund;        // der Kompiler wirft einen Fehler aus
```

Der ganze Code des Beispiels:

```
//--- Basisstruktur zur Beschreibung von Tier
struct Tier
  {
   int               head;          // Anzahl der Köpfe
   int               legs;          // Anzahl der Beine
   int               wings;         // Anzahl der Flügel
   bool              tail;          // Schwanz
   bool              fly;           // fliegt
   bool              swim;          // schwimmt
   bool              run;           // rennt
  };
//--- Struktur zur Beschreibung von Hund
struct Hund: Tier
  {
   bool              hunting;       // Beutejäger
  };
//--- Struktur der Beschreibung der Katzen
struct Katze: Tier
  {
   bool              home;          // Eigenzüchtung
  };
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Erstellen und beschreiben des Basistyps von Tier
  Tier ein_Tier;
  ein_Tier.head=1;
  ein_Tier.legs=4;
  ein_Tier.wings=0;
   ein_Tier.tail=true;
   ein_Tier.fly=false;
   ein_Tier.swim=false;
```

© 2000-2025, MetaQuotes Ltd.

```
   ein_Tier.run=true;
//--- create objects of child types
  Hund hund;
  Katze katze;
//--- Kann vom Vor- auf den Nachfahren kopiert werden (Tier ==> Hund)
  hund=ein_Tier;
   hund.swim=true;    // Hunde können schwimmen
//--- Objekte der Nachfahren können nicht aufeinander kopiert werden (Hund != Katze)
   //katze=hund;        // der Kompiler wirft einen Fehler aus
//--- Daher können die Elemente nur eines nach dem anderen kopiert werden
  katze.head=hund.head;
  katze.legs=hund.legs;
  katze.wings=hund.wings;
  katze.tail=hund.tail;
  katze.fly=hund.fly;
   katze.swim=false;   // Katzen können nicht schwimmen
//--- Man kann die Werte vom Nach- auf den Vorfahren zu kopieren
  Tier elephant;
  elephant=katze;
   elephant.run=false;// Elefanten können nicht rennen
   elephant.swim=true;// Elefanten können schwimmen
//--- Erstellen eines Arrays
  Tier tiere[4];
  tiere[0]=ein_tier;
  tiere[1]=hund;
  tiere[2]=katze;
  tiere[3]=elephant;
//--- Ausdrucken
   ArrayPrint(animals);
//--- Berechnungsergebnis
/*
        [head] [legs] [wings] [tail] [fly] [swim] [run]
   [0]      1      4        0   true false  false  true
   [1]      1      4        0   true false   true  true
   [2]      1      4        0   true false  false false
   [3]      1      4        0   true false   true false
*/
  }
```

Ein anderer Weg einfache Strukturen zu kopieren ist die Verwendung von unions. Die Objekte der Struktur sollten Mitglieder der gleichen union sein - siehe das Beispiel für union.

Zugang zu Strukturgliedern

Strukturname ist ein neuer Datentyp und ermöglicht Variablen dieses Typs zu vereinbaren. Struktur kann nur einmal im Rahmen dieses Projekts vereinbart werden. Zugriff auf die Strukturmitglieder erhält man durch die Operation Punkt (.) durchgeführt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
struct trade_settings
  {
   double take;          // Preis für Take-Profit
   double stop;          // Preis für Stop-Loss
   uchar  slippage;      // Wert der zulässigen Slippage
  };
//--- Variable des Typs trade_settings ist erzeugt und initialisiert
trade_settings my_set={0.0,0.0,5};
if (input_TP>0) my_set.take=input_TP;
```

'pack' zum Ausrichten von Struktur- und Klassenfeldern

Das spezielle Attribut pack ermöglicht die Ausrichtung von Struktur- oder Klassenfeldern.

```
pack([n])
```

wobei n einer der folgenden Werte ist: 1, 2, 4, 8 oder 16. Es kann auch fehlen.

Beispiel:

```
   struct pack(sizeof(long)) MyStruct
     {
      // Strukturmitglieder müssen auf eine 8-Byte-Grenze ausgerichtet werden
     };
oder
   struct MyStruct pack(sizeof(long))
     {
      // Strukturmitglieder müssen auf eine 8-Byte-Grenze ausgerichtet werden
     };
```

'pack(1)' wird standardmäßig auf Strukturen angewendet. Das bedeutet, dass sich die Strukturelemente nacheinander im Speicher befinden und die Strukturgröße gleich der Summe der Größe ihrer Elemente ist.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Einfache Struktur ohne Ausrichtung
   struct Simple_Structure
     {
      char              c; // sizeof(char)=1
      short             s; // sizeof(short)=2
      int               i; // sizeof(int)=4
      double            d; // sizeof(double)=8
     };
   //--- Deklaration einer einfachen Struktur
   Simple_Structure s;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Anzeige der Größe von jedem Strukturmitglied
   Print("sizeof(s.c)=",sizeof(s.c));
   Print("sizeof(s.s)=",sizeof(s.s));
   Print("sizeof(s.i)=",sizeof(s.i));
   Print("sizeof(s.d)=",sizeof(s.d));
//--- Sicherstellen, dass die Größe der POD-Struktur gleich der Summe der Mitgliedergr
   Print("sizeof(simple_structure)=",sizeof(simple_structure));
/*
  Ergebnis:
   sizeof(s.c)=1
   sizeof(s.s)=2
   sizeof(s.i)=4
   sizeof(s.d)=8
   sizeof(simple_structure)=15
*/
  }
```

Das Ausrichten der Strukturfelder kann erforderlich sein, wenn Daten mit Bibliotheken von Drittanbietern (*.DLL) ausgetauscht werden, bei denen diese Ausrichtung angewendet wird.

Lassen Sie uns anhand einiger Beispiele zeigen, wie die Ausrichtung funktioniert. Wir werden eine Struktur verwenden, die aus vier Mitgliedern besteht und nicht ausgerichtet ist.

```
//--- Einfache Struktur ohne Ausrichtung
   struct Simple_Structure pack() // es wird keine Größe angegeben, eine Ausrichtung a
     {
      char              c; // sizeof(char)=1
      short             s; // sizeof(short)=2
      int               i; // sizeof(int)=4
      double            d; // sizeof(double)=8
     };
//--- Deklaration einer einfachen Struktur
   Simple_Structure s;
```

Strukturfelder sind gemäß der Deklarationsreihenfolge und Typengröße nacheinander im Speicher angeordnet. Die Strukturgröße ist 15, während ein Offset zu den Strukturfeldern in den Arrays undefiniert ist.



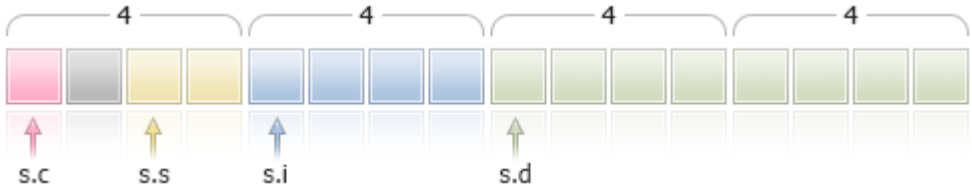
Deklarieren wir nun die gleiche Struktur mit der Ausrichtung von 4 Bytes und führen den Code aus.

```
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Eine einfache Struktur mit einer Ausrichtung von 4 Bytes
```

© 2000-2025, MetaQuotes Ltd.

```
   struct Simple_Structure pack(4)
     {
      char              c; // sizeof(char)=1
      short             s; // sizeof(short)=2
      int               i; // sizeof(int)=4
      double            d; // sizeof(double)=8
     };
   //--- Deklaration einer einfachen Struktur
   Simple_Structure s;
//--- Anzeige der Größe von jedem Strukturmitglied
   Print("sizeof(s.c)=",sizeof(s.c));
   Print("sizeof(s.s)=",sizeof(s.s));
   Print("sizeof(s.i)=",sizeof(s.i));
   Print("sizeof(s.d)=",sizeof(s.d));
//--- make sure the size of POD structure is now not equal to the sum of its members'
   Print("sizeof(simple_structure)=",sizeof(simple_structure));
/*
  Ergebnis:
   sizeof(s.c)=1
   sizeof(s.s)=2
   sizeof(s.i)=4
   sizeof(s.d)=8
   sizeof(simple_structure)=16 // structure size has changed
*/
  }
```

Die Strukturgröße hat sich so verändert, dass alle Mitglieder von 4 Bytes und mehr einen Offset vom Anfang des Strukturfachs von 4 Bytes haben. Kleinere Elemente sind auf ihre eigene Größengrenze auszurichten (z.B. 2 für 'short'). So sieht es aus (das hinzugefügte Byte wird grau dargestellt).



In diesem Fall wird 1 Byte nach dem Mitglied **s.c** hinzugefügt, so dass das Feld **s.s** (sizeof(short)==2) die Grenze von 2 Bytes hat (Ausrichtung für den Typ 'short').

Der Offset zum Anfang der Struktur im Array ist ebenfalls auf die 4-Byte-Grenze ausgerichtet, d.h. die Adressen der Elemente a[0], a[1] und a[n] sollen bei Simple_Structure arr[] ein Vielfaches von 4 Bytes sein.

Betrachten wir zwei weitere Strukturen, die aus ähnlichen Typen mit 4-Byte-Ausrichtung, aber mit einer unterschiedlichen Reihenfolge der Mitglieder. In der ersten Struktur befinden sich die Elemente in aufsteigender Reihenfolge der Typengröße.

```
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- Eine einfache Struktur, ausgerichtet auf eine Begrenzung von 4 Bytes
   struct CharShortInt pack(4)
     {
      char              c; // sizeof(char)=1
      short             s; // sizeof(short)=2
      int               i; // sizeof(double)=4
     };
//--- Deklaration einer einfachen Struktur
   CharShortInt ch_sh_in;
//--- Anzeige der Größe von jedem Strukturmitglied
   Print("sizeof(ch_sh_in.c)=",sizeof(ch_sh_in.c));
   Print("sizeof(ch_sh_in.s)=",sizeof(ch_sh_in.s));
   Print("sizeof(ch_sh_in.i)=",sizeof(ch_sh_in.i));

//--- Sicherstellen, dass die Größe der POD-Struktur gleich der Summe der Mitgliedergr
   Print("sizeof(CharShortInt)=",sizeof(CharShortInt));
/*
  Ergebnis:
   sizeof(ch_sh_in.c)=1
   sizeof(ch_sh_in.s)=2
   sizeof(ch_sh_in.i)=4
   sizeof(CharShortInt)=8
*/
  }
```

Wie wir sehen können, ist die Strukturgröße 8 und besteht aus den beiden 4-Byte-Blöcken. Der erste Block enthält die Felder mit den Typen 'char' und 'short', während der zweite Block das Feld mit dem Typ 'int' enthält.



Nun verwandeln wir die erste Struktur in die zweite, die sich nur in der Feldreihenfolge unterscheidet, indem wir das Element mit dem Typ 'short' an das Ende verschieben.

```
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Eine einfache Struktur, ausgerichtet auf eine Begrenzung von 4 Bytes
   struct CharIntShort pack(4)
     {
      char              c; // sizeof(char)=1
      int               i; // sizeof(double)=4
      short             s; // sizeof(short)=2
     };
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Deklaration einer einfachen Struktur
   CharIntShort ch_in_sh;
//--- Anzeige der Größe von jedem Strukturmitglied
   Print("sizeof(ch_in_sh.c)=",sizeof(ch_in_sh.c));
   Print("sizeof(ch_in_sh.i)=",sizeof(ch_in_sh.i));
   Print("sizeof(ch_in_sh.s)=",sizeof(ch_in_sh.s));
//--- Sicherstellen, dass die Größe der POD-Struktur gleich der Summe der Mitgliedergr
   Print("sizeof(CharIntShort)=",sizeof(CharIntShort));
/*
  Ergebnis:
   sizeof(ch_in_sh.c)=1
   sizeof(ch_in_sh.i)=4
   sizeof(ch_in_sh.s)=2
   sizeof(CharIntShort)=12
*/
  }
```

Obwohl sich der Strukturinhalt nicht geändert hat, hat sich die Größe durch die Änderung der Elementreihenfolge erhöht.



Die Ausrichtung sollte auch bei der Vererbung berücksichtigt werden. Lassen Sie uns dies anhand der einfachen übergeordneten Struktur mit einem einzigen Element vom Typ 'char' demonstrieren. Die Strukturgröße ohne Ausrichtung ist 1.

```
   struct Eltern
     {
      char              c;     // sizeof(char)=1
     };
```

Erstellen wir die abgeleitete Klasse Kind mit dem Mitglied vom Typ 'short' (sizeof(short)=2).

```
   struct Kind pack(2) : Parent
     {
      short             s;    // sizeof(short)=2
     };
```

Im Ergebnis ist die Strukturgröße gleich 4, wenn die Ausrichtung auf 2 Bytes festgelegt wird, obwohl die Größe der Elemente 3 ist. In diesem Beispiel sollen der übergeordneten Klasse 2 Bytes zugeordnet werden, so dass der Zugriff auf das Feld 'short' der Unterklasse auf 2 Bytes ausgerichtet ist.

Das Wissen darüber, wie Speicher für die Strukturmitglieder zugeteilt wird, ist notwendig, wenn eine MQL5-Anwendung mit gelesenen/geschriebenen Daten von Dateien oder Streams arbeitet.

Das MQL5\Include\WinAPI-Verzeichnis der Standardbibliothek enthält die Funktionen zum Arbeiten mit den WinAPI-Funktionen. Diese Funktionen wenden die Strukturen mit einer bestimmten Ausrichtung auf die Fälle an, in denen sie für die Arbeit mit WinAPI erforderlich sind.

© 2000-2025, MetaQuotes Ltd.

offsetof ist ein spezieller Befehl, der sich direkt auf das Attribut pack bezieht. Es ermöglicht uns, das Offset eines Elements am Anfang der Struktur zu abzurufen.

```
//--- Deklaration der Variablen vom Typ Kind
   Children child;
//--- Erkennen des Offsets vom Anfang der Struktur
   Print("offsetof(Children,c)=",offsetof(Children,c));
   Print("offsetof(Children,s)=",offsetof(Children,s));
/*
  Ergebnis:
   offsetof(Children,c)=0
   offsetof(Children,s)=2
*/
```

Modifikator 'final'

Das Vorhandensein des Modifikators 'final' bei der Deklaration der Struktur verbietet weitere Vererbung. Wenn es nicht notwendig ist, die Struktur weiter zu ändern oder wenn Änderungen aus Sicherheitsgründen unzulässig sind, deklarieren Sie diese mit dem Modifikator 'final'. Dadurch werden alle Mitglieder der Struktur implizit als endgültig gelten.

```
struct settings final
  {
  //--- Strukturkörper
  };

struct trade_settings : public settings
  {
  //--- Strukturkörper
  };
```

Bei einem Versuch der Vererbung von einer Struktur mit dem Modifikator 'final', wie im Beispiel oben, wirft der Kompiler einen Fehler aus:

```
cannot inherit from 'settings' as it has been declared as 'final'
see declaration of 'settings'
```

Klassen

Klassen unterscheiden sich von den Strukturen wie folgt:

- Das Schlüsselwort Klasse wird in der Deklaration verwendet;
- Standardmäßig haben alle Klassenmitglieder Zugriff auf die Spezifikation private, sofern nicht anders angegeben. Datenelemente der Struktur haben die standardmäßig die Zugriffsart 'public', sofern nicht anders angegeben;
- Klassenobjekte haben immer eine Tabelle mit virtuellen Funktionen, auch wenn in der Klasse keine virtuellen Funktionen deklariert sind. Strukturen können keine virtuellen Funktionen haben;
- Der Operator new kann auf Klassenobjekte angewendet werden, aber nicht auf Strukturen;
- Klassen können nur von Klassen abgeleitet werden, Strukturen nur von Strukturen.

© 2000-2025, MetaQuotes Ltd.

Klassen und Strukturen können einen expliziten Konstruktor und Destruktor haben. Wenn Ihr Konstruktor explizit definiert ist, ist die Initialisierung einer Struktur- oder Klassenvariablen mit der Initialisierungssequenz nicht möglich.

Beispiel:

```
struct trade_settings
  {
   double take;         // Preis von Take-Profit
   double stop;         // Preis von Stop-Loss
   uchar  slippage;     // Wert der zulässigen Slippage
   //--- Konstruktor
          trade_settings() { take=0.0; stop=0.0; slippage=5; }
   //--- Destruktor
         ~trade_settings() { Print("Das ist Ende"); }
  };
//--- Der Kompiler generiert eine Fehlermeldung, dass eine Initialisierung unmöglich i
trade_settings my_set={0.0,0.0,5};
```

Konstruktoren und Destruktoren

Konstruktor ist eine spezielle Funktion, die automatisch beim Erstellung von Struktur- oder Klassenobjekt aufgerufen. Er wird normalerweise und für Initialisierung von Klassenglieder verwendet. Weitere werden wir nur über die Klassen reden, während das gleiche gilt für Strukturen, soweit nicht anders angegeben. Der Name eines Konstruktors muss mit dem Klassennamen gleich sein. Der Konstruktor hat keinen Rückgabetyp (Sie können den Typ void angeben).

Definierte Klasseglieder - Zeichenketten, dynamische Arrays und Objekte, die Initialisierung erfordern - werden in jedem Fall initialisiert werden, unabhängig davon, ob es ein Konstruktor gibt.

Jede Klasse kann mehrere Konstruktoren mit einer unterschiedlichen Anzahl von Parametern und Initialisierungsliste haben. Ein Konstruktor, der die Angabe von Parametern erfordert, heißt parametrischer Konstruktor.

Ein Konstruktor ohne Parameter ist ein **Standard-Konstruktor**. Wenn keine Konstruktoren in einer Klasse deklariert sind, erstellt der Kompiler einen Standard-Konstruktor während der Kompilierung.

```
//+------------------------------------------------------------------+
//| Klasse für Arbeiten mit einem Datum                              |
//+------------------------------------------------------------------+
class MyDateClass
  {
private:
   int               m_year;          // Jahr
   int               m_month;         // Monat
   int               m_day;           // Tag des Monats
   int               m_hour;          // Stunde an einem Tag
   int               m_minute;        // Minuten
   int               m_second;        // Sekunden
public:
   //--- Standard-Konstruktor
```

© 2000-2025, MetaQuotes Ltd.

```
                     MyDateClass(void);
   //--- parametrischer Konstruktor
                     MyDateClass(int h,int m,int s);
  };
```

Sie können einen Konstruktor in der Klassenbeschreibung deklarieren und dann dessen Körper definieren. Beispielsweise können zwei Konstruktoren der Klasse MyDateClass auf folgende Weise definiert werden:

```
//+------------------------------------------------------------------+
//| Standard-Konstruktor                                             |
//+------------------------------------------------------------------+
MyDateClass::MyDateClass(void)
  {
//---
   MqlDateTime mdt;
   datetime t=TimeCurrent(mdt);
   m_year=mdt.year;
   m_month=mdt.mon;
   m_day=mdt.day;
   m_hour=mdt.hour;
   m_minute=mdt.min;
   m_second=mdt.sec;
   Print(__FUNCTION__);
  }
//+------------------------------------------------------------------+
//| Parametrischer Konstruktor                                       |
//+------------------------------------------------------------------+
MyDateClass::MyDateClass(int h,int m,int s)
  {
   MqlDateTime mdt;
   datetime t=TimeCurrent(mdt);
   m_year=mdt.year;
   m_month=mdt.mon;
   m_day=mdt.day;
   m_hour=h;
   m_minute=m;
   m_second=s;
   Print(__FUNCTION__);
  }
```

Im Standard-Konstruktor werden alle Klassenglieder mit Hilfe der Funktion TimeCurrent() gefüllt. Im parametrischen Konstruktor werden nur die Werte der Stunde übergeben. Die anderen Mitglieder der Klasse (m_year, m_month und m_day) werden automatisch mit dem aktuellen Datum initialisiert.

Der Standard-Konstruktor hat eine besondere Bedeutung bei der Initialisierung eines Arrays von Objekten seiner Klasse. Ein Konstruktor, der für alle Parameter Standardwerte setzt, **ist kein** Standard-Konstruktor. Wir zeigen dies am Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Klasse mit einem Standard-Konstruktor                            |
//+------------------------------------------------------------------+
class CFoo
  {
   datetime          m_call_time;     // Zeit des letzten Aufrufs des Objektes
public:
   //--- Konstruktor mit einem Parameter, der einen Standardwert hat, ist kein Standa
                     CFoo(const datetime t=0){m_call_time=t;};
   //--- Copy-Konstruktor
                     CFoo(const CFoo &foo){m_call_time=foo.m_call_time;};

   string ToString(){return(TimeToString(m_call_time,TIME_DATE|TIME_SECONDS));};
  };
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
// CFoo foo; // diese Option kann nicht verwendet werden - Standard-Konstruktor ist ni
//--- Gültige Optionen für Erstellung des Objekts CFoo
   CFoo foo1(TimeCurrent());     // Ein expliziter Aufruf des parametrischen Konstrukt
   CFoo foo2();                  // Ein expliziter Aufruf des parametrischen Konstrukt
   CFoo foo3=D'2009.09.09';      // Ein impliziter Aufruf des parametrischen Konstrukt
   CFoo foo40(foo1);             // Ein expliziter Aufruf des Copy-Konstruktors
   CFoo foo41=foo1;              // Ein impliziter Aufruf des Copy-Konstruktors
   CFoo foo5;                    // Ein expliziter Aufruf des Standard-Konstruktors (w
                                 // wird ein parametrischer Konstruktor mit Standardpa
//--- Gültige Optionen für Erlangung von Zeigern CFoo
   CFoo *pfoo6=new CFoo();       // dynamische Erstellung des Objekts und Erhaltung de
   CFoo *pfoo7=new CFoo(TimeCurrent());// Eine andere Variante der dynamischen Erstell
   CFoo *pfoo8=GetPointer(foo1); // pfoo8 zeigt nun auf Objekt foo1
   CFoo *pfoo9=pfoo7;            // pfoo9 und pfoo7 zeigen auf das gleiche Objekt
   // CFoo foo_array[3];         // diese Option kann nicht verwendet werden - Standa
//--- Ableiten Werte m_call_time
   Print("foo1.m_call_time=",foo1.ToString());
   Print("foo2.m_call_time=",foo2.ToString());
   Print("foo3.m_call_time=",foo3.ToString());
   Print("foo4.m_call_time=",foo4.ToString());
   Print("foo5.m_call_time=",foo5.ToString());
   Print("pfoo6.m_call_time=",pfoo6.ToString());
   Print("pfoo7.m_call_time=",pfoo7.ToString());
   Print("pfoo8.m_call_time=",pfoo8.ToString());
   Print("pfoo9.m_call_time=",pfoo9.ToString());
//--- Dynamisch erstellte Objekte löschen
   delete pfoo6;
   delete pfoo7;
   //delete pfoo8;  // Es ist nicht notwendig pfoo8 explizit zu löschen, da es auf das
   //delete pfoo9;  // Es ist nicht notwendig pfoo9 explizit zu löschen, da es auf das
```

© 2000-2025, MetaQuotes Ltd.

```
  }
```

Wenn Sie in diesem Beispiel diese Zeile entkommentieren

```
  //CFoo foo_array[3];     // diese Variante kann nicht verwendet werden - es wurde ke
```

oder

```
  //CFoo foo_dyn_array[];  // diese Variante kann nicht verwendet sein - es wurde kein
```

dann wird der Kompiler den Fehler "default constructor is not defined" zurückgeben.

Wenn die Klasse einen Konstruktor hat, der vom Benutzer definiert ist, wird der Standard-Konstruktor vom Kompiler nicht generiert. Dies bedeutet, dass, wenn ein parametrischer Konstruktor in einer Klasse deklariert worden ist, aber kein Standard-Konstruktor, kann man die Arrays von Objekten dieser Klasse nicht deklarieren. Der Kompiler für diesen Skript einen Fehler auswerfen:

```
//+------------------------------------------------------------------+
//| Eine Klasse ohne Standard-Konstruktor                            |
//+------------------------------------------------------------------+
class CFoo
  {
   string            m_name;
public:
                     CFoo(string name) { m_name=name;}
  };
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Fehler "default constructor is not defined" wird beim Kompilieren zurückgegeben
   CFoo badFoo[5];
  }
```

In diesem Beispiel hat die Klasse CFoo einen deklarierten parametrischen Konstruktor - in solchen Fällen erstellt der Kompiler keinen Standard-Konstruktor während der Kompilierung. Gleichzeitig wird bei der Deklaration eines Arrays von Objekten davon ausgegangen, dass alle Objekte automatisch angelegt und initialisiert werden sollen. Bei der automatischen Initialisierung eines Objekts ist es notwendig, einen Standard-Konstruktor aufzurufen, aber da der Standard-Konstruktor nicht explizit deklariert und nicht automatisch vom Kompiler generiert wird, ist es unmöglich, ein solches Objekt zu erstellen. Aus diesem Grund erzeugt der Kompiler bei der Kompilierung einen Fehler.

Es gibt eine spezielle Syntax, um Objekte durch dem Konstruktor zu initialisieren. Konstruktor-Initialisierer (Sonderkonstruktionen zur Initialisierung) für die Mitglieder einer Struktur oder Klasse können in der Initialisierungsliste angegeben werden.

Eine Initialisierungsliste ist eine durch Kommata getrennte Liste von Initialisatoren, die nach dem Doppelpunkt nach der Liste der Parameter eines Konstruktors steht und dem Körper vorangestellt wird (steht vor einer öffnenden Klammer). Es gibt mehrere Anforderungen:

- Initialisierungslisten können nur in Konstruktoren verwendet werden.
- Mitglieder von Eltern können nicht in der Initialisierungsliste initialisiert werden

© 2000-2025, MetaQuotes Ltd.

- Nach der Initialisierungsliste muss Definition (Implementierung) der Funktion folgen.

Hier ist ein Beispiel für mehrere Konstruktoren zur Initialisierung von Klassenmitgliedern.

```
//+------------------------------------------------------------------+
//| Klasse um den Namen einer Person zu speichern                    |
//+------------------------------------------------------------------+
class CPerson
  {
   string            m_first_name;     // Vorname
   string            m_second_name;    // Familienname
public:
   //--- Ein leerer Standard-Konstruktor
                     CPerson() {Print(__FUNCTION__);};
   //--- Parametrischer Konstruktor
                     CPerson(string full_name);
   //--- Ein Konstruktor mit der Initialisierungsliste
                     CPerson(string surname,string name): m_second_name(surname), m_fi
   void PrintName(){PrintFormat("Name=%s Surname=%s",m_first_name,m_second_name);};
  };
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
CPerson::CPerson(string full_name)
  {
   int pos=StringFind(full_name," ");
   if(pos>=0)
     {
      m_first_name=StringSubstr(full_name,0,pos);
      m_second_name=StringSubstr(full_name,pos+1);
     }
  }
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Erhalten eine Fehlermeldung "default constructor is not defined"
   CPerson people[5];
   CPerson Tom="Tom Sawyer";                        // Tom Sawyer
   CPerson Huck("Huckleberry","Finn");              // Huckleberry Finn
   CPerson *Pooh = new CPerson("Winnie","Pooh");  // Winnie the Pooh
   //--- Werte ausgeben
   Tom.PrintName();
   Huck.PrintName();
   Pooh.PrintName();

   //--- Das dynamisch erstellte Objekt löschen
   delete Pooh;
  }
```

© 2000-2025, MetaQuotes Ltd.

In diesem Fall hat die Klasse CPerson drei Konstruktoren:

1. Einen expliziten Standard-Konstruktor, der das Erstellen eines Arrays von Objekten dieser Klasse ermöglicht.
2. Ein Konstruktor mit einem Parameter, der einen vollständigen Namen als Parameter erhält und ihn entsprechend der gefundenen Anordnung in den Namen und Vornamen teilt;
3. Ein Konstruktor mit zwei Parametern, der Initialisierungsliste enthält. Initialisieren - m_second_name(surname) und m_first_name(name).

Beachten Sie, wie die Initialisierung mit einer Liste die Zuweisung ersetzt. Die einzelnen Mitglieder werden wie folgt initialisiert:

```
   Klassenglieder (Liste von Ausdrücken)
```

In der Initialisierungsliste können die Mitglieder in beliebiger Reihenfolge stehen, aber alle Mitglieder der Klasse werden nach der Reihenfolge der Deklaration initialisiert. Dies bedeutet, dass im dritten Konstruktor zuerst das Mitglied m_first_name initialisiert, da es an erster Stelle steht, und danach m_second_name. Dies sollte in Fällen berücksichtigt werden, in denen die Initialisierung von einigen Klassenmitgliedern von den Werten der anderen Klassenmitglieder abhängt.

Wenn kein Standard-Konstruktor in der Basisklasse deklariert ist, und ein oder mehrere Konstruktoren mit Parametern deklariert sind, sollten Sie immer einen der Basisklassenkonstruktoren in der Initialisierungsliste anrufen. Sie ist eine Kommata getrennte, reguläre Mitgliederliste und wird bei der Initialisierung des Objekts unabhängig von der Position in der Initialisierungsliste zuerst aufgerufen.

```
//+------------------------------------------------------------------+
//| Basisklasse                                                      |
//+------------------------------------------------------------------+
class CFoo
  {
   string            m_name;
public:
   //--- Konstruktor mit der Initialisierungsliste
                     CFoo(string name) : m_name(name) { Print(m_name);}
  };
//+------------------------------------------------------------------+
//| Nachfolger der Klasse CFoo                                       |
//+------------------------------------------------------------------+
class CBar : CFoo
  {
   CFoo              m_member;      // Klassenglieder ist ein Objekt von Parent
public:
   //--- Standard-Konstruktor ruft den Konstruktor von Eltern in der Initialisierungsl
                     CBar(): m_member(_Symbol), CFoo("CBAR") {Print(__FUNCTION__);}
  };
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   CBar bar;
```

© 2000-2025, MetaQuotes Ltd.

```
   }
```

In diesem Beispiel während der Erstellung des Objekts bar wird Standard-Konstruktor CBar() aufgerufen. In diesem Standard-Konstruktor erst wird Konstruktor für den Vorfahr CFoo, dann wird Konstruktor für Klassenglied m_member augerufen.

Destruktor ist eine Sonderfunktion, die authomatisch aufgerufen wird, wenn das Klassenobjekt beseitigt wird. Destruktorname wird so geschrieben, wie der Klassenname mit Tilde(~). Zeilen, dynamische Felde und Objekte, die initialisiert werden müssen, werden auf jeden Fall deinitialisiert, unabhängig von der Anwesenheit des Destruktors. Wenn es einen Destruktor gibt, werden diese Handlungen nach Destruktoraufruf durchgeführt werden.

Destruktoren sind immer virtuell, unabhängig davon, ob sie mit dem Schlüuesselwort virtual erklärt werden oder nicht.

Methodenbestimmung der Klassen

Funktionen-Methoden der Klasse können sowohl innerhalb, als auch ausserhalb der Klassenerklärung bestimmt werden. Wenn die Methode innerhalb der Klasse bestimmt wird, folgt Ihr Körper unmittelbar nach der Methodenerklärung.

Beispiel:

```
class CTetrisShape
  {
protected:
   int               m_type;
   int               m_xpos;
   int               m_ypos;
   int               m_xsize;
   int               m_ysize;
   int               m_prev_turn;
   int               m_turn;
   int               m_right_border;
public:
   void              CTetrisShape();
   void              SetRightBorder(int border) { m_right_border=border; }
   void              SetYPos(int ypos)          { m_ypos=ypos;           }
   void              SetXPos(int xpos)          { m_xpos=xpos;           }
   int               GetYPos()                  { return(m_ypos);        }
   int               GetXPos()                  { return(m_xpos);        }
   int               GetYSize()                 { return(m_ysize);       }
   int               GetXSize()                 { return(m_xsize);       }
   int               GetType()                  { return(m_type);        }
   void              Left()                     { m_xpos-=SHAPE_SIZE;    }
   void              Right()                    { m_xpos+=SHAPE_SIZE;    }
   void              Rotate()                   { m_prev_turn=m_turn; if(++m_turn>3) r
   virtual void      Draw()                     { return;                }
   virtual bool      CheckDown(int& pad_array[]);
   virtual bool      CheckLeft(int& side_row[]);
   virtual bool      CheckRight(int& side_row[]);
  };
```

Funktionen mit SetRightBorder(int border) nach Draw() werden vereinbart und bestimmt innerhalb der Klasse CTetrisShape.

© 2000-2025, MetaQuotes Ltd.

Konstruktor CTetrisShape() und Methoden CheckDown(int& pad_array[]), CheckLeft(int& side_row[]) und CheckRight(int& side_row[]) werden nur innerhalb der Klasse erklärt, aber noch nicht bestimmt. Bestimmungen dieser Funktionen müssen weiter nach Kode folgen. Um die Methode ausser der Klasse zu bestimmen, wird Operation der Kontexterlaubnis verwendet , als Kontext funktioniert der Klassenname.

Beispiel:

```
//+------------------------------------------------------------------+
//| Konstruktor der Basisklasse                                      |
//+------------------------------------------------------------------+
void CTetrisShape::CTetrisShape()
  {
   m_type=0;
   m_ypos=0;
   m_xpos=0;
   m_xsize=SHAPE_SIZE;
   m_ysize=SHAPE_SIZE;
   m_prev_turn=0;
   m_turn=0;
   m_right_border=0;
  }
//+------------------------------------------------------------------------+
//| Kontrolle der Möglichkeit nach unten zu bewegen (für Stab und Kubus) |
//+------------------------------------------------------------------------+
bool CTetrisShape::CheckDown(int& pad_array[])
  {
   int i,xsize=m_xsize/SHAPE_SIZE;
//---
   for(i=0; i<xsize; i++)
     {
      if(m_ypos+m_ysize>=pad_array[i]) return(false);
     }
//---
   return(true);
  }
```

Zugangsmodifikatoren public, protected und private

Bei der Ausarbeitung der neuen Klasse ist es empfohlen, den Zugang zu Aussengliedern zu begrenzen. Für diese Zwecke werden Schlüsselwörter private oder protected verwendet. In diesem Fall wird der Zugang zu diesen versteckten Daten nur aus Funktionen-Methoden derselben Klasse erfolgen. Wenn das Schluesselwort protected verwendet wird, kann der Zugang zu versteckten Daten auch aus Methoden der Klassen erfolgen - Nachfolger dieser Klasse. In gleicher Weise kann der Zugang auch zu Funktionen-Metoden der Klasse begrenzt werden.

Wenn es notwendig ist, den Zugang zu Mitgliedern und/oder Methoden der Klasse vollkommen zu eröffnen, wird das Schlüsselwort publicverwendet.

Beispiel:

```
class CTetrisField
  {
private:
   int               m_score;                        // Stand
```

© 2000-2025, MetaQuotes Ltd.

```
   int               m_ypos;                              // laufende Figurlage
   int               m_field[FIELD_HEIGHT][FIELD_WIDTH]; // Dom Matrix
   int               m_rows[FIELD_HEIGHT];                // Numerierung von DOM Reihen
   int               m_last_row;                          // Letzte freie Reihe
   CTetrisShape     *m_shape;                             // Tetrisfigur
   bool              m_bover;                             // Spiel beendet
public:
   void              CTetrisField() { m_shape=NULL; m_bover=false; }
   void              Init();
   void              Deinit();
   void              Down();
   void              Left();
   void              Right();
   void              Rotate();
   void              Drop();
private:
   void              NewShape();
   void              CheckAndDeleteRows();
   void              LabelOver();
  };
```

Alle Mitglieder und Methoden der Klasse, definiert nach dem Spezifikator **public:** (und bis nächsten Spezifikator), sind für jeden Zugriff auf ein Objekt dieser Klasse zugänglich. In diesem Beispiel sind es folgende Mitglieder: Die Funktionen CTetrisField(), Init(), Deinit(), Down(), Left(), Right(), Rotate() und Drop().

Alle Mitglieder der Klasse, deklariert nach dem Spezifikator **private:** (und bis nächsten Spezifikator) sind nur für die Funktions-Mitglieder dieser Klasse zugänglich. Spezifikatoren für einen Zugriff auf die Elemente werden immer mit einem Doppelpunkt (:) und können in der Klassenbestimmung mehrmals auftreten.

Alle Klassenmitglieder, die nach **'protected' deklariert wurden:** Zugriffsspezifikation (und bis zum nächsten Zugriffsspezifikation) sind nur für Mitgliederfunktionen dieser Klasse und Mitgliederfunktionen der abgeleiteten Klassen verfügbar. Bei einem versuch auf die Mitglieder zu zuzugreifen, die innerhalb der Spezifikatoren 'private' oder 'protected' deklariert wurden, erhalten wir einen Fehler bei der Kompilierung. Beispiel:

```
class A
  {
protected:
   //--- der Kopieroperator ist nur innerhalb der Klasse A und ihrer Nachkommen verfüg
   void operator=(const A &)
     {
     }
  };
class B
  {
   //--- Klassenobjekt A deklarieren
   A                 a;
  };
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   //--- zwei Variablen des Typs B deklarieren
   B b1, b2;
   //--- Versuch, das eine Objekt in das andere zu kopieren
   b2=b1;
  }
```

Beim Kompilieren des Codes wird eine Fehlermeldung ausgeworfen — ein Versuch, die ein Objekt auf anderes zu kopieren:

```
attempting to reference deleted function 'void B::operator=(const B&)'   trash3.mq5
```

Die zweite Zeile unten bietet eine detailliertere Beschreibung — der Kopieroperator in Klasse B wurde explizit gelöscht, da der nicht verfügbare Kopieroperator der Klasse A aufgerufen wird:

```
   function 'void B::operator=(const B&)' was implicitly deleted because it invokes in
```

Der Zugriff auf die Mitglieder der Basisklasse kann während Vererbung in abgeleiteten Klassen neu definiert werden.

Zugang zu Gliedern der Basisklasse kann bei Vererbung in abgeleiteten Klassen neu definiert werden.

'delete' Spezifikator

Der Spezifikator delete markiert Funktionen einer Klasse, die nicht verwendet werden können. Das heißt, wenn das Programm explizit oder implizit auf eine solche Funktion verweist, wird der Fehler bereits bei der Kompilierung ausgeworfen. Mit diesem Spezifikator können Sie beispielsweise Methoden der Eltern in einer Unterklasse nicht verfügbar machen. Das gleiche Ergebnis kann erreicht werden, wenn wir die Funktion im privaten Bereich der Elternklasse deklarieren (Deklarationen im Abschnitt private). Hier macht die Verwendung von delete den Code auf der Ebene der Nachkommen lesbarer und verwaltbarer.

```
class A
  {
public:
                     A(void) {value=5;};
   double            GetValue(void) {return(value);}
private:
   double            value;
  };
class B: public A
  {
   double            GetValue(void)=delete;
  };
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Variable vom Typ A deklarieren
   A a;
   Print("a.GetValue()=", a.GetValue());
//--- Versuch, den Wert der Variablen des Typs B abzufragen
   B b;
   Print("b.GetValue()=", b.GetValue()); // Der Kompiler zeigt für diese Zeile einen F
  }
```

Die Kompilernachricht

```
attempting to reference deleted function 'double B::GetValue()'
   function 'double B::GetValue()' was explicitly deleted here
```

Der Spezifikator 'delete' erlaubt es, das Auto-Casting oder den Kopierkonstruktor zu deaktivieren, die sonst auch im Abschnitt 'private' versteckt werden müssten. Beispiel:

```
class A
  {
public:
   void              SetValue(double v) {value=v;}
   //--- alle Aufrufe für den Typ int verhindern
   void              SetValue(int) = delete;
   //--- den Kopieroperator verhindern
   void              operator=(const A&) = delete;
private:
   double            value;
  };
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- zwei Variablen des Typs A deklarieren
   A a1, a2;
   a1.SetValue(3);      // error!
   a1.SetValue(3.14);   // OK
   a2=a1;               // error!
  }
```

Während der Kompilierung erhalten wir die Fehlermeldungen:

```
attempting to reference deleted function 'void A::SetValue(int)'
   function 'void A::SetValue(int)' was explicitly deleted here
attempting to reference deleted function 'void A::operator=(const A&)'
   function 'void A::operator=(const A&)' was explicitly deleted here
```

Modifikator 'final'

Das Vorhandensein des Modifikators 'final' bei der Deklaration der Struktur verbietet weitere Vererbung. Wenn es nicht notwendig ist, die Struktur weiter zu ändern oder wenn Änderungen aus

© 2000-2025, MetaQuotes Ltd.

Sicherheitsgründen unzulässig sind, deklarieren Sie diese mit dem Modifikator 'final'. Dabei werden alle Glieder der Struktur implizit auch als final gelten.

```
class CFoo final
  {
  //--- Körper der Klasse
  };

class CBar : public CFoo
  {
  //--- Körper der Klasse
  };
```

Bei einem Versuch der Vererbung von einer Struktur mit dem Modifikator 'final', wie im Beispiel oben, wirft der Kompiler einen Fehler aus:

```
cannot inherit from 'CFoo' as it has been declared as 'final'
see declaration of 'CFoo'
```

Unions (union)

Eine Union stellt einen besonderen Datentyp dar und setzt sich aus mehreren Variablen zusammen, die sich denselben Speicherbereich teilen. Daraus folgt, dass eine Union die Interpretation einer und derselben Reihenfolge der Bytes auf zwei (oder mehr) verschiedenen Weisen ermöglicht. Die Deklaration einer Union ist gleich der Deklaration einer Struktur und beginnt mit dem Schlüsselwort union.

```
union LongDouble
{
  long   long_value;
  double double_value;
};
```

Im Gegensatz zu Strukturen, gehören verschiedene Komponenten einer Union zu demselben Speicherbereich. In diesem Beispiel wurde die LongDouble Union deklariert, in welcher sich der Wert vom Typ long und der Wert vom Typ double einen Speicherbereich teilen. Es ist wichtig zu verstehen, die Union kann nicht gleichzeitig den ganzzahligen Wert *long* und den reellen Wert *double* speichern (wie es in Strukturen der Fall ist), denn die Variablen long_value und double_value überschneiden sich (im Speicher). Das MQL5-Programm kann aber jederzeit die Information, die in der Union enthalten ist, als einen ganzzahligen (long) oder reellen Wert (double) bearbeiten. Daraus folgt, dass die Union die Interpretation einer und derselben Reihenfolge von Daten auf zwei (oder mehr) verschiedenen Weisen ermöglicht.

Bei der Deklaration der Union reserviert der Compiler einen Speicherbereich, der für das Speichern des größten nach dem Volumen Typs in der Union der Variablen ausreichend ist. Für den Zugang zu einem Element der Union wird dieselbe Syntax wie für Strukturen verwendet - der Point-Operator.

```
union LongDouble
{
  long   long_value;
  double double_value;
};
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   LongDouble lb;
//--- erhalten wir die ungültige Zahl -nan(ind) und geben sie aus
   lb.double_value=MathArcsin(2.0);
   printf("1.  double=%f                integer=%I64X",lb.double_value,lb.long_value);
//--- die größte normalisierte Zahl (DBL_MAX)
   lb.long_value=0x7FEFFFFFFFFFFFFF;
   printf("2.  double=%.16e  integer=%I64X",lb.double_value,lb.long_value);
//--- die kleinste positive normalisierte Zahl (DBL_MIN)
   lb.long_value=0x0010000000000000;
   printf("3.  double=%.16e  integer=%.16I64X",lb.double_value,lb.long_value);
  }
/*  Das Ergebnis der Ausführung
    1.  double=-nan(ind)                integer=FFF8000000000000
    2.  double=1.7976931348623157e+308  integer=7FEFFFFFFFFFFFFF
    3.  double=2.2250738585072014e-308  integer=0010000000000000
*/
```

Da Unions es dem Programm erlauben, dieselben Daten im Speicher unterschiedlich zu interpretieren, werden sie häufig in den Fällen verwendet, wenn eine ungewöhnliche Typumwandlung benötigt wird.

Auf Unions kann keine Vererbung angewendet werden und sie können per Definition keine statischen Members haben. Sonst verhält sich *union* wie eine Struktur, deren Mitglieder eine Nullpunktverschiebung haben. Die folgenden Typen können nicht Mitglieder einer Union sein:

- dynamische Arrays
- Strings
- Pointer auf Objekte und Funktionen
- Objekte von Klassen
- Objekte der Strukturen, die Konstruktoren und Destruktoren haben
- Objekte der Strukturen, die Mitglieder der Punkte 1-5 beinhalten

Eine Union kann, so wie Klassen, Konstruktoren und Destruktoren und Methoden haben. Standardmäßig haben die Members einer Union den Zugriffstyp public, für die Erstellung privater Elemente ist das Schlüsselwort private zu verwenden. Alle diese Möglichkeiten sind in einem Beispiel vorgestellt. Das Beispiel zeigt, wie man die Farbe mit dem Typ color in ARGB umwandeln kann, wie dies die ColorToARGB() Funktion macht.

```
//+------------------------------------------------------------------+
//| Union für die Umwandlung von color(BGR) in ARGB      |
//+------------------------------------------------------------------+
union ARGB
  {
   uchar             argb[4];
   color             clr;
   //--- Konstruktoren
                     ARGB(color col,uchar a=0){Color(col,a);};
```

© 2000-2025, MetaQuotes Ltd.

```
                    ~ARGB(){};
   //--- öffentliche Methoden
public:
   uchar    Alpha(){return(argb[3]);};
   void     Alpha(const uchar alpha){argb[3]=alpha;};
   color    Color(){ return(color(clr));};
   //--- private Methoden
private:
   //+------------------------------------------------------------------+
   //| Setzen der Farbe und des Wertes des Alpha-Kanals                 |
   //+------------------------------------------------------------------+
   void     Color(color col,uchar alpha)
     {
      //--- Farbe dem clr Member setzen
      clr=col;
      //--- den Wert der Alpha Komponente setzen - Transparenzlevel
      argb[3]=alpha;
      //--- die Bytes R und B (Red und Blue) miteinander tauschen
      uchar t=argb[0];argb[0]=argb[2];argb[2]=t;
     };
  };
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- 0x55 bedeutet 55/255=21.6 % (0% - völlig durchsichtig)
   uchar alpha=0x55;
//--- der color Typ wird als 0x00BBGGRR dargestellt
   color test_color=clrDarkOrange;
//--- hier erhalten wir die Werte der Bytes aus der ARGB Union
   uchar argb[];
   PrintFormat("0x%.8X - so sieht der color Typ für %s aus, BGR=(%s)",
               test_color,ColorToString(test_color,true),ColorToString(test_color));
//--- der ARGB Typ wird als 0x00RRGGBB dargestellt, die RR und BB Komponenten haben Pl
   ARGB argb_color(test_color);
//--- kopieren wir das Array der Bytes
   ArrayCopy(argb,argb_color.argb);
//--- so sieht es in der ARGB Darstellung aus
   PrintFormat("0x%.8X - ARGB Darstellung mit dem Alpha-Kanal=0x%.2x, ARGB=(%d,%d,%d,%
               argb_color.clr,argb_color.Alpha(),argb[3],argb[2],argb[1],argb[0]);
//--- fügen wir den Wert der Transparenz hinzu
   argb_color.Alpha(alpha);
//--- versuchen wir ARGB als "color" Typ zu definieren
   Print("ARGB als color=(",argb_color.clr,")  Alpha-Kanal=",argb_color.Alpha());
//--- kopieren wir das Array der Bytes
   ArrayCopy(argb,argb_color.argb);
//--- so sieht es in der ARGB Darstellung aus
   PrintFormat("0x%.8X - ARGB Darstellung mit dem Alpha-Kanal=0x%.2x, ARGB=(%d,%d,%d,%
```

© 2000-2025, MetaQuotes Ltd.

```
                argb_color.clr,argb_color.Alpha(),argb[3],argb[2],argb[1],argb[0]);
//--- vergleichen mit dem Ergebnis der ColorToARGB() Funktion
   PrintFormat("0x%.8X - Ergebnis von ColorToARGB(%s,0x%.2x)",ColorToARGB(test_color,a
                ColorToString(test_color,true),alpha);
  }
/* Das Ergebnis der Ausführung
   0x00008CFF - so sieht der Typ color für clrDarkOrange, BGR=(255,140,0)
   0x00FF8C00 - ARGB Darstellung mit dem Alpha-Kanal=0x00, ARGB=(0,255,140,0)
   ARGB als color=(0,140,255)  Alpha-Kanal=85
   0x55FF8C00 - ARGB Darstellung mit dem Alpha-Kanal=0x55, ARGB=(85,255,140,0)
   0x55FF8C00 - Ergebnis von ColorToARGB(clrDarkOrange,0x55)
*/
```

Schnittstellen (Interfaces)

Eine Schnittstelle gibt an, welche Funktionalität eine Klasse dann umsetzen kann. In der Tat ist sie eine Klasse, die keine Mitglieder enthalten kann und keinen Konstruktor und/oder Destruktor hat. Alle in einer Schnittstelle deklarierten Methoden sind rein virtuell, auch ohne eine explizite Definition.

Eine Schnittstelle wird mit dem Schlüsselwort interface definiert. Zum Beispiel:

```
//--- Basisschnittstelle für die Beschreibung von Tieren
interface ITier
  {
//--- Die Methoden einer Schnittstelle haben public-Zugang
   void Sound();  // Tierlaut
  };
//+------------------------------------------------------------------+
//|  Klasse CKatze ist von der Schnittstelle ITier geerbt            |
//+------------------------------------------------------------------+
class CKatze : public ITier
  {
public:
                     CKatze() { Print("Katze was born"); }
                    ~CKatze() { Print("Katze is dead");  }
   //--- Implementieren die Methode Sound der Schnittstelle ITier
   void Sound(){ Print("meou"); }
  };
//+------------------------------------------------------------------+
//|  Klasse CHund ist von der Schnittstelle ITier geerbt             |
//+------------------------------------------------------------------+
class CHund : public ITier
  {
public:
                     CHund() { Print("Hund was born"); }
                    ~CHund() { Print("Hund is dead");  }
   //--- Implementieren die Methode Sound der Schnittstelle ITier
   void Sound(){ Print("guaf"); }
  };
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ein Array von Zeigern auf Objekte vom Typ ITier
  ITier *animals[2];
//--- Wir erzeugen abgeleiteten Klassen von ITier und speichern Zeiger auf sie in eine
   animals[0]=new CKatze;
   animals[1]=new CHund;
//--- Rufen auf die Methode Sound() der Basisschnittstelle ITier für jede Kindklasse
   for(int i=0;i<ArraySize(animals);++i)
      animals[i].Sound();
//--- Objekte löschen
   for(int i=0;i<ArraySize(animals);++i)
      delete animals[i];
//--- Das Ergebnis
/*
  Katze was born
  Hund was born
   meou
   guaf
  Katze is dead
  Hund is dead
*/
  }
```

Wie im Fall mit abstrakte Klassen, kann man nicht eine Objekt-Schnittstelle ohne Vererbung erstellen. Die Schnittstelle kann nur von anderen Schnittstellen vererbt werden, und auch kann Eltern der Klasse sein. Eine Schnittstelle hat immer die öffentliche Sichtbarkeit.

Die Schnittstelle kann nicht innerhalb einer Klassendeklaration oder Strukturdeklaration deklariert werden, aber man kann den Zeiger auf die Schnittstelle in einer Variablen vom Typ void * speichern. Im Allgemeinen können Sie einen Zeiger auf ein Objekt einer beliebigen Klasse in der Variable vom Typ void * speichern. Um einen void * Zeiger auf einen Zeiger auf ein Objekt einer bestimmten Klasse umzuwandeln, wird der Operator dynamic_cast verwendet. Wenn die Umwandlung nicht möglich ist, wird das Ergebnis der Operation dynamic_cast NULL.

Siehe auch

Objektorientiertes Programmieren

© 2000-2025, MetaQuotes Ltd.

Objekt des dynamischen Arrays

Dynamische Arrays

Maximum kann ein vierdimensionales Array zugelassen. Bei der Erklaerung des dynamischen Arrays (Array mit einem nicht angegebenen Wert im ersten Paar der eckigen Klammern) erzeugt der Compiler automatisch eine Variable der oben angegebenen Struktur (Objekt des dynamischen Feldes) und stellt den Kode für die richtige Initialisierung bereit.

Dynamische Arrays werden automatisch befreit, wenn sie ausserhalb des Blocks gehen, wo sie erklärt werden.

Beispiel:

```
double matrix[][10][20]; // dreidimensionales dynamischer Feld
ArrayResize(matrix,5);   // stellen die Größe der ersten Messung
```

Statische Arrays

Bei der explizeten Angebung aller bedeutenden Dimensionen des Arrays, verteilt der Compiler im voraus die erforderliche Speichergroesse. Ein solches Array wird statisch genannt. Dennoch verteilt der Compiler Speicher für Objekt des dynamischen Arrays, das (Objekt) mit dem vorher verteilten Puffer verbunden ist (Speicherteil für Feldaufbewahren).

Erzeugen eines Objekts des dynamischen Arrays ist durch die moegliche Notwendigkeit bedingt, das angegebene statische Feld in eine Funktion als Parameter zu übertragen.

Beispiele:

```
double stat_array[5]; // eindimensionales statisches Feld
some_function(stat_array);
...
bool some_function(double& array[])
  {
   if(ArrayResize(array,100)<0) return(false);
   ...
   return(true);
  }
```

Arrays als Strukturteile

Bei der Deklaration eines statischen Arrays als Teil einer Struktur wird ein Objekt des dynamischen Arrays nicht erzeugt. Das wird für Kompatibilitaet der Datenstrukturen gemacht, die in Windows API verwendet werden.

Aber statische Felder, die als Strukturglieder erklärt werden, können auch in MQL5-Funktionen übertragen werden. In diesem Fall wird bei Parameterübertragung ein zeitliches Objekt des dynamischen Feldes erzeugt werden, das mit dem statischen Array - Strukturglied verbunden ist.

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Operationen mit Arrays, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Matrizen und Vektoren

Der Typ Vektor ist ein spezieller Datentyp in MQL5, der Operationen mit Vektoren ermöglicht. Ein Vektor ist ein eindimensionales Array vom Typ Double. Er ist eines der grundlegenden Konzepte der linearen Algebra, die in vielen Bereichen der Wissenschaft, einschließlich Physik, Geometrie und anderen, verwendet wird. Vektoren werden zur Lösung linearer Gleichungssysteme, in der 3D-Grafik und in anderen Anwendungsbereichen verwendet. Vektoren können addiert und multipliziert werden. Die Länge oder der Abstand zwischen Vektoren kann durch die Norm ermittelt werden. In der Programmierung werden Vektoren in der Regel durch Arrays homogener Elemente dargestellt, die keine regulären Vektoroperationen ausführen können, d. h., wenn Arrays nicht addiert oder multipliziert werden können und keine Norm haben.

Bei der Arbeit mit Matrizen können Vektoren als Zeilenvektoren und Stringvektoren dargestellt werden. Außerdem verwenden Vektoren in der linearen Algebra die Konzepte der Kovarianz und Kontravarianz. Diese Konzepte machen beim Schreiben eines MQL5-Codes keinen Unterschied, da nur der Programmierer entscheidet, was jedes Objekt des Vektortyps ist. In der 3D-Grafik kann es sich zum Beispiel um einen Rotations-, Verschiebungs- oder Kompressionsvektor handeln.

Allgemein gesprochen ist eine Zahl aus Sicht der linearen Algebra auch ein Vektor, allerdings in einem eindimensionalen Vektorraum. Ein Vektor selbst kann als ein Spezialfall einer Matrix betrachtet werden.

Der Typ Matrix ist ein weiterer spezieller Datentyp in MQL5 zur Darstellung von Matrizen. Eine Matrix ist eigentlich ein zweidimensionales Array vom Typ double. Vektoren und Matrizen wurden in MQL5 eingeführt, um Operationen mit bestimmten Arten von Datensätzen zu erleichtern. Mit ihnen können Entwickler die Möglichkeiten der linearen Algebra in einer einfachen und mathematikähnlichen Form nutzen. Matrizen können verwendet werden, um Systeme von linearen Gleichungen oder Differentialgleichungen kompakt zu schreiben. Die Anzahl der Matrixzeilen entspricht der Anzahl der Gleichungen, während die Anzahl der Spalten gleich der Anzahl der Unbekannten ist. Folglich können lineare Gleichungssysteme durch Matrixoperationen gelöst werden.

Die folgenden algebraischen Operationen sind für die Matrizen definiert:

- Addition von Matrizen gleicher Größe.
- Multiplikation von Matrizen geeigneter Größe: die Anzahl der Spalten der linken Matrix muss gleich der Anzahl der Zeilen der rechten Matrix sein.
- Matrixmultiplikation mit einem Spaltenvektor; Multiplikation eines Zeilenvektors mit einer Matrix nach der Matrixmultiplikationsregel. In diesem Sinne ist der Vektor ein Spezialfall einer Matrix.
- Matrixmultiplikation mit einer Zahl, d. h. mit einem Skalar.

In der Mathematik gibt es viele verschiedene Matrixtypen. Zum Beispiel die Identitätsmatrix, symmetrische, schiefsymmetrische, obere und untere Dreiecksmatrizen und andere Typen. Verschiedene Normalformen spielen eine wichtige Rolle in der Matrixtheorie. Sie stellen eine bestimmte kanonische Form einer Matrix dar, die durch bestimmte Umformungen erhalten werden kann. In der Praxis werden Normalformen verwendet, die zusätzliche Eigenschaften haben, wie z. B. Stabilität.

Die Verwendung von Vektoren und Matrizen, oder besser gesagt, von speziellen Methoden der entsprechenden Typen, ermöglicht die Erstellung eines einfacheren, kürzeren und klareren Codes, der der mathematischen Notation nahe kommt. Mit diesen Methoden müssen Sie keine verschachtelten Schleifen erstellen oder auf die korrekte Indizierung von Arrays in Berechnungen achten. Daher erhöht

© 2000-2025, MetaQuotes Ltd.

die Verwendung dieser Methoden die Zuverlässigkeit und Geschwindigkeit bei der Entwicklung komplexer Programme.

Liste der Matrix- und Vektormethoden

Die Typen matrix und vector enthalten Methoden, die den Bibliotheksmethoden von NumPy entsprechen. Mit diesen Methoden können Sie Algorithmen und Codes mit minimalem Aufwand von Python nach MQL5 übersetzen. Viele Datenverarbeitungsaufgaben, mathematische Gleichungen, neuronale Netze und Aufgaben des maschinellen Lernens können mit vorgefertigten Python-Methoden und -Bibliotheken gelöst werden.

Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
void matrix.Eye(const int rows, const int cols, const int ndiag=0)	eye	Konstruktion einer quadratischen Matrix mit Einsen auf der Hauptdiagonalen und Null an den anderen Stellen.
void matrix.Identity(const int rows)	identity	Konstruktion einer quadratischen Matrix mit Einsen auf der Hauptdiagonalen.
void matrix.Ones(const int rows, const int cols)	ones	Konstruktion einer neuen Matrix mit gegebenen Zeilen und Spalten, die mit Einsen gefüllt sind.
void matrix.Zeros(const int rows, const int cols)	zeros	Konstruktion einer neuen Matrix mit gegebenen Zeilen und Spalten, die mit Nullen gefüllt sind.
void matrix.Full(const int rows, const int cols, const scalar value)	full	Konstruktion einer neuen Matrix mit gegebenen Zeilen und Spalten, gefüllt mit einem Einzelwert.
void matrix.Copy(const matrix a)	copy	Konstruktion einer Kopie der gegebenen Matrix.
void matrix.FromBuffer(const int rows, const int cols, const scalar array[], const int count=-1, const int offset=0)	frombuffer	Konstruktion einer Matrix, die aus einem 1-dimensionalen Array erstellt wurde.
void matrix.FromFile(const int rows, condt int cols, const int file_handle, const int count=-1, const int offset=0)	fromfile	Konstruktion einer Matrix aus Daten in einer Text- oder Binärdatei.
void vector.FromString(const string source, const string sep=" ")	fromstring	Konstruktion eines Vektors, der aus Textdaten in einer Zeichenkette initialisiert wird.

© 2000-2025, MetaQuotes Ltd.

Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
void vector.Arange(const scalar start, const scalar stop, const scalar step=1)	arange	Konstruktion gleichmäßig verteilter Werte innerhalb eines gegebenen Intervalls
void matrix.Diag(const vector v, const int ndiag=0)	diag	Eine Diagonale extrahieren oder eine Diagonalmatrix konstruieren.
void matrix.Tri(const int rows, const int cols, const int ndiag=0)	tri	Konstruktion einer Matrix mit Einsen auf und unterhalb der gegebenen Diagonalen und Nullen an anderen Stellen
void matrix.Tril(const int rows, const int cols, const scalar array[], const int ndiag=0)	tril	Rückgabe der Kopie einer Matrix, bei der die Elemente oberhalb der k-ten Diagonale auf Null gesetzt sind.
void matrix.Triu(const int rows, const int cols, const scalar array[], const int ndiag=0)	triu	Rückgabe der Kopie einer Matrix, bei der die Elemente unterhalb der k-ten Diagonale durch Nullen ersetzt sind.
void matrix.Vander(const vector v, const int cols=-1, const bool increasing=false)	vander	Erzeugen einer Vandermonde-Matrix
vector matrix.Row(const unsigned nrow)		Rückgabe eines Zeilenvektors
vector matrix.Col(const unsigned ncol)		Rückgabe eines Spaltenvektors
unsigned matrix.Rows()		Rückgabe der Anzahl der Zeilen einer Matrix.
unsigned matrix.Cols()		Rückgabe der Anzahl der Spalten einer Matrix.
void matrix.Init()		Initialisieren einer Matrix.
matrix matrix.Transpose()	Transponieren	Umkehrung oder Permutation der Achsen einer Matrix; gibt die geänderte Matrix zurück.
matrix matrix.Dot(const matrix b)	dot	Punktprodukt von zwei Matrizen.
matrix matrix.Inner(const matrix b)	inner	Inneres Produkt von zwei Matrizen.
matrix matrix.Outer(const matrix b)	outer	Äußeres Produkt von zwei Matrizen.
matrix matrix.MatMul(const matrix b)	matmul	Matrixprodukt von zwei Matrizen.
matrix matrix.MatrixPower(const int power)	matrix_power	Erhöhen einer quadratischen Matrix auf die (ganzzahlige) Potenz n.

© 2000-2025, MetaQuotes Ltd.

Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
matrix matrix.Kron(const matrix b)	kron	Rückgabe des Kronecker-Produkts von zwei Matrizen.
bool matrix.Cholesky(matrix& L)	cholesky	Rückgabe der Cholesky-Zerlegung.
bool matrix.QR(matrix& Q, matrix& R)	qr	Berechnen der qr-Faktorisierung einer Matrix.
bool matrix.SVD(matrix& U, matrix& V, vector& singular_values)	svd	Zerlegung in Singulärwerte.
bool matrix.Eig(matrix& eigen_vectors, vector& eigen_values)	eig	Berechnung der Eigenwerte und der rechten Eigenvektoren einer Quadratmatrix.
bool matrix.EigH(matrix& eigen_vectors, vector& eigen_values)	eigh	Rückgabe der Eigenwerte und Eigenvektoren einer hermiteschen Matrix.
bool matrix.EigVals(vector& eigen_values)	eigvals	Berechnung der Eigenwerte einer allgemeinen Matrix.
bool matrix.EigValsH(vector& eigen_values)	eigvalsh	Berechnung der Eigenwerte einer hermiteschen Matrix.
bool matrix.LU(matrix& L, matrix& U)		LU-Zerlegung einer Matrix als Produkt aus einer unteren Dreiecksmatrix und einer oberen Dreiecksmatrix.
bool matrix.LUP(matrix& L, matrix& U, matrix& P)		LUP-Zerlegung mit partieller Pivotisierung, d. h. LU-Zerlegung nur mit Zeilenpermutationen: PA=LU.
double matrix.Norm(const norm)	norm	Rückgabe der Matrix- oder Vektornorm.
double matrix.Cond(const norm)	cond	Berechnen der Zustandszahl einer Matrix.
vector matrix.Spectrum()		Berechnung des Spektrums einer Matrix als die Menge ihrer Eigenwerte aus dem Produkt AT*A.
double matrix.Det()	det	Berechnen der Determinante einer Matrix.
int matrix.Rank()	matrix_rank	Rückgabe des Rangs einer Matrix unter Verwendung der Gaußschen Methode.

© 2000-2025, MetaQuotes Ltd.

Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
int matrix.SLogDet(int& sign)	slogdet	Berechnung von Vorzeichen und Logarithmus der Determinante eines Arrays.
double matrix.Trace()	trace	Rückgabe der Summe entlang der Diagonalen der Matrix.
vector matrix.Solve(const vector b)	solve	Lösen einer linearen Matrixgleichung oder eines Systems linearer algebraischer Gleichungen.
vector matrix.LstSq(const vector b)	lstsq	Rückgabe der Lösung der kleinsten Quadrate von linearen algebraischen Gleichungen (für nicht quadratische oder entartete Matrizen).
matrix matrix.Inv()	inv	Berechnen der (multiplikativen) Inversen einer Matrix.
matrix matrix.PInv()	pinv	Berechnung der Pseudoinverse einer Matrix nach der Moore-Penrose-Methode.
int matrix.Compare(const matrix matrix_c, const double epsilon) int matrix.Compare(const matrix matrix_c, const int digits) int vector.Compare(const vector vector_c, const double epsilon) int vector.Compare(const vector vector_c, const int digits)		Vergleich der Elemente von zwei Matrizen/Vektoren mit der angegebenen Genauigkeit.
double matrix.Flat(const ulong index) bool matrix.Flat(const ulong index,const double value)	flat	Ermöglicht die Adressierung eines Matrixelements über einen Index anstelle von zwei.
double vector.ArgMax() double matrix.ArgMax() vector matrix.ArgMax(const int axis)	argmax	Rückgabe des Index des größten Wertes.
double vector.ArgMin() double matrix.ArgMin() vector matrix.ArgMin(const int axis)	argmin	Rückgabe des Index des kleinsten Wertes.
double vector.Max() double matrix.Max() vector matrix.Max(const int axis)	max	Rückgabe des größten Wertes einer Matrix/Vektors.
double vector.Mean() double matrix.Mean()	mean	Berechnung des arithmetischen Mittels der Elementwerte.

© 2000-2025, MetaQuotes Ltd.

Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
vector matrix.Mean(const int axis)		
double vector.Min() double matrix.Min() vector matrix.Min(const int axis)	min	Rückgabe des kleinsten Wertes einer Matrix/Vektors.
double vector.Sum() double matrix.Sum() vector matrix.Sum(const int axis)	sum	Rückgabe der Summe der Matrix-/Vektorelemente, die auch für die angegebene Achse (Achsen) durchgeführt werden kann.
void vector.Clip(const double min_value,const double max_value) void matrix.Clip(const double min_value,const double max_value)	clip	Limit der Elemente einer Matrix/eines Vektors in einen angegebenen Bereich gültiger Werte.
vector vector.CumProd() vector matrix.CumProd() matrix matrix.CumProd(const int axis)	cumprod	Rückgabe des kumulativen Produkts von Matrix-/Vektorelementen, einschließlich der Elemente entlang der angegebenen Achse.
vector vector.CumSum() vector matrix.CumSum() matrix matrix.CumSum(const int axis)	cumsum	Rückgabe der kumulativen Summe der Matrix-/Vektorelemente, einschließlich derjenigen entlang der angegebenen Achse.
double vector.Prod(const double initial=1) double matrix.Prod(const double initial=1) vector matrix.Prod(const int axis,const double initial=1)	prod	Rückgabe des Produkts der Matrix-/Vektorelemente, das auch für die angegebene Achse durchgeführt werden kann.
void matrix.Reshape(const ulong rows, const ulong cols)	reshape	Ändern der Form einer Matrix, ohne ihre Daten zu ändern.
void matrix.Resize(const ulong rows,const ulong cols)	resize	Rückgabe einer neuen Matrix mit geänderter Form und Größe.
bool matrix.SwapRows(const ulong row1, const ulong row2)		Vertauschen der Zeilen in einer Matrix.
bool matrix.SwapCols(const ulong col1, const ulong col2)		Vertauschen der Spalten in einer Matrix.
double vector.Ptp() double matrix.Ptp() vector matrix.Ptp(const int axis)	ptp	Rückgabe des Wertebereichs einer Matrix/eines Vektors oder der angegebenen Matrixachse, entspricht Max() - Min().
double vector.Percentile(const int percent)	percentile	Rückgabe des angegebenen Perzentils der Werte von

© 2000-2025, MetaQuotes Ltd.

Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
double matrix.Percentile(const int percent) vector matrix.Percentile(const int percent,const int axis)		Matrix-/Vektorelementen oder Elementen entlang der angegebenen Achse. Gültige Werte für den Parameter 'percent' liegen im Bereich [0, 100].
double vector.Quantile(const int percent) double matrix.Quantile(const int percent) vector matrix.Quantile(const int percent,const int axis)	quantil	Rückgabe des angegebenen Quantils der Werte von Matrix-/Vektorelementen oder Elementen entlang der angegebenen Achse zurück. Der Parameter 'percent' nimmt Werte im Bereich [0, 1] an.
double vector.Median() double matrix.Median() vector matrix.Median(const int axis)	median	Berechnung des Medians der Matrix-/Vektor-Elemente. Der Median ist der mittlere Wert, der die höchste Hälfte der Matrix-/Vektorelemente von der niedrigsten Hälfte der Elemente trennt.
double vector.Average() double matrix.Average() vector matrix.Average(const int axis)	average	Berechnung des arithmetische Mittels von Matrix-/Vektorwerten. Die Summe der Gewichte im Nenner darf nicht gleich 0 sein, aber einige Gewichte können 0 sein.
double vector.Std() double matrix.Std() vector matrix.Std(const int axis)	std	Rückgabe der Standardabweichung der Werte von Matrix-/Vektorelementen oder von Elementen entlang der angegebenen Achse.
double vector.Var() double matrix.Var() vector matrix.Var(const int axis)	var	Berechnung der Varianz der Werte von Matrix-/Vektorelementen.
double vector.CorrCoef(const vector& v) matrix matrix.CorrCoef()	corrcoef	Berechnung des Pearson-Korrelationskoeffizientens (linearer Korrelationskoeffizient). Der Korrelationskoeffizient liegt im Bereich [-1, 1].
vector vector.Correlate(const vector& v,enum mode)	correlate	Berechnung der Kreuzkorrelation von zwei Vektoren. Der Parameter 'mode' bestimmt den Berechnungsmodus der linearen Faltung.
vector vector.Convolve(const vector& v, enum mode)	convolve	Rückgabe der diskreten, linearen Faltung zweier Vektoren. Der

© 2000-2025, MetaQuotes Ltd.

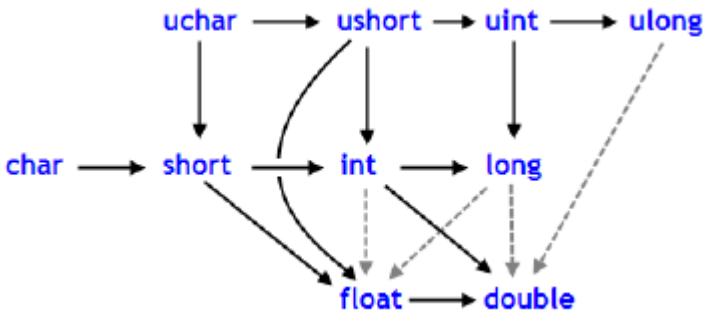
Methode von Matrix/Vektor	Analoge Methode in NumPy	Beschreibung
		Parameter 'mode' bestimmt den Berechnungsmodus der linearen Faltung.
matrix matrix.Cov() matrix vector.Cov(const vector& v); (resulting matrix 2 x 2)	cov	Berechnung der Kovarianzmatrix. Die Kovarianz von zwei Stichproben (zwei Zufallsvariablen) ist ein Maß für ihre lineare Abhängigkeit.

© 2000-2025, MetaQuotes Ltd.

Typenreduzierung

Reduzierung der Zahlentypen

Oft muss man einen Zahlentyp in den anderen umwandeln. Nicht jeden numerischen Typ kann man in einen anderen umwandeln, zulaessige Reduzierungen in MQL5 werden auf dem Schema gezeigt:



Volllinien mit Richtungszeichen bezeichnen Umwandlungen, die ohne Informationsverlust durchgeführt werden. Anstatt des Typs char kann der Typ bool auftreten(die beiden nehmen 1 Byte des Speicherplatzes ein), anstatt des Typs int kann der Typ color verwendet werden (jeder 4 Byte), anstatt des Typs long ist der Typ datetime zulaessig (jeder 8 Byte). Vier graue Strichlinien auch mit Richtungszeichen bezeichnen Umwandlungen, bei denen der Genauigkeitsverlust entstehen kann. ZB. Zifferzahl in der Ganzzahl 123456789 (int) ist mehr als es vom Typ float vertreten werden kann.

```
   int n=123456789;
   float f=n;     // Inhalt f ist 1.234567892E8
   Print("n = ",n,"   f = ",f);
   // Ergebnis n= 123456789    f= 123456792.00000
```

Die Zahl, die in Typ float umgewandelt ist, hat dieselbe Ordnung, aber kleinere Genauigkeit. Die den schwarzen Richtungszeichen entgegengesetzte Umwandlungen können mit Informationsverlust durchgeführt werden. Umwandlungen unter char und uchar, short und ushort, int und uint, long und ulong (gemeint werden gegenseitige Umwandlungen), können zum Informationsverlust führen.

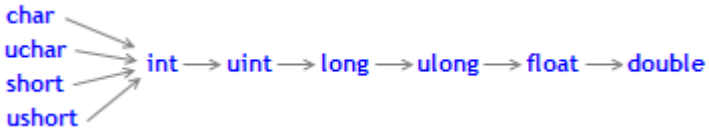
Im Ergebnis der Reduzierung des Fliesspunktwertes zum Bruchwert wird der Bruchteil immer weggelassen. Wen man Fliesspunktzahl zur naechstliegenden Ganzzahl runden muss (in den meisten Faellen ist es nuetzlicher), muss man die Funktion MathRound() verwenden.

Beispiel:

```
//--- Fallbeschleunigung
   double g=9.8;
   double round_g=(int)g;
   double math_round_g=MathRound(g);
   Print("round_g = ",round_g);
   Print("math_round_g = ",math_round_g);
//   Ergebnis:
//   round_g = 9
//   math_round_g = 10
```

© 2000-2025, MetaQuotes Ltd.

Wenn zwei Werte durch Binaeroperator verbunden werden, wird der Operand des jungeren Typ in den Operand des aelteren Typs umgewandelt laut Prioritaet dezeigt auf dem Schema:



Datentypen char, uchar, short und ushort werden zum Typ int reduziert.

Beispiele:

```
   char   c1=3;
//--- erstes Beispiel
   double d2=c1/2+0.3;
   Print("c1/2+0.3 =",d2);
// Ergebnis:   c1/2+0.3 = 1.3

//--- zweites Beispiel
   d2=c1/2.0+0.3;
   Print("c1/2.0+0.3 =",d2);
// Ergebnis:   c1/2.0+0.3 = 1.8
```

Berechnetre Ausdruck besteht aus zwei Operationen. Im ersten Beispiel wird die Variable vom ersten Typ char in die zeitweilige Variable des Typs int umgewandelt, denn der zweite Operand in Division, Konstante 2, hat einen aelteren Typ int. Im Ergebnis der ganzzahliger Division 3/2 wird der Wert 1, den der Typ int hat.

In zweiter Operation des ersten Beispiels tritt als zweiter Operand Konstante 0.3 auf, die zum Typ double gehört, darum wird das Ergebnis nach der ersten Operation in die zeitweilige Variable des Typs double mit dem Wert 1.0 umgewandelt.

Im zweiten Beispiel wird die Variable vom ersten Typ char in die zitweilige Variable des Typs double umgewandelt, denn der zweite Operand in Division, Konstante 2.0 gehört zum Typ double; weitere Umwandlungen werden nicht durchgeführt.

Reduzierung der Zahlentypen

In Ausdrücken der Sprache MQL5 kann explizite und implizite Typenreduzierung verwendet werden. Explizite Typenreduzierung wird auf der Weise geschrieben:

```
var_1 = (Typ)var_2;
```

Als Variable var_2 kann Ausdruck oder Ergebnis der Funktion sein. Zulaessig ist auch Funktionsaufschreibung der expliziten Typenreduzierung:

```
var_1 = Typ(var_2);
```

Betrachten wir explizite Reduzierung am ersten Beispiel:

```
//--- drittes Beispiel
   double d2=(double)c1/2+0.3;
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("(double)c1/2+0.3 =",d2);
// Ergebnis:   (double)c1/2+0.3 = 1.80000000
```

Vor Division wird die Variable c1 explizit in den Typ double umgewandelt. Die derzeit ganzzahlige Konstante 2 wird zum Wert 2.0 des Typs double reduziert, denn während der Umwandlungen bekam der erste Operand den Typ double. Tatsaechlich ist die explizite Typenumwandlung einstellige Operation.

Außerdem kann das Ergebnis bei der Typenreduzierung ausserhalb der zulaessigen Stellenzahl sein. In diesem Fall entsteht Abbruch. ZB.:

```
   char c;
   uchar u;
   c=400;
   u=400;
   Print("c=",c); // Ergebnis c=-112
   Print("u=",u); // Ergebnis u=144
```

Vor den Operationen (ausser Zuordungsoperationen) wird die Umwandlung in den Typ durchgeführt, der die groesste Prioriraet hat, und vor den Zuordnungsoperationen - in den Zieltyp.

Beispiele:

```
   int    i=1/2;         // keine Typenreduzierung, Ergebnis: 0
   Print("i=1/2  ",i);

   int k=1/2.0;          // Ausdruck wird zum Typ double reduziert,
   Print("k=1/2  ",k);   // dann zum Zieltyp int reduziert, Ergebnis: 0

   double d=1.0/2.0;     // keine Typenreduzierung, Ergebnis: 0.5
   Print("d=1/2.0; ",d);

   double e=1/2.0;       // Ausdruck wird zum Typ double reduziert,
   Print("e=1/2.0; ",e);// der mit dem Zieltyp zusammenfaellt, Ergebnis: 0.5

   double x=1/2;         // Ausdruck des Typs int wird zum Zieltyp double reduziert,
   Print("x=1/2; ",x);   // Ergebnis: 0.0
```

Wenn Sie Typ long/ulong in double konvertieren, kann Genauigkeit verloren gehen: wenn das Ganze ist größer als 9223372036854774784 oder kleiner als -9223372036854774784.

```
void OnStart()
  {
   long l_max=LONG_MAX;
   long l_min=LONG_MIN+1;
//--- Finden wir die maximale ganze Zahl, die Genauigkeit beim Gießen zu double nicht
   while(l_max!=long((double)l_max))
      l_max--;
//--- Finden wir die kleinste ganze Zahl, die Genauigkeit beim Gießen zu double nicht
   while(l_min!=long((double)l_min))
      l_min++;
//--- Jetzt zeigen wir das gefundenen Intervall für den ganzen Zahlen
   PrintFormat("Beim Gießen einer ganzen Zahl zu double sollte es "
```

© 2000-2025, MetaQuotes Ltd.

```
                "im folgenden Intervall sein [%I64d, %I64d]",l_min,l_max);
//--- Nun wollen wir sehen, was passiert, wenn die Zahl außerhalb dieses Intervalls is
   PrintFormat("l_max+1=%I64d, double(l_max+1)=%.f, ulong(double(l_max+1))=%I64d",
                l_max+1,double(l_max+1),long(double(l_max+1)));
   PrintFormat("l_min-1=%I64d, double(l_min-1)=%.f, ulong(double(l_min-1))=%I64d",
                l_min-1,double(l_min-1),long(double(l_min-1)));
//--- erhalten eine solche Schlussfolgerung
// Beim Gießen einer ganzen Zahl zu double sollte es im Intervall [-92233720368547747
// l_max+1=9223372036854774785, double(l_max+1)=9223372036854774800, ulong(double(l_ma
// l_min-1=-9223372036854774785, double(l_min-1)=-9223372036854774800, ulong(double(l_
  }
```

Reduzierung für Typ string

Typ string hat die hoechste Prioritaet unter einfachen Typen. Darum wenn einer der Operanden den Typ string hat, wird der andere operand zum Typ string authomatisch reduziert. Man muss in Vormerkung behalten, dass es für den Typ string nur einzige zweistellige Operation zulaessig ist. Explizite Reduzierung der Variable des Typs string zum jeden Zahlentyp ist zulaessig.

Beispiele:

```
   string s1=1.0/8;            // Ausdruck wird zum Typ double reduziert,
   Print("s1=1.0/8; ",s1);     // dann zum Zieltyp string,
// Ergebnis:"0.12500000"(Zeile mit 10 Zeichen)

   string s2=NULL;             // Deinitialisierung der Zeile
   Print("s2=NULL; ",s2);      // Ergebnis: Leerzeile
   string s3="Ticket N"+12345; // Ausdruck wird zum Typ string reduziert
   Print("s=\"Ticket N"+12345,s3);

   string str1="true";
   string str2="0,255,0";
   string str3="2009.06.01";
   string str4="1.2345e2";
   Print(bool(str1));
   Print(color(str2));
   Print(datetime(str3));
   Print(double(str4));
```

Reduzierung der Typen von Anzeigern der Basisklassen zu Anzeigern der sekundaeren Klassen

Objekte der sekundaeren Klasse können auch als Objekte der ihr entsprechenden Basisklasse betrachtet werden. Das führt zu einigen interessanten Folgen. ZB. Gegen die Tatsache, dass Objekte verschiedener Klassen, die von einer Basisklasse entstanden, können sich voneinander unterscheiden, können wir Ihre verbundene Liste erzeugen (List), denn wir betrachten sie als Objekte der Basisklasse.

© 2000-2025, MetaQuotes Ltd.

Aber das Gegenteil stimmt nicht: Objekte der Basisklasse sind nicht automatisch Objekte der sekundaeren Klasse.

Man kann explizite Typenreduzierung verwenden, um die Anzeiger der Basisklasse in die Anzeiger der sekundaeren Klasse umwandeln. Man muss aber ganz sicher sein, dass diese Umwandlung zulaessig ist, sonst entsteht ein kritischer Fehler der Ausführungszeit und das mql5-Programm wird gestoppt.

Dynamische Typumwandlung mithilfe von dynamic_cast

Es besteht die Möglichkeit einer dynamischen Typumwandlung mithilfe der Anweisung dynamic_cast, die nur an Pointer von Klassen angewendet werden kann. Dabei wird die Korrektheit von Typen während der Ausführung des Programms überprüft. Dies bedeutet, der Compiler überprüft nicht den Datentyp für die Umwandlung, wenn dynamic_cast verwendet wird. Wenn der Pointer in den Typ umgewandelt wird, der kein tatsächlicher Objekttyp ist, hat das Ergebnis den Wert NULL.

```
dynamic_cast <type-id> ( expression )
```

Der Parameter *type-id* in spitzen Klammern muss Pointer auf einen früher definierten Typ der Klasse sein. Der *expression* Operand kann von jedem Typ sein (im Vergleich zu C++) außer void.

Beispiel:

```
class CBar { };
class CFoo : public CBar { };

void OnStart()
  {
   CBar bar;
//--- dynamische Typumwandlung des Pointers *bar zum *foo erlaubt
   CFoo *foo = dynamic_cast<CFoo *>(&bar); // kein kritischer Fehler der Ausführung
   Print(foo);                             // foo=NULL
//--- Versuch verboten, die Referenz des Objekttyps Bar in den Objekttyp Foo umzuwande
   foo=(CFoo *)&bar;                       // kritischer Fehler der Ausführung
   Print(foo);                             // diese Zeile wird nicht ausgeführt
  }
```

Sehen Sie auch

Datentypen

© 2000-2025, MetaQuotes Ltd.

Typ void und Konstante NULL

Syntaktisch ist der Typ void ein Basistyp wie die Typen char, uchar, bool, short, ushort, int, uint, color, long, ulong, datetime, float, double und string. Dieser Typ wird als Andeutung verwendet, dass die Funktion keinen Wert rückgibt, oder als Funktionsparameter bezeichnet das Fehlen von Parametern.

Vorbestimmte Konstantvariable **NULL** hat den Typ *void*. Sie kann den Variablen jeder anderen Basistypen ohne Umwandlung zugeordnet werden. Zulaessig ist auch der Vergleich von Variablen der Basistypen mit dem Wert **NULL**

Beispiel:

```
//--- wenn die Zeile nicht initialisiert wird, ordnen wir ihr unsere vorbestimmte Grö
if(some_string==NULL) some_string="empty";
```

NULL kann auch mit Zeigern auf Objekte verglichen werden, erzeugt mit der Anweisung new.

Sehen Sie auch

Variablen, Funktionen

© 2000-2025, MetaQuotes Ltd.

Benutzerdefinierte Typen

Das typedef Schlüsselwort in C++ ermöglicht die Erstellung benutzerdefinierter Typen. Dafür reicht es, einem bereits existierenden Datentyp einen neuen Namen zu geben. Dabei wird kein neuer Datentyp erstellt, es wird nur ein neuer Name einem bereits existierenden Typ gegeben. Dank der Anwendung benutzerdefinierter Typen kann man Programme flexibler machen: dafür reicht es manchmal, typedef-Anweisungen mithilfe von Makros (#define) zu modifizieren. Darüber hinaus erlaubt die Anwendung benutzerdefinierter Typen, die Lesbarkeit des Codes zu verbessern, weil man mithilfe von typedef eigene beschreibende Namen für Standarddatentypen verwenden kann. Das allgemeine Format einer Anweisung für die Erstellung eines benutzerdefinierten Typs:

```
   typedef Typ neuer_Name;
```

Das Element *Typ* bezeichnet hier jeden zulässigen Datentyp, und das Element *neuer_Name* - den neuen Namen für diesen Typ. Es ist wichtig zu betonen, dass der neue Name nur ein Zusatz zum bereits existierenden Namen eines Typs und kein Ersatz ist. In der MQL5 Programmiersprache können mithilfe von typedef Funktionspointer erstellt werden.

Funktionspointer

Ein Funktionspointer hat das folgende Format

```
   typedef Typ_des_Ergebnisses_der_Funktion (*Name_des_Funktionstyps)(Liste_der_Typen_
```

wo nach dem Wort typedef die Signatur der Funktion gesetzt wird: die Anzahl und der Typ der Eingabeparameter sowie der Typ des Ergebnisses der Funktion. Führen wir ein einfaches Beispiel für die Erstellung und Verwendung eines Funktionspointers:

```
//--- deklarieren wir einen Pointer auf die Funktion, die zwei Parameter vom Typ int a
   typedef int (*TFunc)(int,int);
//--- TFunc ist ein Typ, und wir können die Pointer-Variable deklarieren
   TFunc func_ptr; // Funktionspointer
//--- Deklarieren wir die Funktionen, die der TFunc Beschreibung entsprechen
   int sub(int x,int y) { return(x-y); }  // Subtraktion einer Zahl von der anderen
   int add(int x,int y) { return(x+y); }  // Addition von zwei Zahlen
   int neg(int x)       { return(~x);  }  // Invertieren von Bits in der Variablen
//--- in der func_ptr Variablen kann man die Funktionsadresse speichern, um diese spät
   func_ptr=sub;
   Print(func_ptr(10,5));
   func_ptr=add;
   Print(func_ptr(10,5));
   func_ptr=neg;           // Fehler: neg hat nicht den Typ  int (int,int)
   Print(func_ptr(10));    // Fehler: es sind zwei Parameter erforderlich
```

In diesem Beispiel kann man der *func_ptr* Variablen die Funktionen *sub* und *add* zuweisen, denn sie habe je zwei Inputparameter vom Typ int, wie dies in der Definition des *TFunc* Funktionspointers angegeben ist. Die Funktion *neg* kann nicht dem Bezeichner func_ptr zugewiesen werden, denn ihre Signatur ist anders.

Gestaltung von Event-Modellen in der Benutzerschnittstelle

© 2000-2025, MetaQuotes Ltd.

Mithilfe von Funktionspointern ist es einfach, die Verarbeitung von Ereignissen bei der Erstellung einer Benutzerschnittstell zu gestalten. Anhand eines Beispiels aus dem CButton Bereich veranschaulichen wir, wie man Buttons erstellen und ihnen Funktionen für die Verarbeitung von Klicks hinzufügen kann. Zuerst definieren wir den Pointer auf die Funktion *TAction*, die beim Klicken auf den Button aufgerufen wird, und erstellen wir drei Funktionen entsprechend der Beschreibung von *TAction*.

```
//--- erstellen wir einen benutzerdefinierten Funktionstyp
typedef int(*TAction)(string,int);
//+------------------------------------------------------------------+
//|  Öffnet die Datei                                                |
//+------------------------------------------------------------------+
int Open(string name,int id)
  {
   PrintFormat("Funktion %s (name=%s id=%d) aufgerufen",__FUNCTION__,name,id);
   return(1);
  }
//+------------------------------------------------------------------+
//|  Speichert die Datei                                             |
//+------------------------------------------------------------------+
int Save(string name,int id)
  {
   PrintFormat("Funktion %s (name=%s id=%d) aufgerufen",__FUNCTION__,name,id);
   return(2);
  }
//+------------------------------------------------------------------+
//|  Schließt die Datei                                              |
//+------------------------------------------------------------------+
int Close(string name,int id)
  {
   PrintFormat("Funktion %s (name=%s id=%d) aufgerufen",__FUNCTION__,name,id);
   return(3);
  }
```

Danach leiten wir die MyButton Klasse von der CButton Klasse ab, welcher wir *TAction*, den Funktionspointer, hinzufügen.

```
//+------------------------------------------------------------------+
//| Erstellen wir eine eigene Button-Klasse mit der Funktion         |
//+------------------------------------------------------------------+
class MyButton: public CButton
  {
private:
   TAction           m_action;                    // Handler von Chartereignissen
public:
                     MyButton(void){}
                    ~MyButton(void){}
   //--- Konstruktor mit der Angabe des Button-Textes und des Funktionspointers für di
                     MyButton(string text, TAction act)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Text(text);
      m_action=act;
     }
   //--- setzen wir die eigene Funktion, die aus dem OnEvent() Event Handler aufgerufe
   void              SetAction(TAction act){m_action=act;}
   //--- Standard Chart Events Handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons
     {
      if(m_action!=NULL && lparam==Id())
        {
         //--- rufen wir den benutzerdefinierten m_action() Handler auf
         m_action(sparam,(int)lparam);
         return(true);
        }
      else
      //--- geben wir das Ergebnis des Aufrufs des Handlers aus der CButton Basisklass
         return(CButton::OnEvent(id,lparam,dparam,sparam));
     }
  };
```

Erstellen wir die von der CAppDialog Klasse abgeleitete CControlsDialog Klasse, welcher wir das *m_buttons* Array für das Speichern von Buttons vom Typ *MyButton* sowie die Methoden *AddButton(MyButton &button)* und *CreateButtons() hinzufügen*.

```
//+------------------------------------------------------------------+
//| CControlsDialog Klasse                                           |
//| Verwendungszweck: grafisches Panel für die Steuerung             |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CArrayObj         m_buttons;                    // Array für Tasten
public:
                     CControlsDialog(void){};
                    ~CControlsDialog(void){};
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- Button hinzufügen
   bool              AddButton(MyButton &button){return(m_buttons.Add(GetPointer(butto
protected:
   //--- Buttons erstellen
   bool              CreateButtons(void);
  };
//+------------------------------------------------------------------+
//| Erstellung des ControlsDialog Objekts auf dem Chart              |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
   return(CreateButtons());
//---
  }
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Erstellung und Hinzufügung von Buttons auf das CControlsDialog Panel     |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateButtons(void)
  {
//--- Koordinaten der Buttons berechnen
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2;
   int y2=y1+BUTTON_HEIGHT;
//--- Button-Objekte gemeinsam mit Funktionspointern hinzufügen
   AddButton(new MyButton("Open",Open));
   AddButton(new MyButton("Save",::Save));
   AddButton(new MyButton("Close",Close));
//--- erstellen wir Buttons grafisch
   for(int i=0;i<m_buttons.Total();i++)
     {
      MyButton *b=(MyButton*)m_buttons.At(i);
      x1=INDENT_LEFT+i*(BUTTON_WIDTH+CONTROLS_GAP_X);
      x2=x1+BUTTON_WIDTH;
      if(!b.Create(m_chart_id,m_name+"bt"+b.Text(),m_subwin,x1,y1,x2,y2))
        {
         PrintFormat("Failed to create button %s %d",b.Text(),i);
         return(false);
        }
      //--- fügen wir jeden Button in den Container CControlsDialog hinzu
      if(!Add(b))
         return(false);
     }
//--- succeed
   return(true);
  }
```

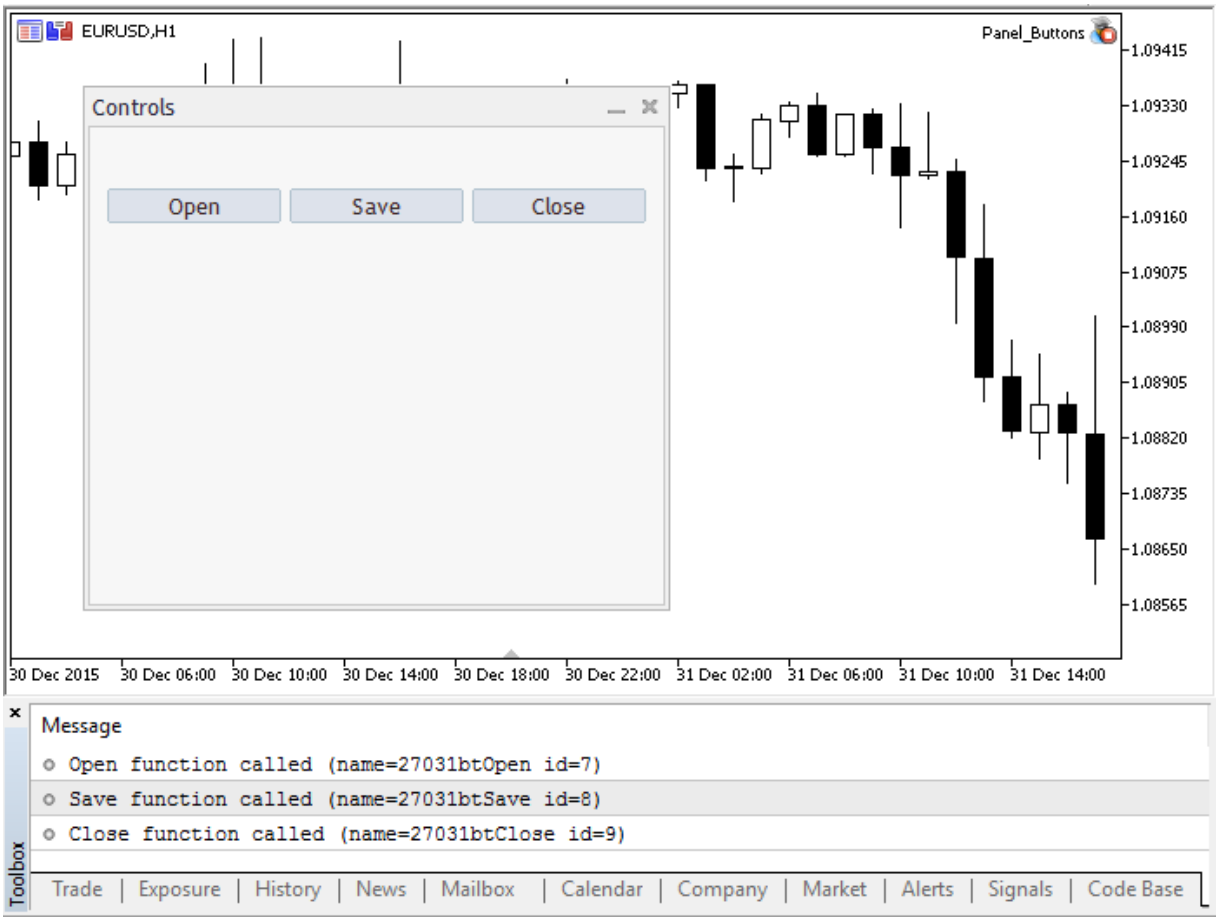
© 2000-2025, MetaQuotes Ltd.

Nun können wir ein Programm unter Verwendung des CControlsDialog Panels schreiben, in welchem 3 Buttons "Open", "Save" und "Close" erstellt werden. Beim Klicken auf einen Button wird die entsprechende Funktion aufgerufen, welche als *TAction* Funktionspointer vorgegeben ist.

```
//--- deklarieren wir dieses Objekt an der globalen Ebene, um es automatisch beim Sta
CControlsDialog MyDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- nun erstellen wir das Objekt auf dem Chart
   if(!MyDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- Applikation starten
   MyDialog.Run();
//--- erfolgreiche Initialisierung der Applikation
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- destroy dialog
   MyDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
//--- für Chartereignisse rufen wir den Event Handler aus der Basislklasse auf (in die
   MyDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

Das Design der gestarteten Applikation und die Ergebnisse der Klicks auf die Buttons sind auf dem Bild unten zu sehen.

© 2000-2025, MetaQuotes Ltd.



Vollständiger Quellcode des Programms

```
//+------------------------------------------------------------------+
//|                                                Panel_Buttons.mq5 |
//|                        Copyright 2017, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+

#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Panel mit mehreren CButton Buttons"
#include <Controls\Dialog.mqh>
#include <Controls\Button.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
```

© 2000-2025, MetaQuotes Ltd.

```
//--- for buttons
#define BUTTON_WIDTH                         (100)      // size by X coordinate
#define BUTTON_HEIGHT                        (20)       // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                          (20)       // size by Y coordinate

//--- erstellen wir einen benutzerdefinierten Funktionstyp
typedef int(*TAction)(string,int);
//+------------------------------------------------------------------+
//|  Öffnet die Datei                                                |
//+------------------------------------------------------------------+
int Open(string name,int id)
  {
   PrintFormat("Funktion %s (name=%s id=%d) aufgerufen",__FUNCTION__,name,id);
   return(1);
  }
//+------------------------------------------------------------------+
//|  Speichert die Datei                                             |
//+------------------------------------------------------------------+
int Save(string name,int id)
  {
   PrintFormat("Funktion %s (name=%s id=%d) aufgerufen",__FUNCTION__,name,id);
   return(2);
  }
//+------------------------------------------------------------------+
//|  Schließt die Datei                                              |
//+------------------------------------------------------------------+
int Close(string name,int id)
  {
   PrintFormat("Funktion %s (name=%s id=%d) aufgerufen",__FUNCTION__,name,id);
   return(3);
  }
//+------------------------------------------------------------------+
//| Erstellen wir eine eigene Button-Klasse mit der Funktion der Verarbeitung von Ever
//+------------------------------------------------------------------+
class MyButton: public CButton
  {
private:
   TAction           m_action;                    // Handler von Chartereignissen
public:
                     MyButton(void){}
                    ~MyButton(void){}
   //--- Konstruktor mit der Angabe des Button-Textes und des Funktionspointers für di
                     MyButton(string text,TAction act)
     {
      Text(text);
      m_action=act;
     }
   //--- setzen wir die eigene Funktion, die aus dem OnEvent() Event Handler aufgerufe
```

© 2000-2025, MetaQuotes Ltd.

```
   void              SetAction(TAction act){m_action=act;}
   //--- Standard Chart Events Handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons
     {
      if(m_action!=NULL && lparam==Id())
        {
         //--- rufen wir den eigenen Handler auf
         m_action(sparam,(int)lparam);
         return(true);
        }
      else
      //--- geben wir das Ergebnis des Aufrufs des Handlers aus der CButton Basisklass
         return(CButton::OnEvent(id,lparam,dparam,sparam));
     }
  };
//+------------------------------------------------------------------+
//| CControlsDialog Klasse                                           |
//| Verwendungszweck: grafisches Panel für die Steuerung             |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CArrayObj         m_buttons;                    // Array für Tasten
public:
                     CControlsDialog(void){};
                    ~CControlsDialog(void){};
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- Button hinzufügen
   bool              AddButton(MyButton &button){return(m_buttons.Add(GetPointer(butto
protected:
   //--- Buttons erstellen
   bool              CreateButtons(void);
  };
//+------------------------------------------------------------------+
//| Erstellung des ControlsDialog Objekts auf dem Chart              |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
   return(CreateButtons());
//---
  }
//+------------------------------------------------------------------+
//| Erstellung und Hinzufügung von Buttons auf das CControlsDialog   |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateButtons(void)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Koordinaten der Buttons berechnen
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2;
   int y2=y1+BUTTON_HEIGHT;
//--- Button-Objekte gemeinsam mit Funktionspointern hinzufügen
   AddButton(new MyButton("Open",Open));
   AddButton(new MyButton("Save",::Save));
   AddButton(new MyButton("Close",Close));
//--- erstellen wir Buttons grafisch
   for(int i=0;i<m_buttons.Total();i++)
     {
      MyButton *b=(MyButton*)m_buttons.At(i);
      x1=INDENT_LEFT+i*(BUTTON_WIDTH+CONTROLS_GAP_X);
      x2=x1+BUTTON_WIDTH;
      if(!b.Create(m_chart_id,m_name+"bt"+b.Text(),m_subwin,x1,y1,x2,y2))
        {
         PrintFormat("Failed to create button %s %d",b.Text(),i);
         return(false);
        }
      //--- fügen wir jeden Button in den Container CControlsDialog hinzu
      if(!Add(b))
         return(false);
     }
//--- succeed
   return(true);
  }
//--- deklarieren wir dieses Objekt an der globalen Ebene, um es automatisch beim Sta
CControlsDialog MyDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- nun erstellen wir das Objekt auf dem Chart
   if(!MyDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- Applikation starten
   MyDialog.Run();
//--- erfolgreiche Initialisierung der Applikation
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- destroy dialog
   MyDialog.Destroy(reason);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,          // event ID
                  const long& lparam,    // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
//--- für Chartereignisse rufen wir den Event Handler aus der Basislklasse auf (in die
   MyDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

Siehe auch

Variablen, Funktionen

© 2000-2025, MetaQuotes Ltd.

Objektanzeiger

MQL5 ermöglicht die dynamische Erzeugung von Objekten komplexen Typs. Dies geschieht mit dem Operator 'new', der einen Deskriptor des erzeugten Objekts zurückgibt. Die Größe des Deskriptors beträgt 8 Bytes. Syntaktisch sind Objektdeskriptoren in MQL5 ähnlich wie C++ Pointer.

Beispiel:

```
MyObject* hobject= new MyObject();
```

Im Gegensatz zu C++ ist die Variable "hobject" aus dem obigen Beispiel kein Zeiger auf den Speicher, sondern ein Objektdeskriptor. Außerdem müssen in MQL5 alle Objekte in Funktionsparametern per Referenz übergeben werden. Die folgenden Beispiele zeigen die Übergabe von Objekten als Funktionsparameter:

```
class Foo
  {
public:
   string            m_name;
   int               m_id;
   static int        s_counter;
   //--- Konstruktor und Destruktor
                     Foo(void){Setup("noname");};
                     Foo(string name){Setup(name);};
                    ~Foo(void){};
   //--- Initialisieren des Objektes Foo
   void              Setup(string name)
     {
      m_name=name;
      s_counter++;
      m_id=s_counter;
     }
  };
int Foo::s_counter=0;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklarieren des Objekts als Variable, mit automatischer Erstellung
   Foo foo1;
//--- Variante der Übergabe eines Objekts per Referenz
   PrintObject(foo1);

//--- einen Zeiger auf ein Objekt deklarieren und es mit dem Operator "new" erstellen
   Foo *foo2=new Foo("foo2");
//--- Variante der Übergabe eines Zeigers auf ein Objekt per Referenz
   PrintObject(foo2); // der Zeiger auf das Objekt wird vom Compiler automatisch umgew

//--- ein Array von Foo-Objekten deklarieren
```

© 2000-2025, MetaQuotes Ltd.

```
   Foo foo_objects[5];
//--- Variante der Übergabe eines Arrays von Objekten
   PrintObjectsArray(foo_objects); // eine separate Funktion zur Übergabe eines Arrays

//--- ein Array von Zeigern auf Objekte des Typs Foo deklarieren
   Foo *foo_pointers[5];
   for(int i=0;i<5;i++)
      foo_pointers[i]=new Foo("foo_pointer");
//--- ein Array von Zeigern auf Objekte des Typs Foo deklarieren
   PrintPointersArray(foo_pointers); // eine separate Funktion zur Übergabe eines Arra

//--- Löschen vor Beendigung, um sicher die als Zeiger angelegten Objekte zu löschen
   delete(foo2);
//--- das Array der Zeiger entfernen
   int size=ArraySize(foo_pointers);
   for(int i=0;i<5;i++)
      delete(foo_pointers[i]);
//---
  }
//+------------------------------------------------------------------+
//|  Objekte werden immer als Referenz übergeben                     |
//+------------------------------------------------------------------+
void PrintObject(Foo &object)
  {
   Print(__FUNCTION__,": ",object.m_id," Object name=",object.m_name);
  }
//+------------------------------------------------------------------+
//| Ein Array von Objekten übergeben                                 |
//+------------------------------------------------------------------+
void PrintObjectsArray(Foo &objects[])
  {
   int size=ArraySize(objects);
   for(int i=0;i<size;i++)
      PrintObject(objects[i]);
  }
//+------------------------------------------------------------------+
//| Ein Array von Objektpointer übergeben                            |
//+------------------------------------------------------------------+
void PrintPointersArray(Foo* &objects[])
  {
   int size=ArraySize(objects);
   for(int i=0;i<size;i++)
      PrintObject(objects[i]);
  }
//+------------------------------------------------------------------+
```

Prüfen des Zeigers vor der Verwendung

© 2000-2025, MetaQuotes Ltd.

Der Versuch, auf einen ungültigen Zeiger zuzugreifen, führt zum kritischen Herunterfahren des Programms. Die Funktion CheckPointer wird verwendet, um einen Zeiger vor der Verwendung zu prüfen. Der Zeiger kann in den folgenden Fällen ungültig sein:

- der Zeiger ist gleich NULL;
- das Objekt wurde mit dem Operator delete zerstört.

Diese Funktion kann verwendet werden, um einen Zeiger zu überprüfen. Ein Wert ungleich Null zeigt an, dass auf Daten an diesem Zeiger zugegriffen werden kann.

```
class CMyObject
 {
protected:
  double              m_value;
public:
                      CMyObject(void);
                      CMyObject(double value) {m_value=value;};
                     ~CMyObject(void){};
  //---
  double              Value(void) {return(m_value);}
 };
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- ein nicht-initialisiertes Objekt erstellen
  CMyObject *pointer;
  if(CheckPointer(pointer)==POINTER_INVALID)
    Print("1. Pointer ist ", EnumToString(CheckPointer(pointer)));
  else
    Print("1. pointer.Value()=", pointer.Value());

//--- den Zeiger initialisieren
  pointer=new CMyObject(M_PI);
  if(CheckPointer(pointer)==POINTER_INVALID)
    Print("2. Pointer ist ", EnumToString(CheckPointer(pointer)));
  else
    Print("2. pointer.Value()=", pointer.Value());

//--- Löschen des Objekts
  delete(pointer);
  if(CheckPointer(pointer)==POINTER_INVALID)
    Print("3. Pointer ist ", EnumToString(CheckPointer(pointer)));
  else
    Print("3. pointer.Value()=", pointer.Value());
 }
/*
   1. pointer is POINTER_INVALID
   2. pointer.Value()=3.141592653589793
```

© 2000-2025, MetaQuotes Ltd.

```
   3. pointer is POINTER_INVALID
*/
```

Um den Zeiger schnell zu überprüfen, können Sie auch den Operator "!" (LNOT) verwenden, der ihn durch einen impliziten Aufruf der Funktion CheckPointer überprüft. Dies ermöglicht einen prägnanteren und klareren Code. Nachfolgend sind die Prüfungen aus dem vorherigen Beispiel aufgeführt:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- ein nicht-initialisiertes Objekt erstellen
  CMyObject *pointer;
  if(!pointer)
    Print("1. Pointer ist ", EnumToString(CheckPointer(pointer)));
  else
    Print("1. pointer.Value()=", pointer.Value());

//--- den Zeiger initialisieren
  pointer=new CMyObject(M_PI);
  if(!pointer)
    Print("2. Pointer ist ", EnumToString(CheckPointer(pointer)));
  else
    Print("2. pointer.Value()=", pointer.Value());

//--- Löschen des Objekts
  delete(pointer);
  if(!pointer)
    Print("3. Pointer ist ", EnumToString(CheckPointer(pointer)));
  else
    Print("3. pointer.Value()=", pointer.Value());
 }
/*
   1. pointer is POINTER_INVALID
   2. pointer.Value()=3.141592653589793
   3. pointer is POINTER_INVALID
*/
```

Der Operator "==" wird für eine schnelle Prüfung auf NULL verwendet. Zum Beispiel: ptr==NULL oder ptr!=NULL.

Sehen Sie auch

Variablen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Referenzen. Modifikator & und Schluesselwort this

Übertragung der Parameter durch Referenz

In MQL5 können Parameter einfacher Typen nach dem Wert und durch Referenz übertragen werden, während Parameter komplizierter Typen werden immer durch Referenz übertragen. Für Andeutung dem Compiler auf die Notwendigkeit, einen Parameter durch Referenz zu übertragen wird vor dem Namen des Parameters das Zeichen Ampersand **&** gestellt.

Übertragung des Parameters durch Referenz bedeutet die Übertragung der Adresse einer Variable, darum werden alle Veränderungen des übertragenen durch Referenz Parameters werden sofort in der Ausgangsvariable dargestellt. Beim Verwenden von Parameterübertragung durch Referenz kann man gleichzeitige Rückkehr mehrerer Ergebnisse aus der Funktion ermöglichen. Man muss den Modifikator const verwenden, um der Veränderung des durch Referenz übertragenen Parameters zu.

So wenn der Eingabeparameter der Funktion ein Feld, Objekt einer Struktur oder einer Klasse ist, wird im Header der Funktion nach dem Typ der Variable und vor ihrem Namen das Symbol '&' gestellt.

Beispiel

```
class CDemoClass
  {
private:
   double            m_array[];

public:
   void              setArray(double &array[]);
  };
//+------------------------------------------------------------------+
//| Feldausfuellung                                                  |
//+------------------------------------------------------------------+
void  CDemoClass::setArray(double &array[])
  {
   if(ArraySize(array)>0)
     {
     ArrayResize(m_array,ArraySize(array));
     ArrayCopy(m_array, array);
     }
  }
```

In dem angeführten Beispiel ist die Klasse CDemoClass erklärt, die ein privates Glied enthält - Feld m_array[] des Typs double. Die Funktion setArray(), in die durch Referenz das Feld array[] übertragen wird, ist erklärt. Wenn Header dieser Funktion ohne Andeutung der Übertragung durch Referenz geschrieben wird, d.h. das Symbol Ampersand wird weggenommen, wird beim Versuch, diesen Kode zu compilieren, eine Fehlernachricht erscheinen.

Abgesehen davon, dass das Feld durch Referenz übertragen wird, können wir nicht ein Feld einem anderen Feld zuordnen. Man muss den Inhalt von Feld-Quelle ins Feld-Rezipient elementenweise kopieren. Das Symbol & für Felder und Strukturen bei der Übertragung als Parameter der Funktion ist bei der Funktionsbeschreibung notwendig.

Schluesselwort this

© 2000-2025, MetaQuotes Ltd.

Variable des Typs Klasse (Objekt) kann sowohl durch Referenz, als auch durch Anzeiger übertragen werden. Anzeiger, wie auch Referenz verwendet man, um Zugang zum Objekt zu bekommen. Nach der Erklaerung von Objektanzeiger muss man die Anweisung new für seine Erzeugung und Initialisierung verwenden.

Das reservierte Wort **this** ist für Erhaltung der Referenz des Objektes auf sich selbst bestimmt. Sie ist innerhalb der Klassenmethoden oder Strukturenmethoden zugaenglich. **this** verweist immer auf das Objekt, in dessen Methode es verwendet wird, und der Ausdruck GetPointer(this) gibt Anzeiger des Objektes, dessen Glied die Funktion ist, in der die Funktion GetPointer() aufgerufen wurde. In MQL5 können die Funktionen keine Obkekte rückgeben, aber sie können Objektanzeiger rückgeben.

D.h. wenn es notwendig ist, dass die Funktion das Objekt rückgibt, koenen wir den Anzeiger dieses Objektes als GetPointer(this) rückgeben. Fuegen wir in die Beschreibung der Klasse CDemoClassdie Funktion getDemoClass() hinzu, die Objektanzeiger dieser Klasse rückgibt.

```
class CDemoClass
  {
private:
   double            m_array[];

public:
   void              setArray(double &array[]);
   CDemoClass       *getDemoClass();
  };
//+------------------------------------------------------------------+
//| Feldausfuellung                                                  |
//+------------------------------------------------------------------+
void  CDemoClass::setArray(double &array[])
  {
   if(ArraySize(array)>0)
     {
      ArrayResize(m_array,ArraySize(array));
      ArrayCopy(m_array,array);
     }
  }
//+------------------------------------------------------------------+
//| gibt eigenen Anzeiger zurück                                     |
//+------------------------------------------------------------------+
CDemoClass *CDemoClass::getDemoClass(void)
  {
   return(GetPointer(this));
  }
```

Strukturen haben keine Anzeiger, man kann die Anweisungen *new* und *delete zu ihnen nicht anwenden*, GetPointer(this) kann auch nicht verwendet werden.

Sehen Sie auch

Objektanzeiger, Erzeugung und Entfernung von Objekten, Sichtbarkeitsbereich und Lebensdauer der Variablen

© 2000-2025, MetaQuotes Ltd.

Operationen und Ausdrücke

Einigen Symbolen und Symbolfolgen wird ein besonderer Wert gegeben. Das sind sogenannte Operationssymbole, ZB.:

```
+ - * / %      Symbole arithmetischer Operationen
&& ||          Symbole logischer Operationen
= += *=        Symbole der Zuordnungsoperationen
```

Operationssymbole werden in Ausdrücken verwendet und dann sinnvoll, wenn ihnen entsprechende Operanden gegeben sind. Ein besonderer Wert wird auch den Interpunktionszeichen gegeben. Interpunktionszeichen schliessen runde Klammern, Figurklammern, Komma, Doppelpunkt und Semikolon ein.

Operationssymbole, Interpunktionszeichen und Spaces werden verwendet, um Sprachelemente abzugliedern.

In diesem Abschnitt werden folgende Themen betrachtet:

- Ausdrücke
- Arithmetische Operationen
- Zuordnungsoperationen
- Vergleichsoperationen
- Logische Operationen
- Bitorganisierte Operationen
- Andere Oerationen
- Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Ausdrücke

Ausdruck besteht aus einem oder vielen Operanden und Operationssymbolen. Kann in viele Zeile geschrieben werden.

Beispiele:

```
a++; b = 10;          // mehrere Ausdrücke liegen auf einer Zeile
//---  ein Ausdruck wird in mehrere Zeilen geteilt
x = (y * z) /
    (w + 2) + 127;
```

Ausdruck mit Semikolon beendet (;), ist der Operator.

Sehen Sie auch

Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Arithmetische Operationen

Zu arithmetischen Operationen gehören additive und Multiplikationsoperationen:

```
Summe von Werten                          i = j + 2;
Substrahieren von Werten                  i = j - 3;
Zeichenänderung                          x = - x;
Multiplikation von Werten                 z = 3 * x;
Quotient euer Devision                    i = j / 5;
Rest einer Division                       minutes = time % 60;
Addieren von 1 zum Variablenwert          i++;
Addieren von 1 zum Variablenwert          ++i;
Substrahieren von 1 vom Variablenwert     k--;
Substrahieren von 1 vom Variablenwert        --k;
```

Inkrement- und Dekrementoperationen werden nur Variablen gegenüber vewendet, nicht Konstanten gegenüber. Praefixinkrement (++i) und praefixdekrement (--k) werden variable gegenüber unmittelbar vor der Verwendung dieser Variable im Ausdruck.

Postfixinkrement (i++) und Postfixdekrement (k--) werden der Variable gegenüber verwendet sofort nach der Verwendung dieser Variable im Ausdruck.

Wichtiger Hinweis

```
int i=5;
int k = i++ + ++i;
```

Rechnerische Probleme können auftreten, während Sie den obigen Ausdruck von einer Programmierumgebung zu einem anderen (wie zB von Borland C++ in MQL5). Im Allgemeinen hängt die Reihenfolge der Auswertung auf der Compilerimplementierung. In der Praxis gibt es zwei Möglichkeiten, um Post-Dekrement (Post-Inkrement) zu implementieren:

1. Die Post-Dekrement (Post-Inkrement) wird der Variablen nach der Berechnung der ganze Ausdruck angewendet;
2. Die Post-Dekrement (Post-Inkrement) wird der Variablen sofort bei der Operation angewendet.

In MQL5 derzeit ist die erste Methode zur Berechnung der Post-Dekrement (Post-Inkrement) implementiert. Aber selbst mit diesem Wissen ist es besser mit der Nutzung dieser Subtilität nicht zu experimentieren.

Beispiele:

```
int a=3;
a++;             // richtiger Ausdruck
int b=(a++)*3;   // falscher Ausdruck
```

Sehen Sie auch

Prioritäten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Zuordnungsoperationen

Der Ausdruckswert, dessen Teil eine Zuordnungsoperation ist, der Wert des linken Operanden nach der Zuordnung:

```
Zuordnung der Größe x der Variable y                              y = x;
```

Folgende Operationen verbinden arithmetische oder bitorganisierte Operationen mit der Zuordnungsoperation:

```
Anstieg der Variable y um x                                       y += x;
Erniedrigung der Variable y um x                                  y -= x;
Multiplikation der Variable y mit x                               y *= x;
Division der Variable y durch x                                   y /= x;
Rest der Division der Variable y durch x                          y %= x;
Verschiebung der Binaerdarstellung y rechts um x Bit              y >>= x;
Verschiebung der Binaerdastellung y links um x Bit                y <<= x;
Bitorganisierte Operation UND der Binaerdarstellungen y und x     y &= x;
Bitorganisierte Operation ODER der Binaerdarstellungen y und x    y |= x;
Bitorganisierte Operation ausschliessendes ODER
                  Binaerdarstellungen y und x                     y ^= x;
```

Bitorganisierte Operationen werden nur mit Ganzzahlen durchgeführt. Bei der Operationsdurchführung logische Verschiebung der dastellung y rechts/links um x Bit werden 5 Binaerstellen mit niedrigem Stellenwert des Wertes x verwendet, hoeherwertige Stellen werden weggelassen, d.h. die Verschiebung wird um 0-31 Bit durchgeführt.

Bei der Operationsdurchführung %= (Wert y des Moduls x) Ergebniszeichen faellt mit dem Zeichen des Dividenden zusammen.

Im Ausdruck kann der Zuordnungsoperator mehrmals auftreten. In diesem Fall wird die Ausdruckbearbeitung von rechts nach links durchgeführt:

```
y=x=3;
```

Zuerst wird der Variable x der Wert 3 zugeordnet, dann wird der Variable y der Wert von Variable x zugeordnet, d.h. auch 3.

Sehen Sie auch

Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Vergleichsoperationen

Logischer Wert FALSCH wird von Null-Wert dargestellt, und der Wert RICHTIG wird von jedem Nichtnull-Wert dargestellt.

Der Wert von Ausdrücken, die Vergleichsoperationen oder logische Operationen enthalten, sind FALSCH(0) oder RICHTIG(1).

```
Richtig, wenn a gleich b                        a == b;
Richtig, wenn a ungleich b                      a != b;
Richtig, wenn a kleiner als b                   a < b;
Richtig, wenn a groesser als b                  a > b;
Richtig, wenn a kleiner als oder b              a <= b;
Richtig, wenn a groesser als oder gleich b      a >= b;
```

Man kann nicht zwei Realzahlen vergleichen. In meisten Faellen können zwei annehmlich gleiche Zahlen nicht gleich sein wegen des Unterschiedes im 15. Zeichen nach dem Komma. Für korrekten Vergleich der zwei Realzahlen muss man normalisierte Differenz dieser Zahlen mit dem Nullwert vergleichen.

Beispiel:

```
bool CompareDoubles(double number1,double number2)
  {
   if(NormalizeDouble(number1-number2,8)==0) return(true);
   else return(false);
  }
void OnStart()
  {
   double first=0.3;
   double second=3.0;
   double third=second-2.7;
   if(first!=third)
     {
      if(CompareDoubles(first,third))
         printf("%.16f  %.16f   dennoch sind sie gleich ",first,third);
     }
  }
// Ergebnis: 0.3000000000000000  0.2999999999999998   dennoch gleich
```

Sehen Sie auch

Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Logische Operationen

Logische Negation NICHT(!)

Operand der logischen Negation NICHT (!) muss den arithmetischen Typ haben. Das Ergebnis ist RICHTIG(1), wenn der Wert des Operanden FALSCH(0) ist, und FALSCH(0), wenn der Operand nicht gleich FALSCH(0 ist).

```
if(!a) Print("nicht 'a'");
```

Logische Operation ODER (||)

Logische Operation ODER (||) der Werte x und y. Der Ausdruckswert ist RICHTIG(1), wenn der Wert von x oder y gueltig (nicht Null) ist. Im Gegenfall - FALSCH (0).

```
if(x<0 || x>=max_bars) Print("out of range");
```

Logische Operation UND (&&)

Logische Operation UND (&&) der Werte x und y. Der Ausdruckswert ist RICHTIG (1), wenn die Werte x und y gueltig sind (nicht Null). Im Gegenfall - FALSCH(0). Logische Ausdrücke werden vollkommen berechnet, d.h. nicht mit dem Schema "Kurzberechnung".

```
if(p!=x && p>y) Print("TRUE");
```

Kurze Einschätzung der logischen Operationen

Logischen Ausdrücken gegenüber wird das Schema der sogenannten "kurzen Einschätzung" angewendet ,d.h, Berechnung des Ausdrucks wird dann beendet, wenn man das Ergebnis des Ausdrucks genau einschätzen kann.

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- das erste Beispiel der kurzen Einschätzung
   if(func_false() && func_true())
     {
      Print("Operation &&: Diese Meldung sehen Sie nie");
     }
   else
     {
      Print("Operation &&: Das Ergebnis des ersten Ausdrucks false, darum wurde der zw
    ;}
//--- das zweite Beispiel der kurzen Einschätzung
   if(!func_false() || !func_true())
     {
      Print("Operation ||: Das Ergebnis des ersten Ausdrucks true, darum wurde der zwe
```

© 2000-2025, MetaQuotes Ltd.

```
    ;}
   else
     {
      Print("Operation ||: Diese Meldung sehen Sie nie");
     }
  }
//+------------------------------------------------------------------+
//| Funktion gibt immer false zurück                                 |
//+------------------------------------------------------------------+
bool func_false()
  {
   Print("Funktion func_false()");
   return(false);
  }
//+------------------------------------------------------------------+
//| Funktion gibt immer true zurück                                  |
//+------------------------------------------------------------------+
bool func_true()
  {
   Print("Funktion func_true()");
   return(true);
  }
```

Sehen Sie auch

Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Bitorganisierte Operationen

1-Komplement

1-Komplement der Variable. Ausdruckswert hat 1 in allen Stellen, wo die Variable 0 hat und o in allen Stellen, wo die Variable 1 hat.

```
b = ~n;
```

Beispiel:

```
   char a='a',b;
   b=~a;
   Print("a = ",a, "  b = ",b);
// Ergebnis wird derartig sein:
// a = 97   b = -98
```

Verschiebung nach rechts

Binaerdarstellung x wird nach rechts um y Stellen verschoben. Wenn der verschobene Wert den zeichenlosen Typ hat, wird die logische Verschiebung nach rechts durchgeführt, d.h. die sich links befreiende Stellen werden mit Nullen ausgefuellt werden.

Wenn aber der verschobene Ausdruck einen Zeichentyp hat, wird arithmetische Verschiebung nach rechts durchgeführt, d.h. die sich links befreiende Stellen werden von Zeichebits ausgefuellt (wenn die Zahl positiv ist, wird der Wert von Zeichenwert 0 sein, wenn die Zahl negativ ist, wird der Wert von Zeichenbit 1 sein).

```
x = x >> y;
```

Beispiel:

```
   char a='a',b='b';
   Print("Before:  a = ",a, "  b = ",b);
//--- verschieben wir nach rechts
   b=a >> 1;
   Print("After:   a = ",a, "  b = ",b);
// Ergebnis wird derartig sein:
// Before:  a = 97   b= 98
// After:   a = 97   b= 48
```

Verschiebung nach links

Binaerdarstellung x wird nach links um y Stellen verschoben; sich befreiende Stellen werden von Nullen ausgefuellt.

```
x = x << y;
```

Beispiel:

```
   char a='a',b='b';
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("Before:  a = ",a, "  b = ",b);
//--- verschieben wir nach links
   b=a << 1;
   Print("After:   a = ",a, "  b = ",b);
// Ergebnis wird derartig sein:
// Before:  a = 97   b = 98
// After:   a = 97   b = -62
```

Es ist nicht empfehlenswert, um mehrere oder gleiche Anzahl der Bits zu verschieben, als die Laenge der verschiebten Variable, denn das Ergebnis der Operation ist unbestimmt.

Bitorganisierte Operation UND

Bitorganisierte Operation UND der Binaerdarstellungen x und y. Der Ausdruckswert hat 1 (RICHTIG) in allen Stellen, in denen x und y keine 0 haben; und 0 (FALSCH) - in allen anderen Stellen.

```
b = ((x & y) != 0);
```

Beispiel:

```
   char a='a',b='b';
//--- Operation UND
   char c=a&b;
   Print("a = ",a,"  b = ",b);
   Print("a & b = ",c);
// Ergebnis wird derartig sein:
// a = 97   b = 98
// a & b = 96
```

Bitorganisierte Operation ODER

Bitorganisierte Operation ODER der Binaerdarstellungen x und y. Ausdruckswert hat 1 in allen Stellen ,wo x oder y keine 0 hat, und 0 - in allen anderen Stellen.

```
b = x | y;
```

Beispiel:

```
   char a='a',b='b';
//--- Operation ODER
   char c=a|b;
   Print("a = " ,a,"  b = ",b);
   Print("a | b = ",c);
// Ergebnis wird derartig sein:
// a = 97   b = 98
// a | b = 99
```

Bitorganisierte Operation ausschliessendes ODER

© 2000-2025, MetaQuotes Ltd.

Bitorganisierte Operation ausschliessendes ODER (eXclusive OR) der Binaerdarstellungen x und y. Der Ausdruckswert hat 1 in den Stellen, in denen x und y verschiedene Binaerwerte haben, und 0 - in allen anderen Stellen.

```
b = x ^ y;
```

Beispiel:

```
   char a='a', b='b';
//--- Operation ausschliessendes ODER
   char c=a^b;
   Print("a = ",a,"  b = ",b);
   Print("a ^ b = ",c);
// Ergebnis wird derartig sein:
// a = 97   b = 98
// a ^ b = 3
```

Bitorganisierte Operationenwerden nur mit Ganzzahlen durchgeführt.

Sehen Sie auch

Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Andere Operationen

Indizieren ([])

Beim Zugriff zum i-Feldelement ist der Ausdruckswert der Wert von Variable mit Laufnummer 1.

Beispiel:

```
array[i] = 3; // dem i-Element des Feldes array den Wert 3 zuordnen.
```

Feldindex kann nur Ganzzahl sein. Zugelassen werden nicht mehr als vierdimensionele Felder. Indizieren jeder Messung wird von 0 bis der Messungsgroesse -1 durchgeführt. Beim eindimensionalen Feld aus 50 Elementen wird der Zugriff zum ersten Element als array [0] aussehen, zum letzten Element - array[49].

Beim Zugriff ausserhalb des Feldes wird das Subsystem einen kritischen Fehler generieren und Durchführung des Programms wird unterbrochen.

Funktionsaufruf mit Argumenten x1, x2,..., xn

Jedes Argument kann Konstante, Variable oder Ausdruck des entsprechenden Typs darstellen. Übergabeargumente werden durch Kommas getrennt und muessen sich innerhalb runder Klammern befinden, linke Klammer muss nach den Namen der Aufrufunktion folgen.

Ausdruckswert ist der Wert, der durch die Funktion zurückgegeben wird. Wenn der Typ des Rückgabewertes einer Funktion void ist, kann der Aufruf dieser Funktion nicht rechts in der Zuordnungsoperation gestellt werden. Bitte beachten Sie, die Reihenfolge der Ausführung der Ausdrücke x1,..., xn wird garantiert.

Beispiel:

```
   int length=1000000;
   string a="a",b="b",c;
//---
   int start=GetTickCount(),stop;
   long i;
   for(i=0;i<length;i++)
     {
      c=a+b;
     }
   stop=GetTickCount();
   Print("time for 'c = a + b' = ",(stop-start)," milliseconds, i = ",i);
```

Operation Komma (,)

Die durch Kommas getrennten Ausdrücke werden von links nach rechts berechnet. Alle Nebeneffekte des linken Ausdrucks können vor der Berechnung des rechten Ausdrucks erscheinen. Typ und

© 2000-2025, MetaQuotes Ltd.

Ergebniswert faellt mit dem Typ und Ergebniswert des rechten Ausdrucks zusammen. Als Beispiel kann die Liste der übertragenen Paameter gesehen werden (s. oben).

Beispiel:

```
for(i=0,j=99; i<100; i++,j--) Print(array[i][j]);
```

Operation Punkt (.)

für direkten Zugriff zu oeffentlichen Gliedern der Strukturen und Klassen wird die Operation Punkt verwendet. Syntax

```
Name der Variable des Typs Strukturen .Name des Gliedes
```

Beispiel:

```
   struct SessionTime
     {
      string sessionName;
      int    startHour;
      int    startMinutes;
      int    endHour;
      int    endMinutes;
     } st;
   st.sessionName="Asian";
   st.startHour=0;
   st.startMinutes=0;
   st.endHour=9;
   st.endMinutes=0;
```

Operation mit Kontexterlaubnis (::)

Jede Funktion im mql5-Programm hat ihr eigenes Durchführungskontext.ZB. die Systemfunktion Print() wird im Globalkontext durchgeführt. Importfunktionen werden im kontext des entsprechenden Imports aufgerufen. Funktionen-Methoden der Klassen haben Kontext der entsprechenden Klasse. Syntax der Operation der Kontextfreigabe:

```
[Kontextname]::Funktionsname(Parameter)
```

Wenn der Kontextname fehlt, ist es explizites Zeichen vom Verwenden des globalen Kontextes. Wenn die Operation der Kontextfreigabe fehlt, wird die Funktion im naechsten Kontext gesucht. Wenn die Funktion im lokalen Kontext fehlt, wird die Suche im globalen Kontext durchgeführt.

Die Operation der Kontextfreigabe wird auch für Definieren der Funktion -Glied einer Klasse verwendet.

```
Typ Klassenname::Funktionsname(Parameterbeschreibung)
   {
// Funktionskoerper
   }
```

© 2000-2025, MetaQuotes Ltd.

Verwendung von mehreren Funktionen mit dem gleichen Namen von verschiedenen Ausführungskontexte in einem Programm kann Mehrdeutigkeit führen. Die Prioritätsreihenfolge der Funktionsaufrufe ohne ausdrückliche Spezifikation von Umfang ist wie folgt:

1. Klassenmethoden. Wenn es keine Funktion mit dem angegebenen Namen in der Klasse gibt, gehen Sie zum nächsten Level.
2. MQL5-Funktionen. Wenn die Sprache nicht eine solche Funktion hat, gehen Sie zum nächsten Level.
3. Benutzerdefinierte globale Funktionen. Wenn keine Funktion mit solchem Namen gefunden ist, gehen Sie zum nächsten Level.
4. Importierte Funktionen. Wenn keine Funktion mit dem angegebenen Namen gefunden wird, gibt der Compiler einen Fehler zurück.

Um die Mehrdeutigkeit der Funktionsaufrufe zu vermeiden, geben Sie immer den Funktionsumfang mit Hilfe der Kontexterlaubnisoperationen.

Bespiel:

```
#property script_show_inputs
#import "kernel32.dll"
   int GetLastError(void);
#import

class CCheckContext
  {
   int          m_id;
public:
               CCheckContext() { m_id=1234; }
protected:
   int         GetLastError() { return(m_id); }
  };
class CCheckContext2 : public CCheckContext
  {
   int          m_id2;
public:
               CCheckContext2() { m_id2=5678; }
   void        Print();
protected:
   int         GetLastError() { return(m_id2); }
  };
void CCheckContext2::Print()
  {
   ::Print("Terminal GetLastError",::GetLastError());
   ::Print("kernel32 GetLastError",kernel32::GetLastError());
   ::Print("parent GetLastError",CCheckContext::GetLastError());
   ::Print("our GetLastError",GetLastError());
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   CCheckContext2 test;
   test.Print();
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

Operation der Größenentnahme des Datentyps oder der Größe des Objektes jedes Datentyps (sizeof)

Mittels der Operation sizeof kann die Speichergroesse bestimmt werden, die dem Identifikatoren oder dem Typ entspricht. Operation sizeof hat folgendes Format:

Beispiel:

```
sizeof(Ausdruck)
```

Als Ausdruck kann jeder Identifikator oder Typenname in Klammern verwendet werden. Merken wir an, dass der Typenname void nicht verwendet werden kann, und Identifikator kann nicht zum Bitfeld gehören oder Funktionsname sein.

Wenn als Ausdruck der Name des statistischen Feldes angegeben wird (d.h. die erste Dimensionierung vorgegebenwird), wird als Ergebnis die Größe des ganzen Feldes sein (d.h. Produkkt der Multiplikation der Elementenzahl mit Typenlaenge). wenn als Ausdruck der Name des dymnamischen Feldes angegeben wird (erste Dimensionalisierung nicht vorgegeben), wird als Ergebnis die Größe des Objektes des dynamischen Feldes.

Wenn sizeof zum Strukturnamen oder Klassenname gegenüber verwendet wird oder dem Identifikatoren, der den Typ des Struktur oder Klasse hat, wird a;s Ergebnis die tatsaechliche Größe der Struktur oder Klasse sein.

Beispiel:

```
struct myStruct
  {
   char   h;
   int    b;
   double f;
  } str;
Print("sizeof(str) = ",sizeof(str));
Print("sizeof(myStruct) = ",sizeof(myStruct));
```

Größenberechnung wird in der Etappe der Compilierung durchgeführt.

Sehen Sie auch

Prioritaeten und Operationsordnung

© 2000-2025, MetaQuotes Ltd.

Prioritaeten und Operationsordnung

für jede Gruppe der Operationen ist die Prioritaet in der Tabelle gleich. Je hoehe Prioritaet von Gruppe der Operationen ist, desto hoeher sich die Gruppe in der Tabelle befindet. Reihenfolge der Befehle bestimmt Gruppieren von Operationen und Operanden.

Achtung: Prioritaet der Operationsdurchfuehrung in Sprache MQL5 enspricht der Prioritaet in Sprache C++, und unterscheidet sich von der Prioritaet vorgegeben in MQL4.

Operation	Beschreibung	Reihenfolge der Befehle
() [] .	Funktionsaufruf Hervorheben des Feldelementes Hervorheben des Strukturelementes	Von links nach rechts
! ~ - ++ -- (Typ) sizeof	Logische Verneinung Bitorganisierte Verneinung (complement) Zeichenänderung Erhoehung um eins (increment) Erniedrigung um eins (decrement) Typumwandlung Bestimmung der Größe in Bytes	von rechts nach links
* / %	Multiplikation Division Division modulo	von links nach rechts
+ -	Addition Substrahieren	von links nach rechts
<< >>	Verschiebung nach links Verschiebung nach rechts	von links nach rechts
< <= > >=	kleiner als kleiner als oder gleich groesser als groesser als oder gleich	von links nach rechts
== !=	Gleich Ungleich	von links nach rechts
&	Bitorganisierte Operation UND	von links nach rechts
^	Bitorganisierte Operation ausschliessendes ODER (eXclude OR)	von links nach rechts
|	Bitorganisierte operation ODER	von links nach rechts
&&	Logische operation UND	von links nach rechts
||	Logische Operation ODER	von links nach rechts
?:	Bedingungsoperation	Von rechts nach links
=	Zuordnung	Von rechts nach links

© 2000-2025, MetaQuotes Ltd.

Operation	Beschreibung	Reihenfolge der Befehle
*= /= %= += -= <<= >>= &= ^= |=	Multiplikation mit Zuordnung Division mit Zuordnung Division modulo Addition mit Zuordnung mit Zuordnung Substrahieren mit Zuordnung Verschiebung nach links mit Zuordnung Verschiebung nach rechts mit Zuordnung Bitorganisiertes UND mit Zuordnung Ausgeschlossenes ODER mit Zuordnung Bitorganisiertes ODER mit Zuordnung	
,	Komma	Von links nach rechts

für die Veränderung der Reihenfolge werden runde Klammern verwendet, die die hoechste Prioritaet haben.

© 2000-2025, MetaQuotes Ltd.

Operatoren

Operatoren der Sprache beschreiben einige algorithmische Handlungen, die man für die Aufgabenloesung durchführen muss. Programmkörper ist die Folge solches Operators. Die aufeinander folgenden Operatoren werden durch Semikolon getrennt.

Operator	Beschreibung
Zusammengesetzter Operator {}	eine oder mehrere Operatoren des Typs in geschweiften Klammern { }
Operator-Ausdruck (;)	Jeder Ausdruck der mit Semikolon beendet (;)
Operator return	Unterbricht die laufende Funktion und gibt die Steuerung dem Aufrufprogramm zurück.
Konditionaloperator if-else	Verwendet bei der Notwendigkeit, die Wahl zu treffen
Wenn-Anweisung ?:	Einfacherer Analog der Anweisung if-else
Auswahlanweisung switch	Gibt die Steuerung der Anweisung, die dem Ausdruckswert entspricht
Zyklusanweisung while	Anweisung wird durchgeführt, bis der Ausdruck falsch ist. Ausdruckspruefung wird vor jeder Iteraion durchgeführt.
Zyklusanweisung for	Anweisung wird durchgeführt, bis der Ausdruck falsch ist. Ausdruckspruefung wird vor jeder Iteraion durchgeführt
Zyklusanweisung do-while	Anweisung wird durchgeführt, bis der Ausdruck falsch ist. Pruefung der Schlussbedingung wird am Ende, nach jedem Zyklusgang durchgeführt. Zykluskoerper wird immer wenigstens einmal durchgeführt.
Anweisung break	Unterbricht die Durchführung der naechsten eingebettenen Aussenanweisung switch, while, do-while oder for
Anweisung continue	Gibt die Steuerung am Ende der naechsten Aussenzyklusanweisung while, do-while oder for
Matrixproduktoperator @	Der MQL5-Operator @ implementiert die Matrixmultiplikation nach den Regeln der linearen Algebra. Er ermöglicht die Multiplikation von Matrizen und Vektoren sowie die Skalarmultiplikation von Vektoren.
Anweisung new	Erzeugt das Objekt entsprechender Größe und gibt Descriptor des erzeugten Objektes zurück.
Anweisung delete	Entfernt das von Anweisung new geschaffenes Objekt

Anweisung kann eine oder mehrere Zeilen besetzen. Zwei oder mehrere Anweisungen können sich auf einer Zeile befinden. Anweisungen, Die die Reihenfolge der Befehle regeln (if, if-else, switch, while und for) können in einander eingebettet sein.

Beispiel:

```
if(Month() == 12)
```

© 2000-2025, MetaQuotes Ltd.

```
if(Day() == 31) Print("Happy New Year!");
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Zusammengesetzter Operator

Zusammengesetzter Operator (Block) besteht aus einem oder mehreren Operatoren jedes Typs in geschweiften Klammern { }. Nach der rechten geschweiften Klammer muss kein Semikolon stehen (;).

Beispel:

```
if(x==0)
  {
   Print("invalid position x=",x);
   return;
  }
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Operator-Ausdruck

Jeder Ausdruck, der mit Semikolon (;) beendet, ist ein Operator. Weiter folgen Beispiele von Operatoren-Ausdrücke.

Operator der Zuordnung:

Identifikator = Ausdruck;

```
x=3;
y=x=3;
bool equal=(x==y);
```

Im Ausdruck kann Operator der Zuordnung mehrmals auftreten. In diesem Fall wird der Ausdruck von rechts nach links bearbeitet.

Operator des Funktionsaufrufs:

Funktionsname (Argument1,..., ArgumentN);

```
FileClose(file);
```

Leerer Operator

Besteht ausschliesslich aus Semikolon (;) und wird verwendet für die Bezeichnung von Leerkoeper der Steueranweisung.

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Rücksetzoperator return

Operator return stoppt die laufende Funktion und gibt Steuerung dem Aufrufprogramm zurück. Ergebnis der Berechnung wird der Aufruffunktion zurückgegeben. Ausdruck kann einen Operator der Zuordnung haben.

Beispiel:

```
int CalcSum(int x, int y)
  {
   return(x+y);
  }
```

In Funktionen mit dem Typ des Rücksatzwertes void muss man den Operator return ohne Ausdruck verwenden:

```
void SomeFunction()
  {
   Print("Hello!");
   return;     // dieser Operator kann entfernt werden.
  }
```

Rechte geschweifte Klammer setzt explizite Durchführung des Operators return ohne Ausdruck voraus.

Man kann einfache Typen, einfache Strukturen, Objektanzeiger rückgeben. Durch den Operator *return* können keine Felder, Klassenobjekte, Variablen der komplizierten Strukturen zurückgegeben werden.

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Konditionaloperator if-else

Operator IF - ELSE wird bei der Notwendigkeit, eine Wahl zu treffen, verwendet. Vorschriftmaessig sieht die Syntax so aus:

```
if (Ausdruck)
     Operator1
else
     Operator2
```

Wenn der Ausdruck richtig ist, wird der Operator1 durchgeführt und die Steuerung wird an den Operator gegeben, die dem Operator2 folgt (d.h. die Operator2 wird nicht durchgeführt). Wenn der Ausdruck falsch ist, wird die Operator2 durchgeführt.

Teil else des Operators if kann weggelassenwerden. Darum kann in eigebetteten Operatoren if mit weggelassenem Teil else Mehdeutigkeit entstehen. In diesem Fall verbindet sich else mit der naechsten vorangehenden Operator if in demselben Block, die keinen Teil else hat.

Beispiele:

```
//--- Teil else gehört zum zweiten Operator if:
if(x>1)
   if(y==2) z=5;
else     z=6;
//--- Teil else gehört zum ersten Operator if
if(x>1)
  {
   if(y==2) z=5;
  }
else        z=6;
//--- Eingebettete Operatoren
if(x=='a')
  {
   y=1;
  }
else if(x=='b')
  {
   y=2;
   z=3;
  }
else if(x=='c')
  {
   y=4;
  }
else Print("ERROR");
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Ternärer Operator ?:

Gesamtform des ternären Operators sieht so aus:

```
Ausdruck1? Ausdruck2:Ausdruck3
```

Als erster Operand - "Ausdruck1" - kann jeder Ausdruck verwendet werden, dessen Ergebnis der Wert bool ist. Wenn das Ergebnis true ist, wird die Anweisung durchgeführt, die vom zweiten Operator vorgegeben wird, d.h. "der Ausdruck2".

Wenn aber der erste Operand false wird der dritte Operand - "Ausdruck3" durchgeführt. Der zweite und der dritte Operand muessen Werte eines Typs Typs zurückgeben und den Typ void nicht haben. Ergebnis der Durchführung des Wenn-Operators ist das Ergebnis des Ausdrucks2 oder Ergebnis des Ausdrucks3, abhängig vom Ergebnis des Ausdrucks1.

```
//--- normieren wir Differenz unter Eröffnungspreisen und Schlusspreisen im Tagsausmas
double true_range = (High==Low)?0:(Close-Open)/(High-Low);
```

Diese Aufzeichnung ist der nächsten äquivalent

```
   double true_range;
   if(High==Low)true_range=0;                // wenn High und Low gleich sind
   else true_range=(Close-Open)/(High-Low); // wenn Ausmass nicht Null ist
```

Einschränkungen bezüglich der Verwendung des Operators

Basierend auf dem Wert von "Ausdruck1", muss der Operator einen der zwei Werte zurückgeben - entweder "Ausdruck2" oder "Ausdruck3". Es gibt einige Einschränkungen auf diese Ausdrücke:

1. Verwechseln Sie nicht den benutzerdefinierten Typ mit dem einfachen Typ oder Enumeration. NULL kann fur den Zeiger verwendet werden.
2. Wenn Typen von Werten einfach sind, wird der Operator des maximalen Typs sein (siehe Typ-Umwandlung).
3. Wenn einer der Werte eine Enumeration ist und die zweite von einem numerischen Typ ist, wird die Enumeration durch int ersetzt und die zweite Regel angewendet wird.
4. Wenn beide Werte Enumerationen sind, müssen ihre Typen identisch sein, und der Operator wird vom Enumeration-Typ sein.

Einschränkungen für die benutzerdefinierte Typen (Klassen oder Strukturen):

a) Typen müssen identisch sein oder sollte eine von dem anderen abgeleitet werden.

b) Wenn Typen nicht identisch sind (Vererbung), dann wird das Kind implizit zum übergeordneten Typ umgewandelt, dh der Operator wird von der übergeordneten Typ sein.

c) Verwechseln Sie nicht das Objekt und den Zeiger - beide Ausdrücke sind entweder Objekte oder Zeiger. NULL kann für den Zeiger verwendet werden..

Hinweis

© 2000-2025, MetaQuotes Ltd.

Seien Sie vorsichtig bei der Verwendung des Wenn-Operators als Argument für eine überladene Funktion, da der Typ des Resultats des Wenn-Operators zum Zeitpunkt der Kompilation des Programms definiert wird. Und dieser Typ ist wie die größeren vom Typen "Ausdruck2" und "Ausdruck3" definiert.

Beispiel:

```
void func(double d) { Print("double argument: ",d); }
void func(string s) { Print("string argument: ",s); }

bool   Expression1=true;
double Expression2=M_PI;
string Expression3="3.1415926";

void OnStart()
  {
   func(Expression2);
   func(Expression3);

   func(Expression1?Expression2:Expression3);   // erhalten Compiler-Warnung über expl
   func(!Expression1?Expression2:Expression3);  // erhalten Compiler-Warnung über expl
  }

//   Ergebnis:
//   double argument: 3.141592653589793
//   string argument: 3.1415926
//   string argument: 3.141592653589793
//   string argument: 3.1415926
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Umschalteroperator switch

Vergleicht den Ausdruckswert mit Konstanten in allen Arten *case* und gibt die Steurung des Operator, die dem Ausdruckswert entspricht. Jede Art kann von ganzzahliger Konstante, Symbolkonstante oder Konstantausdruck notiert werden. Der Ausdruck des Operators *switch* muss ganzzahlig sein - int oder uint.

```
switch(Ausdruck)
  {
   case Konstante: Operatoren
   case Konstante: Operatoren
      ...
   default: Operatoren
  }
```

Operatoren, verbunden mit Vermerk *default*, werden durchgeführt, wenn keine Konstante in Operatoren *case* dem Ausdruckswert nicht gleich ist. Art *default* muss nicht unbedingt erklärt werden und nicht unbedingt am letzten sein. Wenn keine Konstante dem Ausdruckswert entspricht und die Art *default* fehlt, werden keine Handlungen durchgeführt.

Das Schluesselwort *case* mit der Konstante sind einfach Vermerke, wenn Operatoren für die Art *case* durchgeführt werden, werden weiter Operatoren aller folgenden Arten durchgeführt, bis der Operator *break* auftritt. Das ermöglicht eine Aufeinanderfolge von Operatoren mit mehreren Varianten zu verbinden.

Konstantausdruck wird in der Etappe der Compilierung durchgeführt. Keine zwei Konstanten können in einem Operator-Umschalter gleiche Werte haben.

Beispiele:

```
//--- erstes Beispiel
switch(x)
  {
   case 'A':
      Print("CASE A");
      break;
   case 'B':
   case 'C':
      Print("CASE B or C");
      break;
   default:
      Print("NOT A, B or C");
      break;
  }

//---  zweites Beispiel
   string res="";
   int i=0;
   switch(i)
     {
      case 1:
```

© 2000-2025, MetaQuotes Ltd.

```
         res=i;break;
      default:
         res="default";break;
      case 2:
         res=i;break;
      case 3:
         res=i;break;
     }
   Print(res);
/*
   Ergebnis
   default
*/
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Loop-Operator while

Operator while besteht aus einem geprüften Ausdruck und einem Operator, der erfüllt werden muss.

```
while(Ausdruck)
  Operator;
```

Wenn der Ausdruck richtig ist, wird der Operator durchgeführt, bis der Ausdruck falsch wird. Wenn der Ausdruck falsch ist, wird die Steuerung des nächsten folgenden Operators gegeben. Ausdruckswert wird vor der Operatorsdurchführung berechnet. So wenn der Ausdruck vom Anfang an falsch ist, wird der Operator nicht durchgeführt.

Hinweis

Wenn voraussichtlich eine große Anzahl von Iterationen in einem Zyklus behandelt wird, ist es ratsam, dass Sie die Tatsache der erzwungenen Beendigung des Programms mit der Funktion IsStopped() überprüfen.

Beispiel:

```
while(k<n && !IsStopped())
  {
   y=y*x;
   k++;
  }
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Loop-Operator for

Operator for besteht aus drei Ausdrücken und durchgeführten Anweisung:

```
for(Ausdruck1; Ausdruck2; Ausdruck3)
   Anweisung;
```

Ausdruck1 beschreibt Zyklusinitialisierung. Ausdruck2 - Prüfung der Bedingung des Zyklussschlusses. Wenn die Bedingung richtig ist, wird der Operator des Körperzyklus for durchgeführt. Alles wiederholt sich, bis der Ausdruck2 falsch wird. Wenn er falsch ist, ist der Zyklus zu Ende und die Steuerung wird an den nächsten Operator weitergegeben. Ausdruck3 wird nach jeder Iteration berechnet.

Operator for ist der folgenden Folge der Operatoren äquivalent:

```
Ausdruck1;
while(Ausdruck2)
  {
   Operator;
   Ausdruck3;
  };
```

Jeder der drei oder alle drei Ausdrücke im Operator for können fehlen, aber man darf nicht die sie trennenden Semikolon(;) weglassen. Wenn der Ausdruck2 weggelassen wird, gilt der immer für richtig. Operator for(;;) stellt infiniten Zyklus dar, äquivalent des Operators while(1) dar. Jeder Ausdruck1 und Ausdruck3 kann aus mehreren Ausdrücken bestehen, verbunden durch drn Operator Komma ','.

Hinweis

> Wenn voraussichtlich eine große Anzahl von Iterationen in einem Zyklus behandelt wird, ist es ratsam, dass Sie die Tatsache der erzwungenen Beendigung des Programms mit der Funktion IsStopped() überprüfen.

Beispiele:

```
for(x=1;x<=7000; x++)
  {
   if(IsStopped())
      break;
   Print(MathPower(x,2));
  }
//--- anderes Beispiel
for(;!IsStopped();)
  {
   Print(MathPower(x,2));
   x++;
   if(x>10) break;
  }
//--- drittes Beispiel
for(i=0,j=n-1;i<n && !IsStopped();i++,j--) a[i]=a[j];
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Loop-Operator do while

Zyklen for und while pruefen den Abschluss am Anfang und nicht am Ende des Zyklus. Dritter Loop-Operator do - while prueft die Abschlussbedingung am Ende, nach jedem Zyklusgang. Zykluskoerper wird immer wenigstens einmal durchgeführt.

```
do
   Operator;
while(Ausdruck);
```

Zuerst wird der Operator durchgeführt, dann wird de Ausdruck berechnet. Wenn der richtig ist, wird der Operator wieder und wieder durchgeführt. Wenn der Ausdruck falsch wird, ist der Zyklus zu Ende.

Hinweis

Wenn voraussichtlich eine große Anzahl von Iterationen in einem Zyklus behandelt wird, ist es ratsam, dass Sie die Tatsache der erzwungenen Beendigung des Programms mit der Funktion IsStopped() überprüfen.

Beispiel:

```
//--- Berechnung der Aufeinanderfolge von Zahlen Fibonatschi
   int counterFibonacci=15;
   int i=0,first=0,second=1;
   int currentFibonacciNumber;
   do
     {
      currentFibonacciNumber=first+second;
      Print("i = ",i,"  currentFibonacciNumber = ",currentFibonacciNumber);
      first=second;
      second=currentFibonacciNumber;
      i++; // ohne diesen Operator wird infiniter Zyklus!!
     }
   while(i<counterFibonacci && !IsStopped());
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Endoperator break

Operator break unterbricht die Durchführung der naechsten eingebetteten Aussenoperator switch, while, do-while oder for. Steuerung wird dem Operator gegeben, die dem beendenden Operator folgt. Einer der Zwecke dieses Operators - Durchführung des Zyklus zu beenden bei der Werzuordnung einer Variable.

Beispiel:

```
//--- Suche des ersten Nullelementes
for(i=0;i<array_size;i++)
  if(array[i]==0)
    break;
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Forsetzungsoperator continue

Operator continue gibt die Steuerung am Anfang der Aussenoperator des Zykles while, do-while oder for, und dabei ruft Anfang der folgenden Iteration auf. Dieser Operator ist dem Operator break der Wirkung nach gegenübergestellt.

Beispiel:

```
//--- Summe aller Nicht-Nullobjekte
int func(int array[])
  {
   int array_size=ArraySize(array);
   int sum=0;
   for(int i=0;i<array_size; i++)
     {
      if(a[i]==0) continue;
      sum+=a[i];
     }
   return(sum);
  }
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Matrixproduktoperator @

Der MQL5-Operator @ implementiert die Matrixmultiplikation nach den Regeln der linearen Algebra. Er ermöglicht die Multiplikation von Matrizen und Vektoren sowie die Skalarmultiplikation von Vektoren.

Unterstützte Elementtypen:

- float
- double
- complex<float>
- complex<double>

Wichtig: Die Elementtypen der linken und rechten Operanden müssen übereinstimmen.

Anwendungsbeispiele:

1. Matrixmultiplikation (matrix × matrix)

```
matrix A(2, 3);
matrix B(3, 2);
matrix C = A @ B; // Ergebnis: Matrix C mit der Größe [2,2]
```

2. Matrixmultiplikation (matrix × vector)

```
matrix M(2, 3);
vector  V(3);
vector  R = M @ V; // Ergebnis: Vektor R mit 2 Elementen
```

3. Matrixmultiplikation (vector x matrix)

```
matrix M(2, 3);
vector  V(1, 2);
vector  R = V @ M; // Ergebnis: Vektor R mit 3 Elementen
```

4. Skalarmultiplikation (vector × vector)

```
vector V1(1, 3), V2(1, 3);
double r = V1 @ V2; // Ergebnis: Scalar
```

Hinweis

Die Dimensionen müssen den Multiplikationsregeln entsprechen: Anzahl der Spalten im ersten Operanden = Anzahl der Zeilen im zweiten.

Im Falle eines Dimensionsfehlers wird zur Laufzeit ein Fehler ausgelöst.

Vektoren auf der linken Seite gelten als horizontal (1×n).

© 2000-2025, MetaQuotes Ltd.

Vektoren auf der rechten Seite gelten als vertikal (n×1).

Die Priorität der @-Operation entspricht der Priorität der Multiplikation, d. h. für einen Eintrag von D=C+A@B wird zuerst die Matrixmultiplikation T=A@B ausgeführt, gefolgt von der elementweisen Addition D=C+T.

Siehe auch

MatMul, GeMM, Kron, Dot,

© 2000-2025, MetaQuotes Ltd.

Operator der Objekterzeugung new

Operator new erzeugt automatisch das Objekt entsprechender Größe, ruft Objektentwurf auf und gibt Descriptor des erzeugten Objektes zurück. Beim Misserfolg gibt der Operator einen Null-Descriptor zurück, der mit der Konstante NULL verglichen werden kann.

Operator *new* kann nur für Objekte der Klasse verwendet werden, für Strukturen kann sie nicht verwendet werden. Operator new kann nur für Objekte komplizierter Typs verwendet werden (Strukturen und Klassen).

Operator wird nicht für Erzeugung von Objektarrays angewendet. Dafür muss man die Funktion ArrayResize() benutzen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Figurerzeugung                                                   |
//+------------------------------------------------------------------+
void CTetrisField::NewShape()
  {
   m_ypos=HORZ_BORDER;
//--- Zufallsmaessig erzeugen wir eine der 7 moeglicher Figuren
   int nshape=rand()%7;
   switch(nshape)
     {
      case 0: m_shape=new CTetrisShape1; break;
      case 1: m_shape=new CTetrisShape2; break;
      case 2: m_shape=new CTetrisShape3; break;
      case 3: m_shape=new CTetrisShape4; break;
      case 4: m_shape=new CTetrisShape5; break;
      case 5: m_shape=new CTetrisShape6; break;
      case 6: m_shape=new CTetrisShape7; break;
     }
//--- zeichnen wir
   if(m_shape!=NULL)
     {
      //--- Ersteinstellungen
      m_shape.SetRightBorder(WIDTH_IN_PIXELS+VERT_BORDER);
      m_shape.SetYPos(m_ypos);
      m_shape.SetXPos(VERT_BORDER+SHAPE_SIZE*8);
      //--- zeichnen wir
      m_shape.Draw();
     }
//---
  }
```

Es muss bemerkt werden, das der Objektdescriptor kein Speicheranzeiger ist.

Objekt, erzeugt vom Operator new, muss explizit vom Operator delete entfernt werden.

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Operator der Objektenfernung delete

Operator delete enfernt das vom Operator new geschaffene Objekt, ruft Destruktor entsprechender Klasse und befreit den vom Objekt bestzten Speicherplatz. Als Operand wird der gueltiger Beschreiber des wirklichen Objektes verwendet. Nach der Operationsdurchführung delete wird der Objektdescriptor ungueltig.

Beispiel:

```
      //--- entfernen wir die liegende Figur
      delete m_shape;
      m_shape=NULL;
      //--- erzeugen wir die neue Figur
      NewShape();
```

Sehen Sie auch

Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Funktionen

Jeder Task kann in Subtasks geteilt werden, jeder von denen enwede als Kode dargestellt werden kann oder in noch kleinere Subtasks geteilt werden kann. Dieses Verfahren heisst *gestufte Konkretisierung*. Funktionen dienen zur Aufzeichnung des Programmkodes diser zu geloesten Aufgaben. Kode, der beschreibt das, was die Funktion macht, heisst Funktionsdefinition:

```
Funktionstitel
  {
   Anweisungen
  }
```

Alles, was vor der ersten geschweiften Klammer steht, ist der Titel der Funktionsdefinition, und das, was innerhalb der geschweiften Klammern steht, ist *Koerper* der Funkionsdefinition. Funktionstitel besteht aus der Typbeschreibung des Rückgabewertes, dem Namen (Identifikator) und den formellen Parametern . Zahl der Parameter, gegeben der Funktion, ist begrenzt und kann nicht mehr als 64 sein.

Funktion kann aus anderen Progarmmteilen so oft aufgerufen, wie es notwendig ist. Tatsaechlich bilden de Rückgabetyp, Funktionsidentifikator und Parametertypen *Funktionprototyp*.

Funktionsprototyp ist die Erklaerung der Funktion, nicht aber Funktionsdefinition. Wegen der expliziten Erklaerung des Rückgabetyps und der Liste von Argumenttypen, sind beim Zugriff zur Funktion strikte Typenpruefung und implizite Typenumwandlungen moeglich. Besonders oft werden die Funktionserklärungen in Klassen für die Verbesserung des Kodelesbarkeit vewendet.

Funktionsdefinition muss ihrer Erklaerung voellig entsprechen. Jede erklärte Funktion muss definiert werden.

Beispiel:

```
double                          // Typ des Rückgabewertes
linfunc (double a, double b) // Funktionsname und Parameterliste
  {
                                // Zusammengesetzte Anweisung
   return (a + b);              // Rückgabewert
  }
```

Anweisung return kann den Ausdruckswert, der in derselben Anweisung steht, zurückgeben. Ausdruckswert wird in den Typ des Funktionsergebnisses umgewandelt. Man kann einfache Typen, einfache Strukturen, Objektanzeiger rückgeben. Durch den Operator *return* können keine Felder, Klassenobjekte, Variablen der komplizierten Strukturen zurückgegeben werden.

Funktion, die den Wert nicht rückgibt, muss al Funktion mit dem Typ void beschrieben werden.

Beispiel:

```
void errmesg(string s)
  {
   Print("error: "+s);
  }
```

© 2000-2025, MetaQuotes Ltd.

Parameter, gegeben der Funktion, können vorgegebene Werte haben, die von Konstanten des entsprechenden Typs vorgegeben werden.

Beispiel:

```
int somefunc(double a,
             double d=0.0001,
             int n=5,
             bool b=true,
             string s="passed string")
  {
   Print("Pflichtparameter a = ",a);
   Print("Folgende Parameter übertragen werden: d = ",d," n = ",n," b = ",b," s = ",s)
   return(0);
  }
```

Wenn einem Parameter Default-Wert vorgegeben wird, muessen alle folgenden Parameter auch Default-Werte haben.

Beispiel der nicht korrekten Erklaerung :

```
int somefunc(double a,
             double d=0.0001,    // Default-Wert ist erklärt 0.0001
             int n,              // Default-Wert nicht angegeben !
             bool b,             // Default-Wert nicht angegeben !
             string s="passed string")
  {
  }
```

Sehen Sie auch

Überladen, Virtuelle Funktionen, Polymorphismus

© 2000-2025, MetaQuotes Ltd.

Funktionsaufruf

Wenn einer Name, der früher nicht beschrieben wurde, im Ausdruck erscheint, und linke runde Klammer ihm folgt, gilt der nach dem Kontext der Name einer Funktion.

```
Funktionsname (x1, x2,..., xn)
```

Argumente (formelle Parameter) werden nach der Bedeutung übertragen, d.h. jede Bedeutung x1, . . . , xn wird berechnet und der Wert wird der Funktion übertragen. Reihenfolge der Ausdruckberechnungen und Reihenfolge des Wertladens ist nicht garantiert. Waehrend der Durchführung wird die Pruefung der Zahl und des Typs von Argumenten durchgeführt, die der Funktion übertragen wurden. Dieses Verfahren des Funktionszugriffs heisst Wertaufruf.

Funktionsaufruf ist Ausdruck dessen Wert der Wert ist, der von der Funktion zurückgegeben wird. Der beschriebene Funktionstyp muss dem Typ des Rückgabewertes entsprechen. Funktion kann erklärt oder beschrieben werden in jeder Programmstelle auf dem globalen Niveau, d.h. ausser anderer Funktionen. Funktion kann nicht innerhalb der anderen Funktion erklärt oder beschrieben werden.

Beispiele:

```
int start()
  {
   double some_array[4]={0.3, 1.4, 2.5, 3.6};
   double a=linfunc(some_array, 10.5, 8);
   //...
  }
double linfunc(double x[], double a, double b)
  {
   return (a*x[0] + b);
  }
```

Beim Funktionsaufruf mit Default-Werten kann die Liste der übertragenen Parameter vor dem ersten Parameter mit Default-Wert nicht begrenzt werden.

Beispiele:

```
void somefunc(double init,
              double sec=0.0001, //Default-Werte sind bestimmt
              int level=10);
//...
somefunc();                      // falscher Aufruf. der erste Pflichtparameter muss s
somefunc(3.14);                  // richtiger Aufruf
somefunc(3.14,0.0002);           // richtiger Aufruf
somefunc(3.14,0.0002,10);        // richtiger Aufruf
```

Beim Funktionsaufruf darf man nicht Default-Werte auslassen, auch wenn sie Default-Werte haben:

```
somefunc(3.14, , 10);           // falscher Aufruf -> zweiter Parameter ist ausgelasse
```

Sehen Sie auch

überladen, Virtuelle Funktionen, Polymorphismus

© 2000-2025, MetaQuotes Ltd.

Parameterübertragung

Es gibt zwei Verfahren, durch die Computersprache das Argument dem Subprogramm (der Funktion) übertragen kann. Das erste Verfahren - Parameterübertragung nach der Bedeutung. Dieses Verfahren kopiert den Wert von Argument in den formellen Funktionsparameter. Darum haben verschiedene Änderungen dieses Paramerters keinen Einfluss auf das entsprechende Aufrufsargument.

```
//+------------------------------------------------------------------+
//| Parameterübergabe nach dem Wert                                  |
//+------------------------------------------------------------------+
double FirstMethod(int i,int j)
  {
   double res;
//---
   i*=2;
   j/=2;
   res=i+j;
//---
   return(res);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   int a=14,b=8;
   Print("a und b vor dem Aufruf:",a," ",b);
   double d=FirstMethod(a,b);
   Print("a und b nach dem Aufruf:",a," ",b);
  }
//--- Ergebnis der Scriptdurchführung
//  a und b vor dem Aufruf: 14 8
//  a und b nach dem Aufruf: 14 8
```

Zweites Verfahren - Parameterübertragung durch Referenz. In diesem Fall wird der Verweis auf Parameter (nicht aber auf seine Bedeutung_ dem Funktionsparameter übertragen. Innerhalb der Funktion wird der Verweis verwendet, um zum tatsaechlichen Parameter, angegebenen im Aufruf, zu greifen. Das bedeutet, dass Parameteränderungen das Argument beeinflussen werden, das für Funktionsaufruf verwendet wurde.

```
//+------------------------------------------------------------------+
//| Parameterübergabe nach Verweis                                   |
//+------------------------------------------------------------------+
double SecondMethod(int &i,int &j)
  {
   double res;
//---
   i*=2;
   j/=2;
```

© 2000-2025, MetaQuotes Ltd.

```
   res=i+j;
//---
   return(res);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   int a=14,b=8;
   Print("a und b vor dem Aufruf:",a," ",b);
   double d=SecondMethod(a,b);
   Print("a und b nach dem Aufruf:",a," ",b);
  }
//+------------------------------------------------------------------+
//--- Ergebnis der Scriptdurchführung
//  a und b vor dem Aufruf: 14 8
//  a und b nach dem Aufruf: 28 4
```

MQL5 verwendet beide Verfahren, mit Ausnahme von: Felder, Variablen des Typs Strukturen und Klassenobjekte werden immer durch Verweis übertragen. Um Änderungen der tatsaechlichen Parameter auszuschliessen (Argumente, übertragen beim Funktionsaufruf), muss man Zugriffsspezifikator const verwenden.Beim Versuch, Inhalt der Variable zu verändern, die mit dem Spezifikator *const* erklärt wurde, generiert der Compiler einen Fehler.

Bemerkung

Es muss behalten werden, dass Parameter der Funktion umgekehrt übertragen werden, d.h. am Anfang wird der letzte Patameter berechnet und übertragen, dann vorletzte usw. Am letzten wird der Parameter berechnet und übertragen, der am ersten nach den Klammern steht.

Beispiel:

```
void OnStart()
  {
//---
   int a[]={0,1,2};
   int i=0;

   func(a[i],a[i++],"Erste Aufruf(i="+string(i)+")");
   func(a[i++],a[i],"Zweiter Aufruf(i="+string(i)+")");
// Ergebnis:
// Erste Aufruf(i=0) : par1 = 1     par2 = 0
// Zweiter Aufruf(i=1) : par1 = 1     par2 = 1

  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void func(int par1,int par2,string comment)
  {
   Print(comment,": par1 =",par1,"    par2 =",par2);
  }
```

Beim ersten Aufruf im angeführten Beispiel nimmt die Variable i am Anfang an der Zahlenkonkatenanz teil.

```
   "Erster Aufruf(i="+string(i)+")"
```

Dabei verändert sich ihr Wert nicht. Dann beteiligt sich die Variable **i** an der Berechnung des Elementes vom Feld ***a[i++]***, d.h. nach der Entnahme des i-Elementes des Feldes wird die Variable **i** inkrementiert. Und erst danach wird der erste Parameter mit dem veränderten Wert der Variable **i** berechnet.

Im zweiten Aufruf bei der Berechnung aller drei Parameter wird derselbe Wert i verwendet, der beim ersten Funktionsaufruf verändert wurde, und erst nach der Berechnung des ersten Parameters verändert sich die Variable *i* wieder.

Sehen Sie auch

Sichtbereich und Lebensdauer der Variablen, überladen, Virtuelle Funktionen, Polymorphismus

© 2000-2025, MetaQuotes Ltd.

Überladung der Funktionen

Gewoehnlich versucht man im Namen der Funktion ihre Hauptbedeutung darzustellen. In der Regel haben lesbare Programme verschiedenartige und gut passende Identifikatoren. Manchmal werden verschiedene Funktionen für dieselbe Zwecke verwendet. Betrachten wir zB die Funktion, die den Durchschnittwert des Feldes von Zahlen mit Doppelgenauigkeit berechnet und dieselbe Funktion, die aber mit Feldern der Ganzzahlen operiert. Sowie Die erste, als auch die zweite benenne wir AverageFromArray:

```
//+------------------------------------------------------------------+
//| Berechnung des Durchschnittwertes für Feld der Art double        |
//+------------------------------------------------------------------+
double AverageFromArray(const double & array[],int size)
  {
   if(size<=0) return 0.0;
   double sum=0.0;
   double aver;
//---
   for(int i=0;i<size;i++)
     {
      sum+=array[i];    // Addition für double
     }
   aver=sum/size;       // Dividieren wir einfach die Summe durch die Zahl
//---
   Print("Berechnung des Durchschnittwertes für Feld der Art double");
   return aver;
  }
//+------------------------------------------------------------------+
//| Berechnung des Durchschnittwertes für Feld der Art int           |
//+------------------------------------------------------------------+
double AverageFromArray(const int & array[],int size)
  {
   if(size<=0) return 0.0;
   double aver=0.0;
   int sum=0;
//---
   for(int i=0;i<size;i++)
     {
      sum+=array[i];     // Addition für int
     }
   aver=(double)sum/size;// reduzieren wir die Summe zum Typ double und dividieren
//---
   Print("Berechnung des Durchschnittwertes für Feld der Art int");
   return aver;
  }
```

Jede Funktion hat Nachrichtenausgabe mittels der Funktion Print();

```
   Print("Berechnung des Durchschnittwertes für Feld der Art int");
```

© 2000-2025, MetaQuotes Ltd.

Compilierer waehlt die notwendige Funktion der Argumententypen und dern Zahl entsprechend. Regel, nach der diese Wahl durchgeführt wird, heisst *Algorithmus der Signaturübereinstimmung*. Unter Signatur wird die Typenliste verstanden, die in der Funktionserklärung verwendet wird.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   int    a[5]={1,2,3,4,5};
   double b[5]={1.1,2.2,3.3,4.4,5.5};
   double int_aver=AverageFromArray(a,5);
   double double_aver=AverageFromArray(b,5);
   Print("int_aver=",int_aver,"   double_aver=",double_aver);
  }
//--- Ergebnis der Scriptarbeit
// Berecnung des Durchschnittswertes für Feld der Art int
// Berechnung des Durchschnittwertes für Feld der Art double
// int_aver= 3.00000000    double_aver= 3.30000000
```

überladung ist Zuordnung drr mehreren Werte einer Funktion. Die Auswahl des konkreten Wertes hängt von Argumententypen ab, die die Funktion erhalten hat. Bestimmte Funktion wird abhängig davon ausgewaehlt, ob die Liste der Argumente beim Funktionsaufruf der Liste der Parameter in Funktionserkaerung entspricht oder nicht.

Wenn überladene Funktion aufgerufen wird, muss der Compiler Algorithmus für Auswahl notwendiger Funktion haben. Algorithmus, der diese Wahl trifft, hängt von dem Umwandlungstyp ab. Die beste übereinstimmung muss einzigartig sein. Sie muss am besten für wenigstens ein Argument sein und so gut, wie andere übereinstimmungen für alle anderen Argumente sein.

Unten ist Algorithmus der übereinstimmung für jedes Argument angegeben.

Algorithmus der Wahl von überladener Funktion

1. strikte übereinstimmung vewenden (wenn es moeglich ist).
2. Standarderhoehung des Typs probieren.
3. Standardumwandlung des Typs probieren.

Standarderhoehung des Typs ist besser, als andere Standartumwandlungen. Erhoehung ist Umwandlung von float in double, aber auch von bool, char, short oder enum in int. Außerdem gehört zu Standardumwandlunen Umwandlungen der Felder von aehnlichen Ganzzahliger Typen. Ganzzahlige Typen sind: bool, char, uchar, denn alle drei Typen sind ganzzahlige 1-Byte Typen; 2-Byte ganzzahlige Typen short und ushort; 4-Byte ganzzahlige Typen int, unint und color; long, ulong und datetime.

Bestimmt ist strikte übereinstimmung am besten. Für diese übereinstimmung können Typenreduzierungen verwendet werden. Compilierer wird nicht die doppeldeutige Situation schaffen. Darum soll man nicht auf feine Unterschiede unter Typen und explizite Umwandlungen Wert legen, die gestaute Funktion unklar machen.

© 2000-2025, MetaQuotes Ltd.

Wenn Sie zweifeln, verwenden Sie explizite Umwandlungen, um strikte übereinstimmung zu gewaehrleisten.

Beispiele überladener Funktionen in MQL5 können Sie am Beispiel der Funktionen ArrayInitialize() beobachten.

überladungsregeln der Funktion ist für überladung der Klassenmethoden anwendbar.

überladen der Sysemfunktionen ist zulaessig, dabei muss man verfolgen, dass Compiler die noetige Funktion genau waehlen kann. ZB können wir Systemfunktion fmax() mittels der drei vershiedenen Verfahren stauen, aber nur zwei von ihnen werden korrekt sein.

Beispiel:

```
// Stauung ist zulaessig - unterscheidet sich nach der Zahl der Parameter
double fmax(double a,double b,double c);

// Stauung mit Fehler
// Parameterzahl ist verschieden, aber der letzte hat Default-Wert
// das führt zum Kapseln der Systemfunktion beim Aufruf, was unzulaessug ist
double fmax(double a,double b,double c=DBL_MIN);

// normale überladung nach der Parameterzahl
int fmax(int a,int b);
```

Sehen Sie auch

überladen, Virtuelle Funktionen, Polymorphismus

© 2000-2025, MetaQuotes Ltd.

Überladen von Operatoren

Zur leichteren Lesen und Schreiben von Code ist Überladen einiger Operationen erlaubt. Ein Überladenoperator wird mit dem Schlüsselwort Betreiber geschrieben. Die folgenden Operatoren können überladen werden:

- binäre +,-,/,*,%,<<,>>,==,!=,<,>,<=,>=,=,+=,-=,/=,*=,%=,&=,|=,^=,<<=,>>=,&&,||,&,|,^;
- unäre +,-,++,--,!,~;
- Zuweisungsoperator =;
- Indizierungsoperator [].

Überladen von Operatoren ermöglicht den Einsatz des Betriebssystems Notation (geschrieben in der Form von einfachen Ausdrücken) zu komplexen Objekte - Strukturen und Klassen. Schreiben von Ausdrücken mit überladenen Operationen vereinfacht den Blick auf den Quellcode, weil eine komplexere Implementierung verborgen ist.

Betrachten wir zum Beispiel komplexe Zahlen, die aus Real- und Imaginärteil bestehen. Sie sind weit verbreitet in der Mathematik weit verbreitet. Die MQL5 Sprache hat keinen Datentyp für Darstellung der komplexen Zahlen, aber es ist möglich, einen neuen Datentyp in Form einer Struktur oder Klasse zu erstellen. Wir deklarieren die Struktur complex und definieren vier Methoden in dieser Struktur, die die vier arithmetischen Operationen implementieren:

```
//+------------------------------------------------------------------+
//| Struktur für Operationen mit komplexen Zahlen                    |
//+------------------------------------------------------------------+
struct complex
  {
   double            re; // Realteil
   double            im; // Imaginärteil
   //--- Konstruktoren
                     complex():re(0.0),im(0.0) {  }
                     complex(const double r):re(r),im(0.0) {  }
                     complex(const double r,const double i):re(r),im(i) {  }
                     complex(const complex &o):re(o.re),im(o.im) { }
   //--- Rechenoperationen
   complex           Add(const complex &l,const complex &r) const;  // Addition
   complex           Sub(const complex &l,const complex &r) const;  // Subtraktion
   complex           Mul(const complex &l,const complex &r) const;  // Multiplikation
   complex           Div(const complex &l,const complex &r) const;  // Division
  };
```

Jetzt können wir Variablen, die komplexen Zahlen vertreten, in unserem Code deklarieren und mit ihnen arbeiten.

Zum Beispiel:

```
void OnStart()
  {
//--- Deklarieren und initialisieren Variablen eines komplexen Typs
   complex a(2,4),b(-4,-2);
```

© 2000-2025, MetaQuotes Ltd.

```
   PrintFormat("a=%.2f+i*%.2f,   b=%.2f+i*%.2f",a.re,a.im,b.re,b.im);
//--- Addition zweier Zahlen
   complex z;
   z=a.Add(a,b);
   PrintFormat("a+b=%.2f+i*%.2f",z.re,z.im);
//--- Multiplikation zweier Zahlen
   z=a.Mul(a,b);
   PrintFormat("a*b=%.2f+i*%.2f",z.re,z.im);
//--- Division zweier Zahlen
   z=a.Div(a,b);
   PrintFormat("a/b=%.2f+i*%.2f",z.re,z.im);
//---
  }
```

Aber es wäre bequemer für die üblichen arithmetischen Operationen mit komplexen Zahlen die üblichen Operatoren "+", "-", "*" und "/" zu verwenden.

Das Schlüsselwort operator wird verwendet um die Memberfunktion, die eine Typkonvertierung führt, zu definieren. Unäre und binäre Operationen für Variable-Klassenobjekte können als nicht-statische Memberfunktionen überladen werden. Sie implizit auf das Objekt der Klasse wirken.

Die meisten binären Operatoren können wie normale Funktionen, die eine oder beide Argumente als Variable der Klasse oder als ein Zeiger auf ein Objekt dieser Klasse nehmen, überladen werden. Für unseren Typ complex wird Überladen im Deklarieren wie folgt aussehen:

```
   //--- Operatoren
   complex operator+(const complex &r) const { return(Add(this,r)); }
   complex operator-(const complex &r) const { return(Sub(this,r)); }
   complex operator*(const complex &r) const { return(Mul(this,r)); }
   complex operator/(const complex &r) const { return(Div(this,r)); }
```

Ein komplettes Beispiel des Skripts:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- deklarieren und initialisieren Variablen eines komplexen Typs
   complex a(2,4),b(-4,-2);
   PrintFormat("a=%.2f+i*%.2f,   b=%.2f+i*%.2f",a.re,a.im,b.re,b.im);
   //a.re=5;
   //a.im=1;
   //b.re=-1;
   //b.im=-5;
//--- addieren zwei Zahlen
   complex z=a+b;
   PrintFormat("a+b=%.2f+i*%.2f",z.re,z.im);
//--- Multiplikation zweier Zahlen

   z=a*b;
   PrintFormat("a*b=%.2f+i*%.2f",z.re,z.im);
//--- Division zweier Zahlen
   z=a/b;
   PrintFormat("a/b=%.2f+i*%.2f",z.re,z.im);
//---
  }
//+------------------------------------------------------------------+
//| eine Struktur für Operationen mit komplexen Zahlen               |
//+------------------------------------------------------------------+
struct complex
  {
   double            re; // Realteil
   double            im; // Imaginärteil
   //--- Konstruktoren
                     complex():re(0.0),im(0.0) {  }
                     complex(const double r):re(r),im(0.0) {  }
                     complex(const double r,const double i):re(r),im(i) {  }
                     complex(const complex &o):re(o.re),im(o.im) { }
   //--- arithmetische Operationen
   complex           Add(const complex &l,const complex &r) const;  // Addition
   complex           Sub(const complex &l,const complex &r) const;  // Subtraktion
   complex           Mul(const complex &l,const complex &r) const;  // Multiplikation
   complex           Div(const complex &l,const complex &r) const;  // Division
   //--- binäre Operatoren
   complex operator+(const complex &r) const { return(Add(this,r)); }
   complex operator-(const complex &r) const { return(Sub(this,r)); }
   complex operator*(const complex &r) const { return(Mul(this,r)); }
   complex operator/(const complex &r) const { return(Div(this,r)); }
  };
//+------------------------------------------------------------------+
//| Addition                                                         |
//+------------------------------------------------------------------+
complex complex::Add(const complex &l,const complex &r) const
  {
   complex res;
//---
   res.re=l.re+r.re;
   res.im=l.im+r.im;
//--- Ergebnis
   return res;
  }
//+------------------------------------------------------------------+
//| Subtraktion                                                      |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
complex complex::Sub(const complex &l,const complex &r) const
  {
   complex res;
//---
   res.re=l.re-r.re;
   res.im=l.im-r.im;
//--- Ergebnis
   return res;
  }
//+------------------------------------------------------------------+
//| Multiplikation                                                   |
//+------------------------------------------------------------------+
complex complex::Mul(const complex &l,const complex &r) const
  {
   complex res;
//---
   res.re=l.re*r.re-l.im*r.im;
   res.im=l.re*r.im+l.im*r.re;
//--- Ergebnis
   return res;
  }
//+------------------------------------------------------------------+
//| Division                                                         |
//+------------------------------------------------------------------+
complex complex::Div(const complex &l,const complex &r) const
  {
//--- leere komplexe Zahl
   complex res(EMPTY_VALUE,EMPTY_VALUE);
//--- prüfen zu Null
   if(r.re==0 && r.im==0)
     {
      Print(__FUNCTION__+": number is zero");
      return(res);
     }
//--- Hilfsvariable
   double e;
   double f;
//--- Wahl der Rechenvariante
   if(MathAbs(r.im)<MathAbs(r.re))
     {
      e = r.im/r.re;
      f = r.re+r.im*e;
      res.re=(l.re+l.im*e)/f;
      res.im=(l.im-l.re*e)/f;
     }
   else
     {
      e = r.re/r.im;
      f = r.im+r.re*e;
      res.re=(l.im+l.re*e)/f;
      res.im=(-l.re+l.im*e)/f;
     }
//--- Ergebnis
   return res;
  }
```

© 2000-2025, MetaQuotes Ltd.

Die meisten unären Operationen für Klassen können als die üblichen Funktionen, die ein einziges Argument-Klassenobjekt oder einen Zeiger auf es nehmen, überladen werden. Fügen wir eine Überladen unäre Operationen "-" und "!".

```
//+------------------------------------------------------------------+
//| Eine Struktur für Operationen mit komplexen Zahlen               |
//+------------------------------------------------------------------+
struct complex
  {
   double            re;       // Realteil
   double            im;       // Imaginärteil
...
   //--- unäre Operatoren
   complex operator-()  const; // unäres Minus
   bool    operator!()  const; // Negation
  };
...
//+------------------------------------------------------------------+
//| Überladen von Operator "unäres Minus"                            |
//+------------------------------------------------------------------+
complex complex::operator-() const
  {
   complex res;
//---
   res.re=-re;
   res.im=-im;
//--- Ergebnis
   return res;
  }
//+------------------------------------------------------------------+
//| Überladen von Operator "logische Negation"                       |
//+------------------------------------------------------------------+
bool complex::operator!() const
  {
//--- Sind die realen und imaginären Teile der komplexen Zahl gleich Null?
   return (re!=0 && im!=0);
  }
```

Jetzt können wir den Wert einer komplexen Zahl durch Null prüfen und einen negativen Wert erhalten:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- deklarieren und initialisieren Variablen eines komplexen Typs
   complex a(2,4),b(-4,-2);
   PrintFormat("a=%.2f+i*%.2f,   b=%.2f+i*%.2f",a.re,a.im,b.re,b.im);
//--- Division zweier Zahlen z=a/b;
   complex z=a/b;
   PrintFormat("a/b=%.2f+i*%.2f",z.re,z.im);
//--- Eine komplexe Zahl wird standardmäßig Null gleich (im Standardkonstruktor re==0
   complex zero;
   Print("!zero=",!zero);
//--- einen negativen Wert anweisen
   zero=-z;
   PrintFormat("z=%.2f+i*%.2f,  zero=%.2f+i*%.2f",z.re,z.im, zero.re,zero.im);
   PrintFormat("-zero=%.2f+i*%.2f",-zero.re,-zero.im);
//--- prüfen wir zu Null noch einmal
   Print("!zero=",!zero);
//---
  }
```

Bitte beachten Sie, dass wir nicht in diesem Fall den Zuweisungsoperator "=" überladen haben, wie Strukturen von einfachen Typen können eins in andere direkt kopiert werden. So können wir nun den Code für Berechnungen mit komplexen Zahlen in der üblichen Weise schreiben.

Überladen von Index-Operator ermöglicht Erhalten von im Objekt eingeschlossenen Arrays auf einfache und gewohnte Art, und dies trägt auch zu einem besseren Verständnis und Lesbarkeit des Quellcodes. Zum Beispiel müssen wir den Zugang zu einem Symbol im String an der angegebenen Position vorsehen. Ein String in der Sprache MQL5 ist ein eigener Typ string, der nicht ein Array von Symbole ist, sondern mit Hilfe einer überladenen Index-Operation in der generierten Klasse CString können wir eine einfache und transparente Arbeit stellen:

```
//+------------------------------------------------------------------+
//| Klasse, um Symbole im String als Array in Symbole zu zugreifen   |
//+------------------------------------------------------------------+
class CString
  {
   string            m_string;

public:
                     CString(string str=NULL):m_string(str) { }
   ushort operator[] (int x) { return(StringGetCharacter(m_string,x)); }
  };
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Array für Erhalten der Symbole aus einem String
   int     x[]={ 19,4,18,19,27,14,15,4,17,0,19,14,17,27,26,28,27,5,14,
```

© 2000-2025, MetaQuotes Ltd.

```
                  17,27,2,11,0,18,18,27,29,30,19,17,8,13,6 };
   CString str("abcdefghijklmnopqrstuvwxyz[ ]CS");
   string  res;
//--- Bilden einen Satz mit Symbolen aus der Variable str
   for(int i=0,n=ArraySize(x);i<n;i++)
     {
      res+=ShortToString(str[x[i]]);
     }
//--- Ausgabe von Ergebnisse
   Print(res);
  }
```

Ein weiteres Beispiel für ein Überladen der Index-Operation ist Arbeit mit Matrizen. Die Matrix ist ein zweidimensionales dynamisches Array. Größe des Arrays sind nicht im Voraus definiert. So kann man nicht ein Array von Form array[][] ohne Angabe der Größe der zweiten Dimension deklarieren, und dann dieses Array als Parameter übergeben. Eine mögliche Lösung ist eine spezielle Klasse CMatrix, die ein Array von Objekten der Klasse CRow enthält.

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Operationen der Addition und Multiplikation von Matrizen
   CMatrix A(3),B(3),C();
//--- bereiten die Arrays für Strings
   double a1[3]={1,2,3}, a2[3]={2,3,1}, a3[3]={3,1,2};
   double b1[3]={3,2,1}, b2[3]={1,3,2}, b3[3]={2,1,3};
//--- füllen die Matrizen
   A[0]=a1; A[1]=a2; A[2]=a3;
   B[0]=b1; B[1]=b2; B[2]=b3;
//--- Ausgabe von Matrizen in "Experts" Journal
   Print("---- Elemente der Matrix A");
   Print(A.String());
   Print("---- Elemente der Matrix B");
   Print(B.String());

//--- Multiplikation von Matrizen
   Print("---- Multiplikation von Matrizen A und B");
   C=A+B;
//--- Ausgabe der formatierten String-Darstellung
   Print(C.String());

//---Multiplikation von Matrizen
   Print("---- Produkt von Matrizen A und B");
   C=A*B;
   Print(C.String());

//--- jetzt zeigen wir wie Werte im Stil des dynamischen Arrays matrix[i][j] zu erhalt
   Print("die Werte der Matrix C elementweise ausgeben");
//--- Durchschleifen der Strings der Matrix - Objekte CRow
   for(int i=0;i<3;i++)
     {
      string com="| ";
      //--- bilden Strings aus der Matrix für den Wert
      for(int j=0;j<3;j++)
        {
         //--- erhalten ein Element der Matrix aus Nummer von Zeile und Spalte
         double element=C[i][j];// [i] - Zugriff zu CRow im Array m_rows[] ,
                                // [j] - Überladener Index-Operator in CRow
         com=com+StringFormat("a(%d,%d)=%G ; ",i,j,element);
        }
      com+="|";
      //--- Die Werte von der Zeile ausgeben
      Print(com);
     }
  }
//+------------------------------------------------------------------+
//| Klasse "Zeile"                                                   |
//+------------------------------------------------------------------+
class CRow
  {
private:
   double            m_array[];
public:
   //--- Konstruktoren und Destruktor
                     CRow(void)           { ArrayResize(m_array,0);     }
                     CRow(const CRow &r)  { this=r;                     }
                     CRow(const double &array[]);
```

© 2000-2025, MetaQuotes Ltd.

```
                    ~CRow(void){};
   //--- Anzahl der Elementen in der Zeile
   int               Size(void) const    { return(ArraySize(m_array));}
   //--- Gibt den String mit Werte zurück
   string            String(void) const;
   //--- Index-Operator
   double            operator[](int i) const  { return(m_array[i]);   }
   //--- Zuweisungsoperatoren
   void              operator=(const double  &array[]); // Array
   void              operator=(const CRow & r);         // ein anderes Objekt CRow
   double            operator*(const CRow &o);          // Objekt CRow für Multiplikat
  };
//+------------------------------------------------------------------+
//| Konstruktor zur Initialisierung einer Zeile mit einem Array      |
//+------------------------------------------------------------------+
void  CRow::CRow(const double &array[])
  {
   int size=ArraySize(array);
//--- Wenn das Array nicht leer ist
   if(size>0)
     {
      ArrayResize(m_array,size);
      //--- Füllen mit Arrays
      for(int i=0;i<size;i++)
         m_array[i]=array[i];
     }
//---
  }
//+------------------------------------------------------------------+
//| Zuweisungsoperation für das Array                                |
//+------------------------------------------------------------------+
void CRow::operator=(const double &array[])
  {
   int size=ArraySize(array);
   if(size==0) return;
//--- füllen das Array mit Werten
   ArrayResize(m_array,size);
   for(int i=0;i<size;i++) m_array[i]=array[i];
//---
  }
//+------------------------------------------------------------------+
//| Zuweisungsoperation für CRow                                     |
//+------------------------------------------------------------------+
void CRow::operator=(const CRow  &r)
  {
   int size=r.Size();
   if(size==0) return;
//--- füllen das Array mit Werten
   ArrayResize(m_array,size);
   for(int i=0;i<size;i++) m_array[i]=r[i];
//---
  }
//+------------------------------------------------------------------+
//| Operator der Multiplikation auf einer anderen Zeile              |
//+------------------------------------------------------------------+
double CRow::operator*(const CRow &o)
  {
   double res=0;
//--- Prüfung
   int size=Size();
   if(size!=o.Size() || size==0)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print(__FUNCSIG__,": Fehler bei der Multiplikation zweier Matrizen, Die Größe si
       return(res);
      }
//--- multiplizieren die Arrays elementweise und addieren die Produkte
   for(int i=0;i<size;i++)
      res+=m_array[i]*o[i];
//--- Ereignis
   return(res);
  }
//+------------------------------------------------------------------+
//| Gibt die formatierte String-Darstellung zurück                   |
//+------------------------------------------------------------------+
string CRow::String(void) const
  {
   string out="";
//--- wenn die Größe des Arrays mehr als Null ist
   int size=ArraySize(m_array);
//--- arbeiten nur mir Nicht-Null Anzahl der Elemente im Array
   if(size>0)
     {
      out="{";
      for(int i=0;i<size;i++)
        {
         //--- sammeln Werte in einen String
         out+=StringFormat(" %G;",m_array[i]);
        }
      out+=" }";
     }
//--- Ergebnis
   return(out);
  }
//+------------------------------------------------------------------+
//| Klasse "Matrix"                                                  |
//+------------------------------------------------------------------+
class CMatrix
  {
private:
   CRow              m_rows[];

public:
   //--- Konstruktoren und Destruktor
                     CMatrix(void);
                     CMatrix(int rows)  { ArrayResize(m_rows,rows);             }
                    ~CMatrix(void){};
   //--- Erhalten von Größen der Matrix
   int               Rows()       const { return(ArraySize(m_rows));          }
   int               Cols()       const { return(Rows()>0? m_rows[0].Size():0); }
   //--- gibt den Wert der Spalte als String CRow zurück
   CRow              GetColumnAsRow(const int col_index) const;
   //--- gibt einen String mit den Werten der Matrix zurück
   string            String(void) const;
   //--- Index-Operator gibt eine Zeile auf ihrer Nummer zurück
   CRow *operator[](int i) const        { return(GetPointer(m_rows[i]));      }
   //--- Additionoperator
   CMatrix           operator+(const CMatrix &m);
   //--- Multiplikationsoperator
   CMatrix           operator*(const CMatrix &m);
   //--- Zuweisungsoperator
   CMatrix          *operator=(const CMatrix &m);
  };
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Standardkonstruktor, erstellt Array von Strings mit Null-Größe   |
//+------------------------------------------------------------------+
CMatrix::CMatrix(void)
  {
//--- Null Zeilen in der Matrix
   ArrayResize(m_rows,0);
//---
  }
//+------------------------------------------------------------------+
//| Gibt den Wert der Spalte als String CRow zurück                  |
//+------------------------------------------------------------------+
CRow  CMatrix::GetColumnAsRow(const int col_index) const
  {
//--- Variable, um die Werte aus Spalte zu erhalten
   CRow row();
//--- Anzahl der Zeile in der Matrix
   int rows=Rows();
//--- Wenn die Anzahl der Zeilen größer als Null ist, führen die Operation
   if(rows>0)
     {
      //--- Array für Erhalten von Werte der Spalte mit Index col_index
      double array[];
      ArrayResize(array,rows);
      //--- füllen das Array
      for(int i=0;i<rows;i++)
        {
         //--- Prüfung der Spaltennummer für Zeile i, ob es über die Grenzen des Array
         if(col_index>=this[i].Size())
           {
            Print(__FUNCSIG__,": Fehler! Nummer der Spalte ",col_index,"> Größe der Ze
            break; // row bleibt ein nicht initialisiertes Objekt
           }
         array[i]=this[i][col_index];
        }
      //--- erstellen wir eine Zeile CRow basierend auf den Werten des Arrays
      row=array;
     }
//--- Ergebnis
   return(row);
  }
//+------------------------------------------------------------------+
//| Multiplikation zweier Matrizen                                   |
//+------------------------------------------------------------------+
CMatrix CMatrix::operator+(const CMatrix &m)
  {
//--- die Anzahl der Zeilen und Spalten in der übergebene Matrix
   int cols=m.Cols();
   int rows=m.Rows();
//--- Matrix, um das Ergebnis der Addition zu erhalten
   CMatrix res(rows);
//--- Dimensionen der Matrix müssen identisch sein
   if(cols!=Cols() || rows!=Rows())
     {
      //--- kann nicht addieren
      Print(__FUNCSIG__,": Fehler bei der Addition zweier Matrizen, die Größe sind nic
      return(res);
     }
//--- Hilfsarray
   double arr[];
   ArrayResize(arr,cols);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Iteration durch die Zeilen für Addition
   for(int i=0;i<rows;i++)
     {
      //--- wir schreiben das Ergebnis der Addition von Zeilen der Matrix in ein Array
      for(int k=0;k<cols;k++)
        {
         arr[k]=this[i][k]+m[i][k];
        }
      //--- stellen das Array in der Zeile der Matrix
      res[i]=arr;
     }
//--- Ergebnis der Addition zweier Matrizes zurückgeben
   return(res);
  }
//+------------------------------------------------------------------+
//| Multiplikation zweier Matrizen                                   |
//+------------------------------------------------------------------+
CMatrix CMatrix::operator*(const CMatrix &m)
  {
//--- Anzahl der Spalten der ersten Matrix Anzahl der Zeilen in der übergebenen Matrix
   int cols1=Cols();
   int rows2=m.Rows();
   int rows1=Rows();
   int cols2=m.Cols();
//--- Matrix, um das Ergebnis der Addition zu erhalten
   CMatrix res(rows1);
//--- Matrizes sollten abgestimmt werden
   if(cols1!=rows2)
     {
      //--- Kann nicht multiplizieren
      Print(__FUNCSIG__,": Fehler bei der Multiplikation zweier Matrizen, ist das Form
            "- Anzahl der Spalten in den ersten Faktor sollte gleich der Anzahl der Ze
      return(res);
     }
//--- Hilfsarray
   double arr[];
   ArrayResize(arr,cols1);
//--- füllen Zeilen in der Matrix von Multiplikation
   for(int i=0;i<rows1;i++)// iterieren durch die Zeilen
     {
      //--- Array-Empfänger auf Null setzen
      ArrayInitialize(arr,0);
      //--- iterieren Elemente in der Zeile
      for(int k=0;k<cols1;k++)
        {
         //--- Nehmen Werte der Spalte k der Matrix m als ein String CRow
         CRow column=m.GetColumnAsRow(k);
         //--- multiplizieren die beiden Zeilen und schreibt das Ergebnis der Skalarmu
         arr[k]=this[i]*column;
        }
      //--- setzen Array arr[] in der Zeile i der Matrix
      res[i]=arr;
     }
//--- Produkt zweier Matrizen zurückgeben
   return(res);
  }
//+------------------------------------------------------------------+
//| Zuweisungsoperation                                              |
//+------------------------------------------------------------------+
CMatrix *CMatrix::operator=(const CMatrix &m)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- finden und definieren die Anzahl der Zeilen
   int rows=m.Rows();
   ArrayResize(m_rows,rows);
//--- füllen unsere Strings mit den Werten der Zeilen der übergebenen Matrix
   for(int i=0;i<rows;i++) this[i]=m[i];
//---
   return(GetPointer(this));
  }
//+------------------------------------------------------------------+
//| Eine String-Darstellung der Matrix                               |
//+------------------------------------------------------------------+
string CMatrix::String(void) const
  {
   string out="";
   int rows=Rows();
//--- bilden Zeile für Zeile
   for(int i=0;i<rows;i++)
     {
      out=out+this[i].String()+"\r\n";
     }
//--- Ergebnis
   return(out);
  }
```

Sehen Sie auch

Überladen, Arithmetische Operationen, Überladen von Funktionen, Prioritäten und die Reihenfolge der Operationen

© 2000-2025, MetaQuotes Ltd.

Beschreibung der Aussenfunktionen

Die externe Funktionen, die in einem anderen Modul definiert sind, sollen explizit beschrieben werden. Die Beschreibung enthält einen Rückgabetyp, den Funktionsnamen und eine Reihe von Eingabeparametern mit ihren Typen. Fehlen solcher Beschreibung kann zu Fehlern bei der Compilierung, Verknuepfung oder Programmverarbeitung führen. Bei der Beschreibung des Aussenobjektes verwenden Sie Schluesselwort werden *#import* mit dem Modulandeutung.

Beispiele:

```
#import "user32.dll"
  int     MessageBoxW(int hWnd ,string szText,string szCaption,int nType);
  int     SendMessageW(int hWnd,int Msg,int wParam,int lParam);
#import "lib.ex5"
  double  round(double value);
#import
```

Mittels des Imports kann man sehr einfach Funktionen beschreiben, aufgerufen von externen DLL oder kompilierten EX5 Bibliotheken. Bibliotheken EX5 sind kompilierte ex5-Dateien, die die Eigenschaft library haben. Aus EX5 Bibliotheken können nur Funktionen importiert werden, die mit dem Modifikator export beschrieben werden.

Bei der gemeinsame Nutzung der DLL und EX5-Bibliotheken erinnern Sie daran, dass die Bibliotheken verschiedene Namen (unabhängig von Platzierung ihrer Verzeichnisse) haben müssen. Alle importierten Funktionen erhalten Scope "Dateiname" der Bibliothek.

Beispiel:

```
#import "kernel32.dll"
   int GetLastError();
#import "lib.ex5"
   int GetLastError();
#import

class CFoo
  {
public:
   int            GetLastError() { return(12345); }
   void           func()
     {
      Print(GetLastError());           // Aufruf der Klassenmethode
      Print(::GetLastError());         // Aufruf der MQL5-Funktion
      Print(kernel32::GetLastError()); // Aufruf der Funktion kernel32.dll
      Print(lib::GetLastError());      // Aufruf der Funktion lib.ex5
     }
  };

void OnStart()
  {
   CFoo foo;
   foo.func();
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Sehen Sie auch

überladen, Virtuelle Funktionen, Polymorphismus

© 2000-2025, MetaQuotes Ltd.

Exportieren der Funktionen

Es gibt die Moeglichkeit im mql5-Programm die Funktion zu benutzen, erklärt im anderen mql5-Programm mit dem Postmodifikator *export*. Diese Funktion heisst Exportfunktion und sie kann aus anderen Programmen nach der Compilierung aufgerufen werden.

```
int Function() export
  {
  }
```

Dieser Modifikator befiehlt dem Compiler, die Funktion in die Tabelle der EX5-Funktionen einzutragen, die von der ex5 Dstei exportiert werden. Nur die Funktionen mit diesem Modifikator werden aus anderen mql5-Programmen zugaenglich ("sichtbar").

Eigenschaft library deutet dem Compiler hin, dass die vorliegende EX5-Datei Bibliothek sein wird und Compilierer stellt das in Titel EX5.

Alle Funktionen geplant als Exportfunktionen muessen von dem Modifikator *export* vermerkt werden.

Sehen Sie auch

überladen, Virtuelle Funktionen, Polymorphismus

© 2000-2025, MetaQuotes Ltd.

Ereignisbearbeiter

In der Sprache MQL5 ist die Bearbeitung einiger vorbestimmten Ereignissen vorausgesehen. Funktionen für die Bearbeitung dieser Ereignisse müssen im Programm MQL5 bestimmt werden; Funktionsname, Typ des Rückgabewertes, Parametervorrat (wenn sie vorhanden sind) und ihre Typen müssen der Beschreibung der Funktion-Bearbeiter des Ereignisses strikt entsprechen.

Nämlich nach dem Typ des Rückgabewertes und Typen der Parameter identifiziert der Bearbeiter von Ereignissen des Client-Terminals die Funktionen, die verschiedene Ereignisse bearbeiten. Wenn die entsprechende Funktion andere angegebene Parameter hat, die den folgenden Beschreibungen nicht entsprechen oder ein anderer Typ des Rückgabewertes angegeben wird, wird diese Funktion für die Bearbeitung des Ereignisses nicht benutzt werden.

OnStart

Funktion OnStart() ist der Bearbeiter des Ereignisses Start, wird **nur** für ablaufende **Scripts** automatisch generiert. Muss den Typ void haben, hat keine Parameter:

```
void OnStart();
```

für die Funktion OnStart() ist es zugänglich, den Typ des Rückgabewertes int anzugeben.

OnInit

Funktion OnInit() ist der Bearbeiter des Ereignisses Init. Kann den Typ void oder int, haben, hat keine Parameter:

```
void OnInit();
```

Ereignis Init wird sofort nach Der Expert- oder Indikatorladung generiert werden, für Scripts wird dieses Ereignis nicht generiert. Funktion OnInit() wird für die Initialisierung verwendet. Wenn OnInit() Rückgabewert des Typs int hat, bedeutet der Nichtnullrückgabekode verfehlte Initialisierung und generiert Ereignis Deinit mit dem Kode des Deinitialisierungsgrundes REASON_INITFAILED.

Wenn INIT_FAILED zurückgegeben wird, wird der EA zwangsweise aus dem Chart entfernt.

Wenn INIT_FAILED zurückgegeben wird, wird der Indikator nicht aus dem Chart entfernt. Der im Chart verbleibende Indikator ist nicht funktionsfähig — die Ereignisbehandlung im Indikator wird nicht aufgerufen.

Für Optimierung von Eingabeparametern wird es empfohlen Werte aus ENUM_INIT_RETCODE als Rückgabecode zu verwenden. Diese Werte werden für die Organisation der Verlauf der Optimierung, einschließlich der Auswahl der am besten geeigneten Testagenten eingesetzt. Während der Initialisierung eines Expert Advisor vor Beginn des Tests können Sie Informationen über die Konfiguration und die Ressourcen eines Agenten (die Anzahl der Kerne, Menge des freien Speichers, etc.) unter Verwendung des TerminalInfoInteger()-Funktion anfordern. Basierend auf diesen Informationen können Sie entweder Verwendung des Agenten erlauben oder ablehnen bei der Optimierung dieses Expert Advisors.

ENUM_INIT_RETCODE

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
INIT_SUCCEEDED	Erfolgreiche Initialisierung, Test des Expert Advisor kann fortgesetzt werden. Dieser Code bedeutet das gleiche wie ein Null-Wert - der Expert Advisor wurde erfolgreich im Tester initialisiert.
INIT_FAILED	Initialisierung fehlgeschlagen, es gibt keinen Sinn den Test fortzusetzen wegen der fatale Fehler. Zum Beispiel konnte nicht einen Indikator erstellen, der für die Arbeit des Expert Advisors erforderlich ist. Dieser Rückgabewert bedeutet das gleiche wie ein anderer Wert als Null - Initialisierung des Expert Advisors im Tester gescheitert.
INIT_PARAMETERS_INCORRECT	Mit diesem Wert kann der Programmierer einen falschen Satz von Eingabeparameter beziehen. Auf die gesamte Optimierung-Tabelle wird die Zeile des Ergebnisses diesem Return-Code in rot hervorgehoben. Test für die gegebenen Satz von Parametern des Expert Advisors nicht ausgeführt wird, ist der Agent frei, um eine neue Aufgabe zu erhalten. Nach Erhalt dieses Wert wird Strategie Tester garantiert diese Aufgabe an andere Agenten für erneut Ausführen nicht übertragen.
INIT_AGENT_NOT_SUITABLE	Es gibt keine Fehler während der Initialisierung, aber aus irgendeinem Grund ist der Agent nicht geeignet für Test. Zum Beispiel, nicht genug Speicher, kein OpenCL-Unterstützung etc. Nach der Rückkehr von diesem Code wird der Agent Aufgaben bis zum Ende dieser Optimierung nicht empfangen.

Funktion OnInit() des Typs void bedeutet immer erfolgreiche Initialisierung.

OnDeinit

Funktion OnDeinit() wird bei der Deinitialisierung aufgerufen und ist der Bearbeiter des Ereignisses Deinit . Muss mit dem Typ void erklärt werden und einen Parameter des Typs const int haben, der Kode des Deinitialisierungsgrundes hat. Wenn ein anderer Typ erklärt wird, gibt... Für Scripts wird das Ereignis Deinit nicht generiert, darum darf man nicht die Funktion OnDeinit() in Scripts verwenden.

```
void OnDeinit(const int reason);
```

Ereignis Deinit wird für Experten und Indikatoren in folgenden Fällen generiert:

- vor Uminitialisierung wegen der Änderung des Symbols oder Chartperiode, zu dem mql5-Programm angeschlossen ist;
- vor Neuinitialisierung wegen der Veranderung von Inputparametern;
- vor Entladung des mql5-Programms.

OnTick

© 2000-2025, MetaQuotes Ltd.

Ereignis NewTick wird nur **für Experten generiert** beim Eingang des neuen Ticks nach Symbol, zu dessen Chart Expert angeschlossen ist. Es ist sinnlos, die Funktion OnTick() im Benutzerindikator oder Script zu bestimmen, denn das Ereignis NewTick wird für sie nicht generiert.

Ereignis Tick wird nur für Experten generiert, das bedeutet aber nicht, dass Experten die Funktion OnTick() haben muessen, denn für Experten werden nicht nur Ereignisse NewTick, sondern auch Ereignisse Timer, BookEvent und ChartEvent generiert. Muss mit dem Typ void, erklärt werden hat keine Parameter:

```
void OnTick();
```

OnTimer

Funktion OnTimer() wird beim Erscheinen des Ereignisses Timer aufgerufen, das vom Systemtimer nur für Experten und Anzeiger generiert wird - in Skripts kann sie nicht verwendet werden. Funktion OnTimer() wird nur beim Eintritt des Ereignisses Timer aufgerufen, das vom Systemtimer nur für Experten generirt wird - man muss sie nicht in Scripts oder Indikatoren verwenden. Periode des Eintritt dieses Ereignisses wird bei der Subskription auf Benachrichtigungsempfang über dieses Ereignis durch die Funktion EventSetTimer() eingesetzt.

Annulieren der Aufnahme der Timerereignisse für jeden konkreten Experten erfolgt durch die Funktion EventKillTimer(). Funktion muss mit dem Typ void bestimmt werden, hat keine Parameter:

```
void OnTimer();
```

Es ist empfehlenswert, die Funktion EventSetTimer() in der Funktion OnInit() einmalig aufzurufen, und Die Funktion EventKillTimer() einmalig in der Funktion OnDeinit() aufzurufen.

Jeder Expert und Anzeiger arbeitet nur mit seinem Timer und erhaelt Ereignisse nur von ihm. Wenn das mql5-Programm zu funktionieren stoppt, wird Timer zwangslaeufig entfernt, wenn er durch die Funktion EventKillTimer() erzeugt aber nicht entfernt wurde.

OnTrade

Funktion wird beim Eintritt von Trade aufgerufen, das bei der Änderung der Liste der aufgestellten Order und offenen Positionen, Ordergeschichte und Dealgeschichte entsteht. Bei jeder Handelshandlung (Aufstellung de zurückgesetzten Order, Öffnung/Schliessung der Position, Einstellung von Stops, Ausloesung der zurückgesetzten Ordern usw.) verändert sich entsprechenderweise die Geschichte der Ordern und Deals und/oder Liste der Positionen und laufender Ordern.

```
void OnTrade();
```

Benutzer muss selbsstaendig, Pruefung des Standes vom Handelskonto beim Erhalten dieses Ereignisses zu realisieren (wenn es nach Bedingungen der Handelsstrategie notwendig ist). Wenn der Funktionsaufruf OrderSend() erfolgreich beendet und den Wert true zurückgibt - bedeutet es, dass Handelsserver Order in Warteschlange eingereiht hat und ihr die Ticketnummer zugeordnet hat. Sobald der Server diesen Befehl bearbeitet, wird das Ereignis Trade generiert werden. Wenn der Benutzer den Ticketwert im Kopf behaelt, kann er bei der Bearbeitung des Ereignisses OnTrade() aufklaeren, was mit der Order geschehen ist.

OnTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Als Ergebnis der Durchführung von bestimmten Aktionen auf dem Handelskonto, ändert sich der Kontostand. Zu solchen Aktionen gehören:

- Senden der Handelsanfrage aus einer beliebigen MQL5-Anwendung in dem Client-Terminal mittels der Funktionen OrderSend und OrderSendAsync und ihre nachfolgende Ausführung;
- Senden der Handelsanfrage über die grafische Oberfläche des Terminals und ihre nachfolgende Ausführung;
- Auslösen von aufgeschobenen Ordern und Stop-Ordern auf dem Server;
- Ausführen von Operationen auf der Seite der Handel-Server.

Als Ergebnis dieser Aktionen werden die folgenden Handelstransaktionen durchgeführt:

- Die Verarbeitung der Handelsanfrage;
- Änderung der offenen Aufträge;
- Änderung der Ordergeschichte;
- Änderung der Dealgeschichte;
- Änderung der Position.

Zum Beispiel, beim Senden Marktkauforder, wird Marktkauforder behandelt und eine entsprechende Marktkauforder für das Konto erstellt. Ist die Order durchgeführt, wird sie aus der Auftragsliste entfernt und zur Ordergeschichte hinzugefügt. Dann wird der entsprechende Deal zur Geschichte hinzugefügt und eine neue Position wird erstellt. Alle diese Aktionen sind Handelstransaktionen. Ankunft jeder solchen Transaktion im Terminal ist ein Ereignis TradeTransaction. Es ruft den Handler OnTradeTransaction auf

```
void  OnTradeTransaction(
   const MqlTradeTransaction&    trans,        // Struktur der Handelstransaktion
   const MqlTradeRequest&        request,      // Struktur der Anfrage
   const MqlTradeResult&         result        // Struktur der Antwort
   );
```

Der Handler enthält drei Parameter:

- **trans** - in diesen Parameter wird die Struktur MqlTradeTransaction übergeben, die die auf das Handelskonto angewendete Handelstransaktion beschreibt;
- **request** - in diesen Parameter wird die Struktur MqlTradeRequest übergeben, die eine Handelsanfrage beschreibt;
- **result** - in diesen Parameter wird die Struktur MqlTradeResult übergeben, die das Ergebnis der Ausführung einer Handelsanfrage beschreibt.

Zwei letzte Parameter **request** und **result** werden mit den Werten nur für Transaktion des Typs TRADE_TRANSACTION_REQUEST ausgefüllt, die Information über Transaktion kann man aus dem Parameter *type* der Variable **trans** erhalten. Beachten Sie, dass in diesem Fall das Feld *request_id* in der Variable **result** den Identifikator der Handelsanfrage **request** enthält, als Ergebnis dessen Ausführung die in der Variable trans beschriebene **Handelstransaktion** durchgeführt wurde. Das Vorhandensein des Identifikators der Anfrage erlaubt die ausgeführte Aktion (Aufruf der Funktion OrderSend oder OrderSendAsync) mit dem in die OnTradeTransaction() übergebenen Ergebnis dieser Aktion zu verbinden.

Eine Handelsanfrage, die aus dem Terminal manuell oder durch die Handelsfunktionen OrderSend()/OrderSendAsync() gesendet ist, kann mehrere konsequente Handelstransaktionen auf dem Handel-Server auslösen. Dabei ist die Reihenfolge des Eingangs dieser Transaktionen in das

© 2000-2025, MetaQuotes Ltd.

Terminal nicht gewährleistet, deshalb darf man seinen Handelsalgorithmus auf der Erwartung des Eingangs einiger Handelstransaktionen nach der Ankunft der anderen nicht bauen.

- Alle Typen der Handelstransaktionen werden in der Enumeration ENUM_TRADE_TRANSACTION_TYPE beschrieben.
- Die Struktur MqlTradeTransaction, die eine Handelstransaktion beschreibt, wird auf unterschiedlicher Weise je nach Typ der Transaktion ausgefüllt. Zum Beispiel, für Transaktionen des Typs TRADE_TRANSACTION_REQUEST muss nur type Feld (Typ der Handelstransaktion) analysiert werden. Für zusätzliche Informationen muss man die zweiten und dritten Parameter der Funktion OnTradeTransaction (request und result) analysieren. Weitere Informationen finden Sie im Abschnitt "Struktur der Handelstransaktion".
- In der Beschreibung der Handelstransaktion werden nicht alle verfügbaren Informationen über Aufträge, Deals und Positionen (zum Beispiel, Kommentar) übergeben. Für erweiterte Informationen sollten die Funktionen OrderGet *, HistoryOrderGet *, HistoryDealGet * und PositionGet * verwendet werden.

Nach der Anwendung der Handelstransaktionen auf das Kundenkonto werden sie in die Warteschlange der Handeltransaktionen des Terminals konsequent gestellt, woher in den Eingangspunkt OnTradeTransaction in der Reihenfolge der Ankunft in das Terminal schon konsequent übergeben werden.

Während der Verarbeitung der Handelstransaktionen von einem Expert Advisor mit Hilfe des Handlers OnTradeTransaction bearbeitet das Terminal neu ankommenden Handelstransaktionen weiter. Somit kann sich der Zustand des Handelskontodes während des OnTradeTransaction schon ändern. Zum Beispiel, während das MQL5-Programm ein Ereignis des Hinzufügens einer neuen Order bearbeitet, kann sie ausgeführt werden, aus der offenen Auftragsliste entfernt werden und in die Geschichte übertragen werden. Im Folgenden wird das Programm über alle diesen Ereignisse benachrichtigt sein.

Länge der Warteschlange umfasst 1024 Elemente. Falls OnTradeTransaction die nächste Transaktion zu lange bearbeiten wird, können die alten Transaktionenin in der Warteschlange durch die neueren verdrängt sein.

- Im Allgemeinen gibt es kein genaues Verhältnis nach der Anzahl der Aufrufe OnTrade und OnTradeTransaction. Ein Anruf OnTrade entspricht einem oder mehreren Aufrufen OnTradeTransaction.
- OnTrade wird nach den entsprechenden Aufrufen OnTradeTransaction aufgerufen.

OnTester

Funktion OnTester() ist Bearbeiter des Ereignisses Tester, das nach Abschluss des historischen Testen des Experten für das vorgegebene Dateninterval. Funktion muss mit dem Typ void bestimmt werden, hat keine Parameter:

```
double OnTester();
```

Funktion wird unmittelbar vor dem Aufruf der Funktion OnDeinit() aufgerufen und hat den Typ des Rückgabewertes double. Funktion OnTester() kann nur in Experten beim Testen verwendet werden und ist in erster Linie für Berechnung bestimmtes Wertes bestimmt, der als Kriterium Custom max bei genetischer Optimierung der Eingabeparameter verwendet.

© 2000-2025, MetaQuotes Ltd.

Bei genetischer Optimierung wird Sortieren der Ergebnisse innerhalb einer Generation absteigend durchgeführt. D.h. die besten Ergebnisse vom Standpunkt des Optimierungskriterium sind die mit dem größten Wert (für Optimierungkriterium Custom max werden Werte beachtet, die durch die Funktion OnTester zurückgegeben werden). Die schlechtesten Werte bei diesem Sortieren werden am Ende gestellt und danach weggeworfen und an Formieren der nächsten Generation nicht beteiligen.

OnTesterInit

Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterInit auftritt, um notwendige Aktionen vor der Optimierung im Strategie-Tester durchzuführen. Es gibt zwei Versionen dieser Funktion.

Die **Version, die das Ergebnis zurück gibt**

```
int  OnTesterInit(void);
```

Rückgabewert

Vom Typ int, Null bedeutet die erfolgreiche Initialisierung eines auf einem Chart gestarteten EAs vor dem Start der Optimierung.

Der Aufruf OnTesterInit(), der das Ausführungsergebnis zurückgibt, wird zur Verwendung empfohlen, da er nicht nur die Programminitialisierung erlaubt, sondern auch einen Fehlercode im Falle eines frühen Optimierungsstopps zurückgibt. Die Rückgabe eines anderen Wertes als INIT_SUCCEEDED (0) bedeutet einen Fehler, es wird keine Optimierung gestartet.

Die Version ohne Ergebnisrückgabe wird nur aus Kompatibilitätsgründen mit alten Codes belassen. Ein Verwenden wird nicht empfohlen.

```
void  OnTesterInit(void);
```

Ein Expert Advisor, der den Handler OnTesterDeinit() oder OnTesterPass() hat, wird beim Start der Optimierung automatisch auf einem gesonderten Chart des Terminals geladen mit dem Symbol und der Periode, die im Tester angegeben wurden, und ein Ereignis TesterInit empfängt. Die Funktion soll einen Expert Advisor vor Optimierung initialisieren um dann die Ergebnisse der Optimierung zu verarbeiten.

OnTesterPass

Die Funktion OnTesterPass() ist ein Eventhandler von TesterPass, das automatisch vor dem Start der Optimierung eines Expert Advisors generiert wird. Die Funktion muss mit dem Typ void definiert werden, ohne Parameter:

```
void OnTesterPass();
```

Ein Expert Advisor mit Handler OnTesterPass() wird automatisch auf einem gesonderten Chart des Terminals geladen mit dem für ein Test angegebenen Symbol und der Periode, und erhält Ereignisse von TesterPass beim Erhalten von einem Frame bei der Optimierung. Die Funktion wird für die dynamische Verarbeitung von Optimierungsergebnisse direkt "on the fly" ausgelegt, ohne für die Vollendung der Optimierung zu warten. Frames werden durch Funktion FrameAdd() hinzugefügt. Diese Funktion kann nach einem Durchgang im Handler OnTester() aufgerufen werden.

OnTesterDeinit

© 2000-2025, MetaQuotes Ltd.

Die Funktion Boxing Day() ist ein Eventhandler von TesterDeinit, das automatisch nach der Optimierung eines Expert Advisors generiert wird. Die Funktion muss mit dem Typ void definiert werden, ohne Parameter:

```
void OnTesterDeinit();
```

Ein Expert Advisor mit dem Handler TesterDeinit() wird automatisch auf dem Chart beim Start der Optimierung geladen, und erhält ein Ereignis TesterDeinit nach ihrer Fertigstellung. Die Funktion wird für die abschließende Bearbeitung aller Optimierungsergebnisse verwendet.

OnBookEvent

Funktion OnBookEvent() ist der Bearbeiter von Ereignis BookEvent. Ereignis BookEvent wird nur für Experten und Anzeiger bei der Veränderung von DOM (Depth of Market generiert). Muss den Typ void und einen Parameter des Typs string haben:

```
void OnBookEvent (const string& symbol);
```

Um Ereignisse BookEvent für jedes Symbol zu bekommen, reicht es, voraussichtlich auf Erhalten dieser Ereignisse für dieses Symbol durch die Funktion MarketBookAdd() zu subskribieren. Um die Subskription auf Erhalten des Ereignisses BookEvent für konkretes Symbol zu annullieren, muss man die Funktion MarketBookRelease() aufrufen.

Zum Unterschied von den anderen Ereignissen, ist das Ereignis BookEvent broadcast. Das bedeutet, wenn ein Expert Auf Erhalten des Ereignisses BookEvent durch die Funktion MarketBookAdd subskribiert, werden alle anderen Experten mit Bearbeiter OnBookEvent() dieses Ereignis bekommen. Darum muss der Symbolname, der in Bearbeiter als Parameter *const string& symbol* übertragen wird, analysieren.

OnChartEvent

OnChartEvent() ist der Bearbeiter von Gruppe der Ereignisse ChartEvent:

- CHARTEVENT_KEYDOWN — Ereignis des Keyboardbetätigens, wenn das Chartfenster im Fokus steht;
- CHARTEVENT_MOUSE_MOVE — Ereignisse der Maus-Bewegung und Mausklicks (wenn die Eigenschaft CHART_EVENT_MOUSE_MOVE=true ist eingegeben);
- CHARTEVENT_OBJECT_CREATE — Ereignis der Erzeugung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_CREATE=true ist eingegeben);
- CHARTEVENT_OBJECT_CHANGE — Veranderung der Objekteigenschaften durch Dialog der Eigenschaften;
- CHARTEVENT_OBJECT_DELETE — Ereigniss der Entfernung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_DELETE=true ist eingegeben);
- CHARTEVENT_OBJECT_CLICK — Ereignis des Mausklickens auf dem graphischen Objekt, das zum Chart gehört;
- CHARTEVENT_OBJECT_DRAG — Ereignis der übertragung des graphischen Objekts mit der Maus;
- CHARTEVENT_OBJECT_ENDEDIT — Ereignis des Abschlusses von Texteditieren im Eingabefeld des graphischen Objekts LabelEdit;
- CHARTEVENT_CHART_CHANGE — Ereignis von Änderungen des Charts;
- CHARTEVENT_CUSTOM+n — Identifikator des Benutzerereignisses, wo sich n im Bereich von 0 bis 65535 befindet.

© 2000-2025, MetaQuotes Ltd.

- CHARTEVENT_CUSTOM_LAST — letzter zugaenglicher Identifikator des Benutzerereignisses (CHARTEVENT_CUSTOM+65535).

Funktion kann nur in Experten und Indikatoren aufgerufen werden und muss den Typ void und 4 Parameter haben:

```
void OnChartEvent(const int id,          // Identifikator des Ereignisses
                  const long& lparam,    // Parameter des Ereignisses des Typs long
                  const double& dparam, // Parameter des Ereignisses des Typs double
                  const string& sparam  // Parameter des Ereignisses des Typs string
  );
```

Für jeden Typ des Ereignisses haben Eingabeparameter der Funktion OnChartEvent() vorbestimmte Werte, die für Verarbeitung dieses Ereignisses notwendig sind. In der Tabelle sind Ereignisse und Werte aufgezählt, die durch Parameter übertragen werden.

Ereignis	Wet des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
Ereignis des Keyboard Klickens	CHARTEVENT_KEYDOWN	Kode der betätigten Taste	Die Anzahl der Tastenanschläge, die generiert werden, während diese Taste in gedrücktem Zustand war	Der String-Wert eines Bit-Maske, die den Status der Tastatur-Tasten beschreibt
Ereignisse der Maus-Bewegung und Mausklicks (wenn die Eigenschaft CHART_EVENT_MOUSE_MOVE=true ist für das Chart eingegeben)	CHARTEVENT_MOUSE_MOVE	X-Koordinate	Y-Koordinate	Der String-Wert einer Bit-Maske, der den Status der Maustasten beschreibt
Ereignis der Erzeugung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_CREATE=true ist für ein Chart eingegeben)	CHARTEVENT_OBJECT_CREATE	—	—	Name des erzeugten graphischen Objektes
Ereignis der Veränderung der Eigenschaften des Objektes	CHARTEVENT_OBJECT_CHANGE	—	—	Name des veränderten graphischen Objektes

© 2000-2025, MetaQuotes Ltd.

Ereignis	Wet des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
durch Dialog der Eigenschaften				
Ereignis der Entfernung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_DELETE=true ist für ein Chart eingegeben)	CHARTEVENT_OBJECT_DELETE	—	—	Name des entfernten graphischen Objektes
Ereignis des Mausklicks auf der graphischen Darstellung	CHARTEVENT_CLICK	X-Koordinate	Y-Koordinate	—
Ereignis des Mausklicks auf dem graphischen Objekt	CHARTEVENT_OBJECT_CLICK	X-Koordinate	Y-Koordinate	Name des graphischen Objektes, auf dem das Ereignis geschehen ist
Ereignis der Verschiebung des graphischen Objektes per Mausklick	CHARTEVENT_OBJECT_DRAG	—	—	Name des verschobenen graphischen Objektes
Ereignis der Beendung des Texteditierens im Eingabefeld des graphischen Objektes "Eingabefeld"	CHARTEVENT_OBJECT_ENDEDIT	—	—	Name des graphischen Objektes "Eingabefeld", in dem Editieren des Textes beendet wurde
Ereignis der Chart-Veränderung	CHARTEVENT_CHART_CHANGE	—	—	—
Benutzerereignis mit Nummer N	CHARTEVENT_CUSTOM+N	Wert, vorgegeben durch die Funktion EventChartCustom()	Wert, vorgegeben durch die Funktion EventChartCustom()	Wert, vorgegeben durch die Funktion EventChartCustom()

© 2000-2025, MetaQuotes Ltd.

OnCalculate

Funktion OnCalculate() wird nur in Benutzerindikatoren aufgerufen werden bei der Notwendigkeit, Berechnung der Indikatorenwerte am Ereignis Calculate durchzuführen. Gewöhnlich erfolgt es bei der Aufnahme des neuen Ticks für Symbol, für das der Indikator berechnet wird. Dabei braucht der Indikator nicht zum einen Preischart dieses Symbols angehängt werden.

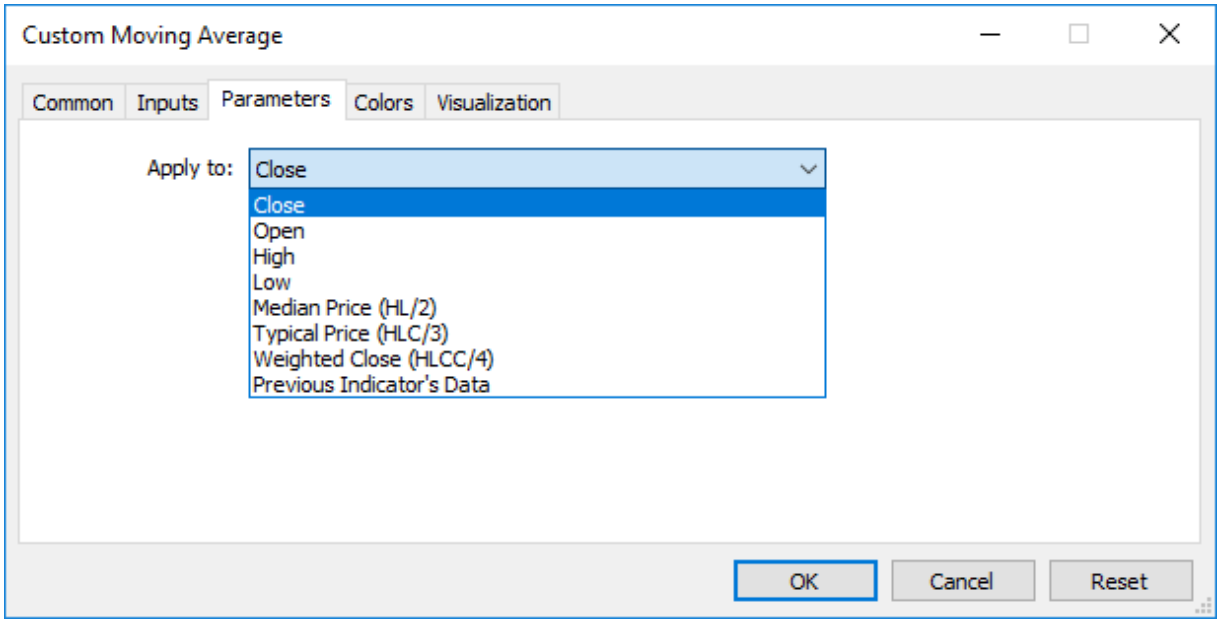
Funktion OnCalculate() muss den Typ des Rückgabewertes int haben. Es gibt zwei Definitionsvarianten. Innerhalb eines Indikators können beide Varianten der Funktion nicht verwendet werden.

Die erste Aufrufform ist für die Indikatoren bestimmt, die auf einem Datenpuffer berechnet werden können. Das Beispiel dieses Indikators - Custom Moving Average.

```
int OnCalculate (const int rates_total,      // Feldgroesse price[]
                 const int prev_calculated,  // verarbeitete Bars im früheren Aufruf
                 const int begin,            // woher bedeutsame Daten anfangen
                 const double& price[]       // Feld für Berechnung
   );
```

Als Feld price[] kann eine der Preistimeserien oder berechneter Buffer irgendwelches Anzeigers übertragen werden. Um Indizierenrichtung im Feld price[] zu bestimmen, muss man die Funktion ArrayGetAsSeries() aufrufen. Um von Default-Werten nicht abzuhängen, muss man die Funktion ArraySetAsSeries() für die Felder, mit denen es zu arbeiten zu erwarten, aufrufen.

Auswahl der notwendigen Timeserie oder des Indikators als Feld price[] macht der Benutzer beim Start des Indikators auf Registerblatt "Parameters". Dafür muss das notwendige Element in der drop-down Liste des Feldes "Apply to" angegeben werden.



für Erhalten der Werte des Benutzerindikators aus anderen mql5-Programmen wird die Funktion iCustom() verwendet, , die handle des Indikators für folgende Operationen rückgibt. Dabei kann auch das notwendige Feld price[] oder handle des anderen Indikators angegeben werden. Dieser Parameter muss als letzte in der Liste von Variablen des Benutzerindikators übertragen werden.
Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//---
   string terminal_path=TerminalInfoString(STATUS_TERMINAL_PATH);
   int handle_customMA=iCustom(Symbol(),PERIOD_CURRENT, "Custom Moving Average",13,0,
   if(handle_customMA>0)
      Print("handle_customMA = ",handle_customMA);
   else
      Print("Cannot open or not EX5 file '"+terminal_path+"\\MQL5\\Indicators\\"+"Cust
  }
```

In diesem Beispiel ist der letzte Parameter der Wert PRICE_TYPICAL (aus Enumeration ENUM_APPLIED_PRICE), der deutet, dass Benutzerindikator auf typischen Preisen gebaut werden wird, erhalten als (High+Low+Close)/3. Wenn der Parameter nicht angegeben wird, wird Indikator auf Grudlage der Werte PRICE_CLOSE gebaut,d.h. auf Schlusspreisen jede Bar.

Ein anderes Beispiel das übergabe des handles des Indikators als letzter Parameter demonstriert für Festlegung des Feldes price[], ist in der Funktionbeschreibung iCustom() angegeben.

Die zweite Aufrufform ist für alle anderen Indikatoren bestimmt, die für Berechnung mehr als eine Zeitreihe verwenden.

```
int OnCalculate (const int rates_total,      // Größe der Eingabezeitreihen
                 const int prev_calculated,  // Bars, verarbeitete im früheren Aufruf
                 const datetime& time[],     // Time
                 const double& open[],       // Open
                 const double& high[],       // High
                 const double& low[],        // Low
                 const double& close[],      // Close
                 const long& tick_volume[],  // Tick Volume
                 const long& volume[],       // Real Volume
                 const int& spread[]         // Spread
   );
```

Parameter open[], high[], low[] und close[] enthalten Felder mit Öffnungspreisen, dem maximalen und minimalen Preis und Schlusspreisen des laufenden Timeframes. Parameter time [] enthält Feld mit Eroeffnungszeit, Parameter spread[] - Feld mit der Geschichte der Spreads (wenn Spread für diesen Typ (wenn Spread für dieses Handelsinstrument vorausgesehen wird). Parameter volume[] und tick_volume[] enthalten die Geschichte des Handels-und Tickvolumen.

Zu bestimmen die Richtung des Indizierenz in Feldern time[], open[], high[], low[], close[], tick_volume[], volume[] und spread[], muss man die Funktion ArrayGetAsSeries() aufrufen. Damit man von Default-Werten nicht abhängt, muss man die Funktion ArraySetAsSeries() für die Felder aufrufen, mit denen es zu arbeiten erwartet.

Der erste Parameter rates_total enthält die Zahl von Bars, zugängliche dem Indikator für Berechnungrund entspricht der Baranzahl, zugängliche auf dem Chart.

Wir muessen über den Zusammenhang zwischen dem Rückgabewert von OnCalculate() und dem zweiten Eingabeparameter prev_calculated erwaehnen. Parameter prev_calculated enthält beim Funktionsaufruf den Wert, den die Funktion OnCalculate() beim **früheren** Aufruf **rückgegeben** hat. Das ermöglicht rationelle Berechnungsalgorithmen des Benutzerindikators zu realisieren, um

© 2000-2025, MetaQuotes Ltd.

Wiederberechnungen für die Bars, die seit früheren Start dieser Funktion nicht verandert haben, zu vermeiden.

Dafür reicht es gewoehnlich, Parameterwert rates_total rückzugeben, der die Baranzahl beim laufenden Funktionsaufruf enthält. Wenn vom Moment des letzten Aufrufes der Funktion OnCalculate() Preisdaten nicht verändert wurden (tiefere Geschichte heruntergeladen sind oder Auslassungen der Geschichte ausgefuellt sind),wird der Wert des Eingabeparameters prev_calculated vom Terminal Null gesetzt werden.

Bemerkung: Wenn die Funktion OnCalculate Nullwert rückgibt, werden die Werte des Indikators im Fenster DataWindow des Client-Terminals nicht gezeigt.

für bessere Verständigung, ist es empfehlenswert, Indikator ablaufen zu lassen, dessen Kode unten angegeben wird.

Beispiel des Indikators:

```
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Line
#property indicator_label1  "Line"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrDarkBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- indicator buffers
double         LineBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,LineBuffer,INDICATOR_DATA);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime& time[],
                const double& open[],
                const double& high[],
                const double& low[],
                const double& close[],
                const long& tick_volume[],
                const long& volume[],
                const int& spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- bekommen die Anzahl der zugänglichen Bars für laufendes Symbol und laufende Peri
   int bars=Bars(Symbol(),0);
   Print("Bars = ",bars,", rates_total = ",rates_total,", prev_calculated = ",prev_cal
   Print("time[0] = ",time[0]," time[rates_total-1] = ",time[rates_total-1]);
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Sehen Sie auch

Programmausführung, Ereignisse des Client-Terminals, Arbeit mit Ereignissen

© 2000-2025, MetaQuotes Ltd.

Variablen

Erklaerung der Variablen

Variablen muessen vor der Vewendung erklärt werden. Für die Identifizierung der Variablen werden unikale Namen verwendet. Beschreibungen der Variablen werden für ihre Definition und Typenerklärung verwendet. Beschreibung ist keine Anweisung.

Einfache Typen sind:

- char, short, int, long, uchar, ushort, uint, ulong - Ganzzahlen;
- color - Ganzzahl, repraesentiert RGB-Farbe;
- datetime - Datum und Zeit, zeichenlose Ganzzahl, enthaehlt Sekundenzahl, die seit 0 Uhr 1 Januar 1970 vergangen sind;
- bool - logische Werte true und false;
- double - Zahlen doppelter Genauigkeit mit Fliesspunkt;
- float -Zahlen mit einfacher Genauigkeit mit Fliesspunkt;
- string - Symbolzeilen.

Beispiele:

```
string szInfoBox;
int    nOrders;
double dSymbolPrice;
bool   bLog;
datetime tBegin_Data   = D'2004.01.01 00:00';
color    cModify_Color = C'0x44,0xB9,0xE6';
```

Zusammengesetzte Typen:

Strukturen - sind zusammengesetzte Datentypen, gebildet mit Verwendung von anderen Typen.

```
struct MyTime
  {
   int hour;    // 0-23
   int minute;  // 0-59
   int second;  // 0-59
  };
...
MyTime strTime; // Variable des Typs der früher erklärten Struktur MyTime
```

Vor der Strukturerklärung kann man nicht Variablen des Typs Strukturen erklären.

Array

Array - indizierte Menge gleichartiger Daten:

```
int    a[50];       // eindimensionales Feld aus 50 Ganzzahlen.
double m[7][50];    // zweidimensionales Feld aus sieben Feldern,
                    // jedes von denen aus 50 Zahlen besteht.
```

© 2000-2025, MetaQuotes Ltd.

```
MyTime t[100];       // Feld mit den Elementen der Art MyTime
```

Arrayindex kann nur Ganzzahl sein. Es werden nicht mehr als vierdimensionale Arrays zugelassen. Die Nummerierung der Elemente des Arrays faengt mit Null an. Das letzte Element des eindimensionalen Arrays ist um eine Nummer kleiner, als die Feldgroesse, d.h. der Zugriff zum letzten Element des Arrays aus 50 Ganzzahlen wird als a[49] aussehen. Dasselbe bezieht sich auf vieldimensionale Arrays - Indizieren einer Dimension erfolgt von 0 bis -1. Das letzte Element des zweidimensionalen Arrays aus dem Beispiel wird als m[6][49] aussehen.

Statistische Arrays können nicht als Timeserien dargestellt werden, d.h. die Funktion ArraySetAsSeries(), die den Zugang zu Feldelementen vom Ende zum Anfang des Arrays festlegt, ist für sie nicht anwendbar. Wenn es notwendig ist, den Zugang zum Feld in Timeserien zu gewaehrleisten, benutzen Sie Objekt des dynamischen Arrays.

Beim Zugriff ueber die Grenzen des Arrays hinaus generiert das vollziehende System ein kritisches Fehler und die Durchführung des Programms wird unterbrochen.

Integrierte Methoden für die Arbeit mit Arrays

Die Funktionen aus dem Abschnitt Array-Funktionen, sowie die integrierten Methoden können verwendet werden, um mit Arrays zu arbeiten:

Methode	Äquivalente	Beschreibung
void array.Fill(const scalar value, const int start_pos=0, const int count=-1);	ArrayFill, ArrayInitialize.	Füllt das Array mit dem angegebenen Wert.
void array.Free();	ArrayFree	Gibt den dynamischen Array-Puffer frei und setzt die Größe der Null-Dimension auf 0 (Null).
int array.Resize(const int range0_size, const int reserve); int array.Resize(const int range_sizes[], const int reserve);	ArrayResize	Legt eine neue Größe in der ersten Dimension des Arrays fest.
int array.Print();	ArrayPrint	Druckt die Werte des Arrays mit einfachen Typen im Journal aus.
int array.Size(const int range=-1);	ArraySize, ArrayRange	Gibt die Anzahl der Elemente des gesamten Arrays (range=-1) oder der angegebenen Array-Dimension zurück.
bool array.IsDynamic();	ArrayIsDynamic	Prüft, ob das Array dynamisch ist.
bool array.IsIndicatorBuffer();		Prüft, ob das Array ein Indikatorpuffer ist.
bool array.IsSeries();	ArrayIsSeries	Prüft, ob das Array eine Zeitreihe ist.

© 2000-2025, MetaQuotes Ltd.

Methode	Äquivalente	Beschreibung
bool array.AsSeries();	ArrayGetAsSeries	Prüft die Indexrichtung des Arrays.
bool array.AsSeries(const bool as_series);	ArraySetAsSeries	Legt die Indexrichtung im Array fest.
int array.Copy(const src_array[], const int dst_start, const int src_start, const int cnt);	ArrayCopy	Kopiert die Array-Werte in ein anderes Array.
int array.Compare(const src_array[], const int dst_start, const int src_start, const int cnt);	ArrayCompare	Gibt das Ergebnis des Vergleichs zweier einfacher Arrays oder nutzerdefinierter Strukturen zurück.
int array.Insert(const src_array[], const int dst_start, const int src_start, const int cnt);	ArrayInsert	Fügt die angegebene Anzahl von Elementen aus einem Quell-Array in ein Empfangs-Array ein, beginnend mit einem angegebenen Index.
int array.Remove(const int start_pos, const int count);	ArrayRemove	Entfernt die angegebene Anzahl von Elementen aus dem Array, beginnend mit einem angegebenen Index.
int array.Reverse(const int start_pos, const int count);	ArrayReverse	Kehrt die angegebene Anzahl von Elementen im Array um, beginnend mit einem angegebenen Index.
bool array.Swap(array& arr[]);	ArraySwap	Tauscht den Inhalt mit einem anderen dynamischen Array desselben Typs.
void array.Sort(sort_function);	ArraySort	Sortiert numerische Arrays nach der ersten Dimension.
int array.Search(scalar value, search_function);	ArrayBsearch	Gibt den Index des ersten gefundenen Elements der ersten Dimension des Arrays zurück.
int array.Find((scalar value, search_function);		Führt eine Suche im Array mit der übergebenen Funktion durch und gibt den Index des ersten gefundenen Elements zurück.
array array.Select(scalar value, search_function);		Führt eine Suche im Array mit der übergebenen Funktion durch und gibt das Array mit allen gefundenen Elementen zurück.

Zugangsspezifikatoren

© 2000-2025, MetaQuotes Ltd.

Zugangsspezifikatoren deuten dem Compilierer darauf, wie der Zugang zu Variable, Struktur- und Klassen-gliedern durchgeführt werden kann.

Spezifikator const erklärt die Variable als Konstante und erlaubt nicht den Wert dieser Variable bei der Programmdurchführung zu verändern. Es wird eimalige Initialisierung der Variable bei ihrer Erklaerung zugelassen.

Beispiel

```
int OnCalculate (const int rates_total,      // Feldgroesse price[]
                 const int prev_calculated,  // Anzahl der verarbeiteter Bars im voric
                 const int begin,            // Anfangsposition der bedeutsamen Daten
                 const double& price[]       // Feld für Berechnung
   );
```

für Zugang zu Struktur- und Klassenglieder werden folgende Spezifikatoren verwendet:

- public - erlaubt unbegrenzten Zugang zur Variable oder Klassenmethoden;
- protected - erlaubt Zugang seitens Methoden der vorliegenden Klasse und seitens Methoden der oeffentlich heritierten Klassen. Anderer Zugang ist unmoeglich;
- private - ermöglicht Zugang zu Variablen und Klassenmethoden ausschliesslich aus Metoden der vorliegenden Klasse.
- virtual - nur für Klassenmethoden anwendbar (nicht aber für Strukturmethoden) und meldet dem Compilierer an, dass diese Metode in der Tabelle virtueller Klassenfunktionen aufgestellt werden muss.

Speicherklassen

Es gibt drei Speicherklassen: static, input und extern. Diese Modifikatoren der Speicherklasse deuten dem Compilierer darauf, dass entsprechende Variablen im vorbestimmten Speicherplatz eingeordnet werden, der als globaler Pool bezeichnet wird. Dabei deuten diese Modifikatoren auf besondere Bearbeitung dieser Variablen hin.

Wenn eine Variable, die auf dem lokalen Niveau erklärt wurde, nicht statisch ist, erfolgt Speicherverteilung für diese Variable automatisch auf dem Programmstack. Speicher, der nicht für ein statisches Feld verteilt wurde, erfolgt auch automatisch beim Ausgang vom Sichtbarkeitsbeich des Blocks, in dem das Feld erklärt wurde.

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte, Statische Klassenmitglieder

© 2000-2025, MetaQuotes Ltd.

Lokale Variablen

Variable, die innerhalb einer Funktion erklärt wird ist lokal. Sichtbereich der lokalen Variable ist von Funktion begrenzt, innerhalb deren sie erklärt wird. Lokale Variable kann initialisiert werden mittels jedes Ausdrucks. Initialisierung der lokalen Variable wird jedesmal beim Aufruf entsprechender Funktion durchgeführt. Lokale Variablen werden im zeitlichen Speicherbereich entsprechender Funktion aufgestellt.

Beispiel:

```
int somefunc()
  {
   int ret_code=0;
   ...
   return(ret_code);
  }
```

Operationsbereich (oder Sichtbereich) der Variable ist Programmteil , in dem auf die Variable verwiesen werden kann. Variable, die innerhalb des Blocks erklärt werden (auf innerem Niveau) haben Block als Operationsbereich. Operationsbereich Block faengt mit der Erklaerung der Variable und beendet mit der rechten geschweiften Klammer.

Lokale Variablen, erklärt am Funktionsanfang haben Operationsbereich Block sowi Funktionsparameter, die lokale Variablen sind. Jeder Block kann Variablenerklärungen enthalten. Wenn Blocks eingebettet sind und Identifikator im Aussenblock den selben Namen, wie Identifikator im innenblock hat, ist der Identifikator des Aussenblocks "unsichtbar" (verhuellt) bis Fertigstellung des Innenblocks.

Beispiel:

```
void OnStart()
  {
//---
   int i=5;       // lokale Variable der Funktion
     {
      int i=10;  // Variable der Funktion
      Print("im Block i = ",i); // Ergebnis  i=10;
     }
   Print("ausser Block i = ",i);  // Ergebnis  i=5;
  }
```

Das bedeutet, dass sobald der Innenblock durchgeführt wird, sieht er die Werte seiner eigenen lokalen Identifikatoren, und nicht die Werte der Identifikatoren mit identischen Namen im umfassenden Block.

Beispiel:

```
void OnStart()
  {
//---
   int i=5;       // lokale Variable der Funktion
   for(int i=0;i<3;i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("Innerhalb for i = ",i);
   Print("Ausserhalb i = ",i);
  }
/* Ergebnis der Durchführung
innerhalb for i = 0
innerhalb for i = 1
innerhalb for i = 2
ausserhalb des Blocks i = 5
*/
```

Lokale Variablen erklärt als static, haben Operationsbereich Block, obwohl sie seit Anfang der Programmdurchführung existieren.

Stack

In jedem MQL5 Programm, ist ein spezieller Speicherbereich genannt zur Speicherung von automatisch erstellten lokalen variable Funktionen verteilt. Es wird ein Stack für alle Funktionen reserviert, standardmäßig beträgt seine Größe für Indikatoren 1 Mb. In Expert Advisors und Skripts kann man die Stack-Größe per Compiler-Direktive #property stacksize steuern (setzt die Stack-Größe in Bytes), standardmäßig werden für sie 8Mb reserviert.

Statische lokale Variablen werden an der gleichen Stelle, wo andere statische und globale Variablen gespeichert werden gespeichert - In einem speziellen Speicherbereich, der separat aus dem Stack besteht. Dynamisch erstellte Variablen auch verwenden ein separater Speicherbereich (getrennt von der Stack).

Mit jedem Aufruf der Funktion, wird ein Platz auf dem Stack für interne nicht-statische Variablen zugewiesen. Nach dem Verlassen der Funktion ist der Speicher wieder verfügbar.

Wenn aus der ersten Funktion dis zweite Funktion aufgerufen wird, dann nimmt die zweite Funktion die erforderliche Größe aus dem verbleibenden Stack-Speicher für ihre Variablen. Damit bei der Verwendung von enthaltenden Funktionen, wird Stack-Speicher sequentiell für jede Funktion belegt werden. Dies kann zu einem Mangel an Arbeitsspeicher während einer der Funktionsaufrufe führen, ist eine solche Situation Stack-Überlauf genannt.

Daher ist für große lokale Daten besser den dynamischen Speicher zu verwenden - bei der Eingabe einer Funktion, den Speicher für die lokalen Bedürfnisse im System zu reservieren (new, ArrayResize()), und beim Verlassen der Funktion den Speicher freizugeben (delete, ArrayFree()).

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Formale Parameter

Die in Funktion übertragenden Parameter sind lokal. Sichtbereich ist Funktionsblock. Formale Parameter muessen sich durch Namen von Aussenparametern und lokalenVariablen unterscheiden, bestimmt innerhalb der Funktion. Im Funktionsblock können formalen Parametern bestimmte Werte zugeordnet werden. Wenn der formale Parameter mit dem Modifikator const erklärt wird, kann sein Wert innerhalb der Funktion nicht verändert werden.

Beispiel:

```
void func(const int & x[], double y, bool z)
  {
   if(y>0.0 && !z)
      Print(x[0]);
   ...
  }
```

Formale Parameter können von Konstante initialisiert werde. In diesem Fall ist initialisierende Wert ein Default-Wert. Parameter, die dem initialisierten Parameter folgen, muessen auch initialisiert werden.

Beispiel:

```
void func(int x, double y = 0.0, bool z = true)
  {
   ...
  }
```

Beim Aufruf solcher Funktion können initialisierte Parameter ausgelassen werden. Statt ihnen werden Default-Werte eingesetzt.

Beispiel:

```
func(123, 0.5);
```

Parameter einfacher Typen werden nach der Bedeutung übertragen, d.h. Änderungen der entsprechenden lokalen Variable dieses Typs innerhalb aufgerufener Funktion nicht in aufrufender Fuktion spiegeln wird. Felder jeder Typen und Daten des Typs Strukturen werden immer mit Link übertragen. Wenn es notwendig ist, veränderung des Feldes oder Strukturinhalt zu verbieten, muessen Parameter dieser Typen mit Schluesselwort const erklärt werden.

Es ist moeglich, Parameter einfacher Typen durch Referenz zu übertragen. In diesem Fall ist Modifikation solcher Parameter entsprechende Variablen in aufgerufener Funktion beeinflusst, die durch Referenz übertragen wurden. Für Andeutung, dass der Parameter durch Referenz übertragen wird, muss nach dem Datentyp Modifikator & gestellt werden.

Beispiel:

```
void func(int& x, double& y, double & z[])
  {
   double calculated_tp;
   ...
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<OrdersTotal(); i++)
     {
      if(i==ArraySize(z))        break;
      if(OrderSelect(i)==false) break;
      z[i]=OrderOpenPrice();
     }
   x=i;
   y=calculated_tp;
  }
```

Parameter die durch Refeernz übertragen werden, können nicht mit Default-Werten initialisiert werden.

In Funktion können nicht mehr als 64 Paramerter übertragen werden.

Sehen Sie auch

Input Variablen, Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Statische Variablen

Speicherklasse static bezeichnet statistische Variable. Modifikator static wird vor dem Datentyp angegeben.

Beispiel:

```
int somefunc()
  {
   static int flag=10;
   ...
   return(flag);
  }
```

Statische Variable kann durch die ihrem Typ entsprechende Konstante oder Konstantausdruck initialisiet werden, zum Unterschied von einfacher lokalen Variable, die von jedem Ausdruck initialisiert werden kann.

Statische Variablen existieren seit Programmdurchführung und werden einmalig vor dem Aufruf spezialisierter Funktion OnInit() initialisiert. Wenn Wenn Anfangswerte nicht angegeben werden, nehmen Variablen der statistischen Klasse Nullwerte an.

Lokale Variablen, erklärt mit Schlüsselwort static bewahren ihre Werte innerhalb der ganzen Lebensdauer der Funktion. Beim der Funktion haben lokale Variablen die Werte, die sie beim vorletzten Aufruf hatten.

Alle Variablen im Block, ausser formaler Parameter der Funktion können als statisch bezeichnet werden.

Beispiel:

```
int Counter()
  {
   static int count;
   count++;
   if(count%100==0) Print("Funktion Counter ist schon aufgerufen ",count," mal");
   return count;
  }
void OnStart()
  {
//---
   int c=345;
   for(int i=0;i<1000;i++)
     {
      int c=Counter();
     }
   Print("c =",c);
  }
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte, Statische Klassenmitglieder

© 2000-2025, MetaQuotes Ltd.

Globale Variablen

Globale Variablen werden durch Unterbringung ihrer Erklaerungen ausser beschreibung bestimmter Funktion erzeugt. Globale Variablen werden auf demselben Niveau, wie Funktionen bestimmt, d.h. sie sind nicht lokal in einem Block.

Beispiel:

```
int GlobalFlag=10;   // globale Variable
int OnStart()
  {
   ...
  }
```

Sichtbereich globaler Variablen - das ganze Programm, globale Variablen werden von allen Funktionen zugänglich, bestimmt im Programm. Werden von Null initialisiert, wenn explizit anderer Anfangswert nicht zugeordnet wird. Globale Variable kann nur von der entsprechenden ihrem Typ Konstante oder Konstantausdruck initialisiert werden.

Globale Variablem werden nur einmal initialisiert, nachdem das Programm in den Speicher von Client-Terminal geladen ist und vor der ersten Behandlung des Init-Ereignisses. Wenn die die Klassenobjekte darstellende globale Variablen initialisiert werden, werden die entsprechenden Konstruktoren aufgerufen. In Skripten werden globale Variablen vor der Behandlung des Start-Ereignisses initialisiert.

Bemerkung: Man soll nicht Variablen, erklärt auf dem globalen Niveau mit globalen Variablen des Client-Terminals verwechseln, der Zugang zu denen mittels der Funktionen GlobalVariable...() durchgeführt wird.

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Eingabevariablen

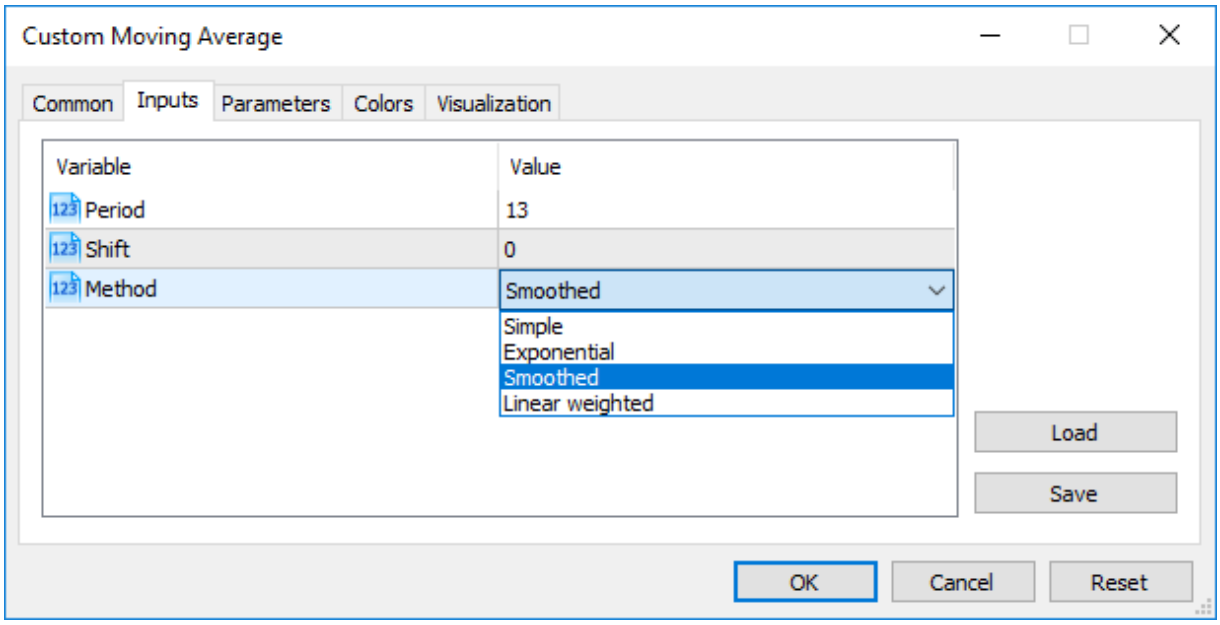
Speicherklasse input bestimmt die externen Variablen. Modifikator input wird vor dem Datentyp angegeben. Den Wert von Variable mit dem Modifikator input innerhalb mql5-Programm können nicht verändert werden, diese Variable sind nur lesbar. Den Wert von input-Variablen kann der Benutzer im Fenster der Programmeigenschaften ändern. Externe Variablen werden vor dem OnInit() Aufruf immer neu initialisiert.

Die maximale Länge der Namen von Eingabevariablen beträgt 63 Zeichen. Für den Eingabeparameter vom Typ string kann die **maximale** Länge (Länge der Zeichenkette) von 191 bis 253 Zeichen betragen (siehe Bemerkung). Die minimale Länge ist 0 Zeichen (der Wert wird nicht gesetzt).

Bespiel:

```
//--- Eingabeparameter
input int             MA_Period=13;
input int             MA_Shift=0;
input ENUM_MA_METHOD MA_Method=MODE_SMMA;
```

Input Variablen bestimmen die Eingabeparameter des Programms, sie sind im Fenster der Programmeigenschaften zugänglich.



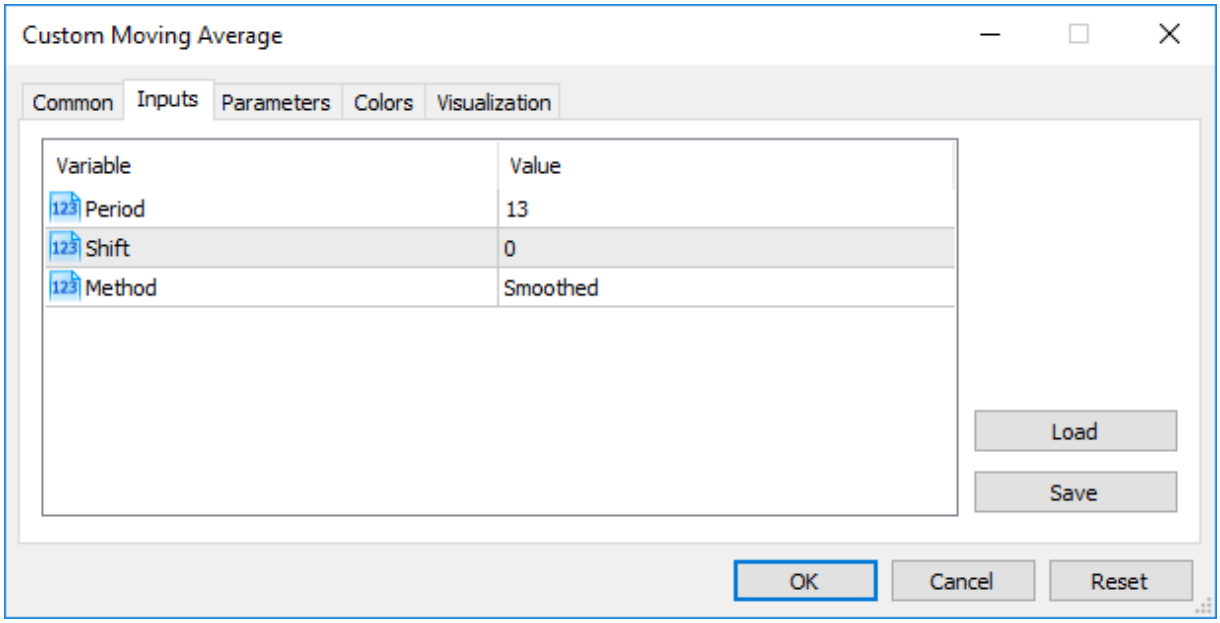
In der Registerkarte "Inputs" kann eine andere Art der Anzeige der Namen der Eingangsparameter eingestellt werden. Verwenden Sie dazu einen Zeilenkommentar, der nach der Beschreibung des Eingabeparameters in derselben Zeile stehen soll. Damit können den Eingabeparameter benutzerfreundliche Namen zugeordnet werden.

Beispiel:

```
//--- Eingabeparameter
input int             InpMAPeriod=13;         // Glättungslänge
input int             InpMAShift=0;           // horizontaler Versatz der Linie
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_MA_METHOD InpMAMethod=MODE_SMMA;  // Glättungsverfahren
```



Bemerkung: Felder und Variablen zusammengesetzter Typen können nicht als input-Variablen auftreten.

Bemerkung: Die Länge eines Kommentartextes für eine Eingabevariable darf 63 Zeichen nicht überschreiten.

Bemerkung: Für Eingabevariablen vom Typ string wird die Begrenzung der Wertlänge (Länge der Zeichenkette) durch folgende Bedingungen festgelegt:

- der Parameterwert wird durch die Zeichenkette "parameter_name=parameter_wert" dargestellt ('=' wird berücksichtigt),
- maximale Darstellungslänge von 255 Zeichen (*total_length_max*=255 oder 254 Zeichen ohne '='),
- Maximale Länge des String-Parameters *parameter_name_length* = 63 Zeichen.

Die maximale Stringgröße für einen String-Parameter wird also nach folgender Gleichung berechnet:

```
parameter_value_length=total_length_max-parameter_name_length=254-parameter_name_lengt
```

Damit liegt die maximale Länge der Zeichenkette zwischen 191 (parameter_name_length=63) und 253 Zeichen (parameter_name_length=1).

Parameterübergabe beim Aufruf der Benutzerindikatoren aus mql5-Programmen

Benutzerindikatoren werden mittels der Funktion iCustom()aufgerufen. Dabei müssen nach dem Namen des Nutzerindikators die Parameter folgen, in strenger Übereinstimmung mit Deklaration der input-Variablen des vorgegebenen Nutzerindikators. Wenn die Anzahl der angegebener Parameter kleiner ist als die Anzahl der erklärten input-Variablen im aufgerufenen Benutzerindikator, werden die fehlenden Parameter von Werten ausgefüllt, die bei der Variablendeklaration angegeben wurden.

© 2000-2025, MetaQuotes Ltd.

Wenn im Nutzerindikator die Funktion OnCalculate erster Art (d.h. der Indikator ist auf einem Datenfeld) verwendet wird, muss als letzter Parameter beim Aufruf solches Nutzerindikators einer der Wete ENUM_APPLIED_PRICE oder das Handle eines anderen Indikators auftreten. Gleichzeitig müssen alle Parameter, die den input-Variablen entsprechen, explizit angegeben werden.

Enumerationen als input-Parameter

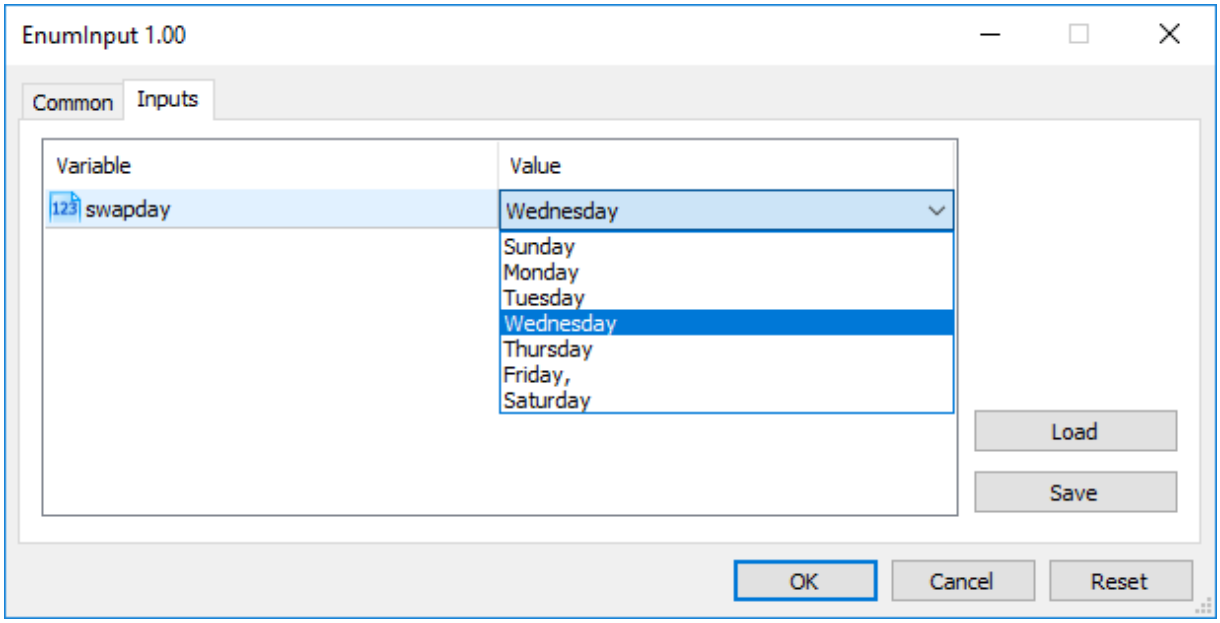
Als input-Variablen (Eingabeparameter für mql5-Programme) können nicht nur die von der Sprache MQL5 vorausgesehene eingebettete Enumerationen verwendet werden, sondern auch die vom Nutzer vorbestimmten Enumerationen. ZB. können wir die Enumeration dayOfWeek erzeugen, die Wochentage beschreibt und input-Variable für Angeben konkretes Wochentages nicht als Ziffer, sondern in der dem Nutzer gewöhnten Art verwenden.

Beispiel:

```
#property script_show_inputs
//--- day of week
enum dayOfWeek
  {
   S=0,      // Sunday
   M=1,      // Monday
   T=2,      // Tuesday
   W=3,      // Wednesday
   Th=4,     // Thursday
   Fr=5,     // Friday,
   St=6,     // Saturday
  };
//--- Eingabeparameter
input dayOfWeek swapday=W;
```

Damit der Nutzer beim Scriptstart den nötigen Wert aus Eigenschaftsfenster wählen kann, verwenden wir Preprozessorbefehl #property script_show_inputs. Dann aktivieren wir Script und können aus der Liste einen der Enumerationswerte dayOfWeek wählen. Dann aktivieren wir Script EnumInInput und übergehen der Registerkarte "Parameter". Ein Standardwert des Parameters swapday ist Mittwoch (W=3), aber wir können einen anderen Wert zuordnen und diesen Wert für Änderung der Programmarbeit verwenden.

© 2000-2025, MetaQuotes Ltd.



Die Anzahl der möglichen Enumerationswerte ist begrenzt. Darum wird Auswahl des Eingabewertes Auswahlliste verwendet. Als Werte gezeigt in der Liste werden mnemonische Namen der Enumerationsglieder verwendet. Wenn dem mnemonischen Namen Kommentar gegenübergestellt wird, wie es in unserem Beispiel gezeigt ist, wird anstatt des mnemonischen Namens Kommentarinhalt verwendet.

Jeder Wert der Enumeration dayOfWeek hat seinen eigenen Wert von 0 bis 6, aber die Liste der Parameter zeigt den Kommentar für jeden Wert. Dies gibt zusätzliche Flexibilität beim Schreiben von Programmen mit klaren Beschreibungen der Eingabeparameter.

Variablen mit dem Modifikator sinput

Variablen mit dem Modifikator input erlauben nicht nur die Werte der externen Parameter bei dem Start der Programme anzugeben, sondern auch eine grosse Rolle bei der Optimierung der Handelsstrategien im Tester spielen. Jede im Expert Advisor erklärte input-Variable, mit Ausnahme des Typs string, kann an der Optimierung teilnehmen.

In einigen Fällen ist es notwendig, einige externe Parameter des Programms von der Erstellung des Bereichs von allen möglichen Durchgängen im Tester auszuschließen. Speziell für solche Fälle gibt es einen Modifikator von Speicher sinput. sInput ist eine Kurzform der Erklärung von statischer globaler Variable: sInput = static input. Es bedeutet die folgende Erklärung im Code des Expert-Advisors

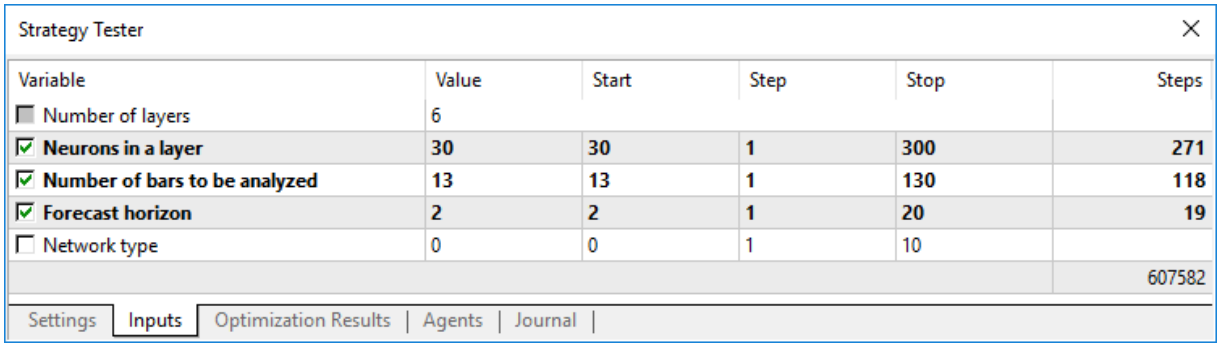
```
sinput       int layers=6;   // Number of layers
```

wird gleichbedeutend mit der vollständigen Erklärung sein

```
static input int layers=6;   // Number of layers
```

Die mit dem Modifikator sinput erklärte Variable ist ein Eingabeparameter des MQL5-programms, der Wert dieses Parameters kann bei seinem Start geändert werden. Dabei nimmt diese Variable nicht an dem Optimierungsprozess der Eingabeparameter teil, das heißt werden ihre Werte bei der Suche des besten Satzes der Parameter nach dem angegebenen Kriterium nicht durchgesucht.

© 2000-2025, MetaQuotes Ltd.



Die Abbildung zeigt, dass der Expert-Advisor fünf externe Parameter hat. Der Parameter "Anzahl von Ebenen" wird als sinput erklärt und ist gleich 6. Dieser Parameter kann nicht während einer Optimierung der Handelsstrategie geändert werden. Man kann für ihn den notwendigen Wert angeben, der weiter verwendet wird. Felder für Start, Stop und Schritt für diese Variable sind für die Einstellung der Werte nicht verfügbar.

Nachdem wir den Modifikator sinput für Variable angegeben haben, verbieten wir dem Benutzer diesen Parameter zu optimisieren. Es bedeutet, dass der Terminal Benutzer für sie die Anfangs-und Endwerte für automatisches Durchsuchen in einem bestimmten Bereich bei der Optimierung im Strategie-Tester nicht angeben kann.

Aber dabei gibt es eine Ausnahme von dieser Regel - sinput-Variablen kann man in Optimierungsaufgaben mit Hilfe der Funktion ParameterSetRange() variieren. Diese Funktion wurde für die Programmsteuerung des Raums der verfügbaren Werte für jede input-Variable speziell erstellt, einschließlich jener, die für static input (sinput) erklärt sind. Eine weitere Funktion ParameterGetRange() erlaubt es Ihnen (im Handler OnTesterInit ()) die Werte der input-Variablen zu erhalten, wenn Optimierung gestartet wird, und falls notwendig, den Schritt der Veränderung und den Bereich neu anzugeben, innerhalb dessen der Wert des optimisierten Parameters durchgesucht wird.

Daher erlaubt die Kombination von sinput Modifikator mit zwei Funktionen für die Arbeit mit input-Parametern die flexiblen Regeln für die Einstellung der Optimierungsintervallen von einigen input-Variablen je nach der Werten der anderen input-Variablen zu schaffen.

Gruppieren der Eingabeparameter

Um das Arbeiten mit MQL5-Programmen zu erleichtern, können die Eingabeparameter mit dem Schlüsselwort group in benannte Blöcke unterteilt werden. Dies ermöglicht eine visuelle Trennung einiger Parameter von anderen, basierend auf der in ihnen zugehörigen Logik.

```
input group           "Group name"
input int             variable1 = ...
input double          variable2 = ...
input double          variable3= ...
```

Nach einer solchen Deklaration werden alle Eingabeparameter visuell zu einer benannten Gruppe zusammengefasst, die die Parameterkonfiguration für MQL5-Nutzer beim Start in einem Chart oder im Strategietester vereinfacht. Die Spezifikation jeder Gruppe ist gültig, bis eine neue Gruppendeklaration erscheint:

```
input group           "Group name #1"
```

© 2000-2025, MetaQuotes Ltd.

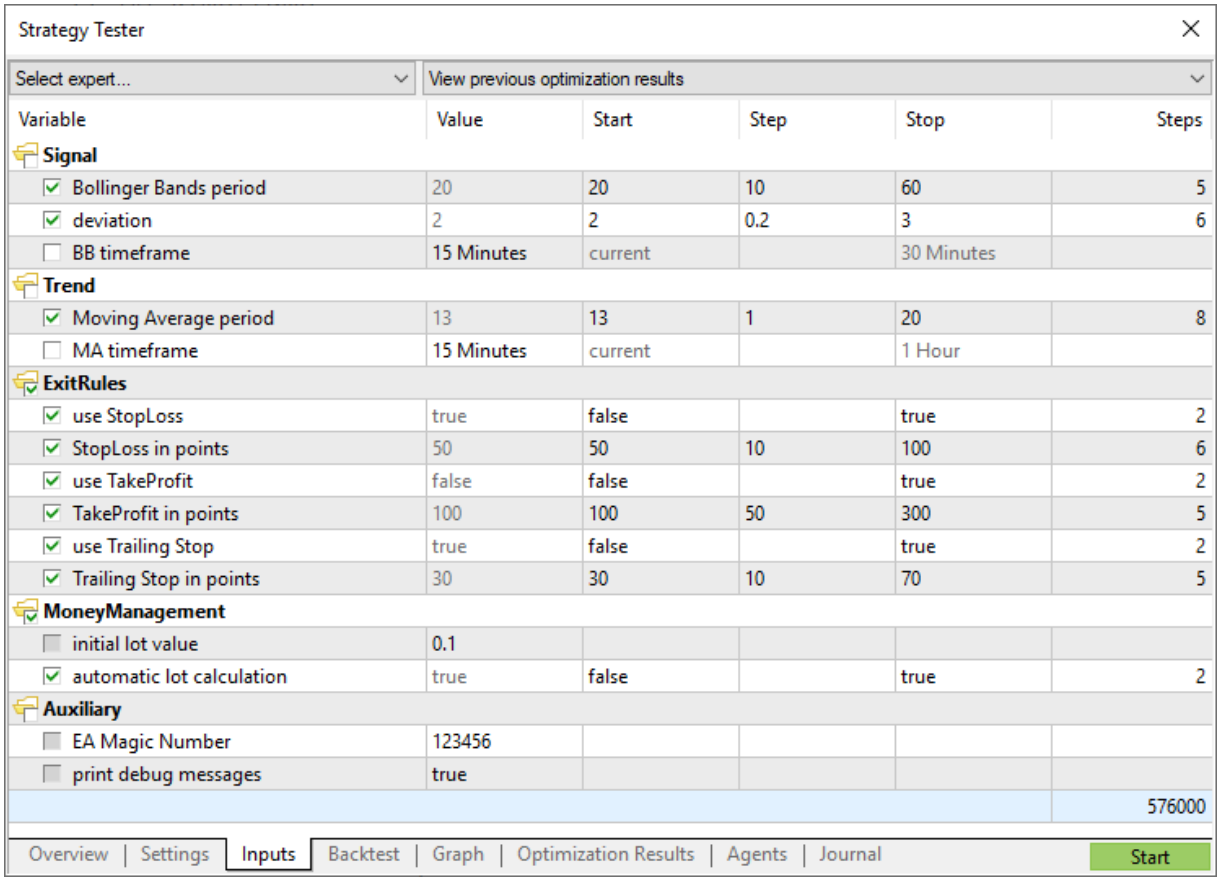
```
input int              group1_var1 = ...
input double           group1_var2 = ...
input double           group1_var3 = ...

input group            "Group name #2
input int              group2_var1 = ...
input double           group2_var2 = ...
input double           group2_var3 = ...
```

Ein Beispiel EA mit dem Block von Eingabeparameter, die nach ihrem Zweck getrennt sind:

```
input group            "Signal"
input int              ExtBBPeriod   = 20;       // Bollinger Bands period
input double           ExtBBDeviation= 2.0;      // deviation
input ENUM_TIMEFRAMES ExtSignalTF=PERIOD_M15;   // BB timeframe

input group            "Trend"
input int              ExtMAPeriod   = 13;       // Moving Average period
input ENUM_TIMEFRAMES ExtTrendTF=PERIOD_M15;    // MA timeframe

input group            "ExitRules"
input bool             ExtUseSL      = true;     // use StopLoss
input int              Ext_SL_Points = 50;       // StopLoss in points
input bool             ExtUseTP      = false;    // use TakeProfit
input int              Ext_TP_Points = 100;      // TakeProfit in points
input bool             ExtUseTS      = true;     // use Trailing Stop
input int              Ext_TS_Points = 30;       // Trailing Stop in points

input group            "MoneyManagement"
sinput double          ExtInitialLot = 0.1;      // initial lot value
input bool             ExtUseAutoLot = true;     // automatic lot calculation

input group            "Auxiliary"
sinput int             ExtMagicNumber = 123456;  // EA Magic Number
sinput bool            ExtDebugMessage= true;    // print debug messages
```

Wenn Sie einen solchen EA im Strategietester starten, können Sie mit einem Doppelklick auf einen Gruppennamen den Eingabeblock ein/ausblenden und mit einem Klick auf das Gruppenkästchen alle Parameter für die Optimierung ab-/auswählen.

© 2000-2025, MetaQuotes Ltd.



Siehe auch

iCustom, Enumerationen, Programmeigenschaften

© 2000-2025, MetaQuotes Ltd.

Extern Variablen

Schlueeselwort extern wird verwendet, um Identifikatoren der Variablen als Identifikatoren statischer Speicherklasse mit globaler Lebensdauer zu erklären. Solche Variable existieren seit Anfang der Programmdurchführung und für sie wird Speicher sofort nach Anfang der Programmdurchführung verteilt und initialisiert.

Man kann Programme erzeugen, die aus mehreren Ausgangsdateien bestehen. Dafür wird die Anweisung des Preprozessors #include verwendet. Variablen, die als extern erklärt werden mit demselben Typ und Identifikator, können in verschiedenen Ausgangsdateien eines Projekts existieren.

Bei Compilierung des ganzen Projektes werden alle extern-Variablen, die denselben Typ und Identifikator haben, werden mit einem. Extern-Variablen sind für getrennte Compilierung der Ausgangsdateien nuetzlich. Extern-Variablen können initialisiert werden, aber nur einmal - es ist nicht zulaessig, dass mehrere initialisierte extern-Variablen eines Typs und mit demselben Identifikator existieren.

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Initialisierung der Variablen

Jede Variable kann bei der Definition initialisiert werden. Wenn explizite Initialisierung der Variable nicht durchgefuhrt wird, kann der Wert, gespeichert in der Variable, ganz verschiden sein. Implizite Initialisierung wird nicht durchgeführt.

Globale und statische Variablen können nur von Konstante des entsprechenden Typs oder Konstantausdruck initialisiert werden. Lokale Variablen können von jedem Ausdruck initialisiert werden, nicht nur von der Konstante.

Initialisierung on globalen und statischen Variablen wird einmalig durchgeführt. Initialisierung lokaler Variablen wird jedesmal beim Aufruf entsprechender Funktionen durchgeführt.

Beispiele:

```
int    n       = 1;
string s       = "hello";
double f[]     = { 0.0, 0.236, 0.382, 0.5, 0.618, 1.0 };
int    a[4][4] = { {1, 1, 1, 1}, {2, 2, 2, 2}, {3, 3, 3, 3}, {4, 4, 4, 4 } };
//--- aus Tetris
int    right[4]={WIDTH_IN_PIXELS+VERT_BORDER,WIDTH_IN_PIXELS+VERT_BORDER,
                 WIDTH_IN_PIXELS+VERT_BORDER,WIDTH_IN_PIXELS+VERT_BORDER};
//--- Initialisierung aller Felder in der Struktur mit der NULL-Werte
MqlTradeRequest request={};
```

Liste von Werten der Feldelemente muss in geschweiften Klammern stehen. Ausgelassene initialisierende Folgen gelten als die der Null gleichen.

Wenn die Größe des zu initialisierten Feldes nicht angegeben wird, wird die vom Compilierer festgestellt, ausgehend von der groesse der initialisierenden Folge.

Beispiele:

```
struct str3
  {
   int               low_part;
   int               high_part;
  };
struct str10
  {
   str3              s3;
   double            d1[10];
   int               i3;
  };
void OnStart()
  {
   str10 s10_1={{1,0},{1.0,2.1,3.2,4.4,5.3,6.1,7.8,8.7,9.2,10.0},100};
   str10 s10_2={{1,0},{},100};
   str10 s10_3={{1,0},{1.0}};
//---
   Print("1.  s10_1.d1[5] = ",s10_1.d1[5]);
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("2.  s10_2.d1[5] = ",s10_2.d1[5]);
   Print("3.  s10_3.d1[5] = ",s10_3.d1[5]);
   Print("4.  s10_3.d1[0] = ",s10_3.d1[0]);
  }
```

Für Variablen des Typs Strukturen ist teilweise Initialisierung erlaubt, gilt dies auch für statische Arrays (mit einer expliziten Größe). Sie können eine oder mehrere der ersten Elemente einer Struktur oder Array initialisieren, werden die restlichen Elemente in diesem Fall mit Nullen initialisiert werden.

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Sichtbereich und Lebensdauer der Variablen

Es gibt zwei Typen von Sichtbereich: lokaler Sichtbereich und globaler Sichtbereich.

Variable, die ausser aller Funktionen erklärt wurde, wird in globalen Sichtbereich eingestellt.Zugang zu diesen Variablen kann aus jedem Programmplatz erfolgen. Solche Variablen befinden sich im globalen Pool des Speichers, darum faellt ihre Lebensdauer mit der Programmlebensdauer.

Variable, erklärt innerhalb des Blocks (Codeteil in geschweiften Klammern) gehört dem lokalen Sichtbereich. Solche Variable ist unsichtbar (darum unzugaenglich) ausser Block, in dem sie erklärt wird. Die verbreiteste lokale Erklaerung - Variable, erklärt innerhalb der Funktion. Variable, die lokal erklärt wid befindet sich im Stoss und ihre Lebensdauer faellt mit der Lebensdauer der Funktion zusammen.

Da Sichtbereich der lokalen Variable Block, in dem sie erklärt wurde, ist, gibt es die Moeglichkeit, Variablen mit Namen zu erklären, der mit den Variablennamen, erklärt in anderen Blocks, und auch erklärt auf den hoeheren Niveaus bis zum globalen, zusammenfaellt.

Beispiel:

```
void CalculateLWMA(int rates_total,int prev_calculated,int begin,const double &price[]
  {
   int        i,limit;
   static int weightsum=0;
   double     sum=0;
//---
   if(prev_calculated==0)
     {
      limit=MA_Period+begin;
      //--- set empty value for first limit bars
      for(i=0; i<limit; i++) LineBuffer[i]=0.0;
      //--- calculate first visible value
      double firstValue=0;
      for(int i=begin; i<limit; i++)
        {
         int k=i-begin+1;
         weightsum+=k;
         firstValue+=k*price[i];
        }
      firstValue/=(double)weightsum;
      LineBuffer[limit-1]=firstValue;
     }
   else
     {
      limit=prev_calculated-1;
     }

   for(i=limit;i<rates_total;i++)
     {
      sum=0;
```

© 2000-2025, MetaQuotes Ltd.

```
      for(int j=0; j<MA_Period; j++) sum+=(MA_Period-j)*price[i-j];
      LineBuffer[i]=sum/weightsum;
     }
//---
  }
```

Legen Sie Wert auf Variable i, erklärt in der Zeile:

```
      for(int i=begin; i<limit; i++)
        {
         int k=i-begin+1;
         weightsum+=k;
         firstValue+=k*price[i];
        }
```

Ihr Sichtbereich ist ausschliesslich der Zyklus for, ausser dieses Zyklus fungiert eine andere Variable mit demselben Namen, erklärt am Anfang der Funktion. Außerdem wird im Zykluskoerper die Variable k erklärt, deren Sichtbereich Zykluskoerper ist.

Lokale Variablen können mit dem Zugangsspezifikator static erklärt werden. In diesem Fall stellt der Compilierer diese Variable in globalenSpeicherpool. Darum faellt Lebensdauer statischer Variabler mit der Lebensdauer der Programmlebensdauer zusammen. Dabei wird Sichtbereich dieser Variable vom Block, in dem sie erklärt wurde, begrenzt.

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Erzeugung und Entfernung der Objekte

Nach der Ladung des mql5-Programms wird für jede Variable Speicher entsprechend dem Typ der Variable verteilt. Variablen werden in zwei Typen nach dem Zugang aufgeteilt - globale Variablen und lokale Variablen, und nach den Speicherklassen: Eingabeparameter des mql5-Programms, statische und automatische. Jede Variable wird wenn notwendig durch den entsprechenden Wert initialisiert. Nach der Verwendung wird die Variable deinitialisiert und der von ihr verwendete Speicher kehrt zum ausführenden System MQL5.

Initialisierung und Deinitialisierung der globalen Variablen

Initialisierung der globalen Variablen erfolgt automatisch nach dem Laden des mql5-Programms und vor dem Aufruf jeder Funktion. Bei der Initialisierung erfolgt automatische Zuordnung der Initialwerte den Variablen einfacher Typen und wird der Konstrukteur für Objekte aufgerufen (wenn er vorhanden ist). Eingabeparameter werden immer auf dem globalen Niveau erklärt, durch die Werte initialisiert,die von Benutzern im Dialog beim Ausführungslauf des mql5-Programms.

Abgesehen davon, dass statische Variablen gewöhnlich auf dem lokalen Niveau erklärt werden, wird der Speicher für diese Variablen im voraus verteilt und Initialisierung erfolgt sofort nach der Ladung des Programms, wie auch für globale Variablen.

Ordnung der Initialisierung erfolgt der Ordnung der Erklärung der Variable im Programm und Deinitialisierung erfolgt umgekehrt vor der Ausladung des mql5-Programms. Diese Regel gilt nur für diejenigen Variablen, die nicht von der Anweisung new erzeugt wurden. Solche Variablen werden sofort nach der Ladung erzeugt und initialisiert, und unmittelbar vor der Ausladung werden sie deinitialisiert.

Initialisierung und Deinitialisierung der lokalen Variablen

Wenn die Variable, die auf dem lokalen Niveau erklärt wird, nicht statisch ist, erfolgt die Speicherverteilung für eine solche Variable automatisch. Lokale Variablen, wie auch globale werden automatisch in dem Augenblick initialisiert, wenn während der Ausführung des Programms Erklärung der lokalen Variable erfolgt. So entspricht die Ordnung der Initialisierung der Ordnung der Erklärung.

Lokale Variablen werden am Ende des Programmblocks deinitialisiert, in dem sie erklärt wurden, und in der Reihenfolge, die ihrer Erklärung umgekehrt ist . Block eines Programms ist eine zusammengesetzte Anweisung, die ein Teil der Auswahlanweisung switch, Zuklusanweisung (for, while, do-while), Körper der Funktion oder Teil der Anweisung if-else sein kann.

Initialisierung der lokalen Variablen erfolgt nur in dem Moment, wenn während der Initialisierung des Programms Erklärung der Variable erfolgt. Wenn während der Programmausführung das Block, in dem die Variable erklärt wurde, nicht ausgeführt wurde, wird eine solche Variable nicht initialisiert.

Initialisierung und Deinitialisierung der dynamisch verteilten Objekte

Ein Besonderer Fall stellen Objektanzeiger dar, denn Erklärung des Anzeigers führt nicht zur Initialisierung des entsprechenden Objektes. Die dynamisch verteilten Objekte werden nur im Augenblick der Erzeugung des Klassenexemplares durch die Anweisung new. Initialisierung des Objektes setzt Aufruf des Konstruktors einer entsprechenden Klasse voraus. Wenn es keinen

© 2000-2025, MetaQuotes Ltd.

entsprechenden Konstruktor in der Klasse gibt, werden seine Glieder des einfachen Typs, automatisch nicht initialisiert; Glieder der Typen Zeile, dynamisches Feld und zusammengesetztes Objekt werden automatisch initialisiert.

Anzeiger können auf dem globalen und auf dem lokalen Niveau erklärt werden, und dabei durch einen leeren Wert NULL oder durch den Wert eines Anzeiger desselben oder des heritierten Typs initialisiert werden können. Wenn für den Anzeiger, der auf dem lokalen Niveau erklärt wurde, die Anweisung *new* aufgerufen wird, muss auch die Anweisung *delete* für diesen Anzeiger vor dem Ausgang vom diesen Niveau ausgeführt. Anderenfalls wird der Anzeiger verlorengehen und das Objekt kann nicht explizit entfernt werden.

Alle Objekte, die mittels des Ausdrucks *Anzeiger_des Objektes*=new *Name_der Klasse* erzeugt wurden, müssen im nachhinein mittels der Anweisung delete(*Anzeiger_des Objektes*) entfernt werden. Wenn eine Variable dieser Art nach dem Beenden des Programms aus irgendwelchen Gründen von der Anweisung delete nicht entfernt wurde, wird darüber eine Meldung im Journal "Experten" ausgegeben. Man kann mehrere Variablen erklären und allen den Anzeiger eines Objektes zuordnen.

Wenn ein dynamisch erzeugtes Objekt Konstruktor hat, wird dieser Konstruktor aufgerufen werden während die Anweisung *new* ausgeführt wird . Wenn Objekt einen Destructor hat, wird er aufgerufen werden, während die Anweisung *delete* ausgeführt wird.

So werden dynamisch verteilte Objekte nur während der Ausführung durch die Anweisung *new* erzeugt und werden bestimmt entweder von der Anweisung *delete* oder vom automatisch ausführenden System MQL5 wenn das Programm ausgeladen wird. Ordnung der Erklärung der Anzeiger der dynamisch erzeugten Objekte beeinflusst die Ordnung ihrer Initialisierung nicht. Die Ordnung der Initialisierung und die Ordnung der Deinitialisierung werden vollständig vom Programmierer kontrolliert.

Arbeit mit dynamischen Speicher

Beim Arbeiten mit dynamischen Arrays wird der freigegebene Speicher unmittelbar an das System zurückgegeben.

Bei der Erstellung eines dynamischen Klassenobjekt durch new, wird der Speicher zuerst im Pool von Klassenspeicher, mit dem der Speicher-Manager arbeitet, gesucht, und wenn der Pool nicht genug Speicher hat, wird der Speicher im System angefordert. Wenn Sie ein dynamisches Objekt durch delete löschen, wird Speicher, der vom Objekt besetzt ist, in den Pool von Klassenspeicher zurückgegeben.

Der Speicher-Manager gibt den Speicher an das System zurück sofort nach der Ausgang aus der folgenden Event-Handler-Funktionen: OnInit(), OnDeinit(), OnStart(), OnTick(), OnCalculate(), OnTimer(), OnTrade(), OnTester(), OnTesterInit(), OnTesterPass(), OnTesterDeinit(), OnChartEvent(), OnBookEvent().

Kurzcharakteristik der Variablen

Hauptinformation über Erzeugung, Entfernung, Aufruf der Konstruktors und Destruktors werden in der Kurztabelle angegeben.

© 2000-2025, MetaQuotes Ltd.

	Globale automatische Variable	Lokale automatische Variable	Dynamisch erzeugtes Objekt
Initialisierung	sofort nach der Ladung des mql5-Programms	Beim Reichen des Kodes der Zeile, wo sie erklärt ist	bei Ausführung der Anweisung **new**
Ordnung der Initialisierung	in Form von Erklärung	in Form von Erklärung	unabhängig von der Ordnung der Erklärung
Deinitialisierung	vor der Ausladung des mql5-Programms	beim Ausgang der Ausführung aus dem Block der Erklärung	bei Ausführung der Anweisung **delete** oder vor Ausladung des mql5-Programms
Ordnung der Deinitialisierung	in der Reihenfolge, die der Initialisierung umgekehrt ist	in der Reihenfolge, die der Initialisierung umgekehrt ist	nicht abhängig von der Ordnung der Initialisierung
Aufruf des Konstruktors	bei Ladung des mql5-Programms	bei der Initialisierung	bei Ausführung der Anweisung *new*
Aufruf des Destruktors	bei Ausladung des mql5-Programms	beim Ausgang aus dem Block, in dem die Variable initialisiert wurde	bei Ausführung der Anweisung *delete*
Meldung über Fehler	Meldung im Journal "Experten" über den Versuch der Entfernung eines automatisch erzeugten Objektes	Meldung im Journal "Experten" über den Versuch der Entfernung eines automatisch erzeugten Objektes	Meldung im Journal "Experten" über nicht entfernte Objekte bei Ausladung des mql5-Programms

Sehen Sie auch

Datentypen, Inkapsulation und Erweiterungsfaehigkeit der Typen, Initialisierung der Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Präprozessor

Präprozessor ist ein spezielles Subsystem des MQL5 Compilers, das den Text einer Quelldatei unmittelbar vor dem Kompilieren des Programms aufbereitet.

Der Präprozessor ermöglicht es, die Lesbarkeit des Quellcodes zu verbessern. Der Codes kann strukturiert werden, indem spezielle Dateien mit Quellcodes von mql5-Programmen miteinbezogen werden. Die Möglichkeit, bestimmten Konstanten mnemonische Namen zuzuweisen, trägt zur Lesbarkeit des Codes bei.

Darüber hinaus erlaubt es der Präprozessor, spezifische Parameter von mql5-Programmen festzulegen:

- Konstanten zu deklarieren
- Programmeigenschaften zu setzen
- Dateien in Programmtext miteinzubeziehen
- Funktionen zu importieren
- Bedingte Kompilierung

Der Compiler verwendet Direktiven des Präprozessors für eine vorbereitende Verarbeitung eines Quellcodes vor der Kompilierung. Die Direktive beginnt immer mit dem Zeichen # (Doppelkreuz), deswegen verbietet der Compiler, dieses Zeichen in den Namen von Variablen, Funktionen usw. zu verwenden.

Jede Direktive wird in einem separaten Eintrag beschrieben und ist bis zum Zeilenumbruch gültig. In einem Eintrag können Sie nur eine Direktive verwenden. Wenn der Eintrag der Direktive zu groß ist, kann er mithilfe des Zeichens '\' in mehrere Zeilen aufgeteilt werden. In diesem Fall gilt die nächste Zeile als eine Fortsetzung des Eintrags dieser Direktive.

```
//+------------------------------------------------------------------+
//|  Pseudo-Operator foreach                                         |
//+------------------------------------------------------------------+
#define ForEach(index, array) for (int index = 0,                    \
   max_##index=ArraySize((array));                                   \
   index<max_##index; index++)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
   string array[]={"12","23","34","45"};
//--- Iteration durch das Array mithilfe von ForEach
   ForEach(i,array)
    {
      PrintFormat("%d: array[%d]=%s",i,i,array[i]);
    }
  }
//+------------------------------------------------------------------+
/* Das Ergebnis
   0: array[0]=12
   1: array[1]=23
```

© 2000-2025, MetaQuotes Ltd.

```
   2: array[2]=34
   3: array[3]=45
*/
```

Für den Compiler sehen alle drei Zeilen der Direktive #define als eine lange Zeile aus. In diesem Beispiel wird auch das Zeichen ## verwendet, das als Merge-Operator bezeichnet wird und für das Zusammenführen von zwei Tokens zu einem in den #define Makros verwendet wird. Der Merge-Operator kann nicht der erste oder der letzte in der Definition eines Makros sein.

© 2000-2025, MetaQuotes Ltd.

Makrosubstitution (#define, #undef)

Der Compiler verwendet Direktiven des Präprozessors für eine vorbereitende Verarbeitung eines Quellcodes vor der Kompilierung. Die Direktive beginnt immer mit dem Zeichen # (Doppelkreuz), deswegen verbietet der Compiler, dieses Zeichen in den Namen von Variablen, Funktionen usw. zu verwenden.

Jede Direktive wird in einem separaten Eintrag beschrieben und ist bis zum Zeilenumbruch gültig. In einem Eintrag können Sie nur eine Direktive verwenden. Wenn der Eintrag der Direktive zu groß ist, kann er mithilfe des Zeichens '\' in mehrere Zeilen aufgeteilt werden. In diesem Fall gilt die nächste Zeile als eine Fortsetzung des Eintrags dieser Direktive.

Befehlende Anweisung #define kann verwendet werden, um mnemonische Namen den zuzuordnen. Es gibt zwei Formen:

```
#define identifier expression                    // parameterfreie Form
#define identifier(par1,... par8) expression     // parametrische Form
```

Befehlsanweisung #define ersetzt **expression** anstatt aller folgenden gefundenen Eintragungen von identifier im Ausgangstext. identifier wird erst dann ersetzt, wenn er einzelner token darstellt. identifier wird nicht ersetzt, wenn er ein Teil des Kommentars, der Zeile oder eines anderen laengeren Identifikators ist.

für den Identifikator der Konstante gelten dieselbe Regeln, wie für die Konstantennamen. Der wert kann jeder Art sein:

```
#define ABC               100
#define PI                3.14
#define COMPANY_NAME      "MetaQuotes Software Corp."
...
void ShowCopyright()
  {
   Print("Copyright  2001-2009, ",COMPANY_NAME);
   Print("https://www.metaquotes.net");
  }
```

expression kann aus mehreren tokens bestehen, wie Schluesselwoerter, Konstanten, Konstant- und Nichtkonstantausdrücke. **expression** beendetmit dem Zeilenende und kann nicht auf die naechste Zeile ubertragen werden.

Beispiel:

```
#define TWO         2
#define THREE       3
#define INCOMPLETE TWO+THREE
#define COMPLETE   (TWO+THREE)
void OnStart()
  {
   Print("2 + 3*2 = ",INCOMPLETE*2);
   Print("(2 + 3)*2 = ",COMPLETE*2);
  }
/* Ergebnis
```

© 2000-2025, MetaQuotes Ltd.

```
   2+3*2 = 8
   (2+3)*2 = 10
*/
```

Parametrische Form #define

Mit der parametrischen Form, werden alle folgenden gefunden Einträge von identifier werden durch expression auf der Grundlage der aktuellen Parameter ersetzt werden. Zum Beispiel,

```
 // Beispiel mit zwei Parameter a und b
#define A 2+3
#define B 5-1
#define MUL(a, b) ((a)*(b))

double c=MUL(A,B);
Print("c=",c);
/*
Expression double c=MUL(A,B);
ist gleichbedeutend mit double c=((2+3)*(5-1));
*/
// Ergebnis
// c=20
```

Achten Sie darauf, dass Parameter in Klammern eingeschlossen sind, wenn Sie den Parameter in expression verwenden, da dies nicht-offensichtlichen Fehlern, die schwer zu finden sind, vermeidet. Wenn wir den Code neu schreiben, ohne die Klammern, wird das Ergebnis anders sein:

```
 // Beispiel mit zwei Parameter a und b
#define A 2+3
#define B 5-1
#define MUL(a, b) a*b

double c=MUL(A,B);
Print("c=",c);
/*
Expression double c=MUL(A,B);
ist gleichbedeutend mit double c=2+3*5-1;
*/
// Ergebnis
// c=16
```

Bei Verwendung der parametrischen Form sind maximal 8 Parameter erlaubt.

```
 // richtige parametrische Form
#define LOG(text)  Print(__FILE__,"(",__LINE__,") :",text)   // ein Parameter - 'text'

 // falsche parametrische Form
#define WRONG_DEF(p1, p2, p3, p4, p5, p6, p7, p8, p9)   p1+p2+p3+p4 // mehr als 8 Para
```

Direktive #undef

Direktive #undef annulliert das früher definierten Makro.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#define MACRO

void func1()
  {
#ifdef MACRO
   Print("MACRO is defined in ",__FUNCTION__);
#else
   Print("MACRO is not defined in ",__FUNCTION__);
#endif
  }

#undef MACRO

void func2()
  {
#ifdef MACRO
   Print("MACRO is defined in ",__FUNCTION__);
#else
   Print("MACRO is not defined in ",__FUNCTION__);
#endif
  }

void OnStart()
  {
   func1();
   func2();
  }

/* Ergebnis:
 MACRO is defined in func1
 MACRO is not defined in func2
*/
```

Sehen Sie auch

Identifikatoren, Symbolkonstanten

© 2000-2025, MetaQuotes Ltd.

Programmeigenschaften (#property)

Bei jedem mql5-Programm können zusätzliche spezifische Parameter *#property* angegeben werden, die dem Client-Terminal helfen, Programme richtig zu bedienen. In erster Linie betrifft es Außeneinstellungen der Identifikatoren. Eigenschaften, beschrieben in Schaltdateien, werden völlig ignoriert. Eigenschaften müssen in der Haupt mql5-Datei vorbestimmt werden.

```
#property Identifikator Wert
```

Compiler schreibt in Einstellungen des ausführenden Moduls erklärte Werte.

Konstante	Typ	Beschreibung
icon	string	Pfad zur Datei mit dem Bild, das als ein Icon des EX5-Programms verwendet werden soll. Pfadangaberegel sind die gleichen wie für die Ressourcen. Die Eigenschaft muss in dem Hauptmodul mit dem MQL5 Quellcode angegeben werden. Die Icondatei muss im ICO-Format sein.
link	string	Verweis auf die Site der Gesellschaft-Hersteller
copyright	string	Name des Autors/Rechteinhabers
version	string	Programmversion, nicht mehr als 31 Zeichen
description	string	Kurze Textbeschreibung des mql5-Programms. Sie kann mehrere Textzeilen umfassen. Allerdings darf die gesamte Länge, einschließlich der Zeilenumbrüche, nicht 511 Zeichen überschreiten.
stacksize	int	Die Stack Größe eines MQL5 Programms. Ein Stack ausreichender Größe wird benötigt, wenn Funktionen rekursiv aufgerufen werden. Beim Start eines Skripts oder Expert Advisors wird ein Stack in Höhe von 8Mb reserviert, für Indikatoren gilt diese Eigenschaft nicht - das Stack ist immer gleich 1Mb. Beim Start im Strategietester wird immer ein Stack von 16 MB reserviert.
library		Bibliothek; wird von keiner Startfunktion zugeordnet, Funktionen mit dem Modifikator export können in andere mql5-Programme importiert werden
indicator_applied_price	int	gibt Default-Wert für Feld "Apply to" vor. Kann einer der Enumerationswerte ENUM_APPLIED_PRICEvorgegeben werden. Wenn die Eigenschaft nicht vorgegeben wird, wird PRICE_CLOSE als Default-Wert verwendet
indicator_chart_window		Indikator im Chartfenster ausgeben

© 2000-2025, MetaQuotes Ltd.

Konstante	Typ	Beschreibung
indicator_separate_window		Indikator in einem separaten Fenster ausgeben
indicator_height	int	Feste Höhe des Indikator-Unterfensters in Pixel (Eigenschaft INDICATOR_HEIGHT)
indicator_buffers	int	Anzahl der Puffer für Indikatorberechnung
indicator_plots	int	Anzahl der graphischen Reihen im Indikator
indicator_minimum	double	der untere Grenzwert der Skala für ein separates Fenster des Indikators
indicator_maximum	double	der obere Grenzwert der Skala für ein separates Fenster des Indikators
indicator_labelN	string	gibt Marke für N- graphische Serie vor, die im Fenster DataWindow dargestellt wird. Für graphische Serien, die mehrere Indikatorpuffer erfordern (DRAW_CANDLES, DRAW_FILLING und andere), werden die Markennamen durch Begrenzer vorgegeben ';'.
indicator_colorN	color	Farbe für Linienausgabe N, wo N - Nummer einer graphischen Serie; Nummerierung fängt mit 1 an
indicator_widthN	int	Linienbreite in der graphischen Serie, wo N - Nummer der graphischen Serie; Numerierung mit 1
indicator_styleN	int	Linienstil in der graphischen Serie, der mittels des Werte aus ENUM_LINE_STYLE angegeben wird. N - Nummer einer graphischen Serie, Nummerierung fängt mit 1 an
indicator_typeN	int	Typ der graphischen Darstellung, der durch den Wert aus ENUM_DRAW_TYPE angegeben wird . N - Nummer einer graphischen Serie, Nummerierung fängt mit 1 an
indicator_levelN	double	horizontaler Level N im eigenen Fenster des Indikators
indicator_levelcolor	color	Farbe der horizontalen Levels des Indikators
indicator_levelwidth	int	Breite der horizontalen Levels des Indikators
indicator_levelstyle	int	Stil der horizontalen Niveaus des Indikators
script_show_confirm		Ausgabe des Bestätigungsfensters vor dem Scriptablauf
script_show_inputs		Ausgabe des Fensters mit Eigenschaften vor dem Scriptablauf und Verbot, Bestätigungsfenster auszugeben

© 2000-2025, MetaQuotes Ltd.

Konstante	Typ	Beschreibung
tester_indicator	string	Name des Benutzerindikators im Format "*Name_des Indikators.ex5*". Die für Testenerforderliche Indikatoren werden automatisch aus Aufruf der Funktionen iCustom() bestimmt, wenn der entsprechende Parameter von der Konstantzeile vorgegeben ist. Für andere Fälle (Verwendung der Funktion IndicatorCreate() oder Verwendung einer nicht Konstantzeile im Parameter, der den Namen des Indikator vorgibt) ist diese Eigenschaft erforderlich.
tester_file	string	Dateiname für Tester mit Angabe der Erweiterung, in Doppelanführungszeichen (als Konstantzeile). Die angegebene Datei wird dem Tester für Arbeit übertragen. Eingabeparameter für Testen müssen immer angegeben werden (wenn sie erforderlich sind)
tester_library	string	Name der Bibliothek mit Erweiterung in doppelten Anführungszeichen. Eine Bibliothek kann sowohl die Erweiterung dll, als auch die Erweiterung ex5 haben. Die für das Testen erforderlichen Bibliotheken werden automatisch definiert. Aber wenn eine der Bibliotheken von einem benutzerdefinierten Indikator verwendet wird, muss man diese Eigenschaft verwenden.
tester_set	string	Der Name der Set-Datei mit den Werten und der Schrittweite der Input-Parameter. Die Datei wird dem Tester vor dem Testen oder der Optimierung übergeben. Die Dateiname muss samt einer Erweiterung und in doppelten Anführungszeichen als ein Constant-String angegeben werden. Wenn Sie den Namen des Expert Advisors und die Nummer der Version im Namen einer Set-Datei als "<expert_name>_<number>.set" angeben, wird diese dem Download-Menü der Versionen der Parameter unter der Nummer der Version <number> automatisch hinzugefügt. Der Name "MACD Sample_4.set" bedeutet zum Beispiel, dass dies eine Set-Datei für den Expert Advisor "MACD Sample.mq5" mit der Versionsnummer 4 ist. Um das Format zu untersuchen, ist es empfehlenswert, Einstellungen des Testens/der Optimierung im Strategietester zu speichern, und dann die auf solche Weise erstellte Set-Datei zu öffnen.

© 2000-2025, MetaQuotes Ltd.

Konstante	Typ	Beschreibung
tester_no_cache	string	Während einer Optimierung speichert der Strategietester alle Ergebnisse der ausgeführten Durchläufe im Optimierungs-Cache, in dem das Testergebnis für jeden Satz von Eingangsparameter gespeichert wird. Dies ermöglicht es, die bereits erzielten Ergebnisse bei einer erneuten Optimierung mit den gleichen Parametern zu verwenden, ohne Zeit durch eine erneute Berechnung zu verschwenden. Bei einigen Aufgaben (z.B. bei mathematischen Berechnungen) kann es jedoch notwendig sein, Berechnungen unabhängig von der Verfügbarkeit bereits erzielter Ergebnisse im Optimierungs-Cache durchzuführen. In diesem Fall sollte die Datei die Eigenschaft *tester_no_cache* enthalten. Die Testergebnisse werden weiterhin im Cache gespeichert, so dass Sie im Strategietester alle Daten zu den durchgeführten Durchläufen sehen können.
tester_everytick_calculate	string	Im Strategietester werden Indikatoren erst beim Zugriff auf ihre Daten berechnet, d.h. wenn die Werte von Indikatorpuffern angefordert werden. Dies ermöglicht eine deutlich schnellere Test- und Optimierungsgeschwindigkeit, da nicht bei jedem Tick Indikatorwerte bereitgestellt werden müssen. Durch die Angabe der Eigenschaft *tester_everytick_calculate* können Sie die erzwungene Berechnung des Indikators bei jedem Tick aktivieren. Die Indikatoren im Strategietester werden auch in den folgenden Fällen für jeden Tick zwingend berechnet: • Wenn Sie im visuellen Modus; testen. • Wenn der Indikator über eine der folgenden Funktionen verfügt: EventChartCustom, OnChartEvent, OnTimer; • Wenn der Indikator mit dem Compiler mit build Nummer kleiner als 1916 erstellt wurde. Diese Funktion gilt nur für den Strategietester, während in den Terminals die Indikatoren immer bei jedem empfangenen Tick berechnet werden.

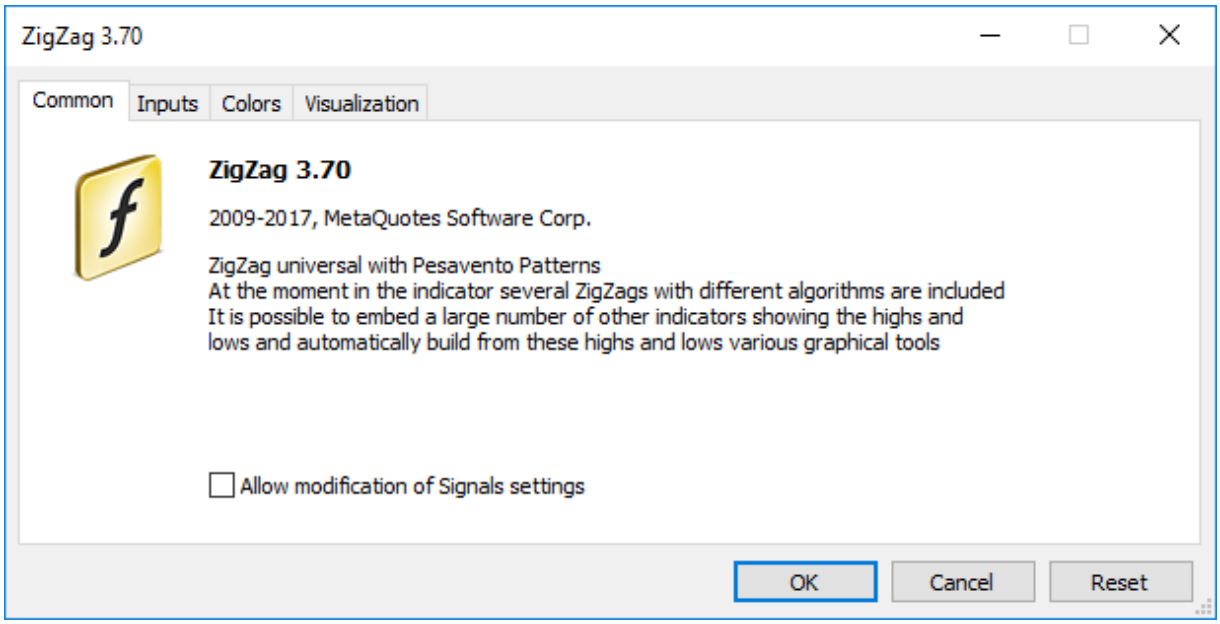
© 2000-2025, MetaQuotes Ltd.

Konstante	Typ	Beschreibung
optimization_chart_mode	string	Gibt den Diagrammtyp und die Namen von zwei Eingabeparameter an, die für die Visualisierung von Optimierungsergebnissen verwendet werden. Beispielsweise bedeutet "3d, InpX, InpY", dass die Ergebnisse in einem 3D-Diagramm mit den Koordinatenachsen basierend auf den getesteten InpX- und InpY-Parameterwerten dargestellt werden. Somit ermöglicht die Eigenschaft die Angabe von Parametern, die zur Anzeige des Optimierungsdiagramms und des Diagrammtyps verwendet werden, direkt im Programmcode. Mögliche Optionen: • "3d, input_parameter_name1, input_parameter_name2" means a 3D Visualizierungsdiagramm, das rotiert, vergrößert und verkleinert werden kann. Das Diagramm wird mit zwei Parametern erstellt. • "2d, input_parameter_name1, input_parameter_name2" bedeutet ein 2D-Gitterdiagramm, in dem jede Zelle je nach Ergebnis in einer bestimmten Farbe dargestellt wird. Das Diagramm wird mit zwei Parametern erstellt. • "1d, input_parameter_name1, input_parameter_name2" bedeutet ein lineares Diagramm, in dem die Ergebnisse nach dem angegebenen Parameter gereiht werden. Jeder Durchlauf wird als Punkt angezeigt. Das Diagramm wird auf Basis eines Parameters erstellt. • "0d, input_parameter_name1, input_parameter_name2" bedeutet ein reguläres Diagramm aus Ergebnissen, in der Reihenfolge der Entstehungszeit der Durchlaufergebnisse. Jeder Durchlauf wird als Punkt im Diagramm angezeigt. Eine Parameteranzeige ist nicht erforderlich, aber die angegebenen Parameter können für die manuelle Umschaltung auf andere Diagrammtypen verwendet werden. Optional können Sie nur den Diagrammtyp angeben, ohne einen oder beide Eingabeparameter anzugeben. In diesem Fall wählt das Terminal die erforderlichen Parameter aus, um das Optimierungsdiagramm anzuzeigen.

Beispiel für die Angabe der Beschreibung und Nummer einer Version

© 2000-2025, MetaQuotes Ltd.

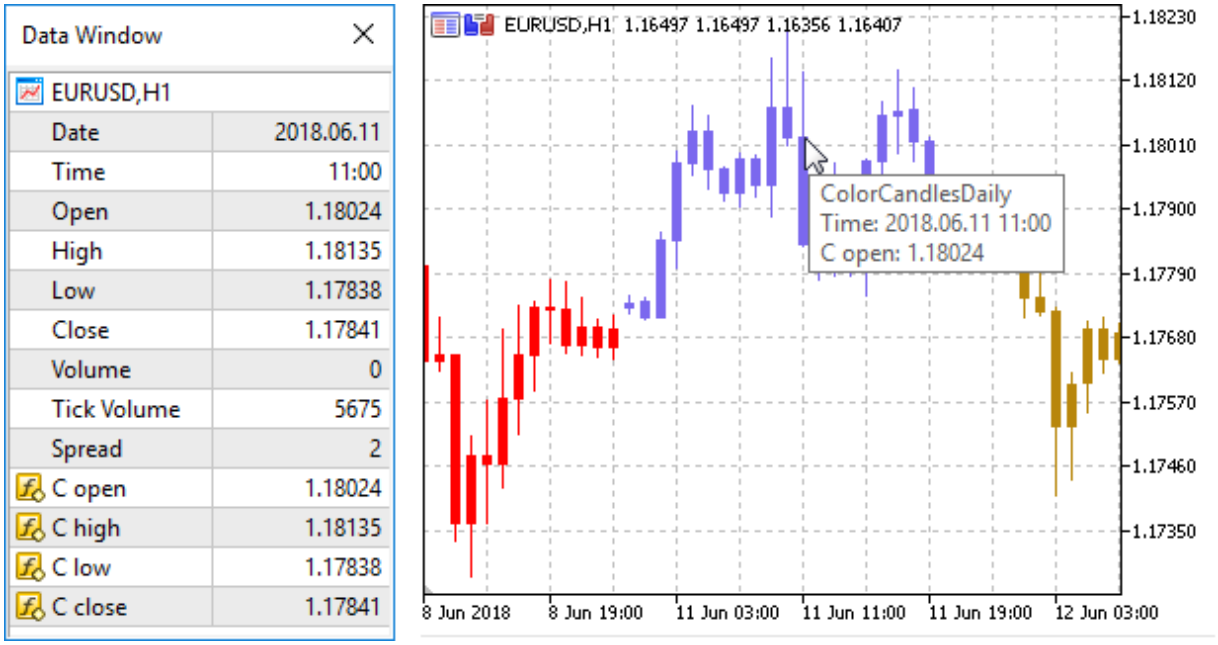
```
#property version     "3.70"      // aktuelle Expertversion
#property description "ZigZag universell mit Pesavento Patterns"
#property description "Zum jetzigen Zeitpunkt werden in Indikator mehrere ZigZag mit v
#property description "Es ist möglich, eine große Anzahl anderer Indikatoren einzubaue
#property description "Minima zeigen und aus diesen Minima und Maxima verschiedene gra
```



Beispiel für die Angabe eines separaten Labels für jeden Indikatorpuffer ("C open;C high;C low;C close")

```
#property indicator_chart_window
#property indicator_buffers 4
#property indicator_plots   1
#property indicator_type1   DRAW_CANDLES
#property indicator_width1  3
#property indicator_label1  "C open;C high;C low;C close"
```

© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

Include Dateien (#include)

Befehlszeile *#include* kann in jeder Programmstelle auftreten, gewoehnlich aber befinden sich alle Einschluesse am Anfang der Datei des Ausgangstextes.

Aufrufformat:

```
#include <Dateiname>
#include "Dateiname"
```

Beispiele:

```
#include <WinUser32.mqh>
#include "mylib.mqh"
```

Preprozessor ersetzt die Zeile *#include* <*Dateiname*> durch den Dateiinhalt WinUser32.mqh. Spitze Klammern bezeichnen, dass die Datei WinUser32.mqh dem Standardkatalog entnommen werden wird (gewoehnlich ist es Katalog_*des Terminals\MQL5\Include*). Laufendes Katalog wird nicht durchgesehen.

Wenn der Dateiname in Anführungszeichen zteht, wird die Suche im laufenden Katalog (wo sich Hauptdatei des Ausgangstextes befindet) durchgeführt. Standardkatalog wird nicht durchgesehen.

Sehen Sie auch

Standardbibliothek, Import der Funktionen

© 2000-2025, MetaQuotes Ltd.

Import der Funktionen (#import)

Import der Funktionen wird von kompilierten Moduls MQL5 (Dateien *.ex5) und aus Moduls des Operationssystems (Dateien *.dll) durchgeführt. Modulname wird in der befehlenden Anweisung *#import* angegeben. Damit der Compiler den Aufruf der importierenden Funktion richtig durchführen und richtige Parameterübergabe gestalten kann, ist völlige Beschreibung der Funktionen notwendig. Funktionsbeschreibungen folgen unmittelbar der befehlenden Anweisung *#import "Modulname"*. Neuer Befehl *#import* (möglich auch ohne Parameter) beendet Beschreibungsblock der importierender Funktionen.

```
#import "Dateiname"
    func1 define;
    func2 define;
    ...
    funcN define;
#import
```

Zu importierende Funktionen können verschiedene Namen haben. Importiert kann aus verschiedenen Modulen der Funktion mit gleichen Namen gleichzeitig werden. Zu importierende Funktionen können Namen haben, die mit den Namen interner Funktionen zusammenfallen. Operation von Kontexterlaubnis bestimmt, welche Funktion aufgerufen muss.

Die Suchreihenfolge nach der Datei, die nach der Schlüsselwort #import angegeben ist, ist in der Sektion Aufruf der Importfunktionen beschrieben.

Da sich zu importierende Funktionen ausser compilierten Moduls befinden, kann der Compiler Richtigkeit der übertragenen Parameter nicht prüfen. Darum um Ausführungsfehler zu vermeiden muss man genau Inhalt und Ordnung der Parameter beschreiben, die in importierte Funktionen übertragen werden. Parameter, übertragen in importierte Funktionen (sowie aus EX5, als auch aus DLL-Modulen) können Default-Werte haben.

In importierten Funktionen können als Parameter nicht verwendet werden:

- Anzeiger (*);
- Verweise auf Objekte, die dynamische Felder und/oder Anzeiger enthalten.

In die aus DLL importierten Funktionen können als Parameter keine Klassen, Zeilenfelder oder komplizierte Objekte übertragen werden, die Zeilen und/oder dynamische Felder verschiedener Typen enthalten.

Beispiele:

```
#import "stdlib.ex5"
string ErrorDescription(int error_code);
int    RGB(int red_value,int green_value,int blue_value);
bool   CompareDoubles(double number1,double number2);
string DoubleToStrMorePrecision(double number,int precision);
string IntegerToHexString(int integer_number);
#import "ExpertSample.dll"
int    GetIntValue(int);
double GetDoubleValue(double);
string GetStringValue(string);
double GetArrayItemValue(double &arr[],int,int);
```

© 2000-2025, MetaQuotes Ltd.

```
bool   SetArrayItemValue(double &arr[],int,int,double);
double GetRatesItemValue(double &rates[][6],int,int,int);
#import
```

für Importieren der Funktionen während der Durchführung eines mql5-Programms wird frühe Bindung verwendet. Das bedeutet, dass Bibliothek während der Ladung des ex5-Programm geladen wird, das sie verwendet.

Es ist nicht empfehlenswert, den vollständig qualifizierten Namen des geladeten Moduls des Typs *Drive:\Directory\FileName.Ext* zu verwenden. Die MQL5 Bibliotheken werden aus dem Ordner *terminal_dir\MQL5\Libraries* geladen.

Wenn die zu importierende Funktion verschiedene Aufrufvarianten für die 32- und 64-Bit-Versionen von Windows hat, müssen beide importiert werden und die benötigte Variante der Funktion muss mithilfe der Variablen _IsX64 explizit aufgerufen werden.

Beispiel:

```
#import "user32.dll"
//--- für das 32-Bit-System
int    MessageBoxW(uint hWnd,string lpText,string lpCaption,uint uType);
//--- für das 64-Bit-System
int    MessageBoxW(ulong hWnd,string lpText,string lpCaption,uint uType);
#import
//+------------------------------------------------------------------+
//| MessageBox_32_64_bit verwendet die richtige MessageBoxW() Version|
//+------------------------------------------------------------------+
int MessageBox_32_64_bit()
 {
   int res=-1;
   //--- wenn wir das 64-Bit-Windows verwenden
   if(_IsX64)
    {
      ulong hwnd=0;
      res=MessageBoxW(hwnd,"Beispiel für den Aufruf der 64-Bit-Version von MessageBoxW
     }
   else  // wir verwenden das 32-Bit-Windows
    {
      uint hwnd=0;
      res=MessageBoxW(hwnd,"Beispiel für den Aufruf der 32-Bit-Version von MessageBoxW
     }
   return (res);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
+//---+
```

© 2000-2025, MetaQuotes Ltd.

```
   int ans=MessageBox_32_64_bit();
   PrintFormat("MessageBox_32_64_bit returned %d",ans);
  }
```

Importieren von Funktionen aus den .NET-Bibliotheken

Um mit .NET-Bibliotheksfunktionen zu arbeiten, importieren Sie einfach die DLL selbst, ohne bestimmte Funktionen zu definieren. MetaEditor importiert automatisch alle Funktionen, mit denen man arbeiten kann:

- Einfache Strukturen (einfache, alte Daten oder POD, plain old data) — Strukturen, die nur einfache Datentypen beinhalten.
- 'Public static' Funktionen mit Parametern, bei denen nur einfache Typen und POD-Strukturen oder deren Arrays verwendet werden.

Um Funktionen aus der Bibliothek aufzurufen, importieren Sie sie einfach:

```
#import "TestLib.dll"
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int x=41;
   TestClass::Inc(x);
   Print(x);
  }
```

Der C#-Code der Funktion Inc() aus TestClass schaut wie folgt aus:

```
public class TestClass
  {
   public static void Inc(ref int x)
     {
      x++;
     }
  }
```

Als Ergebnis der Ausführung gibt das Skript den Wert 42 zurück.

Siehe auch

Include Dateien

© 2000-2025, MetaQuotes Ltd.

Bedingte Kompilierung (#ifdef, #ifndef, #else, #endif)

Der Compiler verwendet Direktiven des Präprozessors für eine vorbereitende Verarbeitung eines Quellcodes vor der Kompilierung. Die Direktive beginnt immer mit dem Zeichen # (Doppelkreuz), deswegen verbietet der Compiler, dieses Zeichen in den Namen von Variablen, Funktionen usw. zu verwenden.

Jede Direktive wird in einem separaten Eintrag beschrieben und ist bis zum Zeilenumbruch gültig. In einem Eintrag können Sie nur eine Direktive verwenden. Wenn der Eintrag der Direktive zu groß ist, kann er mithilfe des Zeichens '\' in mehrere Zeilen aufgeteilt werden. In diesem Fall gilt die nächste Zeile als eine Fortsetzung des Eintrags dieser Direktive.

Durch Direktiven für die bedingte Kompilierung von Präprozessor cessor können Sie Teil des Programms kompilieren oder vermissen, abhängig von der Erfüllung bestimmter Bedingungen.

Der Bedingung kann eine der nachfolgend beschriebenen Formen annehmen.

```
#ifdef identifier
   // Code hier wird kompiliert, wenn identifier bereits für die Präprozessor in Befeh
#endif
```

```
#ifndef identifier
   // Code hier wird kompiliert, wenn identifier für die Präprozessor in Befehl #defin
#endif
```

Jedem der Befehlen der bedingten Kompilierung kann eine beliebige Anzahl von Zeilen folgen, die möglicherweise der Befehl #else mit Ende #endif enthalten. Wenn die Bedingung erfüllt ist, werden die Zeilen zwischen #else und #endif ignoriert. Ist die Bedingung nicht erfüllt, dann alle Zeilen zwischen dem Prüfung und Befehl #else ignoriert werden. Wenn es dieser Befehl gibt nicht, dann zwischen dem Prüfung und dem Befehl #endif.

Beispiel:

```
#ifndef TestMode
   #define TestMode
#endif
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   #ifdef TestMode
      Print("Test mode");
   #else
      Print("Normal mode");
   #endif
  }
```

Abhängig vom Typ des Programms und Standardkompilierungmodus, werden Makros wie folgt definiert:

© 2000-2025, MetaQuotes Ltd.

Makro __ MQL5__ ist bei der Kompilierung der Datei *.mq5 verfügbar. Bei der Kompilierung von *.mq4 ist Makro __ MQL4__ verfügbar.
Makro _DEBUG ist bei der Kompilierung für die Fehlersuche verfügbar.
Makro _Release ist bei der Kompilierung für anders als Fehlersuche verfügbar.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   #ifdef __MQL5__
      #ifdef _DEBUG
         Print("Hello from MQL5 compiler [DEBUG]");
      #else
        #ifdef _RELEASE
           Print("Hello from MQL5 compiler [RELEASE]");
        #endif
     #endif
   #else
      #ifdef __MQL4__
         #ifdef _DEBUG
           Print("Hello from MQL4 compiler [DEBUG]");
        #else
           #ifdef _RELEASE
              Print("Hello from MQL4 compiler [RELEASE]");
           #endif
        #endif
     #endif
   #endif
  }
```

© 2000-2025, MetaQuotes Ltd.

Objektorientiertes Programmieren

Objektorientiertes Programmieren ist datenorientiertes Programmieren, wobei Daten und Verhalten unaufloeslich miteinander verbunden sind. Zusammen sind Daten und Verhalten eine Klasse, und Objekte sind Klassenexemplare.

Bestandteile des objektorientierten Programmieren sind:

- Inkapsulation und Erweiterungsfaehigkeit der Typen
- Vererbung
- Polimorphismus
- überladung
- Virtuelle Funktionen

Objektorientiertes Programmieren betrachtet Berechnungen als Verhaltensmodellierung. Das was modelliert wird sind Objekte, dargestellt durch rechnerische Abstraktionen. ZB. wollen wir das allgemeinbekannte Spiel "Tetris" schreiben, dafür muessen wir lernen, die Erscheinung einer Zufallfigur zu modellieren,die aus vier Kaestchen besteht, miteinander durch Ränder verbunden. Auch muss man die Fallgeschwindigkeit der Figur regeln, Rotierungs- und Schubsoperationen der Figur einstellen. Umsetzungen der FIgur auf Display sind durch Glasgrenzen begrenzt, diese Firderung muessen wir auch modellieren. Außerdem muessen die gefuellten Reihen der Kaestchen im Glas entfernt werden und die erarbeiteten Punkte berechnet werden muessen.

So,fordert dieses einfaches Spiel das Schaffen vo mehreren Modellen - Figurmodell, Glasmodell, Modell der Figurbewegung im Glas usw. Alle diesen Modelle sind Abstraktionen, dargestellt von Berechnungen im Computer. Für Beschreibung dieser Modelle wird der Begriff abstrakter Datentyp ADT (oder komplexer Datentyp) verwendet. Streng gesagt ist Bewegungsmodell der "Figur" im "Glas" kein Datentyp, sondern Gesamtheit von Datenoperationen des Typs "Figur", die Datenbegrenzungen des Typs "Glas" verwenden.

Objekte sind Variablen der Klasse. Objektorientiertes Programmieren ermöglicht abstrakte Datentypen einfach zu erzeugen und zu verwenden. Objektorientiertes Programmieren verwendet Mechanismus von Vererbung. Vererbung ist deswegen vorteilhaft, weil es die Erhaltung von sekundaeren Typen aus den vom Benutzer bestimmten Datentypen ermöglicht. für die Figurerzeugung im Tetris ist es bequem, vor allem Basisklasse Shape zu erzeugen, auf deren Grund sekundaere Typen aller sieben moeglichen Figuren in Tetris erhalten werden. In der Basisklasse ist Figurverhalten bestimmt, in den sekundaeren ist Realisierung des Verhaltens jeder konkreten Figur detailiert.

In OPP sind Objekte für Ihr Verhalten verantwortlich. Entwickler von ADT muss darin Kode für die Beschreibung jedes Verhaltens einbeschliessen, das von den ensprechenden Objekten zu erwarten ist. Die Tatsache, dass das Objekt für sein Verhalten selbst verantwortlich ist, vereinfacht bedeutend das Programmieren für Benutzer dieses Objektes.

Wenn wir auf dem Display Figur zeichen wollen, muessen wir kennen, wo sich ihr Zentrum befunden wird und wie man sie zeichnen muss. Wenn es einzelne Figur genau kennt, wie sie sich zeichnen muss, muss der Programmierer bei der Verwendung dieser Figur dem Objekt die Nachricht "sich zeichnen" übertragen.

Sprache MQL5 ist der Sprache C++ aehnlich und darin ist auch der Mechanismus der Inkapsulation für Realisierung von ADT realisiert. Inkapsulation vereinigt in sich einerseits interne Details der Realisierung des konkreten Typs und andererseits die aussen zugaenglichen Funktionen, die die

© 2000-2025, MetaQuotes Ltd.

Objekte des Typs bewirken können. Realisierungsdetails können für das Programm, das diesen Typ verwendet, unzugaenglich sein.

Zum Begriiff OPP gehört eine Reihe von Konzeptionen einschliesslich folgende:

- Modellieren der Handlungen aus der realen Welt
- Vorhandensein von Datentypen bestimmt vom Benutzer
- Verhuellung der Realisierungsdetails
- Moeglichkeit von mehrmaliger Verwendung durch Vererbung
- Interpretieren der Funktionsaufrufe in der Durchführungsetappe

Einige von diesen Begriffen sind ganz undeutig, einige - abstrakt, andere haben allgemeinen Charakter.

© 2000-2025, MetaQuotes Ltd.

Inkapsulation und Erweiterungsfaehigkeit der Typen

OOP ist balancierter Zugang zum Schreiben von Software. Daten und Verhalten sind zusammen verpackt. Diese Inkapsulation erzeugt die vom Benutzer bestimmten Datentypen, die eigene Sprachdatentypen und mit ihnen zusammenwirkende Typen erweitern. Erweiterungsfaehigkeit der typen ist die Moeglichkeit, die vom Benutzer bestimmten Datentypen der Sprache zuzusetzen, die so einfach verwendet werden können, wie Grundtypen.

Abstrakter Datentyp, zB. die Zeile ist die Beschreibung des idealen, allgemeinbekannten Verhaltens des Typs. Benutzer der Zeile weiss, dass Operationen, wie Konkatenanz oder Drucken bestimmtes Verhalten hat. Operationen on Konkatenanz und Drucken heissen Methoden.

Bestimmte ADT Realisierung kann Beschraenkungen haben; die Laenge von Zeilen kann ZB. begrenzt werden. Diese Beschraenkungen beeinflussen Verhalten, offen für alle. Zu gleicher Zeit beeinflussen interne oder geschlossene Realisierungsdetails das nicht, wie der Benutzer das Objekt sieht. ZB. die Zeile wird oft als Feld realisiert; dabei ist interne Grundadresse der Elemente dieses Feldes und sein Name für Benutzer nicht bedeutend sind.

Inkapsulation ist die Faehigkeit, interne Details zu verhuellen Versorgung mit dem offenen Interface zu dem vom Benutzer bestimmten Typ. In MQL5 werden wie in C++, Definitionen der Klasse und der Struktur (class und struct) mit den Schluesselwoertern des Zuganges private (geschlossen), protected (geschuetzt) und public (offen) verwendet.

Schluesselwort public zeigt, dass Zugang zu Gliedern, die ihm folgen, ist offen ohne Begrenzungen. Ohne dieses Schluesselwort sind Klassenglieder als Standardeinstellung geschlossen. Geschlossene Glieder sind erst den Funktionen-Gliedern der eigenen Klasse.

Geschuetzte Klassenglieder sind nicht nur für die Funktionen-Glieder der eigener Klasse, sondern auch für die Klassen-Nachfolger zugaenglich. Offene Glieder sind für jede Funktion innerhalb des Sichtbereiches der Klassenerklärung zugaenglich. Geschlossenheit ermöglicht Teil der Klassenrealisierung zu verhuellen, und verhindert damit nicht vorausgesehene Änderungen der Datenstruktur. Zugangsbegrezung oder Datenverhuellung ist eine Eigenschaft des objektorientiertes Programmieren.

Gewoehnlich versucht man, Klassenglieder zu schuetzen und sie mit dem Modifikator protected zu erklären, Einstellung und Lesen der Werte dieser Glieder ist durch sogenannte set- und get-Methoden durchgeführt, die vom Zugangsmodifikator , public bestimmt werden

Beispiel:

```
class CPerson
  {
protected:
   string            m_name;                    // Name
public:
   void              SetName(string n){m_name=n;}// stellt Name ein
   string            GetName(){return (m_name);} // gibt Name zurück
  };
```

Solcher Zugang hat eine Reihe der Vorteile. Erstens kann nach dem Namen der Funktion verstanden werden, was sie macht - ob sie den Wert des Klassenglieders einstellt oder erhaelt. Zweitens können

© 2000-2025, MetaQuotes Ltd.

wir in der Zukunft Typ der Variable m_name in der Klasse CPerson oder in einer seiner sekundaeren Klasse verändert werden, wenn erforderlich.

In diesem Fall ist es genug, Realisierung der Funktionen SetName() und GetName() zu verändern, die Objekte der Klasse CPerson können im Programm ohne Änderungen im Kode verwende werden, denn der Benutzer wird nicht wissen, dass sich Datentyp m_name verändert hat.

Beispiel:

```
struct Name
  {
   string            first_name;                  // Vorname
   string            last_name;                   // Name
  };

class CPerson
  {
protected:
   Name              m_name;                      // Vorname
public:
   void              SetName(string n);
   string            GetName(){return(m_name.first_name+" "+m_name.last_name);}
private:
   string            GetFirstName(string full_name);
   string            GetLastName(string full_name);
  };

void CPerson::SetName(string n)
  {
   m_name.first_name=GetFirstName(n);
   m_name.last_name=GetLastName(n);
  }

string CPerson::GetFirstName(string full_name)
  {
   int pos=StringFind(full_name," ");
   if(pos>0) StringSetCharacter(full_name,pos,0);
   return(full_name);
 ;}

string CPerson::GetLastName(string full_name)
  {
   string ret_string;
   int pos=StringFind(full_name," ");
   if(pos>0) ret_string=StringSubstr(full_name,pos+1);
   else      ret_string=full_name;
   return(ret_string);
  }
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Datentypen

© 2000-2025, MetaQuotes Ltd.

Vererbung

Eigenartigkeit von OOP besteht in der Aufmunterung zu Wiederverwendung des Kodes mittels Vererbung. Die neue Klasse wird von der vorausgehenden erzeugt, die Basisklasse genannt wird. Sekundäre Klasse verwendet die Glieder der Basisklasse, kann aber sie verändern und erweitern.

Viele Typen sind Variationen der vorhandenen Typen. Oft ist es ermüdend einen neuen Koden für jeden auszuarbeiten. Außerdem erscheinen mit neuem Kode neue Fehler. Sekundäre Klasse erbt die Beschreibung der Basisklasse und macht es unnötig, Kode wieder auszuarbeiten und zu testen. Vererbungverhalten sind hierarchisch.

Hierarchie ist die Methode, die Elemente in in voller Vielfältigkeit und Kompliziertheit zu kopieren ermöglicht. Sie leitet Objektklassifizierung ein. ZB gibt es im periodischen System von Elementen Gase. Sie haben Eigenschaften, die allen Systemelementen eigen sind.

Neutrale Gase stellen die nächste wichtige Subklasse dar. Hierarchie besteht darin, dass neutrales Gas, zB. Argon ist ein Gas, und Gas ist ein Systemelement. Solche Hierarchie ermöglicht das Verhalten von neutralen Gasen leicht zu interpretieren. Wir wissen, dass ihre Atome Protonen und Elektronen haben, was für andere Elemente auch typisch ist.

Wit wissen, dass sie beim Raumtemperatur gasförmig sind, wie alle Gase. Wir wissen, dass kein Gas aus der Subklasse neutrale Gase mit anderen Elementen reagiert und das ist die Eigenschaft aller neutralen Gase.

Betrachten wir Vererbung am Beispiel mit geometrischen Figuren. Für Beschreibung aller Vielfältigkeit von einfachen Figuren (Kreis, Dreieck, Rechteck, Quadrat usw.) ist es am besten eine Basisklasse zu erzeugen (ADT), die Vorfahr aller sekundären Klassen ist.

Erzeugen wir eine Basisklasse CShape, in der es nur allgemeine Glieder gibt, die die Figur beschreiben. Diese Glieder beschreiben die Eigenschaften, die jeder Figur eigen sind - Figurtyp und Koordinaten des Bezugspunktes.

Beispiel:

```
//--- Basisklasse Figur
class CShape
  {
protected:
   int       m_type;                    // Figurtyp
   int       m_xpos;                    // X - Koordinate des Bezugspunktes
   int       m_ypos;                    // Y - Koordinate des Bezugspunktes
public:
             CShape(){m_type=0; m_xpos=0; m_ypos=0;} // Konstruktor
   void      SetXPos(int x){m_xpos=x;} // stellen wir X ein
   void      SetYPos(int y){m_ypos=y;} // stellen wir Y ein
  };
```

Weiter erzeugen wir von der Basisklasse sekundäre Klassen, in die wir notwendige Felder zufügen, die jede konkrete Klasse präzisieren. Für die Figur Circle(Kreis) ist es notwendig das Glied zuzufügen, das Radiuswert enthält. Figur Quadrate (Quadrat) ist durch den Wert der Quadratseite charakterisiert. Darum werden sekundäre Klassen, die von der Basisklasse CShape geerbt werden, folgenderweise erklärt :

© 2000-2025, MetaQuotes Ltd.

```
//--- sekundaere Klasse Kreis
class CCircle : public CShape        // nach Doppelpunkt wird Basisklasse angegeben,
  {                                    // von der Vererbung erfolgt
private:
   int             m_radius;           // Kreisradius

public:
                   CCircle(){m_type=1;}// Konstruktor, Typ ist 1 gleich
  };
```

für Quadrat sieht Klassenerklärung ähnlicherweise aus:

```
//--- sekundaere Klasse Quadrat
class CSquare : public CShape        // nach Doppelpunkt wird Basisklasse angegeben ,
  {                                    // von der Vererbung erfolgt
private:
   int            m_square_side;       // Quadratseite

public:
                   CSquare(){m_type=2;} // Konstruktor ist 2 gleich
  };
```

Es muss bemerkt werden, dass bei der Ezeugung eines Objektes vor allem Konstruktor einer Basisklasse, und dannKonstruktor einer sekundären Klasse aufgerufen wird. Bei der Entfernung eines Objektes wird zuerst Destruktor einer sekundären Klasse und dann Destruktor einer Basisklasse aufgerufen.

So nach der Erklärung allgemeiner Glieder in einer Basisklasse können wir in sekundäre Klassen zusätzliche Glieder hinzufügen, die eine konkrete Klasse präzisieren. Vererbung erlaubt große Kodebibliotheken zu erzeugen, die mehrmals und wiederholt verwendet werden können.

Syntax der Erzeugung der sekundären Klasse von der schon vorhandenen sieht so aus:

```
class Klassenname :
          (public | protected | private) opt  Name der Basisklasse
  {
    Gliedererklärungen
  };
```

Einer der Aspekte der sekundären Klasse ist Sichtbarkeit (Offenheit) seiner Glieder-Nachfolger. Schlüsselwörter public, protected und private werden verwendet, um anzudeuten, inwieweit Glieder der Basisklasse für Glieder der sekundären Klasse zugänglich sind. Verwendung im Namen der sekundären Klasse der Schlüsselklasse public, der nach Doppelpunkt folgt, bedeutet, dass geschützte und offene (protected und public) Glieder der Basisklasse CShape wie geschützte und offene Glieder der sekundären Klasse CCircle geerbt werden müssen.

Geschlossene Glieder der Basisklasse sind für sekundäre Klasse nicht zugänglich. Offene Vererbung bedeutet ach, dass sekundäre Klassen (CCircle und CSquare) CShape sind. D.h. Quadrat (CSquare) ist eine Figur (CShape), eine Figur mus aber nicht unbedingt ein Quadrat sein.

© 2000-2025, MetaQuotes Ltd.

Sekundäre Klasse ist Modifikation der Basisklasse; sie erbt geschützte und offene Glieder der Basisklasse. Erst Konstruktors und Destruktors der Basisklasse können nicht geerbt werden. Oft werden in die sekundäre Klasse neue Glieder zu Gliedern der Basisklasse hinzugefügt.

Sekundäre Klasse kann Realisierung der Funktionen-Glieder einschließen, die sich von der Basisklasse unterscheidet. Das hat nichts mit Überladung zu tun, wenn der Sinn desselben Funktionsnamens verschieden für verschiedene Signaturen sein kann.

Beim geschützten Vererbung werden offene und geschützte Glieder der Basisklasse zu offenen und geschützten Gliedern der sekundären Klasse. Beim geschlossenen Vererbung werden offene und geschützte Glieder de Basisklasse zu geschlossenen Gliedern der sekundären Klasse.

Beim geschützten und geschlossenen Vererbung stimmt es nicht, dass Objekt der sekundären Klasse Objekt der Basisklasse ist. Geschütztes und geschlossenes Vererbung treten selten auf und müssen sehr vorsichtig verwendet werden.

Es sollte klar sein, dass die Art der Vererbung (public, protected oder private) beeinflusst nicht die Mittel **für den Zugang an die Glieder der Basisklassen in der Vererbungshierarchie der abgeleiteten Klasse (Erben)**. In jeder Art der Vererbung, nur Glieder einer Basisklasse, die mit dem Zugang-Spezifizierer public und protected deklariert worden, sind aus abgeleiteten Klassen erreichbar. Betrachten wir das obige Beispiel:

```
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
//+------------------------------------------------------------------+
//| Beispielklasse mit mehreren Arten des Zugriffs                   |
//+------------------------------------------------------------------+
class CBaseClass
  {
private:              //--- Privater Mitglied ist von den abgeleiteten Klassen nicht zu
   int                m_member;
protected:            //--- Geschützter Mitglied ist von der Basisklasse und ihren abge
   int                Member(){return(m_member);}
public:               // Konstruktor der Klasse ist allen zugänglich
                      CBaseClass(){m_member=5;return;};
private:              // Private Methode für die Zuweisung eines Wertes an den Mitglied
   void               Member(int value) { m_member=value;};

  };
//+------------------------------------------------------------------+
//| Die abgeleitete Klasse mit Fehlern                               |
//+------------------------------------------------------------------+
class CDerivaed: public CBaseClass // Es ist nicht notwendig public-Vererbung anzugebe
  {
public:
   void Func() // In der abgeleiteten Klasse definieren wir eine Funktion mit Aufrufen
     {
      //--- Versuch private Mitglied der Basisklasse zu ändern
      m_member=0;         // Fehler, ist ein privates Mitglied der Basisklasse nicht zu
```

© 2000-2025, MetaQuotes Ltd.

```
      Member(0);          // Fehler, ist ein privates Mitglied der Basisklasse in der a
      //--- Lesen einer Basisklassenmitglied
      Print(m_member);    // Fehler, ist ein privates Mitglied der Basisklasse nicht zu
      Print(Member());    // Kein Fehler, offenes Mitglied der Basisklasse ist immer zu
     }
  };
```

In diesem Beispiel hat Klasse CBaseClass nur eine öffentliche Methode - den Konstruktor. Konstruktoren sind automatisch aufgerufen, wenn ein Objekt der Klasse erstellt wird. So kann man nicht von außen den privaten Mitglieder m_member und die geschützte Methode Member() aufrufen. Aber mit einer offenen (public) Vererbung ist die Methode der Basisklasse Member() von den abgeleiteten Klassen zugänglich.

In der geschützte (protected) Vererbung, werden alle Basisklassenmitglieder mit einem offenen und geschützten Zugriff geschützt. Dies bedeutet, dass, wenn das öffentliche Datenelemente und Methoden der Basisklasse von außen zugänglich sind, dann in geschützter Vererbung sind sie nur aus den Klassen des Kindes und seiner anschließenden abgeleiteten Klassen zugänglich.

```
//+------------------------------------------------------------------+
//| Beispielklasse mit mehreren Arten des Zugriffs                   |
//+------------------------------------------------------------------+
class CBaseMathClass
  {
private:              //--- Privater Mitglied ist von den abgeleiteten Klassen nicht zu
   double             m_Pi;
public:               //--- Erhalten und angeben Wert für m_Pi
   void               SetPI(double v){m_Pi=v;return;};
   double             GetPI(){return m_Pi;};
public:               // Konstruktor der Klasse ist allen zugänglich
                      CBaseMathClass() {SetPI(3.14);  PrintFormat("%s",__FUNCTION__);};
  };
//+------------------------------------------------------------------+
//| Abgeleitete Klasse, in deren m_Pi kann nicht verändert werden    |
//+------------------------------------------------------------------+
class CProtectedChildClass: protected CBaseMathClass // Geschützte Vererbung
  {
private:
   double             m_radius;
public:               //--- Offene Methoden in der abgeleiteten Klasse
   void               SetRadius(double r){m_radius=r; return;};
   double             GetCircleLength(){return GetPI()*m_radius;};
  };
//+------------------------------------------------------------------+
//| Funktion started den Skript                                      |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Bei der Erstellung des Kindes wird Basisklassenkonstruktor automatisch aufgerufe
   CProtectedChildClass pt;
//--- Radius angeben
```

© 2000-2025, MetaQuotes Ltd.

```
   pt.SetRadius(10);
   PrintFormat("Length=%G",pt.GetCircleLength());
//--- Wenn den folgenden String benutzen, bekommen wir eine Fehlermeldung beim Kompil
// pt.SetPI(3);

//--- und jetzt deklarieren wir eine Variable einer Basisklasse und versuchen, eine K
   CBaseMathClass bc;
   bc.SetPI(10);
//--- Ergebnis
   PrintFormat("bc.GetPI()=%G",bc.GetPI());
  }
```

Dieses Beispiel zeigt, dass die Methoden SetPI() und GetPi() in der Basisklasse CBaseMathClass offen sind und von überall angerufen werden können. Aber zur gleichen Zeit für sein Kind CProtectedChildClass können die Aufrufe dieser Methoden nur aus Methoden der Klasse CProtectedChildClass oder ihren Kinder gemacht werden.

Mit einer privaten Vererbung werden alle Mitglieder der Basisklasse mit Zugriff public und protected geschlossen, und mit weiterer Vererbung ist Aufruf dieser Mitglieder unmöglich.

In MQL5 gibt es kein multiple Vererbung.

Sehen Sie auch

Strukturen und Klassen

© 2000-2025, MetaQuotes Ltd.

Polymorphismus

Polymorphismus ist die Moeglichkeit für Objekte verschiedener Klassen verbunden durch Vererbung beim Zugriff zu derselben Funktion-Element verschieden zu reagieren. Das hilft, universelle Mechanismen zu schaffen, die nicht nur das Verhalten der Basisklasse beschreiben, sondern auch das Verhalten der Klassen-Nachfolger.

Setzen wir die Ausarbeitung der Basisklasse CShape fort, in der wir die Funktion-Glied GetArea() definieren, die für Berechnung der Figurflaeche bestimmt ist. In allen Klassen-Nachfolger, erzeugt durch Vererbung von der Basisklasse, definieren wir die Funktion entsprechend den Regeln der Flaechenberechnung für konkrete Figur um.

für Quadrat (Klasse CSquare) wird die Flaeche durch Seiten, für Kreis (Klasse CCircle) durch Radius berechnet usw. Wir können Feld schaffen für Speichern von Objeketen des Typs CShape, in dem sowohl das Objekt der Basisklasse, als auch alle ihre Nachfolger gespeichert werden können. Im Nachhinein können wir dieselbe Funktion für jedes Element dieses Feldes aufrufen.

Beispiel:

```
//--- Basisklasse
class CShape
  {
protected:
   int            m_type;                // Figurtyp
   int            m_xpos;                // X - Koordinate des Bezugspunktes
   int            m_ypos;                // Y - Koordinate des Bezugspunktes
public:
   void           CShape(){m_type=0;};   // Konstruktor, Typ ist Null gleich
   int            GetType(){return(m_type);};// gibt Figurtyp zurück
virtual
   double         GetArea(){return (0); }// gibt Figurflaeche zurück
  };
```

Jetzt haben alle Variablen die Funktion-Glied getArea(), die Nullwert zurückgibt. Realisierung dieser Funktion wird in jedem Nachfolger verschieden sein.

```
//--- sekundaere Klasse Kreis
class CCircle : public CShape            // nach Doppelpunkt wird Basisklasse angegebe
  {                                      // von der Vererbung erfolgt
private:
   int            m_radius;              // Kreisradius

public:
   void           CCircle(){m_type=1;};  // Konstruktor, Typ ist 1 gleich
   void           SetRadius(double r){m_radius=r;};
   virtual double GetArea(){return (3.14*m_radius*m_radius);}// Kreisflaeche
  };
```

Für Quadrat sieht Klassenerklärung ähnlich aus:

```
//--- sekundaere Klasse Quadrat
class CSquare : public CShape            // nach Doppelpunkt wird Basisklasse angegebe
```

© 2000-2025, MetaQuotes Ltd.

```
  {                                        // von der Vererbung erfolgt
private:
   int              m_square_side;         // Quadratseite

public:
   void             CSquare(){m_type=2;}; // Konstruktor, Typ ist 2 gleich
   void             SetSide(double s){m_square_side=s;};
   virtual double GetArea(){return (m_square_side*m_square_side);}//Quadratinhalt
  };
```

Da für Berechnung des Quadrats und des Kreises entsprechende Werte der Gliesder m_radius und m_square_side notwendig sind, haben wir in der Erklärung der entsprechenden Klassen Funktionen SetRadius und SetSide() zugesetzt.

Es wird davon ausgegangen, dass in unserem Programm Objekte der verschiedenen Typen (CCircle und CSquare) abgeleitet von einem Basistyp CShape verwendet werden. Polymorphismus erlaubt die Erstellung eines Arrays von Objekten der Basisklasse CShape, aber bei der Deklaration von diesem Array sind diese Objekte noch nicht bekannt und ihre Typ nicht definiert ist.

Die Entscheidung, welcher Typ von Objekt wird in jedem Element des Arrays enthalten sein werden direkt während der Programmausführung genommen werden. Dies beinhaltet die dynamische Erstellung von Objekten der entsprechenden Klassen, und damit die Notwendigkeit, Objekt Zeiger anstelle von Objekten zu verwenden..

Das new-Array ist für die dynamische Erzeugung von Objekten eingesetzt. Jedes solches Objekt muss einzeln und explizit mit dem Operator delete gelöscht werden. Deshalb erklären wir ein Array den Zeigern von CShape Typ, und erstellen wir ein Objekt einer richtigen Typ für jedes Element (**new Klassname**), wie im folgenden Skript Beispiel gezeigt:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Erklären wir Array von Zeigern auf Objekte der Basistyp
   CShape *shapes[5];   // Array von Zeigern auf Objekte CShape

//--- Hier füllen wir das Array mit abgeleiteten Objekten
//--- Erklären wir Zeigern auf Objekt der CCircle-Typ
   CCircle *circle=new CCircle();
//--- Stellen wir die Eigenschaften von Objekt im circle Zeiger auf
   circle.SetRadius(2.5);
//--- Platzieren wir den Zeigerwert in shapes[0]
   shapes[0]=circle;

//--- Erstellen wir noch ein Objekt CCircle und schreiben sein Zeiger in shapes[1]
   circle=new CCircle();
   shapes[1]=circle;
   circle.SetRadius(5);

//--- Hier haben wir absichtlich "vergessen ", einen Wert für shapes[2] zu geben
```

© 2000-2025, MetaQuotes Ltd.

```
//circle=new CCircle();
//circle.SetRadius(10);
//shapes[2]=circle;

//--- Setzen wir NULL für die nicht genutzte Element
   shapes[2]=NULL;

//--- Erstellen wir Objekt CSquare und schreiben sein Zeiger in shapes[3]
   CSquare *square=new CSquare();
   square.SetSide(5);
   shapes[3]=square;

//--- Erstellen wir CSquare und schreiben sein Zeiger in shapes[4]
   square=new CSquare();
   square.SetSide(10);
   shapes[4]=square;

//--- Es gibt ein Array von Zeiger, erhalten wir sein Größe
   int total=ArraySize(shapes);
//--- Passen wir in der Schleife über alle Zeiger in dem Array
   for(int i=0; i<5;i++)
     {
      //--- Wenn der Zeiger an dem angegebenen Index gültig ist
      if(CheckPointer(shapes[i])!=POINTER_INVALID)
        {
         //--- Speichern wir Tap und Fläche der Figur in Logs
         PrintFormat("Objekt von Typ %d mit Fläche %G",
               shapes[i].GetType(),
               shapes[i].GetArea());
       ;}
      //--- Wenn der Zeiger ist vom Typ POINTER_INVALID
      else
        {
         //--- Melden wir einen Fehler
         PrintFormat("Objekt shapes[%d] ist nicht initialisiert! Sein Zeiger ist %s",
                     i,EnumToString(CheckPointer(shapes[i])));
        }
     }

//--- Wir müssen zerstören alle erstellten dynamischen Objekten
   for(int i=0;i<total;i++)
     {
      //--- Wir können nur die Objekte die den Zeiger vom typ POINTER_DYNAMIC haben lö
      if(CheckPointer(shapes[i])==POINTER_DYNAMIC)
        {
         //--- Melden wir die Löschung
         PrintFormat("Löschen wir shapes[%d]",i);
         //--- Löschen wir ein Objekt durch seinen Zeiger
         delete shapes[i];
```

© 2000-2025, MetaQuotes Ltd.

```
            }
        }
    }
```

Beachten Sie, dass wenn das Object mit dem Operator delete gelöscht wird, der Typ seinen Zeiger überprüfen werden soll. Nur Objekte mit dem Zeiger POINTER_DYNAMIC kann durch delete gelöscht werden. Für andere Typen wird ein Fehler zurückgegeben.

Ausser Umdefinieren der Funktion bei der Vererbung schließt Polymorphismus auch Realisierung derselben Funktion mit verschiedenem Parametervorrat innerhalb einer Klasse. Das bedeutet in der Klasse können mehrere Funktionen mit demselben Namen, aber verschiedenem Parametertyp/Parametervorrat definiert werden. In diesem Fall wird Polymorphismus durch Funktionüberladen realisiert.

Sehen Sie auch

Standardbibliothek

© 2000-2025, MetaQuotes Ltd.

Überladen

Innerhalb einer Klasse können zwei oder mehr ere Methoden festgestellt werden, die zusammen denselben Namen verwenden, aber verschiedene Parameterzahl haben. Wenn es der Fall ist, heißen die Methoden *überladene* Methoden und der Prozess heißt *überladen der Methode*. Überladen der Methode ist eines der Verfahren, mit dessen Hilfe Polymorphismus realisiert wird. Überladen der Methoden in Klassen erfolgt nach denselben Regeln, wie überladen der Funktionen.

Wenn es für zu aufgerufene Funktion kein genaues Passen gibt, sucht der Compiler konsequent die passende Funktion nach drei Niveaus:

1. Suche unter Klassenmethoden;
2. Suche unter Methoden der Basisklassen, konsequent vom nächsten Vorfahren bis zum allerersten;
3. Suche unter anderen Funktionen.

Wenn es auf keinem Niveau genaues Passen gefunden ist, aber einige passende Funktionen auf verschiedenen Niveaus gefunden werden, wird die Funktion verwendet, die auf dem kleinsten Niveau gefunden ist. Innerhalb eines Niveaus kann nicht mehr als eine passende Funktion sein.

In MQL5 gibt es kein überladen von Anweisungen.

Sehen Sie auch

Überladung der Funktionen

© 2000-2025, MetaQuotes Ltd.

Virtuelle Funktionen

Das Schlüsselwort virtual dient als Funktionsspezifikator, der Mechanismus für dynamische Auswahl in der Etappe der Ausführung der passenden Funktion-Glied unter Funktionen der Basis- und sekundaeren Klasse, Strukture können keine virtuelle Funktionen haben. Es kann erst für die Änderung Erklärung der Funktionen-Glieder verwendet werden.

Virtuelle Funktion muss wie eine normale Funktion ausfürenden Körper haben. Beim Aufruf hat sie dieselbe Semantik, wie andere Funktionen. .

Virtuelle Funktion kann in der sekundaeren Klasse substituiert werden. Die Auswahl der Funktionsdefinition erfolgt dynamisch (in der Ausführungsetappe). Typischer Fall - wenn die Basisklasse virtuelle Funktion hat, und sekundaere Klassen ihre eigenen Versionen dieser Funktion haben.

Anzeiger auf die Basisklasse kann entweder auf das Objekt der Basisklasse oder auf das Objekt der sekundaeren Klasse hinweisen. Auswahl der aufgerufenen Funktion-Glied wird in der Ausführungsetappe durchgeführt und wird vom Objektyp und nicht vom Anzeigertyp abhängig. Beim Fehlen des Gliedes des sekundären Typs wird vorbestimmt virtuelle Funktion der Basisklasse verwendet.

Destruktors sind immer virtuell unabhängig davon, ob sie mit dem Schluesselwort virtual erklärt werden oder nicht.

Betrachten wir die Verwendung von virtuellen Funktionen am Beispiel des Programms MT5_Tetris.mq5. In einschliessender Datei MT5_TetrisShape.mqh ist die Basisklasse CTetrisShape mit der virtuellen Funktion Draw (zeichnen) bestimmt.

```
//+------------------------------------------------------------------+
class CTetrisShape
  {
protected:
   int               m_type;
   int               m_xpos;
   int               m_ypos;
   int               m_xsize;
   int               m_ysize;
   int               m_prev_turn;
   int               m_turn;
   int               m_right_border;
public:
   void              CTetrisShape();
   void              SetRightBorder(int border) { m_right_border=border; }
   void              SetYPos(int ypos)          { m_ypos=ypos;           }
   void              SetXPos(int xpos)          { m_xpos=xpos;           }
   int               GetYPos()                  { return(m_ypos);        }
   int               GetXPos()                  { return(m_xpos);        }
   int               GetYSize()                 { return(m_ysize);       }
   int               GetXSize()                 { return(m_xsize);       }
   int               GetType()                  { return(m_type);        }
   void              Left()                     { m_xpos-=SHAPE_SIZE;    }
```

© 2000-2025, MetaQuotes Ltd.

```
   void              Right()                    { m_xpos+=SHAPE_SIZE;     }
   void              Rotate()                   { m_prev_turn=m_turn; if(++m_turn>3) r
   virtual void      Draw()                     { return;                 }
   virtual bool      CheckDown(int& pad_array[]);
   virtual bool      CheckLeft(int& side_row[]);
   virtual bool      CheckRight(int& side_row[]);
  };
```

Weiters wird diese Funktion für jede sekundaere Klasse realisiert entsprechend den Besonderheiten der Klasse-Nachfolger. ZB. hat die erste Figur CTetrisShape1 ihre eigene Realisierung der Funktion Draw():

```
class CTetrisShape1 : public CTetrisShape
  {
public:
   //--- Zeichnung der Figur
   virtual void      Draw()
     {
      int    i;
      string name;
      //---
      if(m_turn==0 || m_turn==2)
        {
         //--- horizontaler Stock
         for(i=0; i<4; i++)
           {
            name=SHAPE_NAME+(string)i;
            ObjectSetInteger(0,name,OBJPROP_XDISTANCE,m_xpos+i*SHAPE_SIZE);
            ObjectSetInteger(0,name,OBJPROP_YDISTANCE,m_ypos);
           }
        }
      else
        {
         //--- senkrechter Stock
         for(i=0; i<4; i++)
           {
            name=SHAPE_NAME+(string)i;
            ObjectSetInteger(0,name,OBJPROP_XDISTANCE,m_xpos);
            ObjectSetInteger(0,name,OBJPROP_YDISTANCE,m_ypos+i*SHAPE_SIZE);
           }
        }
     }
  }
```

Figur Quadrat ist von der Klasse CTetrisShape6 beschrieben und hat ihre eigene Realisierung der Methode Draw():

```
class CTetrisShape6 : public CTetrisShape
  {
public:
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- Figurzeichnung
   virtual void      Draw()
     {
      int    i;
      string name;
      //---
      for(i=0; i<2; i++)
        {
         name=SHAPE_NAME+(string)i;
         ObjectSetInteger(0,name,OBJPROP_XDISTANCE,m_xpos+i*SHAPE_SIZE);
         ObjectSetInteger(0,name,OBJPROP_YDISTANCE,m_ypos);
        }
      for(i=2; i<4; i++)
        {
         name=SHAPE_NAME+(string)i;
         ObjectSetInteger(0,name,OBJPROP_XDISTANCE,m_xpos+(i-2)*SHAPE_SIZE);
         ObjectSetInteger(0,name,OBJPROP_YDISTANCE,m_ypos+SHAPE_SIZE);
        }
     }
  };
```

Abhängig davon, Objekt welcher Klasse erzeugt wird, wird virtuelle Funktion einer oder anderer sekundären Klasse aufgerufen.

```
void CTetrisField::NewShape()
  {
//--- zufallsmässig schaffen wir eine der 7 möglichen Figuren
   int nshape=rand()%7;
   switch(nshape)
     {
      case 0: m_shape=new CTetrisShape1; break;
      case 1: m_shape=new CTetrisShape2; break;
      case 2: m_shape=new CTetrisShape3; break;
      case 3: m_shape=new CTetrisShape4; break;
      case 4: m_shape=new CTetrisShape5; break;
      case 5: m_shape=new CTetrisShape6; break;
      case 6: m_shape=new CTetrisShape7; break;
     }
//--- zeichnen wir
   m_shape.Draw();
//---
  }
```

Modifier Override

Der override Modifier bedeutet, dass die zu deklarierende Funktion die Methode der Basisklasse unbedingt überschreiben muss. Dank diesem Modifier kann man Fehler beim Überschreiben vermeiden, wie z.B. zufällige Änderung der Methodensignatur. In der Basisklasse ist zum Beispiel die Methode func definiert, die die Variable vom Typ int als Argument hat:

© 2000-2025, MetaQuotes Ltd.

```
class CFoo
  {
   void virtual func(int x) const { }
  };
```

Weiter wird die Methode in der abgeleiteten Klasse überschrieben:

```
class CBar : public CFoo
  {
   void func(short x) { }
  };
```

Aber durch einen Fehler ändert sich der Typ des Arguments von int zu short. In diesem Fall wird die Methode nicht mehr überschrieben, sondern überladen. In einigen Situationen kann der Compiler entsprechend dem Algorithmus der überladenen Funktion statt der überschriebenen Methode eine in der Basisklasse definierte Methode auswählen.

Um solche Fehler zu vermeiden, sollte man der Methode den override Modifier hinzufügen.

```
class CBar : public CFoo
  {
   void func(short x) override { }
  };
```

Wenn die Signatur beim Überschreiben geändert wird, kann der Compiler keine Methode mit der gleichen Signatur in der Basisklasse finden und gibt einen Kompilierungsfehler aus:

```
'CBar::func' method is declared with 'override' specifier but does not override any ba
```

Modifier final

Der Modifier final funktioniert umgekehrt: er verbietet, Methoden in abgeleiteten Klassen zu überschreiben. Wenn die Implementierung der Methode abgeschlossen ist, deklarieren Sie diese mit dem Modifier final, damit sie später garantiert nicht verändert wird.

```
class CFoo
  {
   void virtual func(int x) final { }
  };

class CBar : public CFoo
  {
   void func(int) { }
  };
```

Beim Versuch, die Methode mit dem final Modifier zu überschreiben, gibt der Compiler eine Fehlermeldung aus:

```
'CFoo::func' method declared as 'final' cannot be overridden by 'CBar::func'
see declaration of 'CFoo::func'
```

Sehen Sie auch

Standardbibliothek

© 2000-2025, MetaQuotes Ltd.

Statische Mitglieder der Klasse/Struktur

Statische Mitglieder

Die Mitglieder der Klasse können mit der Storage-Klasse Modifier statisch deklariert werden. Diese Datenmitglieder werden von allen Instanzen dieser Klasse geteilt und sind an einem Ort gespeichert. Nicht-statische Datenmitglieder werden für jede Variable-Klassenobject erstellt.

Die Unfähigkeit, statische Mitglieder einer Klasse zu deklarieren, würde auf die Notwendigkeit, diese Daten auf der globalen Ebene des Programms zu deklarieren, geführt haben. Es würde die Beziehung zwischen den Daten und ihrer Klasse brechen und steht nicht im Einklang mit der grundlegenden Paradigma der OOP - die Zusammenführung von Daten und Methoden für Verarbeitung in einer Klasse. Ein statischer Mitglied ermöglicht Klassendaten, die auf eine bestimmte Instanz nicht spezifisch sind, im Gültigkeitsbereich der Klasse zu existieren.

Da eine statische Klasse Mitglied nicht von der jeweiligen Instanz abhängen, ist der Verweis auf ihn wie folgt:

```
class_name::variable
```

wo *class_name* - ist der Name der Klasse, *variable* ist der name des Mitglieds der Klasse.

Ein statisches Klassenmitglied muss explizit mit den gewünschten Wert initialisiert werden. Um dies zu tun, muss es auf globaler Ebene deklariert und initialisiert werden. Die Reihenfolge der Initialisierung von statischen Mitglieder entspricht der Reihenfolge, in der sie auf der globalen Ebene im Quellcode deklariert sind.

Zum Beispiel haben wir eine Klasse *CParser*, die zum Parsen der Text verwendet wird, und wir brauchen die Gesamtzahl der verarbeiteten Wörter und Zeichen zu zählen. Wir müssen nur die notwendigen Klassenmitglieder als statische deklarieren und sie auf globaler Ebene initialisieren. Dann werden alle Instanzen der Klasse den Zähler von Wörtern und Zeichen gemeinsam nutzen.

```
//+------------------------------------------------------------------+
//| Klasse "Text Parser"                                             |
//+------------------------------------------------------------------+
class CParser
  {
public:
   static int        s_words;
   static int        s_symbols;
   //--- Konstruktor und Destruktor
                     CParser(void);
                    ~CParser(void){};
  };
...
//--- Initialisierung von statischen Mitglieder der Klasse Parser auf globaler Ebene
int CParser::s_words=0;
int CParser::s_symbols=0;
```

Ein statisches Klassenmitglied kann mit dem Schlüsselwort *const* deklariert werden. Solche statischen Konstanten müssen auf globaler Ebene mit dem Schlüsselwort *const initialisiert werden:*

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Klasse "Stack" zur Speicherung der Daten, die verarbeitet werden |
//+------------------------------------------------------------------+
class CStack
  {
public:
                     CStack(void);
                    ~CStack(void){};
...
private:
   static const int  s_max_length; // Maximale Kapazität des Stacks
  };

//--- initialisierung der statischen Konstanten der Klasse CStack
const int CStack::s_max_length=1000;
```

Zeiger this

Das Schlüsselwort this bedeutet implizit deklarierten Zeiger auf sich selbst - auf eine bestimmte Instanz der Klasse, in deren Rahmen die Methode ausgeführt wird. Es kann nur in nicht-statische Methoden der Klasse verwendet werden. Zeiger this ist ein implizites nicht-statisches Mitglied einer beliebigen Klasse.

In statischen Funktionen können Sie nur statische Mitglieder/Methoden einer Klasse zugreifen.

Statische Methoden

In MQL5 Mitgliederfunktionen vom Typ static können verwendet werden. Modifikator *static* muss vor dem Rückgabetyp der Funktion in der Deklaration innerhalb einer Klasse gehen.

```
class CStack
  {
public:
   //--- Konstruktor und Destruktor
                     CStack(void){};
                    ~CStack(void){};
   //--- Maximale Kapazität des Stacks
   static int        Capacity();
private:
   int               m_length;     // Anzahl der Elemente im Stack
   static const int  s_max_length; // Maximale Kapazität des Stacks
  };
//+------------------------------------------------------------------+
//| Gibt maximale Anzahl der Elemente im Stack zu speichern zurück   |
//+------------------------------------------------------------------+
int CStack::Capacity(void)
  {
   return(s_max_length);
  }
//--- initialisierung der statischen Konstanten der Klasse CStack
const int CStack::s_max_length=1000;
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- eine Variable vom Typ CStack deklarieren
   CStack stack;
//--- die statische Objektmethode anrufen
   Print("CStack.s_max_length=",stack.Capacity());
//--- es ist auch möglich so anzurufen, da die Methode statisch ist und kein Objekt ei
   Print("CStack.s_max_length=",CStack::Capacity());
 ;}
```

Eine Methode mit Modifikator const heißt konstant und kann nicht implizite Mitglieder ihrer Klasse ändern. Die Deklaration der konstanten Funktionen einer Klasse und konstanten Parametern ist *Kontrolle der Konstanz* (const-correctness). Durch diese Kontrolle können Sie sicher sein, dass der Compiler die Konsistenz der Werte der Objekte gewährleistet und wird einen Fehler während der Kompilierung zurückgeben, wenn etwas falsch ist.

Modifikator const wird nach der Liste von Argumenten innerhalb einer Klassendeklaration platziert. Definition außerhalb einer Klasse sollte auch die Modifikator *const* haben:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Klasse "Rechteck"                                                |
//+------------------------------------------------------------------+
class CRectangle
  {
private:
   double            m_width;      // Breite
   double            m_height;     // Höhe
public:
   //--- Konstruktoren und Destruktor
                     CRectangle(void):m_width(0),m_height(0){};
                     CRectangle(const double w,const double h):m_width(w),m_height(h)
                    ~CRectangle(void){};
   //--- Berechnung der Fläche
   double            Square(void) const;
   static double     Square(const double w,const double h);// { return(w*h); }
  };
//+------------------------------------------------------------------+
//| Gibt die Fläche des Objekts "Rechteck" zurück                    |
//+------------------------------------------------------------------+
double CRectangle::Square(void) const
  {
   return(Square(m_width,m_height));
  }
//+------------------------------------------------------------------+
//| Gibt das Produkt von zwei Variablen zurück                       |
//+------------------------------------------------------------------+
static double CRectangle::Square(const double w,const double h)
  {
   return(w*h);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- erstellen wir ein Rechteck rect mit Seiten 5 und 6
   CRectangle rect(5,6);
//--- finden wir Fläche eines Rechtecks mit der konstanten Methode
   PrintFormat("rect.Square()=%.2f",rect.Square());
//--- finden wir das Produkt der Zahlen mit der statischen Methode der CRectangle-Klas
   PrintFormat("CRectangle::Square(2.0,1.5)=%f",CRectangle::Square(2.0,1.5));
  }
```

Ein zusätzliches Argument für die Verwendung der Konstanz Kontrolle ist die Tatsache, dass in diesem Fall der Compiler eine spezielle Optimierung erzeugt, zum Beispiel, hat ein konstantes Objekt im Nurlesespeicher.

Eine statische Funktion kann nicht mit einem Modifikator const definiert werden, da dies Modifikator die Unveränderlichkeit von den Mitgliedern der Instanz garantiert, wenn diese Funktion aufgerufen wird. Aber, wie oben erwähnt, kann die statische Funktion nicht-statische Mitgliedern der Klasse nicht zugreifen.

Sehen Sie auch

Statische Variablen, Variablen, Referenzen. Modifikator & und Schlüsselwort this

© 2000-2025, MetaQuotes Ltd.

Funktionstemplates

Überladene Funktionen werden häufig verwendet, um ähnliche Operationen auf verschiedene Datentypen durchzuführen. Ein einfaches Beispiel für eine solche Funktion im MQL5 ist ArraySize(), die die Größe des Arrays jeder Art zurück gibt. In der Tat ist dieses System Funktion überladen, und die ganze Umsetzung dieses Überladen wird von der MQL5 Software-Entwickler versteckt:

```
int  ArraySize(
   void&  array[]      // Array zu prüfen
   );
```

MQL5 Compiler MQL5 ersetzt eine rechte Umsetzung für jeden Aufruf dieser Funktion. Zum Beispiel für Arrays von Typ integer:

```
int  ArraySize(
   int&  array[]      // Array mit Elementen vom Typ int
   );
```

Und für einem Array vom Typ MqlRates für Arbeit mit Preise im Format von historischen Daten kann Funktion ArraySize() wie folgt dargestellt werden:

```
int  ArraySize(
   MqlRates&  array[] // Array mit Elementen vom Typ MqlRates
   );
```

Somit ist es sehr praktisch, die gleiche Funktion für verschiedenen Typen zu nutzen, aber Sie müssen Vorarbeiten ausführen - nämlich Überladen der Funktion für alle Datentypen, mit denen sie arbeitet.

Es gibt eine bequeme Lösung - wenn für jeden Datentyp identische Operationen durchgeführt werden sollte, gibt es eine kompakte und praktische Lösung, Funktionsschablonen zu verwenden. In diesem Fall muss der Programmierer nur eine Beschreibung der Funktionsschablone zu schreiben. Für diese Beschreibung der Schablone, müssen Sie nicht einen spezifischen Datentyp, mit dem die Funktion arbeiten soll, als Parameter eingeben, sondern einen formalen Parameter. Basierend auf der Typen der verwendeten Argumente beim Aufruf der Funktion, wird der Compiler automatisch verschiedene Funktionen für die geeignete Behandlung für jeden Typ generieren.

Die Definition einer Funktionsschablone beginnt mit dem Schlüsselwort template wonach in spitzen Klammern ist eine Liste der formalen Parameter. Jeder formale Parameter wird durch das Schlüsselwort typename vorangestellt. Die formalen Parameter-Typen sind eingebaute Typen oder benutzerdefinierte Typen. Sie werden verwendet:

- um die Type der Argumente der Funktion einzugeben
- um den Typ des Rückgabewerts der Funktion einzugeben
- um Variablen innerhalb des Körpers der Beschreibung der Funktion zu erklären

Die Anzahl der Parameter in einer Schablone kann nicht mehr als acht sein. Jeder formale Parameter in der Schablone-Definition sollte mindestens einmal in der Funktion Parameter-Liste erscheinen. Jeder Name vom formalen Parameter muss eindeutig sein.

Hier ist ein Beispiel für ein Funktionsschablone für die Suche nach den maximalen Wert in einem Array von beliebigen numerischen Typs (integer und reelle Zahlen):

© 2000-2025, MetaQuotes Ltd.

```
template<typename T>
T ArrayMax(T &arr[])
  {
   uint size=ArraySize(arr);
   if(size==0) return(0);

   T max=arr[0];
   for(uint n=1;n<size;n++)
      if(max<arr[n]) max=arr[n];
//---
   return(max);
  }
```

Dieses Beispiel beschreibt eine Funktion, die den größten Wert in dem übergebenen Array findet, um es als Ergebnis zurück zu geben. Daran erinnern, dass die eingebaute MQL5 Funktion ArrayMaximum() gibt nur der Index des gefundenen Maximalwert zurück, mit dem der Benutzer kann in der Zukunft den Wert selbst bekommen. Zum Beispiel:

```
//--- Erstellen wir ein Array
   double array[];
   int size=50;
   ArrayResize(array,size);
//---  Füllen es mit zufälligen Werten
   for(int i=0;i<size;i++)
     {
      array[i]=MathRand();
     }

//--- Finden wir die Position der maximalen Element im Array
   int max_position=ArrayMaximum(array);
//--- Bekommen wir das Maximalwert im Array
   double max=array[max_position];
//--- Ausgabe des gefundenen Wertes
   Print("Max value = ",max);
```

So brauchen wir zwei Schritte, um den maximalen Wert in dem Array zu erhalten. Mit Hilfe der Funktionsschablone ArrayMax() kann man das Ergebnis vom gewünschten Typ erhalten, wenn man ein Array des entsprechenden Typs in sie übergibt. Das heißt, anstelle der zwei letzten Zeilen

```
//--- Finden wir die Position der maximalen Element im Array
   int max_position=ArrayMaximum(array);
//--- Bekommen wir das Maximalwert im Array
   double max=array[max_position];
```

Jetzt können wir nur eine einzige Zeile, die ein Ergebnis des gleichen Typs wie der übergebenen Array zurückgibt, benutzen:

```
//--- Finden wir den maximalen Wert
   double max=ArrayMax(array);
```

Der Typ des Ergebnisses, der durch ArrayMax() zurückgegeben wird, wird automatisch mit dem Typ des Arrays gleich sein.

Um mit verschiedenen Typen von Daten zu arbeiten, müssen Sie das Schlüsselwort typename verwenden, um den Typ des Arguments als Zeichen zu erhalten. Wir zeigen dies am Beispiel einer Funktion, die einen Datentyp als String zurückgibt:

© 2000-2025, MetaQuotes Ltd.

```
#include <Trade\Trade.mqh>
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   CTrade trade;
   double d_value=M_PI;
   int i_value=INT_MAX;
   Print("d_value: type=",GetTypeName(d_value), ",   value=", d_value);
   Print("i_value: type=",GetTypeName(i_value), ",   value=", i_value);
   Print("trade: type=",GetTypeName(trade));
//---
  }
//+------------------------------------------------------------------+
//| Gibt den Typ als String zurück                                   |
//+------------------------------------------------------------------+
template<typename T>
string GetTypeName(const T &t)
  {
//--- den Typ als String zurückgeben
   return(typename(T));
//---
  }
```

Funktionsschablonen können auch für Klassenmethode verwendet werden, zum Beispiel:

```
class CFile
  {
   ...
public:
   ...
   template<typename T>
   uint WriteStruct(T &data);
  };

template<typename T>
uint CFile::WriteStruct(T &data)
  {
   ...
   return(FileWriteStruct(m_handle,data));
  }
```

Funktionsschablonen können nicht mit den Schlüsselwörtern export, virtual und #import deklariert werden.

Überladen von Funktionstemplates

In einigen Fällen müssen Funktionen überladen werden. Wir haben zum Beispiel ein Funktionstemplate, das den Wert des zweiten Parameters in den ersten Parameter mithilfe der expliziten Typenumwandlungschreibt. In der MQL5-Programmiersprache ist es verboten, den string Typ in den bool Typ umzuwandeln, wir können das selbst tun - dafür erstellen wir die Überladung des Funktiostempaltes. Zum Beispiel:

```
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Funktionsvorlage                                                 |
//+------------------------------------------------------------------+
template<typename T1,typename T2>
string Assign(T1 &var1,T2 var2)
  {
   var1=(T1)var2;
   return(__FUNCSIG__);
  }
//+------------------------------------------------------------------+
//| Spezielle Überladung für den Fall bool+string                    |
//+------------------------------------------------------------------+
string Assign(bool &var1,string var2)
  {
   var1=(StringCompare(var2,"true",false) || StringToInteger(var2)!=0);
   return(__FUNCSIG__);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int i;
   bool b;
   Print(Assign(i,"test"));
   Print(Assign(b,"test"));
  }
```

Wenn der Code ausgeführt wird, können wir sehen, dass für das Paar int+string das Funktionstemplate Assign() verwendet wurde und bei dem zweiten Aufruf für das Paar bool+string die überladene Version verwendet.

```
string Assign<int,string>(int&,string)
string Assign(bool&,string)
```

Siehe auch

Überladung

© 2000-2025, MetaQuotes Ltd.

Vorteile von Templates

Funktionstemplates werden in den Fällen verwendet, wenn man gleiche Operationen mit Daten verschiedener Typen ausführen muss, zum Beispiel, Suche nach dem maximalen Element in einem Array. Der wichtigste Vorteil der Templates besteht darin, dass der Entwickler keine Überladung für jeden Typ separat schreiben muss. Das heißt statt Überladung für jeden Typ zu deklarieren

```
double ArrayMax(double array[])
  {
   ...
  }
int ArrayMax(int array[])
  {
   ...
  }
uint ArrayMax(uint array[])
  {
   ...
  }
long ArrayMax(long array[])
  {
   ...
  }
datetime ArrayMax(datetime array[])
  {
   ...
  }
```

reicht es, ein Funktionstemplate zu schreiben

```
template<typename T>
T ArrayMax(T array[])
  {
   if(ArraySize()==0)
      return(0);
   uint max_index=ArrayMaximum(array);
   return(array[max_index]);
  }
```

und dieses dann im Code zu verwenden:

```
double high[];
datetime time[];
....
double max_high=ArrayMax(high);
datetime lasttime=ArrayMax(time);
```

Hier gibt es einen formellen Parameter *T*, der den Datentyp setzt, d.h. der Compiler generiert automatisch eine Funktion für jeden Typ – double, datetime und so weiter. Genauso kann man in MQL5 Klassentemplates erstellen und von allen Vorteilen dieser Methode profitieren.

© 2000-2025, MetaQuotes Ltd.

Klassentemplates

Ein Klassentemplate wird mithilfe des Schlüsselwortes template deklariert, nach welchem spitze Klammern stehen <>, in welchen formelle Parameter mit dem Schlüsselwort typename aufgezählt sind. Dies weist den Compiler darauf hin, dass er mit einer allgemeinen Klasse mit dem formellen *T* Parameter zu tun hat, welcher den realen Typ der Variablen bei der Implementierung der Klasse setzt. Zum Beispiel, erstellen wir eine Vector-Klasse für das Speichern eines Arrays mit Elementen vom Typ *T*:

```
#define TOSTR(x) #x+" "   // Makro für die Ausgabe des Objektnamens
//+------------------------------------------------------------------+
//| Vector-Klasse für das Speichern der Elemente vom Typ T           |
//+------------------------------------------------------------------+
template <typename T>
class TArray
  {
protected:
   T                 m_array[];
public:
   //--- der Konstruktor erstellt standardmäßig ein Array für 10 Elemente
   void TArray(void){ArrayResize(m_array,10);}
   //--- Konstruktor für die Erstellung eines Vectors mit der vorgegebenen Arraygröße
   void TArray(int size){ArrayResize(m_array,size);}
   //--- gibt Typ und Anzahl der Daten, die im Objekt vom Typ TArray gespeichert werde
   string Type(void){return(typename(m_array[0])+":"+(string)ArraySize(m_array));};
  };
```

Weiter erstellen wir auf verschiedene Weisen drei *TArray* Objekte für das Arbeiten mit verschiedenen Typen

```
void OnStart()
  {
   TArray<double> double_array;   // standardmäßig beträgt die Vector-Größe 10
   TArray<int> int_array(15);     // Verctor-Größe 15
   TArray<string> *string_array;  // Pointer auf den Vector TArray<string>
//--- erstellen wir ein dynamisches Objekt
   string_array=new TArray<string>(20);
//--- geben wir den Objektnamen, Datentyp und die Vector-Größe ins Journal aus
   PrintFormat("%s (%s)",TOSTR(double_array),double_array.Type());
   PrintFormat("%s (%s)",TOSTR(int_array),int_array.Type());
   PrintFormat("%s (%s)",TOSTR(string_array),string_array.Type());
//--- löschen wir das dynamische Objekt vor dem Beenden des Programms
   delete(string_array);
  }
```

Das Ergebnis der Ausführung des Skriptes:

```
  double_array  (double:10)
  int_array  (int:15)
  string_array  (string:20)
```

© 2000-2025, MetaQuotes Ltd.

Es wurden insgesamt 3 Vectors mit verschiedenen Datentypen erstellt: double, int und string.

Die Klassentemplates passen gut für die Erstellung von Containers - Objekten, die für die Datenkapselung von Objekten jeglichen Typs vorgesehen sind. Die Objekte von Containers stellen Kollektionen dar, die bereits Objekte eines bestimmten Typs beinhalten. In der Regel wird das Arbeiten mit gespeicherten Daten in den Container eingebaut.

Man kann zum Beispiel ein Klassentemplate erstellen, der den Zugriff auf ein Element außerhalb des Arrays verbietet, und auf diese Weise den kritischen Fehler "out of range" vermeiden.

```
//+------------------------------------------------------------------+
//| Klasse für sicheren Zugriff auf ein Element des Arrays           |
//+------------------------------------------------------------------+
template<typename T>
class TSafeArray
  {
protected:
   T                 m_array[];
public:
   //--- standardmäßiger Konstruktor
   void              TSafeArray(void){}
   //--- Konstruktor für die Erstellung eines Arrays mit vorgegebener Größe
   void              TSafeArray(int size){ArrayResize(m_array,size);}
   //--- Arraygröße
   int               Size(void){return(ArraySize(m_array));}
   //--- Änderung der Arraygröße
   int               Resize(int size,int reserve){return(ArrayResize(m_array,size,rese
   //--- den Inhalt des Arrays löschen
   void              Erase(void){ZeroMemory(m_array);}
   //--- Operator für das Zugreifen auf ein Element des Arrays nach dem Index
   T                 operator[](int index);
   //--- Zuweisungsoperator für das Erhalten aller Elemente aus dem Array
   void              operator=(const T  &array[]); // Array vom Typ T
  };
//+------------------------------------------------------------------+
//| Operation des Erhaltens eines Elements nach dem Index            |
//+------------------------------------------------------------------+
template<typename T>
T TSafeArray::operator[](int index)
  {
   static T invalid_value;
//---
   int max=ArraySize(m_array)-1;
   if(index<0 || index>=ArraySize(m_array))
     {
      PrintFormat("%s index %d is not in range (0-%d)!",__FUNCTION__,index,max);
      return(invalid_value);
     }
//---
   return(m_array[index]);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Zuweisungsoperator für das Array                                 |
//+------------------------------------------------------------------+
template<typename T>
void TSafeArray::operator=(const T  &array[])
  {
   int size=ArraySize(array);
   ArrayResize(m_array,size);
//--- der T Typ muss den Copy Operator unterstützen
   for(int i=0;i<size;i++)
      m_array[i]=array[i];
//---
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int copied,size=15;
   MqlRates rates[];
//--- kopieren wir das Array der Kurse
   if((copied=CopyRates(_Symbol,_Period,0,size,rates))!=size)
     {
      PrintFormat("CopyRates(%s,%s,0,%d) Fehlercode %d",
      _Symbol,EnumToString(_Period),size,GetLastError());
      return;
     }
//--- erstellen wir einen Container und fügen ihm das Array der MqlRates Werte hinzu
   TSafeArray<MqlRates> safe_rates;
   safe_rates=rates;
   //--- Index innerhalb des Arrays
   int index=3;
   PrintFormat("Close[%d]=%G",index,safe_rates[index].close);
   //--- Index außerhalb des Arrays
   index=size;
   PrintFormat("Close[%d]=%G",index,safe_rates[index].close);
  }
```

Bitte beachten Sie, dass bei der Beschreibung von Methoden außerhalb der Deklaration der Klasse auch bei der Deklaration des Templates verwendet werden muss

```
template<typename T>
T TSafeArray::operator[](int index)
  {
   ...
  }
template<typename T>
void TSafeArray::operator=(const T  &array[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   ...
  }
```

Klassen- und Funktionstemplates erlauben es, mehrere formelle Parameter durch Komma getrennt anzugeben, zum Beispiel, die Map Kollektion für das Speichern von Paaren "Schlüssel - Wert":

```
template<typename Key, template Value>
class TMap
  {
   ...
  }
```

Siehe auch

Funktionstempaltes, Überladung

© 2000-2025, MetaQuotes Ltd.

Abstrakte Klassen und rein virtuelle Funktionen

Abstrakte Klassen sind für die Erstellung generalisierter Entitäten bestimmt, aufgrund deren später konkretere abgeleitete Klassen erstellt werden. Eine abstrakte Klasse ist die Klasse, die nur als Basisklasse für eine andere Klasse genutzt werden kann. Aus diesem Grund kann man kein Objekt vom Typ abstrakter Klasse erstellen

Eine Klasse, die mindestens eine rein virtuelle Funktion enthält, gilt als abstrakt. Deswegen müssen Klassen, die von einer abstrakten Klasse abgeleitet wurden, alle ihre rein virtuellen Funktionen umsetzen, sonst werden sie auch als abstrakte Klassen gelten.

Eine virtuelle Funktion wird als "reine" mit der Syntax des Reinheit-Spezifikators deklariert. Schauen wir uns die CAnimal-Klasse an, die nur für die Gewährleistung allgemeiner Funktionen erstellt wird - Objekte vom Typ CAnimal sind zu allgemein für eine praktische Anwendung. So ist die CAnimal-Klasse ein guter Kandidat für abstrakte Klasse:

```
class CAnimal
  {
public:
                     CAnimal();      // Konstruktor
   virtual void      Sound() = 0;    // rein virtuelle Funktion
private:
   double            m_legs_count;   // Anzahl der Pfoten
  };
```

Die Sound()-Funktion ist hier rein virtuell, weil sie mit dem Spezifikator der rein virtuellen PURE-Funktion (=0) deklariert wurde.

Als rein virtuelle werden nur die virtuellen Funktionen bezeichnet, für welche der PURE-Spezifikator angegeben wurde, und zwar: (=NULL) oder (=0). Beispiel der Deklaration und Anwendung abstrakter Klassen:

```
class CAnimal
  {
public:
   virtual void      Sound()=NULL;   // PURE method, muss in der abgeleiteten Klasse
  };
//--- abgeleitet von der abstrakten Klasse
class CCat : public CAnimal
 {
public:
  virtual void       Sound() { Print("Myau"); } // PURE umdefiniert, die CCat Klasse
 };

//--- Beispiele für falsche Anwendung
new CAnimal;          // Fehler 'CAnimal' - der Compiler zeigt die Fehlermeldung "canno
CAnimal some_animal; // Fehler 'CAnimal' - der Compiler zeigt die Fehlermeldung "canno

//--- Beispiele für richtige Anwendung
new CCat;  // kein Fehler - die CCat Klasse ist nicht abstrakt
CCat cat;  // kein Fehler - die CCat Klasse ist nicht abstrakt
```

© 2000-2025, MetaQuotes Ltd.

Begrenzungen für Anwendung abstrakter Klasse

Wenn der Konstruktor einer abstrakten Klasse die rein virtuelle Funktion (direkt oder indirekt) aufruft, wird das Ergebnis undefiniert sein.

```
//+------------------------------------------------------------------+
//| Abstrakte Basisklasse                                            |
//+------------------------------------------------------------------+
class CAnimal
  {
public:
   //--- rein virtuelle Funktion
   virtual void      Sound(void)=NULL;
   //--- Funktion
   void              CallSound(void) { Sound(); }
   //--- Konstruktor
   CAnimal()
    {
     //--- impliziter Aufruf einer virtuellen Methode
     Sound();
     //--- expliziter Aufruf (über eine dritte Funktion)
     CallSound();
     //--- Konstruktor bzw. Destruktor ruft immer seine eigenen Funktionen auf,
     //--- trotz der Virtualität und Umdefinierung der aufgerufenen Funktion in einer
     //--- wenn diese Funktion rein virtuell ist,
     //--- führt der Aufruf zum kritischen Ausführungsfehler: "pure virtual function c
    }
  };
```

Jedoch können Konstruktoren und Destruktoren abstrakter Klassen andere Memberfunktionen aufrufen.

© 2000-2025, MetaQuotes Ltd.

Namensraum

Ein Namespace (Namensraum) ist ein speziell deklarierter Bereich, in dem verschiedene IDs definiert sind: Variablen, Funktionen, Klassen, etc. Es wird über das Schlüsselwort namespase definiert:

```
namespace name of_space {
  // Liste der Funktions-, Klassen- und Variablendefinitionen
}
```

Die Verwendung von 'namespace' ermöglicht die Aufteilung des globalen Namensraums in Unterräume. Alle IDs innerhalb des Namensraums stehen einander ohne zusätzliche Angabe zur Verfügung. Der Operator :: (Operator zur Auflösung des Geltungsbereiches) wird verwendet, um von außen auf Mitglieder eines Namensraums zuzugreifen.

```
namespace ProjectData
{
class DataManager
  {
public:
   void              LoadData() {}
  };
void Func(DataManager& manager) {}
}
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- mit dem Namensraum ProjectData arbeiten
   ProjectData::DataManager mgr;
   mgr.LoadData();
   ProjectData::Func(mgr);
  }
```

Namensräume werden verwendet, um einen Code in Form von logischen Gruppen anzuordnen und Namenskonflikte zu vermeiden, die bei der Verwendung mehrerer Bibliotheken in einem Programm auftreten könnten. In solchen Fällen kann jede Bibliothek in ihrem eigenen Namensraum deklariert werden, um explizit auf die notwendigen Funktionen und Klassen jeder Bibliothek zuzugreifen.

Ein Namensraum kann in mehreren Blöcken in einer oder mehreren Dateien deklariert werden. Der Compiler kombiniert alle Teile während einer Vorverarbeitung miteinander und der resultierende Namensraum enthält alle in allen Teilen deklarierten Elemente. Angenommen, wir haben eine Klasse in der Include-Datei Sample.mqh implementiert:

```
//+------------------------------------------------------------------+
//|                                                   Sample.mqh |
//+------------------------------------------------------------------+
class A
  {
public:
                     A() {Print(__FUNCTION__);}
```

© 2000-2025, MetaQuotes Ltd.

```
  };
```

Wir wollen diese Klasse in unserem Projekt verwenden, aber wir haben bereits eine Klasse A. Um beide Klassen verwenden zu können und ID-Konflikte zu vermeiden, binden Sie die eingebundene Datei einfach in einen Namensraum ein:

```
//--- deklarieren der ersten Klasse A
class A
  {
public:
                     A() {Print(__FUNCTION__);}
  };
//--- die Klasse A aus der Datei Sample.mqh in den Namensraum von Library einfügen, ur
namespace Library
{
#include "Sample.mqh"
}
//--- und noch eine weitere Klasse zum Namensraum Library hinzufügen
namespace Library
{
class B
  {
public:
                     B() {Print(__FUNCTION__);}
  };
}
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Klasse A des globalen Namensraums verwenden
   A a1;
//--- die Klassen A und B aus dem Namensraum Library verwenden
   Library::A a2;
   Library::B b;
  }
//+------------------------------------------------------------------+

/*
Ergebnis:
   A::A
   Library::A::A
   Library::B::B
*/
```

Namensräume können verschachtelt werden. Ein verschachtelter Namensraum hat unbegrenzten Zugriff auf Mitglieder seines übergeordneten Bereichs, aber Mitglieder des übergeordneten Bereichs haben keinen unbegrenzten Zugriff auf den verschachtelten Namensraum.

© 2000-2025, MetaQuotes Ltd.

```
namespace General
{
int Func();

namespace Details
{
 int Counter;
 int Refresh()  {return Func(); }
}

int GetBars()   {return(iBars(Symbol(), Period()));};
int Size(int i) {return Details::Counter;}
}
```

Globaler Namensraum

Wenn die ID nicht explizit im Namensraum deklariert wird, gilt sie als impliziter Teil des globalen Namensraums. Um die globale ID explizit festzulegen, verwenden Sie den Operator des Geltungsbereichs ohne Namen. Dadurch können Sie diese ID von jedem anderen Element mit dem gleichen Namen unterscheiden, das sich in einem anderen Namensraum befindet. Zum Beispiel, wenn Sie eine Funktion importieren:

```
#import "lib.dll"
int Func();
#import
//+------------------------------------------------------------------+
//|  Some function                                                   |
//+------------------------------------------------------------------+
int Func()
  {
   return(0);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//+--- Aufruf der importierten Funktion
   Print(lib::Func());
//+--- Aufruf unserer Funktion
   Print(::Func());
  }
```

In diesem Fall wurden alle aus der DLL-Funktion importierten Funktionen in den gleichnamigen Namensraum aufgenommen. Dadurch konnte der Compiler die aufzurufende Funktion eindeutig bestimmen.

Siehe auch

© 2000-2025, MetaQuotes Ltd.

Global Variablen, Lokale Variablen, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

Standardkonstanten, Enumerationen und Strukturen

für die Erleichterung von Programmschreiben und gute Wahrnehmung von Ausgangsprogrammtexten werden in der Sprache MQL5 vorbestimmte Standardkonstanten und Enumerationen vorausgesehen. Außerdem werden für Informationsspeicherung dienstliche Strukturen verwendet.

Standardkonstanten sind Analoge der Makrosubstitutionen und haben den Typ int.

Konstanten sind nach ihrem Zweck gruppiert:

- Chartkonstanten werden bei der Arbeit mit Preischarts benutzt: Öffnung, Navigation, Parametereinstellung;
- Objektkonstanten sind für die Verarbeitung graphischer Objekte bestimmt, die erstellt und in Charten dargestellt werden können;
- Indikatorkonstanten dienen der Arbeit mit Standard- und Benutzerindikatoren;
- Medium Zustand – beschreiben Eigenschaften eines mql5-Programms, geben Information über das Client-Terminal, Handelsinstrument und das laufende Handelskonto;
- Handelskonstanten ermöglichen es verschiedenartige Information während Handelsprozesses zu praezisieren ;
- Benannte Konstanten – Konstanten der Sprache MQL5;
- Datenstrukturen beschreiben Verwendete Formate der Datenspeicherung;
- Fehler- Warnungskoden beschreiben Compilernachrichten und Nachrichten des Handelsservers auf Handelsanforderungen
- Input/Output Konstanten sind für die Arbeit mit Dateifunktionen und Ausgabe von Informationen auf Computerdislay durch die Funktion MessageBox() bestimmt.

© 2000-2025, MetaQuotes Ltd.

Chartkonstanten

Konstanten, die verschiedenen Charteigenschaften beschreiben sind in folgende Gruppen aufgeteilt:

- Typen der Ereignisse – Ereignisse, die bei der Arbeit mit Chart vorkommen;
- Chartperioden – eingebaute Standardperioden;
- Charteigenschaften – Identifikatoren, die als Parameter der Funktionen für Arbeit mit Charts verwendet werden;
- Positionierungskonstanten - Parameterwerte der Funktion ChartNavigate();
- Darstellung von Charts - Einsetzung von Chartaussehen.

© 2000-2025, MetaQuotes Ltd.

Typen der Chartereignisse

Es gibt 11 Typen der Ereignisse, die durch die vorbestimmte Funktion OnChartEvent() bearbeitet werden können. Für Benutzerereignisse werden 65535 Identifikatoren im Bereich von CHARTEVENT_CUSTOM bis CHARTEVENT_CUSTOM_LAST vorausgesehen. Für Generieren des Benutzerereignisses muss man die Funktion EventChartCustom() verwenden.

ENUM_CHART_EVENT

Identifikator	Beschreibung
CHARTEVENT_KEYDOWN	Tasten
CHARTEVENT_MOUSE_MOVE	Das Bewegen der Maus und Mausklicks (wenn die Eigenschaft CHART_EVENT_MOUSE_MOVE=true ist für den Chart eingegeben)
CHARTEVENT_MOUSE_WHEEL	Drücken oder Scrollen des Mausrads (wenn die Eigenschaft CHART_EVENT_MOUSE_WHEEL=true für den Chart gesetzt wurde)
CHARTEVENT_OBJECT_CREATE	Erzeugung vom graphischen Objekt (wenn die Eigenschaft CHART_EVENT_OBJECT_CREATE=true ist für den Chart eingegeben)
CHARTEVENT_OBJECT_CHANGE	Veranderung der Objekteigenschaften durch Dialog der Eigenschaften
CHARTEVENT_OBJECT_DELETE	Entfernung vom graphischen Objekt (wenn die Eigenschaft CHART_EVENT_OBJECT_DELETE=true ist für den Chart eingegeben)
CHARTEVENT_OBJECT_CLICK	Betaetigen der Taste mit der Maus auf dem graphischen Objekt
CHARTEVENT_OBJECT_DRAG	Ziehen des graphischen Objektes
CHARTEVENT_OBJECT_ENDEDIT	Beenden von Texteditieren im graphischen Objekt Edit
CHARTEVENT_CHART_CHANGE	Ereignis der Veränderung der Chart-Größe oder Veränderung der Chart-Eigenschaften durch Dialog von Eigenschaften
CHARTEVENT_CUSTOM	Anfangsnummer des Ereignisses vom Bereich der Benutzerereignisse
CHARTEVENT_CUSTOM_LAST	Endnummer des Ereignisses vom Bereich der Benutzerereignisse.

© 2000-2025, MetaQuotes Ltd.

Für jeden Typ des Ereignisses haben Eingabeparameter der Funktion OnChartEvent() vorbestimmte Werte, die für Verarbeitung dieses Ereignisses notwendig sind. In der Tabelle sind Ereignisse und Werte aufgezählt, die durch Parameter übertragen werden.

Ereignis	Wet des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
Ereignis des Keyboard Klickens	CHARTEVENT_KEYDOWN	Kode der betätigten Taste	Die Anzahl der Tastenanschläge, die generiert werden, während diese Taste in gedrücktem Zustand war	Der String-Wert eines Bit-Maske, die den Status der Tastatur-Tasten beschreibt
Mausereignisse (wenn die Eigenschaft CHART_EVENT_MOUSE_MOVE=true ist für den Chart eingegeben)	CHARTEVENT_MOUSE_MOVE	X-Koordinate	Y-Koordinate	Der String-Wert einer Bit-Maske, die den Status der Maustasten beschreibt
Das Ereignis des Mausrads (wenn die Eigenschaft CHART_EVENT_MOUSE_WHEEL=true für den Chart gesetzt wurde)	CHARTEVENT_MOUSE_WHEEL	Flags des Status der Tasten und Knöpfe der Maus, X- und Y-Koordinaten des Cursors. Die Beschreibung ist im Beispiel unten zu finden	Der Wert Delta des Scrollens des Mausrads	—
Ereignis der Erzeugung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_CREATE=true ist für den Chart eingegeben)	CHARTEVENT_OBJECT_CREATE	—	—	Name des erzeugten graphischen Objektes
Ereignis der Veränderung der Eigenschaften des Objektes durch Dialog der Eigenschaften	CHARTEVENT_OBJECT_CHANGE	—	—	Name des veränderten graphischen Objektes

© 2000-2025, MetaQuotes Ltd.

Ereignis	Wet des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
Ereignis der Entfernung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_DELETE=true ist für den Chart eingegeben)	CHARTEVENT_OBJECT_DELETE	—	—	Name des entfernten graphischen Objektes
Ereignis des Mausklicks auf der graphischen Darstellung	CHARTEVENT_CLICK	X-Koordinate	Y-Koordinate	—
Ereignis des Mausklicks auf dem graphischen Objekt	CHARTEVENT_OBJECT_CLICK	X-Koordinate	Y-Koordinate	Name des graphischen Objektes, auf dem das Ereignis geschehen ist
Ereignis der Verschiebung des graphischen Objektes per Mausklick	CHARTEVENT_OBJECT_DRAG	—	—	Name des verschobenen graphischen Objektes
Ereignis der Beendung des Texteditierens im Eingabefeld des graphischen Objektes "Eingabefeld"	CHARTEVENT_OBJECT_ENDEDIT	—	—	Name des graphischen Objektes "Eingabefeld", in dem Editieren des Textes beendet wurde
Ereignis der Veränderung der Chart-Größe oder Veränderung der Chart-Eigenschaften durch Dialog von Eigenschaften	CHARTEVENT_CHART_CHANGE	—	—	—
Benutzerereignis mit Nummer N	CHARTEVENT_CUSTOM+N	Wert, vorgegeben durch die	Wert, vorgegeben durch die	Wert, vorgegeben durch die

© 2000-2025, MetaQuotes Ltd.

Ereignis	Wet des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
		Funktion EventChartCustom()	Funktion EventChartCustom()	Funktion EventChartCustom()

Beispiel:

```
#define KEY_NUMPAD_5       12
#define KEY_LEFT           37
#define KEY_UP             38
#define KEY_RIGHT          39
#define KEY_DOWN           40
#define KEY_NUMLOCK_DOWN   98
#define KEY_NUMLOCK_LEFT  100
#define KEY_NUMLOCK_5     101
#define KEY_NUMLOCK_RIGHT 102
#define KEY_NUMLOCK_UP    104
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   Print("The expert with name ",MQL5InfoString(MQL5_PROGRAM_NAME)," is running");
//--- enable object create events
   ChartSetInteger(ChartID(),CHART_EVENT_OBJECT_CREATE,true);
//--- enable object delete events
   ChartSetInteger(ChartID(),CHART_EVENT_OBJECT_DELETE,true);
//--- erzwungene Aktualisierung der Charteigenschaften garantiert die Bereitschaft zu
```

© 2000-2025, MetaQuotes Ltd.

```
   ChartRedraw();
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // Event identifier
                  const long& lparam,   // Event parameter of long type
                  const double& dparam, // Event parameter of double type
                  const string& sparam  // Event parameter of string type
                  )
  {
//--- the left mouse button has been pressed on the chart
   if(id==CHARTEVENT_CLICK)
     {
      Print("The coordinates of the mouse click on the chart are: x = ",lparam,"  y =
     }
//--- the mouse has been clicked on the graphic object
   if(id==CHARTEVENT_OBJECT_CLICK)
     {
      Print("The mouse has been clicked on the object with name '"+sparam+"'");
     }
//--- the key has been pressed
   if(id==CHARTEVENT_KEYDOWN)
     {
      switch(lparam)
        {
         case KEY_NUMLOCK_LEFT:  Print("The KEY_NUMLOCK_LEFT has been pressed");   bre
         case KEY_LEFT:          Print("The KEY_LEFT has been pressed");           bre
         case KEY_NUMLOCK_UP:    Print("The KEY_NUMLOCK_UP has been pressed");     bre
         case KEY_UP:            Print("The KEY_UP has been pressed");             bre
         case KEY_NUMLOCK_RIGHT: Print("The KEY_NUMLOCK_RIGHT has been pressed"); bre
         case KEY_RIGHT:         Print("The KEY_RIGHT has been pressed");          bre
         case KEY_NUMLOCK_DOWN:  Print("The KEY_NUMLOCK_DOWN has been pressed");   bre
         case KEY_DOWN:          Print("The KEY_DOWN has been pressed");           bre
         case KEY_NUMPAD_5:      Print("The KEY_NUMPAD_5 has been pressed");       bre
         case KEY_NUMLOCK_5:     Print("The KEY_NUMLOCK_5 has been pressed");      bre
         default:                Print("Some not listed key has been pressed");
        }
      ChartRedraw();
     }
//--- the object has been deleted
   if(id==CHARTEVENT_OBJECT_DELETE)
     {
      Print("The object with name ",sparam," has been deleted");
     }
//--- the object has been created
   if(id==CHARTEVENT_OBJECT_CREATE)
     {
      Print("The object with name ",sparam," has been created");
     }
//--- the object has been moved or its anchor point coordinates has been changed
   if(id==CHARTEVENT_OBJECT_DRAG)
     {
      Print("The anchor point coordinates of the object with name ",sparam," has been
     }
//--- the text in the Edit of object has been changed
   if(id==CHARTEVENT_OBJECT_ENDEDIT)
     {
      Print("The text in the Edit field of the object with name ",sparam," has been ch
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
```

Für die Ereignisse CHARTEVENT_MOUSE_MOVE enthält der Stringparameter sparam eine Zahl, die den Zustand der Tasten übergibt:

Bit	Besreibung
1	Der Zustand der linken Maustaste
2	Der Zustand der rechten Maustaste
3	Der Zustand der SHIFT-Taste
4	Der Zustand der CTRL-Taste
5	Der Zustand der mittleren Maustaste
6	Der Zustand der ersten zusätzlichen Maustaste
7	Der Zustand der zweiten zusätzlichen Maustaste

Beispiel:

```
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- aktivieren Berichte über die Bewegung der Maus auf dem Chartfenster
   ChartSetInteger(0,CHART_EVENT_MOUSE_MOVE,1);
// --- deaktiviere das Kontextmenü des Diagramms (rechts)
   ChartSetInteger(0,CHART_CONTEXT_MENU,0);
// --- deaktiviere das Fadenkreuz (mit der mittleren Taste)
   ChartSetInteger(0,CHART_CROSSHAIR_TOOL,0);
//--- erzwungene Aktualisierung der Charteigenschaften garantiert die Bereitschaft zu
   ChartRedraw();
  }
//+------------------------------------------------------------------+
//| MouseState                                                       |
//+------------------------------------------------------------------+
string MouseState(uint state)
  {
   string res;
   res+="\nML: "   +(((state& 1)== 1)?"DN":"UP");   // mouse left
   res+="\nMR: "   +(((state& 2)== 2)?"DN":"UP");   // mouse right
   res+="\nMM: "   +(((state&16)==16)?"DN":"UP");   // mouse middle
   res+="\nMX: "   +(((state&32)==32)?"DN":"UP");   // mouse first X key
   res+="\nMY: "   +(((state&64)==64)?"DN":"UP");   // mouse second X key
   res+="\nSHIFT: "+(((state& 4)== 4)?"DN":"UP");   // shift key
   res+="\nCTRL: " +(((state& 8)== 8)?"DN":"UP");   // control key
   return(res);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,const long &lparam,const double &dparam,const string &s
  {
   if(id==CHARTEVENT_MOUSE_MOVE)
      Comment("POINT: ",(int)lparam,",",(int)dparam,"\n",MouseState((uint)sparam));
  }
```

Für das Ereignis CHARTEVENT_MOUSE_WHEEL beinhalten die Parameter **lparam** und **dparam** Informationen über den Status der Tasten **Ctrl**, **Shift**, Maustasten, Koordinaten des Cursors und den Scrollwert des Mausrads. Für ein besseres Verständnis starten Sie diesen Expert Advisor auf dem Chart und scrollen Sie das Mausrad, während Sie Knöpfe und Tasten drücken, die im Code beschrieben sind.

Ein Beispiel für die Verarbeitung des Ereignisses CHARTEVENT_MOUSE_WHEEL:

```
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
init OnInit()
  {
//--- Aktivieren der Meldungen über das Scrollen des Mausrads
   ChartSetInteger(0,CHART_EVENT_MOUSE_WHEEL,1);
//--- erzwungene Aktualisierung der Charteigenschaften garantiert die Bereitschaft zur
   ChartRedraw();
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,const long &lparam,const double &dparam,const string &s
  {
   if(id==CHARTEVENT_MOUSE_WHEEL)
     {
      //--- erläutern wir den Status der Tasten und des Mausrads für dieses Ereignis
      int flg_keys = (int)(lparam>>32);          // Flag des Status der Tasten Ctrl ur
      int x_cursor = (int)(short)lparam;         // X-Koordinate, in welcher das Ereig
      int y_cursor = (int)(short)(lparam>>16);   // Y-Koordinate, in welcher das Ereig
      int delta    = (int)dparam;                // Gesamtwert des Scrollens des Maus
      //--- Flag verarbeiten
      string str_keys="";
      if((flg_keys&0x0001)!=0) str_keys+="LMOUSE ";
      if((flg_keys&0x0002)!=0) str_keys+="RMOUSE ";
      if((flg_keys&0x0004)!=0) str_keys+="SHIFT ";
      if((flg_keys&0x0008)!=0) str_keys+="CTRL ";
      if((flg_keys&0x0010)!=0) str_keys+="MMOUSE ";
      if((flg_keys&0x0020)!=0) str_keys+="X1MOUSE ";
```

© 2000-2025, MetaQuotes Ltd.

```
      if((flg_keys&0x0040)!=0) str_keys+="X2MOUSE ";

      if(str_keys!="")
         str_keys=", keys='"+StringSubstr(str_keys,0,StringLen(str_keys)-1) + "'";
      PrintFormat("%s: X=%d, Y=%d, delta=%d%s",EnumToString(CHARTEVENT_MOUSE_WHEEL),x_
     }
  }
//+------------------------------------------------------------------+ /*
   Ausgabebeispiel
   CHARTEVENT_MOUSE_WHEEL: Ctrl pressed: X=193, Y=445, delta=-120
   CHARTEVENT_MOUSE_WHEEL: Shift pressed: X=186, Y=446, delta=120
   CHARTEVENT_MOUSE_WHEEL:  X=178, Y=447, delta=-120
   CHARTEVENT_MOUSE_WHEEL:  X=231, Y=449, delta=120
   CHARTEVENT_MOUSE_WHEEL: MiddleButton pressed: X=231, Y=449, delta=120
   CHARTEVENT_MOUSE_WHEEL: LeftButton pressed: X=279, Y=320, delta=-120
   CHARTEVENT_MOUSE_WHEEL: RightButton pressed: X=253, Y=330, delta=120  */
```

Sehen Sie auch

Funktionen der Bearbeitung von Ereignissen, Arbeit mit Ereignissen

© 2000-2025, MetaQuotes Ltd.

Chartperioden

Alle vordefinierten Zeitrahmen haben eindeutige Identifikatoren. Der Identifikator PERIOD_CURRENT bezeichnet den aktuellen Zeitrahmen eines Charts, auf welchem das mql5-Programm läuft.

ENUM_TIMEFRAMES

Identifikator	Beschreibung
PERIOD_CURRENT	Aktueller Zeitrahmen
PERIOD_M1	1 Minute
PERIOD_M2	2 Minuten
PERIOD_M3	3 Minuten
PERIOD_M4	4 Minuten
PERIOD_M5	5 Minuten
PERIOD_M6	6 Minuten
PERIOD_M10	10 Minuten
PERIOD_M12	12 Minuten
PERIOD_M15	15 Minuten
PERIOD_M20	20 Minuten
PERIOD_M30	30 Minuten
PERIOD_H1	1 Stunde
PERIOD_H2	2 Stunden
PERIOD_H3	3 Stunden
PERIOD_H4	4 Stunden
PERIOD_H6	6 Stunden
PERIOD_H8	8 Stunden
PERIOD_H12	12 Stunden
PERIOD_D1	1 Tag
PERIOD_W1	1 Woche
PERIOD_MN1	1 Monat

Beispiel:

```
   string chart_name="test_Object_Chart";
   Print("Versuchen wir das Objekt Chart mit dem Namen zu erstellen",chart_name);
//--- wenn dieses Objekt fehlt, erstellen wir es
   if(ObjectFind(0,chart_name)<0)ObjectCreate(0,chart_name,OBJ_CHART,0,0,0,0,0);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Symbol definieren
   ObjectSetString(0,chart_name,OBJPROP_SYMBOL,"EURUSD");
//--- X-Koordinate für den Ankerpunkt vorgeben
   ObjectSetInteger(0,chart_name,OBJPROP_XDISTANCE,100);
//--- Y-Koordinate für den Ankerpunkt vorgeben
   ObjectSetInteger(0,chart_name,OBJPROP_YDISTANCE,100);
//--- Breite setzen
   ObjectSetInteger(0,chart_name,OBJPROP_XSIZE,400);
//--- Höhe setzen
   ObjectSetInteger(0,chart_name,OBJPROP_YSIZE,300);
//--- Zeitrahmen setzen
   ObjectSetInteger(0,chart_name,OBJPROP_PERIOD,PERIOD_D1);
//--- Maßstab setzen ( von 0 bis 5)
   ObjectSetDouble(0,chart_name,OBJPROP_SCALE,4);
//--- Hervorheben mit der Maus deaktivieren
   ObjectSetInteger(0,chart_name,OBJPROP_SELECTABLE,false);
```

Identifikatoren der Zeitreihen

Die Identifikatoren der Zeitreihen werden in den Funktion iHighest() und iLowest() verwendet. Sie können gleich einem der Werte der Aufzählung sein

ENUM_SERIESMODE

Identifier	Beschreibung
MODE_OPEN	Eröffnungspreis
MODE_LOW	Tief
MODE_HIGH	Hoch
MODE_CLOSE	Schlusspreis
MODE_VOLUME	Tick-Volumen
MODE_REAL_VOLUME	Echtes Volumen
MODE_SPREAD	Spread

Siehe auch

PeriodSeconds, Period, Datum und Zeit, Objektsichtbarkeit

© 2000-2025, MetaQuotes Ltd.

Charteigenschaften

Identifikatoren der Enumerationsfamilie ENUM_CHART_PROPERTY verwenden als Parameter Funktionen für Abeit mit Charts. Initialkurzwort r/o in der Spalte "Typ der Eigenschaft" bedeutet, dass diese Eigenschaft erst für Lesen bestimmt ist und kann nicht verändert werden. Abkürzung w/o in der Spalte "Typ der Eigenschaft" bedeutet, dass diese Eigenschaft write-only ist und kann nicht erhalten werden. Beim Zugriff zu einigen Eigenschaften ist es notwendig, zusätzlichen Parameter-Modifikator (modifier) anzugeben, der der Andeutung der Nummer von Chartsubfenster dient.

Funktionen, die Eigenschaften des Charts setzen, werden in der Tat verwendet, um Befehle zu ändern zu senden. Der erfolgreiche Erfüllung dieser Funktionen wird der Befehl in die allgemeine Warteschlange der Chartereignisse platziert. Der Chart wird im Zuge der Verarbeitung der Ereigniswarteschlange von diesem Chart geändert.

Aus diesem Grund sollte man nicht erwarten, dass der Chart sofort nach dem Aufruf dieser Funktionen visuell aktualisiert wird. Im Allgemeinen ist der Chart durch das Terminal automatisch nach Änderung-Ereignisse aktualisiert - die Ankunft der neuen Quote, Änderung von Chartfenstergröße, etc.

Um Aussehen des Charts zu aktualisieren, verwenden Sie ChartRedraw().

für Funktionen ChartSetInteger() und ChartGetInteger()

ENUM_CHART_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
CHART_SHOW	Zeichnung eines Preischarts. Wenn false, wird die Zeichnung jeglicher Eigenschaften eines Preischarts deaktiviert und alle Randabstände des Charts werden entfernt: Zeit- und Preisskala, die Leiste der schnellen Navigation, Markierungen von Ereignissen im Kalender, Zeichen von Trades, Tooltips von Indikatoren und Balken, Unterfenster von	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Indikatoren, Histogramme der Volumina usw. Die Deaktivierung der Zeichnung stellt eine ideale Lösung für die Erstellung eines eigenen Interfaces des Programms unter Verwendung grafischer Ressourcen dar. Die grafischen Objekte werden immer unabhängig vom Wert der CHART_SHOW Eigenschaft gezeichnet.	
CHART_IS_OBJECT	Attribut um Objekt "Chart" (OBJ_CHART) zu identifizieren - für ein grafisches Objekt gibt true zurück. Für dieses Chart ist die Eigenschaft false	bool r/o
CHART_BRING_TO_TOP	Den Chart über anderen Chart anzeigen	bool
CHART_CONTEXT_MENU	Aktivieren/Deaktivieren des Zugriffs auf das Kontexmenü durch einen Klick auf die rechte Maustaste. Der Wert CHART_CONTEX	bool (standardmäßig true)

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	T_MENU=false deaktiviert nur das Kontextmenü des Charts, dabei ist das Kontexmenü für Objekte auf dem Chart verfügbar.	
CHART_CROSSHAIR_TOOL	Aktivieren/Deaktivieren des Zugriffs auf das Tool "Fadenkreuz" durch das Drücken auf die mittlere Maustaste	bool (standardmäßig true)
CHART_MOUSE_SCROLL	Den Chart mit der linken Maustaste horizontal scrollen. Vertikales Scrollen wird verfügbar sein, wenn der Wert von einer der drei Eigenschaften true ist: CHART_SCALEFIX, CHART_SCALEFIX_11 oder CHART_SCALE_PT_PER_BAR Bei CHART_MOUSE_SCROLL=false ist das Scrollen des Charts mithilfe des Mausrads nicht möglich.	bool
CHART_EVENT_MOUSE_WHEEL	Senden von Meldungen über die Ereignisse des Mausrads	bool (standardmäßig true)

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	(CHARTEVENT_MOUSE_WHEEL) an alle mql5-Programme	
CHART_EVENT_MOUSE_MOVE	Senden alle MQL5-Programme auf dem Chart Benachrichtigungen über Ereignisse der Bewegung und Klicken der Maustasten (CHARTEVENT_MOUSE_MOVE)	bool
CHART_EVENT_OBJECT_CREATE	Senden alle MQL5-Programme auf dem Chart eine Benachrichtigung über Ereignis des neuen Objekts Erstellung (CHARTEVENT_OBJECT_CREATE)	bool
CHART_EVENT_OBJECT_DELETE	Senden alle MQL5-Programme auf dem Chart eine Benachrichtigung über Ereignis des Objekts Löschung (CHARTEVENT_OBJECT_DELETE)	bool
CHART_MODE	Charttyp (Kerzen, Bars, oder Linie)	enum ENUM_CHART_MODE
CHART_FOREGROUND	Preischart auf dem Hintergrund	bool
CHART_SHIFT	Absatz des Preischarts rechtsbündig	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
CHART_AUTOSCROLL	Automatischer Sprung zum rechten Schlussrand des Charts	bool
CHART_KEYBOARD_CONTROL	Erlaubt den Chart mit der Tastatur zu steuern ("Home", "End", "PageUp", "+", "-", "Pfeil nach oben" usw). Wenn man CHART_KEYBOARD_CONTROL auf false setzt, werden das Scrollen und die Skallierung des Charts deaktiviert, dabei kann man die Ereignisse des Drückens dieser Tasten in OnChartEvent() weiterhin erhalten.	bool
CHART_QUICK_NAVIGATION	Es erlaubt dem Chart, Tastendrücke der Leertaste und Eingabetaste abzufangen, um die Leiste der schnellen Navigation zu aktivieren. Die Leiste der schnellen Navigation erscheint automatisch auf dem Chart unten beim Doppelklick oder beim Drücken der	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Leer-/Eingabetaste. Auf diese Weise kann man das Symbol, den Zeitrahmen und das Datum des ersten sichtbaren Balkens schnell ändern.	
CHART_SCALE	Maßstab	int von 0 bis 5
CHART_SCALEFIX	Festmaßstab	bool
CHART_SCALEFIX_11	Maßstab 1:1	bool
CHART_SCALE_PT_PER_BAR	Masßtab in Punkt/Balken	bool
CHART_SHOW_TICKER	Den Symbolticker in der linken, oberen Ecke anzeigen. Wird CHART_SHOW_TICKER auf 'false' gesetzt, wird auch CHART_SHOW_OHLC auf 'false' gesetzt und deaktiviert OHLC.	bool
CHART_SHOW_OHLC	Die Werte von OHLC in der linken, oberen Ecke anzeigen. Wird CHART_SHOW_OHLC auf 'true' gesetzt, wird auch CHART_SHOW_TICKER auf 'true' gesetzt und aktiviert den Ticker	bool
CHART_SHOW_BID_LINE	Darstellung des Wertes Bid als	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	horizontale Linie auf dem Chart	
CHART_SHOW_ASK_LINE	Darstellung des Wertes Ask als horizontale Linie auf dem Chart	bool
CHART_SHOW_LAST_LINE	Darstellung des Wertes Last durch horizontale Linie auf dem Chart	bool
CHART_SHOW_PERIOD_SEP	Darstellung der senkrechten Begrenzungen unter Nachbarperioden	bool
CHART_SHOW_GRID	Darstellung des Zeitrasters auf dem Chart	bool
CHART_SHOW_VOLUMES	Darstellung von Volumen auf dem Chart	enum ENUM_CHART_VOLUME_MODE
CHART_SHOW_OBJECT_DESCR	Textbeschreibung von Objekten anzeigen (nicht für alle Objekte)	bool
CHART_SHOW_TRADE_HISTORY	Darstellung der historischen Handelstätigkeiten durch Pfeile für Positionsanfang und -ende auf einem Chart. Siehe die Beschreibungen der Option "Handelshistorie anzeigen" in den Plattformeinstellungen.	bool
CHART_VISIBLE_BARS	Anzahl der Balken auf dem Chart, zugänglich	int r/o

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	für die Darstellung	
CHART_WINDOWS_TOTAL	Gesamtzahl der Chartfenster, einschließlich Subfenster der Indikatoren	int r/o
CHART_WINDOW_IS_VISIBLE	Sichtbarkeit der Subfenster	bool r/o Modifikator - Nummer des Subfensters
CHART_WINDOW_HANDLE	Handle des Charts (HWND)	int r/o
CHART_WINDOW_YDISTANCE	Der Abstand zwischen dem oberen Rahmen des Unterfensters eines Indikators und dem oberen Rahmen des Hauptfensters des Charts, entlang der vertikalen Y-Achse, in Pixeln. Im Falle eines Maus-Ereignisses, werden die Cursor-Koordinaten im Koordinaten des Hauptfensters übergegeben, während die Koordinaten von grafischen Objekten im Indikator-Unterfenster relativ zur oberen linken Ecke des Unterfensters eingestellt sind. Der Wert ist erforderlich, um	int r/o Modifikator - Nummer des Subfensters

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	die absoluten Koordinaten des Hauptfensters zu den lokalen Koordinaten eines Unterfensters für die korrekte Arbeit mit der grafischen Objekte, deren Koordinaten sind relativ zur oberen linken Ecke des Unterfensters angegeben, umzuwandeln.	
CHART_FIRST_VISIBLE_BAR	Nummer der ersten sichtbaren Balken auf dem Chart. Indizierung von Balken ist das gleiche wie für Zeitreihen.	int r/o
CHART_WIDTH_IN_BARS	Chart-Breite in Balken	int r/o
CHART_WIDTH_IN_PIXELS	Chart-Breite in Pixel	int r/o
CHART_HEIGHT_IN_PIXELS	Chart-Höhe in Pixel	int Modifikator - Nummer des Subfensters
CHART_COLOR_BACKGROUND	Hintergrundfarbe des Charts	color
CHART_COLOR_FOREGROUND	Farbe von Achsen, Skale und Zeile OHLC	color
CHART_COLOR_GRID	Farbe des Zeitrasters	color
CHART_COLOR_VOLUME	Farbe der Volumen und Levels der offenen Positionen	color

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
CHART_COLOR_CHART_UP	Farbe des Balkens oben, des Schattens und der Umrandung des Körpers der Bullenkerze	color
CHART_COLOR_CHART_DOWN	Farbe des Balkens unten, des Schattens und Umrandung des Körpers der Bärenkerze	color
CHART_COLOR_CHART_LINE	Farbe der Chartlinie und der japanischen Kerzen "Dodzhi"	color
CHART_COLOR_CANDLE_BULL	Körperfarbe der der Bullenkerze	color
CHART_COLOR_CANDLE_BEAR	Körperfarbe der Bärenkerze	color
CHART_COLOR_BID	Linienfarbe des Bid-Preises	color
CHART_COLOR_ASK	Linienfarbe des Ask-Preises	color
CHART_COLOR_LAST	Linienfarbe des Preises des letzten abgeschlossenen Deals (Last)	color
CHART_COLOR_STOP_LEVEL	Farbe der Stop-Ordern auf den Chart (Stop Loss und Take Profit)	color
CHART_SHOW_TRADE_LEVELS	Darstellung der Handelsniveaus auf dem Chart (Niveaus der offenen Positionen, Stop Loss, Take Profit und wartender Order)	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
CHART_DRAG_TRADE_LEVELS	Ziehen der Handelsstufen auf dem Chart mit dem Maus erlauben. Standardmäßig ist der Ziehen-Modus aktiviert (true)	bool
CHART_SHOW_DATE_SCALE	Anzeige der Zeit-Skala auf dem Chart	bool
CHART_SHOW_PRICE_SCALE	Anzeige der Preis-Skala auf dem Chart	bool
CHART_SHOW_ONE_CLICK	Anzeige des Panels für schnellen Handel auf dem Chart (Option "Ein-Klick-Panel")	bool
CHART_IS_MAXIMIZED	Chartfenster vergrößert	bool r/o
CHART_IS_MINIMIZED	Chartfenster verkleinert	bool r/o
CHART_IS_DOCKED	Das Chartfenster ist angedockt. Wenn man den Wert false angibt, kann der Chart außerhalb des Terminals verschoben werden	bool
CHART_FLOAT_LEFT	Die linke Koordinate des abgedockten Charts hinsichtlich des virtuellen Bildschirms	int
CHART_FLOAT_TOP	Die obere Koordinate des abgedockten	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Charts hinsichtlich des virtuellen Bildschirms	
CHART_FLOAT_RIGHT	Die rechte Koordinate des abgedockten Charts hinsichtlich des virtuellen Bildschirms	int
CHART_FLOAT_BOTTOM	Die untere Koordinate des abgedockten Charts hinsichtlich des virtuellen Bildschirms	int

für die Funktionen ChartSetDouble() und ChartGetDouble()

ENUM_CHART_PROPERTY_DOUBLE

Identifikator	Beschreibung	Typ der Eigenschaft
CHART_SHIFT_SIZE	Absatzgröße der Nullbar vom rechten Schlussrand	double (von 10 bis 50 Prozant)
CHART_FIXED_POSITION	Position der festen Position des Chart vom linken Rand als Prozentsatz. Feste Position des Charts ist mit einem kleinen grauen Dreieck auf der horizontalen Achse der Zeit gezeichnet und wird nur angezeigt, wenn Autoscroll des Charts mit einem neuen Tick nach	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	rechts deaktiviert ist (siehe Eigenschaft CHART_AUTOSCROLL). Der Bar, der in der feste Position ist, bleibt an der gleiche Stelle mit Zoom-in und Zoom-out.	
CHART_FIXED_MAX	Festes Chartmaximum	double
CHART_FIXED_MIN	Festes Chartminimum	double
CHART_POINTS_PER_BAR	Massstabwert in Punkt/Balken	double
CHART_PRICE_MIN	Chartminimum	double r/o Modifikator - Nummer des Subfensters
CHART_PRICE_MAX	Chartmaximum	double r/o Modifikator - Nummer des Subfensters

für die Funktionen ChartSetString() und ChartGetString()

ENUM_CHART_PROPERTY_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
CHART_COMMENT	Kommentartext auf dem Chart	string
CHART_EXPERT_NAME	Der Name des Expert Advisors, der auf dem Chart mit dem angegebenen chart_id gestartet wurde	string r/o
CHART_SCRIPT_NAME	Der Name des Skripts, der auf dem Chart mit dem angegebenen chart_id gestartet wurde	string r/o

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
int chartMode=ChartGetInteger(0,CHART_MODE);
switch(chartMode)
 {
   case(CHART_BARS):    Print("CHART_BARS");   break;
   case(CHART_CANDLES): Print("CHART_CANDLES");break;
   default:Print("CHART_LINE");
  }
bool shifted=ChartGetInteger(0,CHART_SHIFT);
if(shifted) Print("CHART_SHIFT = true");
else Print("CHART_SHIFT = false");
bool autoscroll=ChartGetInteger(0,CHART_AUTOSCROLL);
if(autoscroll) Print("CHART_AUTOSCROLL = true");
else Print("CHART_AUTOSCROLL = false");
int chartHandle=ChartGetInteger(0,CHART_WINDOW_HANDLE);
Print("CHART_WINDOW_HANDLE = ",chartHandle);
int windows=ChartGetInteger(0,CHART_WINDOWS_TOTAL);
Print("CHART_WINDOWS_TOTAL = ",windows);
if(windows>1)
 {
   for(int i=0;i<windows;i++)
    {
      int height=ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,i);
      double priceMin=ChartGetDouble(0,CHART_PRICE_MIN,i);
      double priceMax=ChartGetDouble(0,CHART_PRICE_MAX,i);
      Print(i+": CHART_HEIGHT_IN_PIXELS = ",height," pixels");
      Print(i+": CHART_PRICE_MIN = ",priceMin);
      Print(i+": CHART_PRICE_MAX = ",priceMax);
    }
 }
```

Siehe auch

Beispiele von Operationen mit dem Chart

© 2000-2025, MetaQuotes Ltd.

Positionieren des Charts

Drei Identifikatoren aus der Liste ENUM_CHART_POSITION sind moegliche Werte des Parameters position für die Funktion ChartNavigate().

ENUM_CHART_POSITION

Identifikator	Beschreibung
CHART_BEGIN	Anfang des Charts (die ältesten Preise)
CHART_CURRENT_POS	Laufende Position
CHART_END	Ende des Charts (die neuesten Preise)

Beispiel:

```
long handle=ChartOpen("EURUSD",PERIOD_H12);
if(handle!=0)
  {
   ChartSetInteger(handle,CHART_AUTOSCROLL,false);
   ChartSetInteger(handle,CHART_SHIFT,true);
   ChartSetInteger(handle,CHART_MODE,CHART_LINE);
   ResetLastError();
   bool res=ChartNavigate(handle,CHART_END,150);
   if(!res) Print("Navigate failed. Error = ",GetLastError());
   ChartRedraw();
  }
```

© 2000-2025, MetaQuotes Ltd.

Darstellung des Charts

Preischarts können durch drei Verfahren dargestellt werden:

- in Form von Bars;
- in Form von Kerzen;
- in Form von einer gebrochenen Linie.

Konkretes Verfahren der Darstellung vom Preischart ist durch die Funktion ChartSetInteger(handle_des Digramms,CHART_MODE, Typ_des_Charts) eingesetzt, wo Typ_des_Charts - einer der Werte von Enumeration ENUM_CHART_MODE.

ENUM_CHART_MODE

Identifikator	Beschreibung
CHART_BARS	Darstellung in Form von Bars
CHART_CANDLES	Darstellung in Form von japanischen Kerzen
CHART_LINE	Darstellung in Form einer Linie, gezogen an den Preisen Close

für Andeutung des Image-Betrieves auf dem Preischart wird die Funktion ChartSetInteger(handle_des Charts, CHART_SHOW_VOLUMES,Chart_Typ) verwendet, wo Chart_Typ einer der Werte von Enumeration ENUM_CHART_VOLUME_MODE ist.

ENUM_CHART_VOLUME_MODE

Identifikator	Beschreibung
CHART_VOLUME_HIDE	Volumen nicht gezeigt
CHART_VOLUME_TICK	Tickvolumen
CHART_VOLUME_REAL	Handelsvolumen

Beispiel:

```
//--- bekommen handle des laufenden Charts
   long handle=ChartID();
   if(handle>0) // wenn es gelungen ist, geben wir zuaetzliche Einstellungen vor
     {
      //--- stellen wir auto-panning ab
      ChartSetInteger(handle,CHART_AUTOSCROLL,false);
      //--- tragen wir den Chart rechtsbündig ein
      ChartSetInteger(handle,CHART_SHIFT,true);
      //--- stellen wir in Form von Kerzen dar
      ChartSetInteger(handle,CHART_MODE,CHART_CANDLES);
      //--- laufen wir um 100 Bars vom Anfang der Geschichte
      //--- stellen wir Darstellung von Tickvolumen ein
      ChartSetInteger(handle,CHART_SHOW_VOLUMES,CHART_VOLUME_TICK);
     }
```

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

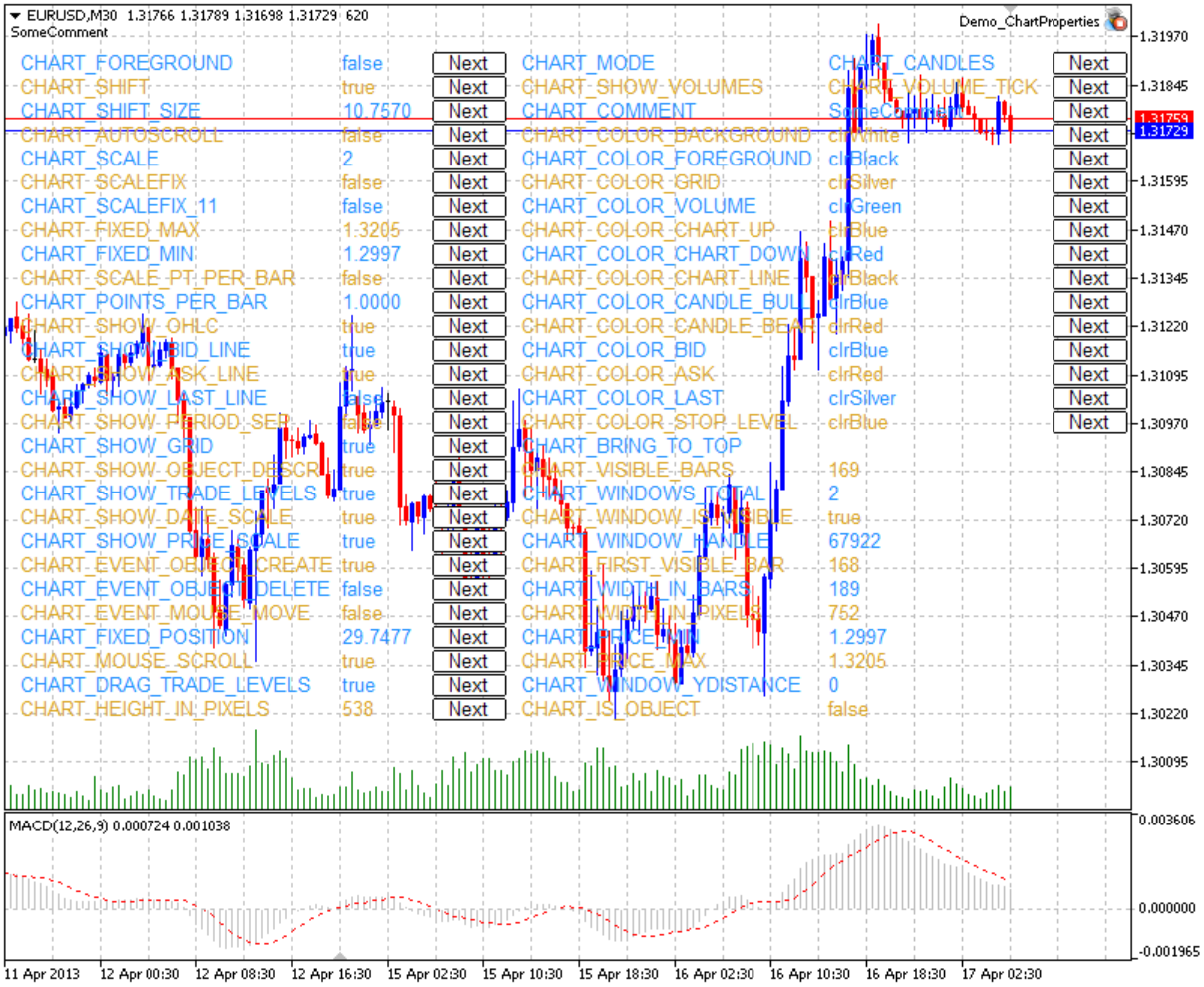
ChartOpen, ChartID

© 2000-2025, MetaQuotes Ltd.

Beispiele von Operationen mit dem Chart

Dieser Abschnitt enthält Beispiele für die Eigenschaften des Charts. Für jede Eigenschaft gibt es hier ein oder zwei komplette Funktionen, die den Wert dieser Eigenschaft eingeben/erhalten erlauben. Diese Funktionen können in Ihrer MQL5 Programme verwendet werden.

Die Abbildung zeigt eine grafische Panel, die zeigt, wie Veränderung des Charts sein Aussehen ändert. Ein Klick auf "Next" erlaubt es Ihnen den neuen Wert der jeweiligen Eigenschaft einzugeben und die Änderungen im Chart sehen.



Quellcode der Panel ist unten.

Chart-Eigenschaften und Beispiele von Funktionen für Arbeit mit ihnen

- **CHART_IS_OBJECT** - bestimmt, ob das Objekt ein Chart oder ein grafisches Objekt ist.

```
//+------------------------------------------------------------------+
//| Die Ermittlung, ob das Objekt ein Chart ist. Wenn es             |
//| ein grafisches Objekt ist, ist das Ergebnis true. Wenn es ein    |
//| Chart ist, wird das Variable result false gesetzt werden.        |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool ChartIsObject(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten die Eigenschaft des Charts
   if(!ChartGetInteger(chart_ID,CHART_IS_OBJECT,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      //--- false zurückgeben
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_BRING_TO_TOP** - zeigt ein Chart über alle anderen.

```
//+------------------------------------------------------------------+
//| Senden Befehl dem Terminal, Chart über alle anderen zu zeigen.   |
//+------------------------------------------------------------------+
bool ChartBringToTop(const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeigt wir den Chart über alle anderen
   if(!ChartSetInteger(chart_ID,CHART_BRING_TO_TOP,0,true))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_MOUSE_SCROLL** - Eigenschaft von Chart-Scroll mit der linken Maustaste.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob Scroll des Charts mit der             |
//| linken Maustaste möglich ist.                                    |
//+------------------------------------------------------------------+
bool ChartMouseScrollGet(bool &result,const long chart_ID=0)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_MOUSE_SCROLL,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Scroll mit der linken Tasten  |
//| möglich ist.                                                     |
//+------------------------------------------------------------------+
bool ChartMouseScrollSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_MOUSE_SCROLL,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_EVENT_MOUSE_MOVE** - Eigenschaft der Sendung von Nachrichten über Ereignisse der Bewegung und Klicken an den MQL5-Programmen (CHARTEVENT_MOUSE_MOVE).

```
//+------------------------------------------------------------------+
//| Werden die Nachrichten über Ereignisse der Bewegung und Klicken  |
//| an allen MQL5-Programmen auf diesem Chart gesendet?              |
//+------------------------------------------------------------------+
bool ChartEventMouseMoveGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_EVENT_MOUSE_MOVE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert das Senden von Nachrichten    |
//| über Ereignisse von Bewegung und Klicken an allen                |
//| MQL5-Programmen auf diesem Chart.                                |
//+------------------------------------------------------------------+
bool ChartEventMouseMoveSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_EVENT_MOUSE_MOVE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_EVENT_OBJECT_CREATE** - Eigenschaft der Sendung von Nachrichten über Ereignisse der Erstellung des grafischen Objektes an den MQL5-Programmen (CHARTEVENT_OBJECT_CREATE).

```
//+------------------------------------------------------------------+
//| Werden die Nachrichten über Ereignisse der Erstellung eines      |
//| Objektes an allen MQL5-Programmen auf diesem Chart gesendet?     |
//+------------------------------------------------------------------+
bool ChartEventObjectCreateGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_EVENT_OBJECT_CREATE,0,value))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- Schreiben die Fehlermeldung in den Log "Experten"
       Print(__FUNCTION__+", Error Code = ",GetLastError());
       return(false);
      }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert das Senden von Nachrichten    |
//| über Ereignisse von Erstellung eines grafischen Objektes an allen|
//| MQL5-Programmen auf diesem Chart.                                |
//+------------------------------------------------------------------+
bool ChartEventObjectCreateSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_EVENT_OBJECT_CREATE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_EVENT_OBJECT_DELETE** - Eigenschaft der Sendung von Nachrichten über Ereignisse der Löschung des grafischen Objektes an den MQL5-Programmen (CHARTEVENT_OBJECT_DELETE).

```
//+------------------------------------------------------------------+
//| Die Nachrichten über Ereignisse der Löschung eines grafischen    |
//| Objektes an allen MQL5-Programmen auf diesem Chart zu senden?    |
//+------------------------------------------------------------------+
bool ChartEventObjectDeleteGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_EVENT_OBJECT_DELETE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert das Senden von Nachrichten    |
//| über Ereignisse von Löschung eines grafischen Objektes an allen  |
//| MQL5-Programmen auf diesem Chart.                                |
//+------------------------------------------------------------------+
bool ChartEventObjectDeleteSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_EVENT_OBJECT_DELETE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_MODE** - Typ des Charts (Kerzen, Bars oder eine Linie).

```
//+------------------------------------------------------------------+
//| Bekommen wir den Typ des Charts (in Form von Kerzen, Bars oder   |
//| einer Linie).                                                    |
//+------------------------------------------------------------------+
ENUM_CHART_MODE ChartModeGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=WRONG_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_MODE,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((ENUM_CHART_MODE)result);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Setzen den Typ des Charts (in Form von Kerzen, Bars oder         |
//| einer Linie).                                                    |
//+------------------------------------------------------------------+
bool ChartModeSet(const long value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_MODE,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_FOREGROUND** - Eigenschaft der Anzeige vom Preischart im Vordergrund.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob das Preischart im Vordergrund         |
//| angezeigt wird.                                                  |
//+------------------------------------------------------------------+
bool ChartForegroundGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_FOREGROUND,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige des Preischarts im    |
//| Vordergrund.                                                     |
//+------------------------------------------------------------------+
bool ChartForegroundSet(const bool value,const long chart_ID=0)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_FOREGROUND,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHIFT** - Modus von der Verschiebung des Preischarts vom rechten Rand.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Anzeigemodus des Charts mit  |
//| Verschiebung vom rechten Rand aktiv ist.                         |
//+------------------------------------------------------------------+
bool ChartShiftGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHIFT,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige des Preischarts mit   |
//| Verschiebung vom rechten Rand.                                   |
//+------------------------------------------------------------------+
bool ChartShiftSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHIFT,0,value))
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_AUTOSCROLL** - Modus der automatischen Scrollen an den rechten Rand des Charts.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob den Modus der automatischen Scrollen  |
//| des Charts nach rechts, wenn neue Ticks empfangen werden,        |
//| aktiv ist.                                                       |
//+------------------------------------------------------------------+
bool ChartAutoscrollGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_AUTOSCROLL,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert die automatische Scrollen     |
//| des Charts nach rechts, wenn neue Ticks empfangen werden.        |
//+------------------------------------------------------------------+
bool ChartAutoscrollSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_AUTOSCROLL,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SCALE** - Eigenschaft der Chart-Skala.

```
//+------------------------------------------------------------------+
//| Bekommen die Skala des Charts (0 bis 5).                         |
//+------------------------------------------------------------------+
int ChartScaleGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SCALE,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
//+------------------------------------------------------------------+
//| Setzen die Skala des Charts (0 bis 5).                           |
//+------------------------------------------------------------------+
bool ChartScaleSet(const long value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SCALE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SCALEFIX** - Modus der festen Skala des Charts.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Modus der festen Skala aktiv ist. |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ChartScaleFixGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SCALEFIX,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert die feste Skala               |
//+------------------------------------------------------------------+
bool ChartScaleFixSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SCALEFIX,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SCALEFIX_11** - Modus der Chartskala 1:1.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Modus der Skala "1:1" aktiv ist.  |
//+------------------------------------------------------------------+
bool ChartScaleFix11Get(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SCALEFIX_11,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert die Skala "1:1"               |
//+------------------------------------------------------------------+
bool ChartScaleFix11Set(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SCALEFIX_11,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SCALE_PT_PER_BAR** - Modus der Eingabe der Skala in Punkten pro Bar.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Eingabemodus der Skala       |
//| in Punkten  pro Bar aktiv ist.                                   |
//+------------------------------------------------------------------+
bool ChartScalePerBarGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SCALE_PT_PER_BAR,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Modus der Eingabe der Skala   |
//| in Punkten pro Bar.                                              |
//+------------------------------------------------------------------+
bool ChartScalePerBarSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SCALE_PT_PER_BAR,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_OHLC** - Eigenschaft der Anzeige der Werten OHLC in der linken oberen Ecke.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Anzeigemodus von OHLC-Werten      |
//| in der linken oberen Ecke aktiv ist.                             |
//+------------------------------------------------------------------+
bool ChartShowOHLCGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_OHLC,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige der Werten OHLC in der|
//| linken oberen Ecke des Charts.                                   |
//+------------------------------------------------------------------+
bool ChartShowOHLCSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_OHLC,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_BID_LINE** - Eigenschaft der Anzeige des Bid-Wertes als eine horizontale Linie auf dem Chart.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt der Modus der Anzeige der Bid-Linie       |
//| auf dem Chart.                                                   |
//+------------------------------------------------------------------+
bool ChartShowBidLineGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_BID_LINE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige der Bid-Linie auf dem |
//| Chart.                                                           |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ChartShowBidLineSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_BID_LINE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_ASK_LINE** - Eigenschaft der Anzeige des Ask-Wertes als eine horizontale Linie auf dem Chart.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt die Modus der Anzeige der Ask-Linie       |
//| auf dem Chart.                                                   |
//+------------------------------------------------------------------+
bool ChartShowAskLineGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_ASK_LINE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige der Ask-Linie auf dem |
//| Chart.                                                           |
//+------------------------------------------------------------------+
bool ChartShowAskLineSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_ASK_LINE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_LAST_LINE** - Eigenschaft der Anzeige des Last-Wertes als eine horizontale Linie auf dem Chart.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Anzeigemodus der Linie für   |
//| den Preis der letzten Transaktion aktiv ist.                     |
//+------------------------------------------------------------------+
bool ChartShowLastLineGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_LAST_LINE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige der Linie des Preises |
//| der letzten Transaktion.                                         |
//+------------------------------------------------------------------+
bool ChartShowLastLineSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_LAST_LINE,0,value))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_PERIOD_SEP** - Eigenschaft der Anzeige der vertikalen Teilern zwischen benachbarten Perioden.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt den Anzeigemodus der vertikalen      |
//| Teilern zwischen benachbarten Perioden.                          |
//+------------------------------------------------------------------+
bool ChartShowPeriodSeparatorGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_PERIOD_SEP,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert die Anzeige der vertikalen    |
//| Teilern zwischen benachbarten Perioden.                          |
//+------------------------------------------------------------------+
bool ChartShowPeriodSepapatorSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_PERIOD_SEP,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_GRID** - Eigenschaft der Anzeige des Chartrasters.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob der Chartraster angezeigt wird.       |
//+------------------------------------------------------------------+
bool ChartShowGridGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_GRID,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert den Chartraster.              |
//+------------------------------------------------------------------+
bool ChartShowGridSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_GRID,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_VOLUMES** - Eigenschaft der Anzeige der Volumen auf dem Chart.

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob die Volumen auf dem Chart angezeigt   |
//| werden (werden nicht angezeigt, Tick-Volumen angezeigt werden,   |
//| reale Volumen angezeigt werden).                                 |
//+------------------------------------------------------------------+
ENUM_CHART_VOLUME_MODE ChartShowVolumesGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=WRONG_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_VOLUMES,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((ENUM_CHART_VOLUME_MODE)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt den Modus der Anzeige der Volumina            |
//| auf dem Chart.                                                   |
//+------------------------------------------------------------------+
bool ChartShowVolumesSet(const long value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_VOLUMES,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_OBJECT_DESCR** - Eigenschaft der Pop-up-Beschreibungen von grafischen Objekten.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob die Pop-up-Beschreibungen der         |
//| grafischen Objekten angezeigt werden, wenn Sie mit dem Mauszeiger|
//| auf ihnen deuten.                                                |
//+------------------------------------------------------------------+
bool ChartShowObjectDescriptionGet(bool &result,const long chart_ID=0)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_OBJECT_DESCR,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert die Anzeige der               |
//| Popup-Beschreibungen der grafischen Objekten wenn Sie mit dem Mauszeiger auf ihner
//| Popup-Beschreibungen der grafischen Objekten wenn Sie mit dem Mauszeiger auf ihner
//+------------------------------------------------------------------+
bool ChartShowObjectDescriptionSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_OBJECT_DESCR,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_VISIBLE_BARS** - ermittelt die Anzahl der Baren auf dem Chart, die zur Anzeige verfügbar sind.

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt die Anzahl der Bars, die angezeigt  wird    |
//| im Chart-Fenster.                                                |
//+------------------------------------------------------------------+
int ChartVisibleBars(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_VISIBLE_BARS,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_WINDOWS_TOTAL** - ermittelt die Gesamtzahl der Chart-Fenster, einschließlich Unterfenster der Indikatoren.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Gesamtzahl der Chart-Fenster             |
//| der Indikatoren.                                                 |
//+------------------------------------------------------------------+
int ChartWindowsTotal(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_WINDOWS_TOTAL,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_WINDOW_IS_VISIBLE** - ermittelt die Sichtbarkeit des Unterfensters.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt,ob ein Fenster oder Unterenster des Charts|
//| sichtbar ist.                                                    |
//+------------------------------------------------------------------+
bool ChartWindowsIsVisible(bool &result,const long chart_ID=0,const int sub_window=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ChartGetInteger(chart_ID,CHART_WINDOW_IS_VISIBLE,sub_window,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_WINDOW_HANDLE** - gibt ein Handle des Charts zurück.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Handle des Charts                        |
//+------------------------------------------------------------------+
int ChartWindowsHandle(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_WINDOW_HANDLE,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_WINDOW_YDISTANCE** - ermittelt den Abstand in Pixel zwischen dem oberen Rahmen des Indikator-Unterfensters und dem oberen Rahmen des Hauptfensters des Charts.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Abstand in Pixel zwischen dem oberen     |
//| Rahmen des Unterfensters und oberen Rahmen des Hauptfensters.    |
//+------------------------------------------------------------------+
int ChartWindowsYDistance(const long chart_ID=0,const int sub_window=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ChartGetInteger(chart_ID,CHART_WINDOW_YDISTANCE,sub_window,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_FIRST_VISIBLE_BAR** - gibt den Nummer des ersten sichtbaren Bars auf dem Chart zurück (Indizierung der Baren wie in Zeitreihe).

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt die Nummer des ersten sichtbaren Bars des Charts
//| Indexing als in Zeitreihe, dass die letzten Bars kleineren Indizes haben
//+------------------------------------------------------------------+
int ChartFirstVisibleBar(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_FIRST_VISIBLE_BAR,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_WIDTH_IN_BARS** - gibt die Breite des Chart in den Baren zurück.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Wert der Breite des Charts in den Baren  |
//+------------------------------------------------------------------+
int ChartWidthInBars(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_WIDTH_IN_BARS,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_WIDTH_IN_PIXELS** - gibt die Breite des Chart in den Pixeln zurück.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Wert der Breite des Charts in Pixeln     |
//+------------------------------------------------------------------+
int ChartWidthInPixels(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_WIDTH_IN_PIXELS,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

- **CHART_HEIGHT_IN_PIXELS** - Eigenschaft der Höhe des Charts in Pixeln.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Wert der Höhe des Charts in Pixeln       |
//+------------------------------------------------------------------+
int ChartHeightInPixelsGet(const long chart_ID=0,const int sub_window=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long result=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_HEIGHT_IN_PIXELS,sub_window,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((int)result);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Die Funktion setzt den Wert der Höhe des Charts in Pixeln        |
//+------------------------------------------------------------------+
bool ChartHeightInPixelsSet(const int value,const long chart_ID=0,const int sub_windo
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_HEIGHT_IN_PIXELS,sub_window,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_BACKGROUND** - Farbe von Hintergrund des Charts.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Hintergrundfarbe des Charts              |
//+------------------------------------------------------------------+
color ChartBackColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir die Hintergrundfarbe des Charts
   if(!ChartGetInteger(chart_ID,CHART_COLOR_BACKGROUND,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Hintergrundfarbe des Charts               |
//+------------------------------------------------------------------+
bool ChartBackColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Setzen wir die Hintergrundfarbe des Charts
   if(!ChartSetInteger(chart_ID,CHART_COLOR_BACKGROUND,clr))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_FOREGROUND** - Farbe der Achse, Skala und OHLC Zeile.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe der Achse, Skala und OHLC Zeile    |
//+------------------------------------------------------------------+
color ChartForeColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir die Farbe der Achse, Skala und OHLC Zeile
   if(!ChartGetInteger(chart_ID,CHART_COLOR_FOREGROUND,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe der Achse, Skala und OHLC Zeile     |
//+------------------------------------------------------------------+
bool ChartForeColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Setzen wir die Farbe der Achse, Skala und OHLC Zeile
   if(!ChartSetInteger(chart_ID,CHART_COLOR_FOREGROUND,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

- **CHART_COLOR_GRID** - Farbe des Chart-Gitters.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe des Gitters des Charts             |
//+------------------------------------------------------------------+
color ChartGridColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten die Farbe des Chart-Gitters
   if(!ChartGetInteger(chart_ID,CHART_COLOR_GRID,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe des Gitters des Charts              |
//+------------------------------------------------------------------+
bool ChartGridColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Setzen wir die Farbe des Chart-Gitters
   if(!ChartSetInteger(chart_ID,CHART_COLOR_GRID,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_VOLUME** - Farbe der Volumen und Ebenen der Öffnung der Positionen.

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt die Farbe Volumen oder Ebenen der Öffnung   |
//| der Positionen.                                                  |
//+------------------------------------------------------------------+
color ChartVolumeColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir die Farbe der Volumen und Ebenen der Öffnung der Positionen
   if(!ChartGetInteger(chart_ID,CHART_COLOR_VOLUME,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe Volumen oder Ebenen der Öffnung     |
//| der Positionen.                                                  |
//+------------------------------------------------------------------+
bool ChartVolumeColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Setzen wir die Farbe der Volumen und Ebenen der Öffnung der Positionen
   if(!ChartSetInteger(chart_ID,CHART_COLOR_VOLUME,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_CHART_UP** - Farbe des Bars nach oben, Schatten und Kanten des Körpers von Bull-Kerze.

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt die Farbe des Bars nach oben, die Farbe     |
//| der Schatten und Kanten des Körpers von Bull-Kerze               |
//+------------------------------------------------------------------+
color ChartUpColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe des Bars nach oben, Schatten und Kanten des Körpers von Bull-
   if(!ChartGetInteger(chart_ID,CHART_COLOR_CHART_UP,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe des Bars nach oben, die Farbe       |
//| der Schatten und Kanten des Körpers von Bull-Kerze               |
//+------------------------------------------------------------------+
bool ChartUpColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe des Bars nach oben, Schatten und Kanten des Körpers von Bull-Ke
   if(!ChartSetInteger(chart_ID,CHART_COLOR_CHART_UP,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_CHART_DOWN** - Farbe des Bars nach unten, Schatten und Kanten des Körpers von Bär-Kerze.

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt die Farbe des Bars nach unten, die Farbe    |
//| der Schatten und Kanten des Körpers von Bär-Kerze                |
//+------------------------------------------------------------------+
color ChartDownColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe des Bars nach unten, Schatten und Kanten des Körpers von Bär-
   if(!ChartGetInteger(chart_ID,CHART_COLOR_CHART_DOWN,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe des Bars nach unten, die Farbe      |
//| der Schatten und Kanten des Körpers von Bär-Kerze                |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ChartDownColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe des Bars nach unten, Schatten und Kanten des Körpers von Bär-Ke
   if(!ChartSetInteger(chart_ID,CHART_COLOR_CHART_DOWN,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_CHART_LINE** - Farbe der Linie des Charts und Kerzen "Doji".

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe der Linie des Charts und Kerzen "Doji"
//+------------------------------------------------------------------+
color ChartLineColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir die Farbe der Linie des Charts und Kerzen "Doji"
   if(!ChartGetInteger(chart_ID,CHART_COLOR_CHART_LINE,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe der Linie des Charts und Kerzen     |
//| "Doji".                                                          |
//+------------------------------------------------------------------+
bool ChartLineColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Setzen wir die Farbe der Linie des Charts und Kerzen "Doji"
   if(!ChartSetInteger(chart_ID,CHART_COLOR_CHART_LINE,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_CANDLE_BULL** - Farbe des Körpers von Bull-Kerze.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe des Körpers von Bull-Kerze         |
//+------------------------------------------------------------------+
color ChartBullColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe des Körpers von Bull-Kerze
   if(!ChartGetInteger(chart_ID,CHART_COLOR_CANDLE_BULL,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe des Körpers von Bull-Kerze          |
//+------------------------------------------------------------------+
bool ChartBullColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe des Körpers von Bull-Kerze
   if(!ChartSetInteger(chart_ID,CHART_COLOR_CANDLE_BULL,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_CANDLE_BEAR** - Farbe des Körpers von Bär-Kerze.

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe des Körpers von Bär-Kerze          |
//+------------------------------------------------------------------+
color ChartBearColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe des Körpers von Bär-Kerze
   if(!ChartGetInteger(chart_ID,CHART_COLOR_CANDLE_BEAR,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe des Körpers von Bär-Kerze           |
//+------------------------------------------------------------------+
bool ChartBearColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe des Körpers von Bär-Kerze
   if(!ChartSetInteger(chart_ID,CHART_COLOR_CANDLE_BEAR,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_BID** - Farbe der Linie des Bid-Preises.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe der Bid-Linie                      |
//+------------------------------------------------------------------+
color ChartBidColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe der Bid-Linie
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ChartGetInteger(chart_ID,CHART_COLOR_BID,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
    ;}
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
 ;}
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe der Bid-Linie                       |
//+------------------------------------------------------------------+
bool ChartBidColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe der Bid-Linie
   if(!ChartSetInteger(chart_ID,CHART_COLOR_BID,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_ASK** - Farbe der Linie des Ask-Preises.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe der Ask-Linie                      |
//+------------------------------------------------------------------+
color ChartAskColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe der Ask-Linie
   if(!ChartGetInteger(chart_ID,CHART_COLOR_ASK,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe der Ask-Linie                       |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ChartAskColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe der Ask-Linie
   if(!ChartSetInteger(chart_ID,CHART_COLOR_ASK,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
    ;}
//--- erfolgreiche Ausführung
   return(true);
 }
```

- **CHART_COLOR_LAST** - Farbe der Linie von letzten Transaktion (Last).

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Farbe der Linie von letzten Transaktion  |
//+------------------------------------------------------------------+
color ChartLastColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe der Linie von letzten Transaktion (Last)
   if(!ChartGetInteger(chart_ID,CHART_COLOR_LAST,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe der Linie von letzten               |
//| Transaktion.                                                     |
//+------------------------------------------------------------------+
bool ChartLastColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe der Linie von letzten Transaktion (Last)
   if(!ChartSetInteger(chart_ID,CHART_COLOR_LAST,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_COLOR_STOP_LEVEL** - Farbe von Stop-Orders (Stop Loss und Take Profit).

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt die Farbe von Stop Loss und Take Profit     |
//+------------------------------------------------------------------+
color ChartStopLevelColorGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um die Farbe zu erhalten
   long result=clrNONE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten die Farbe von Stop Loss und Take Profit.
   if(!ChartGetInteger(chart_ID,CHART_COLOR_STOP_LEVEL,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return((color)result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Farbe von Stop Loss und Take Profit       |
//+------------------------------------------------------------------+
bool ChartStopLevelColorSet(const color clr,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- setzen die Farbe von Stop Loss und Take Profit.
   if(!ChartSetInteger(chart_ID,CHART_COLOR_STOP_LEVEL,clr))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

- **CHART_SHOW_TRADE_LEVELS** - Eigenschaft der Anzeige von Handelsstufen auf dem Chart (Stufen der offenen Positionen, Stop Loss, Take Profit und Pending Orders).

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob die Handelsstufen auf dem Chart angezeigt werden
//+------------------------------------------------------------------+
bool ChartShowTradeLevelsGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_TRADE_LEVELS,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert die Anzeige der Handelsstufen |
//+------------------------------------------------------------------+
bool ChartShowTradeLevelsSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_TRADE_LEVELS,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_DRAG_TRADE_LEVELS** - Eigenschaft von Ziehen der Handelsstufen auf dem Chart mit dem Maus.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob Ziehen von Handelsstufen auf dem Chart|
//| mit dem Maus erlaubt ist.                                        |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ChartDragTradeLevelsGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_DRAG_TRADE_LEVELS,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Ziehen der Handelsstufen      |
//| auf dem Chart mit dem Maus.                                      |
//+------------------------------------------------------------------+
bool ChartDragTradeLevelsSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_DRAG_TRADE_LEVELS,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_DATE_SCALE** - Eigenschaft der Anzeige der Zeitskala auf dem Chart.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob die Zeitskala auf dem Chart angezeigt wird
//+------------------------------------------------------------------+
bool ChartShowDateScaleGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_DATE_SCALE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige der Zeitskala auf dem |
//| Chart.                                                           |
//+------------------------------------------------------------------+
bool ChartShowDateScaleSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_DATE_SCALE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_PRICE_SCALE** - Eigenschaft der Anzeige der Preiseskala auf dem Chart.

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob die Preiseskala auf dem Chart angezeigt wird
//+------------------------------------------------------------------+
bool ChartShowPriceScaleGet(bool &result,const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
   long value;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetInteger(chart_ID,CHART_SHOW_PRICE_SCALE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
     }
//--- speichern den Wert der Chat-Eigenschaft in der Variable
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert Anzeige der Preiseskala       |
//| auf dem  Chart                                                   |
//+------------------------------------------------------------------+
bool ChartShowPriceScaleSet(const bool value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetInteger(chart_ID,CHART_SHOW_PRICE_SCALE,0,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHOW_ONE_CLICK** - Anzeige des Panels für schnellen Handel auf dem Chart (Option "Ein-Klick-Handel").

```
//+------------------------------------------------------------------+
//| Die Funktion bestimmt, ob das Panel für schnellen Handel         |
//| ("Ein-Klick-Handel") auf dem Chart angezeigt wird                |
//+------------------------------------------------------------------+
bool ChartShowOneClickPanelGet(bool &result,const long chart_ID=0)
  {
//--- Variable für die Erhaltung des Eigenschaftswertes vorbereiten
   long value;
//--- den Wert des Fehlers zurücksetzen
   ResetLastError();
//--- Eigenschaftswert erhalten
   if(!ChartGetInteger(chart_ID,CHART_SHOW_ONE_CLICK,0,value))
     {
      //--- Fehlermeldung im Experten-Journal anzeigen
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- den Wert der Charteigenschaft in der Variablen speichern
   result=value;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion aktiviert/deaktiviert den Anzeigemodus              |
//| des Handelspanels auf dem Chart                                  |
//+------------------------------------------------------------------+
bool ChartShowOneClickPanelSet(const bool value,const long chart_ID=0)
  {
//--- den Wert des Fehlers zurücksetzen
   ResetLastError();
//--- Eigenschaftswert setzen
   if(!ChartSetInteger(chart_ID,CHART_SHOW_ONE_CLICK,0,value))
     {
      //--- Fehlermeldung im Experten-Journal  anzeigen
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_SHIFT_SIZE** - Verschiebung des Null-Baren vom rechten Rand in Prozent.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Größe der Verschiebung des Null-Baren    |
//| vom rechten Rand des Charts in Prozent (von 10% bis 50%).        |
//+------------------------------------------------------------------+
double ChartShiftSizeGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_SHIFT_SIZE,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Größe der Verschiebung des Null-Baren     |
//| vom rechten Rand des Charts in Prozent (von 10% bis 50%). Um     |
//| den Modus der Verschiebung zu aktivieren, setzen Sie den Wert    |
//| von CHART_SHIFT auf true.                                        |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ChartShiftSizeSet(const double value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetDouble(chart_ID,CHART_SHIFT_SIZE,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_FIXED_POSITION** - Position der festen Position des Chart vom linken Rand als Prozentsatz.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält die Position der festen Position des Chart   |
//| vom linken Rand als Prozentsatz.                                 |
//+------------------------------------------------------------------+
double ChartFixedPositionGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_FIXED_POSITION,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt die Position der festen Position des Chart    |
//| vom linken Rand als Prozentsatz. Um die Position der feste       |
//| Position auf dem Chart zu sehen, muss man zunähst den Wert der   |
//| Eigenschaft CHART_AUTOSCROLL auf false setzen.                   |
//+------------------------------------------------------------------+
bool ChartFixedPositionSet(const double value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ChartSetDouble(chart_ID,CHART_FIXED_POSITION,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_FIXED_MAX** - die Eigenschaft des festen Maximalwertes des Charts.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den festen Maximalwert des Charts.           |
//+------------------------------------------------------------------+
double ChartFixedMaxGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_FIXED_MAX,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt den festen Maximalwert des Charts.            |
//| Um den Wert dieser Eigenschaft zu ändern, setzen Sie             |
//| zunächst den Wert der Eigenschaft CHART_SCALEFIX auf             |
//| true.                                                            |
//+------------------------------------------------------------------+
bool ChartFixedMaxSet(const double value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetDouble(chart_ID,CHART_FIXED_MAX,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
  }
```

- **CHART_FIXED_MIN** - Eigenschaft des festen Minimalwertes des Charts.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den festen Minimalwert des Charts            |
//+------------------------------------------------------------------+
double ChartFixedMinGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_FIXED_MIN,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt den festen Minimalwert des Charts.            |
//| Um den Wert dieser Eigenschaft zu ändern, setzen Sie             |
//| zunächst den Wert der Eigenschaft CHART_SCALEFIX auf true.       |
//+------------------------------------------------------------------+
bool ChartFixedMinSet(const double value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetDouble(chart_ID,CHART_FIXED_MIN,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_POINTS_PER_BAR** - Skalenwert als Punkten pro Bar.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Skalenwert als Punkten pro Bar           |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
double ChartPointsPerBarGet(const long chart_ID=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_POINTS_PER_BAR,0,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
//+------------------------------------------------------------------+
//| Die Funktion setzt den Skalenwert als Punkten pro Bar.           |
//| Um das Ergebnis von Veränderungen des Wertes dieser              |
//| Eigenschaft zu sehen, setzen Sie zunähst den Wert von            |
//| CHART_SCALE_PT_PER_BAR auf true.                                 |
//+------------------------------------------------------------------+
bool ChartPointsPerBarSet(const double value,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetDouble(chart_ID,CHART_POINTS_PER_BAR,value))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_PRICE_MIN** - gibt den Minimalwert des Charts zurück.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Minimalwert des Charts im Hauptfenster   |
//| oder Untenfenster.                                               |
//+------------------------------------------------------------------+
double ChartPriceMin(const long chart_ID=0,const int sub_window=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_PRICE_MIN,sub_window,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
```

- **CHART_PRICE_MAX** - gibt den Maximalwert des Charts zurück.

```
//+------------------------------------------------------------------+
//| Die Funktion erhält den Maximalwert des Charts im Hauptfenster   |
//| oder Untenfenster.                                               |
//+------------------------------------------------------------------+
double ChartPriceMax(const long chart_ID=0,const int sub_window=0)
  {
//--- Bereiten wir eine Variable, um das Ergebnis zu erhalten
   double result=EMPTY_VALUE;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetDouble(chart_ID,CHART_PRICE_MAX,sub_window,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
     }
//--- Geben den Wert der Eigenschaft zurück
   return(result);
  }
```

- **CHART_COMMENT** - Text der Kommentar auf dem Chart.

```
//+------------------------------------------------------------------+
//| Die Funktion bekommt den Kommentartext in der linken oberen Ecke |
//| des Charts.                                                      |
//+------------------------------------------------------------------+
bool ChartCommentGet(string &result,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erhalten wir den Wert der Eigenschaft
   if(!ChartGetString(chart_ID,CHART_COMMENT,result))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion bekommt den Kommentartext in der linken oberen Ecke |
//| des Charts.                                                      |
//+------------------------------------------------------------------+
bool ChartCommentSet(const string str,const long chart_ID=0)
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Geben wir den Wert der Eigenschaft ein
   if(!ChartSetString(chart_ID,CHART_COMMENT,str))
     {
      //--- Schreiben die Fehlermeldung in den Log "Experten"
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- erfolgreiche Ausführung
   return(true);
  }
```

- **CHART_IS_MAXIMIZED** - Chartfenster maximiert

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob das Chartfenster vergrößert ist       |
//+------------------------------------------------------------------+
bool ChartWindowsIsMaximized(bool &result,const long chart_ID=0)
  {
//--- bereiten wir eine Variable vor, um den Wert der Eigenschaft zu erhalten
   long value;
//--- setzen wir den Fehlerwert zurück
   ResetLastError();
//--- Wert der Eigenschaft erhalten
   if(!ChartGetInteger(chart_ID,CHART_IS_MAXIMIZED))
     {
      //--- Fehlermeldung im Journal "Expert Advisors" ausgeben
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- den Wert der Eigenschaft in der Variablen speichern
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

- **CHART_IS_MINIMIZED** - Chartfenster minimiert

```
//+------------------------------------------------------------------+
//| Die Funktion ermittelt, ob das Chartfenster minimiert ist        |
//+------------------------------------------------------------------+
bool ChartWindowsIsMinimized(bool &result,const long chart_ID=0)
  {
//--- bereiten wir eine Variable vor, um den Wert der Eigenschaft zu erhalten
   long value;
//--- setzen wir den Fehlerwert zurück
   ResetLastError();
//--- den Wert der Eigenschaft erhalten
   if(!ChartGetInteger(chart_ID,CHART_IS_MINIMIZED))
     {
      //--- Fehlermeldung im Journal "Expert Advisor" anzeigen
      Print(__FUNCTION__+", Error Code = ",GetLastError());
      return(false);
     }
//--- den Wert der Eigenschaft in dewr Variablen speichern
   result=value;
//--- erfolgreiche Ausführung
   return(true);
  }
```

Panel für die Charteigenschaften

```
//--- Schließen Sie das Control-Bibliothek
#include <ChartObjects\ChartObjectsTxtControls.mqh>
//--- Vordefinierte Konstanten
#define X_PROPERTY_NAME_1    10  // X-Koordinate der Eigenschaft Name in der ersten Sp
#define X_PROPERTY_VALUE_1   225 // X-Koordinaten des Eigenschaftwertes in der ersten
#define X_PROPERTY_NAME_2    345 // X-Koordinate der Eigenschaft Name in der zweiten u
#define X_PROPERTY_VALUE_2   550 // X-Koordinaten des Eigenschaftwertes in der zweiter
#define X_BUTTON_1           285 // X-Koordinate der Taste in der ersten Spalte
#define X_BUTTON_2           700 // X-Koordinate der Taste in der zweiten Spalte
#define Y_PROPERTY_1         30  // Y-Koordinate des Anfangspunktes von der ersten und
#define Y_PROPERTY_2         286 // Y-Koordinate des Anfangspunktes von der dritten Sp
#define Y_DISTANCE           16  // Der y-axiale Abstand zwischen den Zeilen
#define LAST_PROPERTY_NUMBER 111 // Nummer der letzten grafischen Eigenschaften
//--- Input-Parameter
input color InpFirstColor=clrDodgerBlue; // Farbe der ungeraden Zeilen
input color InpSecondColor=clrGoldenrod; // Farbe der geraden Zeilen
//--- Variablen und Arrays
CChartObjectLabel  ExtLabelsName[];  // Label, um die Namen der Eigenschaften anzuzeig
CChartObjectLabel  ExtLabelsValue[]; // Label, um die Namen der Werte anzuzeigen
CChartObjectButton ExtButtons[];     // Tasten
int                ExtNumbers[];     // Indizes der Eigenschaften
string             ExtNames[];       // Namen der Eigenschaften
uchar              ExtDataTypes[];   // Datentypen von Eigenschaften (integer, double,
uint               ExtGroupTypes[];  // Array, das Informationen über die Zubehör der
uchar              ExtDrawTypes[];   // Array, das Informationen über die Methode der
```

© 2000-2025, MetaQuotes Ltd.

```
double             ExtMaxValue[];    // Maximale Werte der Eigenschaften, die sie in d
double             ExtMinValue[];    // Minimale Werte der Eigenschaften, die sie in d
double             ExtStep[];        // Schritte zum Ändern der Eigenschaften
int                ExtCount;         // Gesamtzahl der Eigenschaften
color              ExtColors[2];     // Array von Farben, um Zeilen anzuzeigen
string             ExtComments[2];   // Array von Kommentaren (für die Eigenschaft CHA
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Kommentar auf dem Chart anzeigen
   Comment("SomeComment");
//--- Farben in das Array speichern, um später zwischen ihnen umzuschalten
   ExtColors[0]=InpFirstColor;
   ExtColors[1]=InpSecondColor;
//--- Kommentare in das Array speichern, um später zwischen ihnen umzuschalten
   ExtComments[0]="FirstComment";
   ExtComments[1]="SecondComment";
//--- Bedienfeld vorbereiten und anzeigen
   if(!PrepareControls())
      return(INIT_FAILED);
//--- erfolgreiche Ausführung
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialization function of the expert                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Den Text des Kommentars aus dem Chart löschen
   Comment("");
  }
//+------------------------------------------------------------------+
//| Chart Event Handler                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
//--- Überprüfen das Ereignis von Klick auf das Objekt es Charts
   if(id==CHARTEVENT_OBJECT_CLICK)
     {
      //--- Aufteilen den Namen des Objekts auf Abscheider
      string obj_name[];
      StringSplit(sparam,'_',obj_name);
      //--- Überprüfen ob das Objekt eine Taste ist
      if(obj_name[0]=="Button")
```

© 2000-2025, MetaQuotes Ltd.

```
         {
          //--- Index der Taste erhalten
          int index=(int)StringToInteger(obj_name[1]);
          //--- Taste in nicht gedrückte Zustand setzen
          ExtButtons[index].State(false);
          //--- Den neuen Wert der Eigenschaft je nach ihr Typ setzen
          if(ExtDataTypes[index]=='I')
             ChangeIntegerProperty(index);
          if(ExtDataTypes[index]=='D')
             ChangeDoubleProperty(index);
          if(ExtDataTypes[index]=='S')
             ChangeStringProperty(index);
         }
      }
//--- Eigenschaftswerten neu zeichnen
   RedrawProperties();
   ChartRedraw();
 ;}
//+------------------------------------------------------------------+
//| Ändern die integrale Eigenschaft des Charts                      |
//+------------------------------------------------------------------+
void ChangeIntegerProperty(const int index)
  {
//--- Erhalten wir den aktuellen Wert der Eigenschaft
   long value=ChartGetInteger(0,(ENUM_CHART_PROPERTY_INTEGER)ExtNumbers[index]);
//--- Definieren den folgenden Wert der Eigenschaft
   switch(ExtDrawTypes[index])
     {
      case 'C':
         value=GetNextColor((color)value);
         break;
      default:
         value=(long)GetNextValue((double)value,index);
         break;
     }
//--- Geben wir den aktuellen Wert der Eigenschaft ein
   ChartSetInteger(0,(ENUM_CHART_PROPERTY_INTEGER)ExtNumbers[index],0,value);
  }
//+------------------------------------------------------------------+
//| Ändern die echte Eigenschaft des Charts                          |
//+------------------------------------------------------------------+
void ChangeDoubleProperty(const int index)
  {
//--- Erhalten wir den aktuellen Wert der Eigenschaft
   double value=ChartGetDouble(0,(ENUM_CHART_PROPERTY_DOUBLE)ExtNumbers[index]);
//--- Definieren den folgenden Wert der Eigenschaft
   value=GetNextValue(value,index);
//--- Geben wir den aktuellen Wert der Eigenschaft ein
   ChartSetDouble(0,(ENUM_CHART_PROPERTY_DOUBLE)ExtNumbers[index],value);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Ändern die String-Eigenschaft des Charts                         |
//+------------------------------------------------------------------+
void ChangeStringProperty(const int index)
  {
//--- Statische Variable um Kommentare ExtComments innerhalb des Arrays zu wechseln
   static uint comment_index=1;
//--- Ändern Index für einen weiteren Kommentar
   comment_index=1-comment_index;
//--- Geben wir den aktuellen Wert der Eigenschaft ein
   ChartSetString(0,(ENUM_CHART_PROPERTY_STRING)ExtNumbers[index],ExtComments[comment_
  }
//+------------------------------------------------------------------+
//| Identifizierung des nächsten Wertes der Eigenschaft              |
//+------------------------------------------------------------------+
double GetNextValue(const double value,const int index)
  {
   if(value+ExtStep[index]<=ExtMaxValue[index])
      return(value+ExtStep[index]);
   else
      return(ExtMinValue[index]);
  }
//+------------------------------------------------------------------+
//| Erhalten die nächste Farbe für die Eigenschaft von Typ color     |
//+------------------------------------------------------------------+
color GetNextColor(const color clr)
  {
//--- Den nächsten Farbwert zurückgeben
   switch(clr)
     {
      case clrWhite: return(clrRed);
      case clrRed:   return(clrGreen);
      case clrGreen: return(clrBlue);
      case clrBlue:  return(clrBlack);
      default:       return(clrWhite);
    ;}
  }
//+------------------------------------------------------------------+
//| Eigenschaftswerten neu zeichnen                                  |
//+------------------------------------------------------------------+
void RedrawProperties(void)
  {
//--- Text des Eigenschaftswertes
   string text;
   long   value;
//--- Zyklus über die Anzahl der Eigenschaften
   for(int i=0;i<ExtCount;i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      text="";
      switch(ExtDataTypes[i])
        {
         case 'I':
            //--- Erhalten wir den aktuellen Wert der Eigenschaft
            if(!ChartGetInteger(0,(ENUM_CHART_PROPERTY_INTEGER)ExtNumbers[i],0,value))
            break;
            //--- Text der Integer-Eigenschaft
            switch(ExtDrawTypes[i])
              {
               //--- Farbeigenschaft
               case 'C':
                  text=(string)((color)value);
                  break;
                  //--- Boolesche Eigenschaft
               case 'B':
                  text=(string)((bool)value);
                  break;
                  //--- Eigenschaft von ENUM_CHART_MODE
               case 'M':
                  text=EnumToString((ENUM_CHART_MODE)value);
                  break;
                  //--- Eigenschaft von ENUM_CHART_VOLUME_MODE
               case 'V':
                  text=EnumToString((ENUM_CHART_VOLUME_MODE)value);
                  break;
                  //--- Nummer er Typ int
               default:
                  text=IntegerToString(value);
                  break;
              }
            break;
         case 'D':
            //--- Text der echten Eigenschaft
            text=DoubleToString(ChartGetDouble(0,(ENUM_CHART_PROPERTY_DOUBLE)ExtNumber
            break;
         case 'S':
            //--- Text der String-Eigenschaft
            text=ChartGetString(0,(ENUM_CHART_PROPERTY_STRING)ExtNumbers[i]);
            break;
        }
      //--- Anzeige der Wert der Eigenschaft
      ExtLabelsValue[i].Description(text);
     }
  }
//+------------------------------------------------------------------+
//| Erstellung einer Platte,um die Eigenschaften des Charts          |
//| zu kontrollieren                                                 |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool PrepareControls(void)
  {
//--- Reservieren Speicher für Arrays mit Reserve
   MemoryAllocation(LAST_PROPERTY_NUMBER+1);
//--- Variablen
   int i=0;     // Zyklus-Variable
   int col_1=0; // Anzahl der Eigenschaften in der erste Spalte
   int col_2=0; // Anzahl der Eigenschaften in der zweite Spalte
   int col_3=0; // Anzahl der Eigenschaften in der dritte Spalte
//--- Die aktuelle Anzahl von Eigenschaften ist 0
   ExtCount=0;
//--- Suche nach Eigenschaften in Zyklus
   while(i<=LAST_PROPERTY_NUMBER)
     {
      //--- Den aktuellen Nummer der Eigenschaft speichern
      ExtNumbers[ExtCount]=i;
      //--- Den Wert der Zyklus-Eigenschaft erhöhen
      i++;
      //--- Prüfen, ob es eine Eigenschaft mit diesem Nummer gibt
      if(CheckNumber(ExtNumbers[ExtCount],ExtNames[ExtCount],ExtDataTypes[ExtCount],Ex
        {
         //--- Steuerelemente für Eigenschaften erstellen
         switch(ExtGroupTypes[ExtCount])
           {
            case 1:
               //--- ein Label und eine Schaltfläche für die Eigenschaft erstellen
               if(!ShowProperty(ExtCount,0,X_PROPERTY_NAME_1,X_PROPERTY_VALUE_1,X_BUTT
               return(false);
               //--- Die Anzahl der Elemente in der ersten Spalte hat erhöht
               col_1++;
               break;
            case 2:
               //--- ein Label und eine Schaltfläche für die Eigenschaft erstellen
               if(!ShowProperty(ExtCount,1,X_PROPERTY_NAME_2,X_PROPERTY_VALUE_2,X_BUTT
               return(false);
               //--- Die Anzahl der Elemente in der zweite Spalte hat erhöht
               col_2++;
               break;
            case 3:
               //--- Nur Label für Eigenschaften erstellen
               if(!ShowProperty(ExtCount,2,X_PROPERTY_NAME_2,X_PROPERTY_VALUE_2,0,Y_PR
               return(false);
               //--- Die Anzahl der Elemente in der dritten Spalte hat erhöht
               col_3++;
               break;
           }
         //--- Den maximalen und minimalen Wert und den Schritt definieren
         GetMaxMinStep(ExtNumbers[ExtCount],ExtMaxValue[ExtCount],ExtMinValue[ExtCount
         //--- Erhöhung der Anzahl der Eigenschaften
```

© 2000-2025, MetaQuotes Ltd.

```
         ExtCount++;
        }
      }
//--- Freien wir den Speicher, der durch Arrays nicht verwendet wird
   MemoryAllocation(ExtCount);
//--- Eigenschaftswerten neu zeichnen
   RedrawProperties();
   ChartRedraw();
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Speicher für Arrays reservieren                                  |
//+------------------------------------------------------------------+
void MemoryAllocation(const int size)
  {
   ArrayResize(ExtLabelsName,size);
   ArrayResize(ExtLabelsValue,size);
   ArrayResize(ExtButtons,size);
   ArrayResize(ExtNumbers,size);
   ArrayResize(ExtNames,size);
   ArrayResize(ExtDataTypes,size);
   ArrayResize(ExtGroupTypes,size);
   ArrayResize(ExtDrawTypes,size);
   ArrayResize(ExtMaxValue,size);
   ArrayResize(ExtMinValue,size);
   ArrayResize(ExtStep,size);
  }
//+------------------------------------------------------------------+
//| Prüfen ob der Index der Eigenschaft zu einem der Enumerationen   |
//| ENUM_CHART_PROPERTIES gehört                                     |
//+------------------------------------------------------------------+
bool CheckNumber(const int ind,string &name,uchar &data_type,uint &group_type,uchar &d
  {
//--- Prüfen, ob die Eigenschaft eine ganze Zahl ist
   ResetLastError();
   name=EnumToString((ENUM_CHART_PROPERTY_INTEGER)ind);
   if(_LastError==0)
     {
      data_type='I';                       // Eigenschaft aus ENUM_CHART_PROPERTY_INTEG
      GetTypes(ind,group_type,draw_type); // Parameter von Anzeige der Eigenschaften d
      return(true);
     }
//--- Prüfen, ob die Eigenschaft echt ist
   ResetLastError();
   name=EnumToString((ENUM_CHART_PROPERTY_DOUBLE)ind);
   if(_LastError==0)
     {
      data_type='D';                       // Eigenschaft aus Enumeration ENUM_CHART_PR
```

© 2000-2025, MetaQuotes Ltd.

```
      GetTypes(ind,group_type,draw_type); // Parameter von Anzeige der Eigenschaften d
      return(true);
     }
//--- Prüfen, ob die Eigenschaft String ist
   ResetLastError();
   name=EnumToString((ENUM_CHART_PROPERTY_STRING)ind);
   if(_LastError==0)
     {
      data_type='S';                      // Eigenschaft aus Enumeration ENUM_CHART_PR
      GetTypes(ind,group_type,draw_type); // Parameter von Anzeige der Eigenschaften d
      return(true);
     }
//--- die Eigenschaft gehört zu keiner Enumeration
   return(false);
  }
//+------------------------------------------------------------------+
//| Definieren, in welcher Gruppe die Eigenschaft sein sollte,       |
//| und ihr Display-Typ                                              |
//+------------------------------------------------------------------+
void GetTypes(const int property_number,uint &group_type,uchar &draw_type)
  {
//--- Prüfen, ob die Eigenschaft zur dritten Grippe gehört
//--- Eigenschaften der dritten Gruppe werden in der zweite Spalte von CHART_BRING_TO_
   if(CheckThirdGroup(property_number,group_type,draw_type))
      return;
//--- Prüfen, ob die Eigenschaft zur zweiten Grippe gehört
//--- Eigenschaften der zweiten Gruppe werden in der zweite Spalte von Anfang angezeig
   if(CheckSecondGroup(property_number,group_type,draw_type))
      return;
//--- Wenn Sie hier sind, bedeutet es, das die Eigenschaft zur ersten Gruppe (erste Sp
   CheckFirstGroup(property_number,group_type,draw_type);
  }
//+------------------------------------------------------------------+
//| Die Funktion überprüft, ob die Eigenschaft zur dritten Gruppe    |
//| gehört, und wenn ja, bestimmt den Display-Typ                    |
//+------------------------------------------------------------------+
bool CheckThirdGroup(const int property_number,uint &group_type,uchar &draw_type)
  {
//--- Prüfen, ob die Eigenschaft zur dritten Grippe gehört
   switch(property_number)
     {
      //--- Boolesche Eigenschaften
      case CHART_IS_OBJECT:
      case CHART_WINDOW_IS_VISIBLE:
         draw_type='B';
         break;
         //--- Integer Eigenschaften
      case CHART_VISIBLE_BARS:
      case CHART_WINDOWS_TOTAL:
```

© 2000-2025, MetaQuotes Ltd.

```
      case CHART_WINDOW_HANDLE:
      case CHART_WINDOW_YDISTANCE:
      case CHART_FIRST_VISIBLE_BAR:
      case CHART_WIDTH_IN_BARS:
      case CHART_WIDTH_IN_PIXELS:
         draw_type='I';
         break;
         //--- echte Eigenschaften
      case CHART_PRICE_MIN:
      case CHART_PRICE_MAX:
         draw_type='D';
         break;
         //--- In der Tat ist diese Eigenschaft eine Instruktion den Chat an der Spitz
         //--- Für dieses Panel ist ihre Anwendung nicht notwendig, da das Fenster ist
         //--- über alle anderen Fenster
      case CHART_BRING_TO_TOP:
         draw_type=' ';
         break;
         //--- Die Eigenschaft gehört nicht zur dritten Gruppe
      default:
         return(false);
     }
//--- Die Eigenschaft gehört zur dritten Gruppe
   group_type=3;
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion überprüft, ob die Eigenschaft zur zweiten Gruppe    |
//| gehört, und wenn ja, bestimmt den Display-Typ                    |
//+------------------------------------------------------------------+
bool CheckSecondGroup(const int property_number,uint &group_type,uchar &draw_type)
  {
//--- Prüfen, ob die Eigenschaft zur zweiten Grippe gehört
   switch(property_number)
     {
      //--- Eigenschaft von Typ ENUM_CHART_MODE
      case CHART_MODE:
         draw_type='M';
         break;
         //--- Eigenschaft von Typ ENUM_CHART_VOLUME_MODE
      case CHART_SHOW_VOLUMES:
         draw_type='V';
         break;
         //--- String-Eigenschaft
      case CHART_COMMENT:
         draw_type='S';
         break;
         //--- Eigenschaft der Farbe
      case CHART_COLOR_BACKGROUND:
```

© 2000-2025, MetaQuotes Ltd.

```
      case CHART_COLOR_FOREGROUND:
      case CHART_COLOR_GRID:
      case CHART_COLOR_VOLUME:
      case CHART_COLOR_CHART_UP:
      case CHART_COLOR_CHART_DOWN:
      case CHART_COLOR_CHART_LINE:
      case CHART_COLOR_CANDLE_BULL:
      case CHART_COLOR_CANDLE_BEAR:
      case CHART_COLOR_BID:
      case CHART_COLOR_ASK:
      case CHART_COLOR_LAST:
      case CHART_COLOR_STOP_LEVEL:
         draw_type='C';
         break;
         //--- Die Eigenschaft gehört nicht zur zweiten Gruppe
      default:
         return(false);
     }
//--- Die Eigenschaft gehört zur zweiten Gruppe
   group_type=2;
   return(true);
  }
//+------------------------------------------------------------------+
//| Diese Funktion wird nur aufgerufen, wenn es bereits bekannt ist, |
//| dass die Eigenschaft nicht zur zweiten und dritten Gruppe        |
//| von Eigenschaften gehört                                         |
//+------------------------------------------------------------------+
void CheckFirstGroup(const int property_number,uint &group_type,uchar &draw_type)
  {
//--- Die Eigenschaft gehört zur ersten Gruppe
   group_type=1;
//--- Typ von Anzeige der Eigenschaft definieren
   switch(property_number)
     {
      //--- Integer Eigenschaften
      case CHART_SCALE:
      case CHART_HEIGHT_IN_PIXELS:
         draw_type='I';
         return;
         //--- echte Eigenschaften
      case CHART_SHIFT_SIZE:
      case CHART_FIXED_POSITION:
      case CHART_FIXED_MAX:
      case CHART_FIXED_MIN:
      case CHART_POINTS_PER_BAR:
         draw_type='D';
         return;
         //--- Nur Boolesche Eigenschaften
      default:
```

© 2000-2025, MetaQuotes Ltd.

```
         draw_type='B';
         return;
      }
   }
//+------------------------------------------------------------------+
//| Ein Label und eine Schaltfläche für die Eigenschaft erstellen    |
//+------------------------------------------------------------------+
bool ShowProperty(const int ind,const int type,const int x1,const int x2,
                  const int xb,const int y,const bool btn)
   {
//--- Statischer Array um innerhalb des Farbe-Arrays ExtColors zu wechseln
   static uint color_index[3]={1,1,1};
//--- Ändern Index für eine andere Farbe
   color_index[type]=1-color_index[type];
//--- Label und Taste (wenn btn=true) für die Eigenschaft anzeigen
   if(!LabelCreate(ExtLabelsName[ind],"name_"+(string)ind,ExtNames[ind],ExtColors[colo
      return(false);
   if(!LabelCreate(ExtLabelsValue[ind],"value_"+(string)ind,"",ExtColors[color_index[t
      return(false);
   if(btn && !ButtonCreate(ExtButtons[ind],(string)ind,xb,y+1))
      return(false);
//--- erfolgreiche Ausführung
   return(true);
   }
//+------------------------------------------------------------------+
//| Label erstellen                                                  |
//+------------------------------------------------------------------+
bool LabelCreate(CChartObjectLabel &lbl,const string name,const string text,
                 const color clr,const int x,const int y)
   {
   if(!lbl.Create(0,"Label_"+name,0,x,y)) return(false);
   if(!lbl.Description(text))             return(false);
   if(!lbl.FontSize(10))                  return(false);
   if(!lbl.Color(clr))                    return(false);
//--- erfolgreiche Ausführung
   return(true);
   }
//+------------------------------------------------------------------+
//| Taste erstellen                                                  |
//+------------------------------------------------------------------+
bool ButtonCreate(CChartObjectButton &btn,const string name,
                  const int x,const int y)
   {
   if(!btn.Create(0,"Button_"+name,0,x,y,50,15)) return(false);
   if(!btn.Description("Next"))                  return(false);
   if(!btn.FontSize(10))                         return(false);
   if(!btn.Color(clrBlack))                      return(false);
   if(!btn.BackColor(clrWhite))                  return(false);
   if(!btn.BorderColor(clrBlack))                return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Den maximalen und minimalen Wert und den Schritt eingeben        |
//+------------------------------------------------------------------+
void GetMaxMinStep(const int property_number,double &max,double &min,double &step)
  {
   double value;
//--- Den neuen Wert der Eigenschaft je nach ihr Typ setzen
   switch(property_number)
     {
      case CHART_SCALE:
         max=5;
         min=0;
         step=1;
         break;
      case CHART_MODE:
      case CHART_SHOW_VOLUMES:
         max=2;
         min=0;
         step=1;
         break;
      case CHART_SHIFT_SIZE:
         max=50;
         min=10;
         step=2.5;
         break;
      case CHART_FIXED_POSITION:
         max=90;
         min=0;
         step=15;
         break;
      case CHART_POINTS_PER_BAR:
         max=19;
         min=1;
         step=3;
         break;
      case CHART_FIXED_MAX:
         value=ChartGetDouble(0,CHART_FIXED_MAX);
         max=value*1.25;
         min=value;
         step=value/32;
         break;
      case CHART_FIXED_MIN:
         value=ChartGetDouble(0,CHART_FIXED_MIN);
         max=value;
         min=value*0.75;
         step=value/32;
```

© 2000-2025, MetaQuotes Ltd.

```
         break;
      case CHART_HEIGHT_IN_PIXELS:
         max=700;
         min=520;
         step=30;
         break;
         //--- Default values
      default:
         max=1;
         min=0;
         step=1;
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Objektkonstanten

44 graphische Objekte werden vorausgesehen, die erzeugt und auf dem Preischart dargestellt werden können. Alle Konstanten für Arbeit mit Objekten werden in 9 Gruppen aufgeteilt:

- Objekttypen - Identifikatoren der graphischen Objekte;
- Objekteigenschaften - Arbeit mit Eigenschaften der graphischen Objekte;
- Methoden von Objektsnap - Konstanten der Objektpositionierung auf dem Chart;
- Snapwinkel - ermöglicht Winkel des Charts einzustellen, demgegenüber Positionieren des Objekts in Pixel erfolgt;
- Sichtbarkeit der Objekte - Festlegung von Timeframes, auf denen das Objekt sichtbar wird;
- Größe von Elliott Wellen - Abstufung der Wellenmarkierung;
- Objekte von Gann - Trendkonstanten für Gann Fan und Gann Grid;
- Bestand von Web-Farben - Konstanten der vorbestimmten Web-Farben; ;
- Wingdings - Symbolenkoden der Schrift Wingdings.

© 2000-2025, MetaQuotes Ltd.

Objekttypen

Bei der Erzeugung des graphischen Objekts durch die Funktion ObjectCreate() muss man Typ des zu erzeugenen Objekts angeben, der einen der Werte der Aufzaelung ENUM_OBJECT annehmen kann. Weitere Praezisierungen der Eigenschaften des zu erzeugenden Objekts ist durch die Funktion für die Arbeit mit graphischen Objekten moeglich.

ENUM_OBJECT

Identifikator		Beschreibung
OBJ_VLINE		Senkrechte Linie
OBJ_HLINE		horizontale Linie
OBJ_TREND		Trendlinie
OBJ_TRENDBYANGLE		Trendlinie nach Winkel
OBJ_CYCLES		Zykluslinien
OBJ_ARROWED_LINE		Objekt "Linie mit einem Pfeil"
OBJ_CHANNEL		Abstandsgleicher Kanal
OBJ_STDDEVCHANNEL		Kanal der Standardabweichung
OBJ_REGRESSION		Kanal der linearen Regression
OBJ_PITCHFORK		Andrews' Pitchfork
OBJ_GANNLINE		Gannlinie
OBJ_GANNFAN		Gannfan
OBJ_GANNGRID		Ganngrid
OBJ_FIBO		Fibonacci Retracement
OBJ_FIBOTIMES		Zeitzonen Fibonacci
OBJ_FIBOFAN		Fibonacci Fan
OBJ_FIBOARC		Fibonacci Arcs
OBJ_FIBOCHANNEL		Kanal Fibonacci
OBJ_EXPANSION		Erweiterung Fibonacci
OBJ_ELLIOTWAVE5		Elliott Motive Wave
OBJ_ELLIOTWAVE3		Elliott Correction Wave
OBJ_RECTANGLE		Rechteck
OBJ_TRIANGLE		Dreieck
OBJ_ELLIPSE		Ellipse
OBJ_ARROW_THUMB_UP		Zeichen "Gut" (Daumen oben)

© 2000-2025, MetaQuotes Ltd.

Identifikator		Beschreibung
OBJ_ARROW_THUMB_DOWN		Zeichen "Schlecht" (Daumen unten)
OBJ_ARROW_UP		Zeichen "Weiche oben"
OBJ_ARROW_DOWN		Zeichen "Weich unten"
OBJ_ARROW_STOP		Zeichen "Stop"
OBJ_ARROW_CHECK		Zeichen "Kontrollmarke"
OBJ_ARROW_LEFT_PRICE		linke Preismarke
OBJ_ARROW_RIGHT_PRICE		Rechte Preismarke
OBJ_ARROW_BUY		Zeichen "Buy"
OBJ_ARROW_SELL		Zeichen "Sell"
OBJ_ARROW		Objekt "Weiche"
OBJ_TEXT		Objekt "Text"
OBJ_LABEL		Objekt "Textmarke"
OBJ_BUTTON		Objekt "Schaltfläche"
OBJ_CHART		Objekt "Chart"
OBJ_BITMAP		Objekt "Bild"
OBJ_BITMAP_LABEL		Objekt "graphische Marke"
OBJ_EDIT		Objekt "Eingabefeld"
OBJ_EVENT		Objekt "Ereignis", das einem Ereignis in der wirtschaftlichen Kalender entspricht
OBJ_RECTANGLE_LABEL		Objekt "Rechteckige Marke" um eigene grafische Benutzeroberfläche zu entwerten.

© 2000-2025, MetaQuotes Ltd.

OBJ_VLINE

Vertikale Linie.



Hinweis

Wenn Sie eine vertikale Linie erstellen, können Sie den Anzeigemodus für Linien in alle Chart-Fenster (Eigenschaft OBJPROP_RAY) angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart eine vertikale Linie. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Vertikale Linie\"."
#property description "Datum des Ankerpunkts wird als Prozentsatz der Breite"
#property description "des Chartfensters in Bars."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="VLine";      // Linienname
input int              InpDate=25;           // Datum der Linie %
input color            InpColor=clrRed;      // Linienfarbe
input ENUM_LINE_STYLE  InpStyle=STYLE_DASH;  // Linienstil
input int              InpWidth=3;           // Linienbreite
input bool             InpBack=false;        // Die Linie im Hintergrund
input bool             InpSelection=true;    // Wählen zu bewegen
input bool             InpRay=true;          // Fortsetzung der Linie nach unten
```

© 2000-2025, MetaQuotes Ltd.

```
input bool            InpHidden=true;      // Ausgeblendet in der Objektliste
input long            InpZOrder=0;         // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt eine vertikale Linie                                    |
//+------------------------------------------------------------------+
bool VLineCreate(const long            chart_ID=0,        // ID des Charts
                 const string          name="VLine",      // Linienname
                 const int             sub_window=0,      // Nummer des Unterfensters
                 datetime              time=0,            // Linienzeit
                 const color           clr=clrRed,        // Linienfarbe
                 const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil
                 const int             width=1,           // Linienbreite
                 const bool            back=false,        // Im Hintergrund
                 const bool            selection=true,    // Wählen um zu bewegen
                 const bool            ray=true,          // Fortsetzung der Linie nac
                 const bool            hidden=true,       // Ausgeblendet in der Objek
                 const long            z_order=0)         // Priorität auf Mausklick
  {
//--- wenn die Zeit der Linie nicht angegeben ist, wird sie auf den letzten Balken lie
   if(!time)
      time=TimeCurrent();
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine vertikale Linie erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_VLINE,sub_window,time,0))
     {
      Print(__FUNCTION__,
            ": Vertikale Linie konnte nicht erstellt werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- Farbe der Linie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Anzeige der Linie in Unterfenster de
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY,ray);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Vertikale Linie bewegen                                          |
//+------------------------------------------------------------------+
bool VLineMove(const long   chart_ID=0,   // ID des Charts
               const string name="VLine", // Linienname
               datetime     time=0)       // Linienzeit
  {
//--- wenn die Zeit der Linie nicht angegeben ist, bewegen wir sie auf den letzten Bal
   if(!time)
      time=TimeCurrent();
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine vertikale Linie bewegen
   if(!ObjectMove(chart_ID,name,0,time,0))
     {
      Print(__FUNCTION__,
            ": Vertikale Linie konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht eine vertikale Linie                                      |
//+------------------------------------------------------------------+
bool VLineDelete(const long   chart_ID=0,   // ID des Charts
                 const string name="VLine") // Linienname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine vertikale Linie löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Vertikale Linie konnte nicht gelöscht werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Array speichert Datumswerte, die verwendet werden,
//--- um die Koordinaten der Ankerpunkten der Linie zu setzen und zu ändern
   datetime date[];
//--- Speicher reservieren
   ArrayResize(date,bars);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Wir definieren Punkte, um die Linie zu zeichnen
   int d=InpDate*(bars-1)/100;
//--- eine vertikale Linie erstellen
   if(!VLineCreate(0,InpName,0,date[d],InpColor,InpStyle,InpWidth,InpBack,
      InpSelection,InpRay,InpHidden,InpZOrder))
      return;
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Linie
//--- Schleifenzähler
   int h_steps=bars/2;
//--- Linie bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!VLineMove(0,InpName,date[d]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.03 Sekunden
      Sleep(30);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Kanal aus dem Chart löschen
   VLineDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_HLINE

Horizontale Linie.



Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart eine horizontale Linie. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Horizontale Linie\
#property description "Preis des Ankerpunkts wird als Prozentsatz"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="HLine";      // Linienname
input int              InpPrice=25;          // Linienpreis in %
input color            InpColor=clrRed;      // Linienfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASH; // Linienstil
input int              InpWidth=3;           // Linienbreite
input bool             InpBack=false;        // Die Linie im Hintergrund
input bool             InpSelection=true;    // Wählen zu bewegen
input bool             InpHidden=true;       // Ausgeblendet in der Objektliste
input long             InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt eine horizontale Linie                                  |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool HLineCreate(const long            chart_ID=0,        // ID des Charts
                 const string          name="HLine",      // Linienname
                 const int             sub_window=0,      // Nummer des Unterfensters
                 double                price=0,           // Linienpreis
                 const color           clr=clrRed,        // Linienfarbe
                 const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil
                 const int             width=1,           // Linienbreite
                 const bool            back=false,        // Im Hintergrund
                 const bool            selection=true,    // Wählen um zu bewegen
                 const bool            hidden=true,       // Ausgeblendet in der Objek
                 const long            z_order=0)         // Priorität auf Mausklick
  {
//--- wenn der Preis nicht angegeben ist, setzen wir ihn gleich dem aktuellen Bid-Prei
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine horizontale Linie erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_HLINE,sub_window,0,price))
     {
      Print(__FUNCTION__,
            ": Horizontale Linie konnte nicht erstellt werden! Fehlercode = ",GetLastE
      return(false);
     }
//--- Farbe der Linie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Horizontale Linie bewegen                                        |
//+------------------------------------------------------------------+
bool HLineMove(const long   chart_ID=0,   // ID des Charts
```

© 2000-2025, MetaQuotes Ltd.

```
                const string name="HLine", // Linienname
                double       price=0)      // Linienfarbe
  {
//--- wenn der Preis der Linie nicht angegeben ist, bewegen wir die Linie auf das Nive
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine horizontale Linie bewegen
   if(!ObjectMove(chart_ID,name,0,0,price))
     {
      Print(__FUNCTION__,
            ": Horizontale Linie konnte nicht bewegt werden! Fehlercode = ",GetLastErr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht eine horizontale Linie                                    |
//+------------------------------------------------------------------+
bool HLineDelete(const long   chart_ID=0,   // ID des Charts
                 const string name="HLine") // Linienname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine horizontale Linie löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Horizontale Linie konnte nicht gelöscht werden! Fehlercode = ",GetLastE
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Größe des Arrays price
   int accuracy=1000;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Array speichert Preiswerte, die verwendet werden,
//--- um die Koordinaten der Ankerpunkten der Linie zu setzen und zu ändern
   double price[];
//--- Speicher reservieren
   ArrayResize(price,accuracy);
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Wir definieren Punkte, um die Linie zu zeichnen
   int p=InpPrice*(accuracy-1)/100;
//--- eine horizontale Linie erstellen
   if(!HLineCreate(0,InpName,0,price[p],InpColor,InpStyle,InpWidth,InpBack,
      InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Linie
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- Linie bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p<accuracy-1)
         p+=1;
      //--- den Punkt bewegen
      if(!HLineMove(0,InpName,price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- die Linie aus dem Chart löschen
   HLineDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
```

© 2000-2025, MetaQuotes Ltd.

```
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_TREND

Trendline.



Hinweis

Für die Trendlinie können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart eine Trendlinie. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Trendlinie\"."
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Trend";      // Linienname
input int             InpDate1=35;          // Datum des ersten Punktes in %
input int             InpPrice1=60;         // Preis des ersten Punktes in %
input int             InpDate2=65;          // Datum des zweiten Punktes in %
input int             InpPrice2=40;         // Preis des zweiten Punktes in %
input color           InpColor=clrRed;      // Linienfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASH; // Linienstil
```

© 2000-2025, MetaQuotes Ltd.

```
input int              InpWidth=2;          // Linienbreite
input bool             InpBack=false;       // Die Linie im Hintergrund
input bool             InpSelection=true;   // Wählen zu bewegen
input bool             InpRayLeft=false;    // Fortsetzung der Linie nach links
input bool             InpRayRight=false;   // Fortsetzung der Linie nach rechts
input bool             InpHidden=true;      // Ausgeblendet in der Objektliste
input long             InpZOrder=0;         // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt eine Trendlinie auf angegebenen Koordinaten |
//+------------------------------------------------------------------+
bool TrendCreate(const long            chart_ID=0,        // ID des Charts
                 const string          name="TrendLine",  // Linienname
                 const int             sub_window=0,      // Nummer des Unterfensters
                 datetime              time1=0,           // Zeit des ersten Punktes
                 double                price1=0,          // Preis des ersten Punktes
                 datetime              time2=0,           // Zeit des zweiten Punktes
                 double                price2=0,          // Preis des zweiten Punktes
                 const color           clr=clrRed,        // Linienfarbe
                 const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil
                 const int             width=1,           // Linienbreite
                 const bool            back=false,        // Im Hintergrund
                 const bool            selection=true,    // Wählen um zu bewegen
                 const bool            ray_left=false,    // Fortsetzung der Linie nac
                 const bool            ray_right=false,   // Fortsetzung der Linie nac
                 const bool            hidden=true,       // Ausgeblendet in der Objek
                 const long            z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeTrendEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine Trendlinien auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_TREND,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": Trendlinie konnte nicht erstellt werden! Fehlercode = ",GetLastError())
      return(false);
     }
//--- Farbe der Linie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
```

© 2000-2025, MetaQuotes Ltd.

```
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung der Linie nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung der Linie nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Trendlinien                        |
//+------------------------------------------------------------------+
bool TrendPointChange(const long   chart_ID=0,       // ID des Charts
                      const string name="TrendLine", // Linienname
                      const int    point_index=0,    // Nummer des Ankerpunktes
                      datetime     time=0,           // Zeitkoordinate des Punktes
                      double       price=0)          // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Ankerpunkt der Trendlinie bewegen
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion löscht die Trendlinie aus dem Chart                 |
//+------------------------------------------------------------------+
bool TrendDelete(const long   chart_ID=0,       // ID des Charts
                 const string name="TrendLine") // Linienname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine Trendlinie löschen
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Trendlinie konnte nicht gelöscht werden! Fehlercode = ",GetLastError())
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Trendlinie              |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeTrendEmptyPoints(datetime &time1,double &price1,
                            datetime &time2,double &price2)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time2)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time1,10,temp);
      //--- den zweiten Punkt 9 Bars nach links vom ersten Punkt setzen
      time2=temp[0];
     }
//--- Wenn der Preis des zweiten Punktes nicht angegeben wird, dann wird es der Preis
   if(!price2)
      price2=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Bezugspunkte der Linie verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Wir definieren Punkte, um die Linie zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- eine Trendlinie erstellen
   if(!TrendCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],InpColor,InpStyle
      InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir die Bezugspunkte der Linie bewegen
//--- Schleifenzähler
   int v_steps=accuracy/5;
//--- Wir bewegen den ersten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
         p1-=1;
      //--- den Punkt bewegen
      if(!TrendPointChange(0,InpName,0,date[d1],price[p1]))
```

© 2000-2025, MetaQuotes Ltd.

```
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- Wir bewegen den zweiten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p2<accuracy-1)
         p2+=1;
      //--- den Punkt bewegen
      if(!TrendPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Schleifenzähler
   int h_steps=bars/2;
//--- Wir bewegen die beiden Bezugspunkte horizontal gleichzeitig
   for(int i=0;i<h_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(d1<bars-1)
         d1+=1;
      if(d2>1)
         d2-=1;
      //--- Die Punkte bewegen
      if(!TrendPointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!TrendPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.03 Sekunden
      Sleep(30);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die Trendlinie löschen
   TrendDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_TRENDBYANGLE

Winkel Trendlinie.



Hinweis

Für die Trendlinie können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Um die Linienneigung anzugeben, können Sie den Winkel und Koordinaten des zweiten Ankerpunktes verwenden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart eine Trendlinie. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Winkel Trendlinie\
#property description "Koordinaten der Ankerpunkten werden in Pixeln angegeben"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="Trend";      // Linienname
input int              InpDate1=50;          // Datum des ersten Punktes in %
input int              InpPrice1=75;         // Preis des ersten Punktes in %
input int              InpAngle=0;           // Winkel der Linie
input color            InpColor=clrRed;      // Linienfarbe
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE InpStyle=STYLE_DASH; // Linienstil
input int             InpWidth=2;          // Linienbreite
input bool            InpBack=false;       // Die Linie im Hintergrund
input bool            InpSelection=true;   // Wählen zu bewegen
input bool            InpRayLeft=false;    // Fortsetzung der Linie nach links
input bool            InpRayRight=true;    // Fortsetzung der Linie nach rechts
input bool            InpHidden=true;      // Ausgeblendet in der Objektliste
input long            InpZOrder=0;         // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt eine Winkel-Trendlinie                                  |
//+------------------------------------------------------------------+
bool TrendByAngleCreate(const long            chart_ID=0,        // ID des Charts
                        const string          name="TrendLine",  // Linienname
                        const int             sub_window=0,      // Nummer des Unterfe
                        datetime              time=0,            // Zeit des Punktes
                        double                price=0,           // Preis des Punktes
                        const double          angle=45.0,        // Winkel
                        const color           clr=clrRed,        // Linienfarbe
                        const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil
                        const int             width=1,           // Linienbreite
                        const bool            back=false,        // Im Hintergrund
                        const bool            selection=true,    // Wählen um zu beweg
                    const bool            ray_left=false,    // Fortsetzung der Linie
                    const bool            ray_right=true,    // Fortsetzung der Linie
                        const bool            hidden=true,       // Ausgeblendet in de
                        const long            z_order=0)         // Priorität auf Maus
  {
//--- Um es bequem wäre, die Trendlinie mit dem Maus zu bewegen, erstellen wir einen z
   datetime time2=0;
   double   price2=0;
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeTrendEmptyPoints(time,price,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine Trendlinie auf zwei Punkte erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_TRENDBYANGLE,sub_window,time,price,time2,price2)
     {
      Print(__FUNCTION__,
            ": Trendlinie konnte nicht erstellt werden! Fehlercode = ",GetLastError())
      return(false);
     }
//--- Den Winkel der Trendlinie ändern; und in den Prozess der Veränderung des Winkels
//--- Punktes der Linie automatisch neu definiert in Übereinstimmung mit dem neuen Wir
   ObjectSetDouble(chart_ID,name,OBJPROP_ANGLE,angle);
//--- Farbe der Linie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung der Linie nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung der Linie nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Koordinaten der Ankerpunkte der Trendlinie            |
//+------------------------------------------------------------------+
bool TrendPointChange(const long   chart_ID=0,       // ID des Charts
                      const string name="TrendLine", // Linienname
                      datetime     time=0,           // Zeitkoordinate des Punktes
                      double       price=0)          // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Ankerpunkt der Trendlinie bewegen
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Winkel der Trendlinie                                 |
//+------------------------------------------------------------------+
bool TrendAngleChange(const long   chart_ID=0,       // ID des Charts
```

© 2000-2025, MetaQuotes Ltd.

```
                      const string name="TrendLine", // Name der Trendlinie
                      const double angle=45)         // Winkel der Trendlinie
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Winkel der Trendlinie ändern
   if(!ObjectSetDouble(chart_ID,name,OBJPROP_ANGLE,angle))
     {
      Print(__FUNCTION__,
            ": Kann den Trendlinie Winkel nicht ändern! Fehlercode = ",GetLastError())
      return(false);
    ;}
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Löscht die Trendlinie                                            |
//+------------------------------------------------------------------+
bool TrendDelete(const long   chart_ID=0,       // ID des Charts
                 const string name="TrendLine") // Linienname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine Trendlinie löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Trendlinie konnte nicht gelöscht werden! Fehlercode = ",GetLastError())
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Trendlinie              |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeTrendEmptyPoints(datetime &time1,double &price1,
                            datetime &time2,double &price2)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- setzen wir die Koordinaten des zweiten Hilfspunktes
//--- der zweite Punkt liegt 9 Balken nach links und hat den gleichen Preis
   datetime second_point_time[10];
```

© 2000-2025, MetaQuotes Ltd.

```
   CopyTime(Symbol(),Period(),time1,10,second_point_time);
   time2=second_point_time[0];
   price2=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
    ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Bezugspunkte der Linie verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
    ;}
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Wir definieren Punkte, um die Linie zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
//--- eine Trendlinie erstellen
   if(!TrendByAngleCreate(0,InpName,0,date[d1],price[p1],InpAngle,InpColor,InpStyle,
      InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,InpZOrder))
     {
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen und drehen wir die Linie
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- bewegen wir den Ankerpunkt und ändern wir den Winkel der Linie
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
         p1-=1;
      //--- den Punkt bewegen
      if(!TrendPointChange(0,InpName,date[d1],price[p1]))
         return;
      if(!TrendAngleChange(0,InpName,18*(i+1)))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- die Linie aus dem Chart löschen
   TrendDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_CYCLES

Zyklische Linien.



Hinweis

Der Abstand zwischen den Linien wird durch die Zeitkoordinaten der zwei Ankerpunkte des Objekts angegeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart zyklische Linien. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt zyklische Linien auf dem Chart."
#property description "Koordinaten der Ankerpunkten werden in Prozente angegeben"
#property description "von der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Cycles";   // Objektname
input int             InpDate1=10;        // Datum des ersten Punktes in %
input int             InpPrice1=45;       // Preis des ersten Punktes in %
input int             InpDate2=20;        // Datum des zweiten Punktes in %
input int             InpPrice2=55;       // Preis des zweiten Punktes in %
input color           InpColor=clrRed;    // Farbe der zyklischen Linien
input ENUM_LINE_STYLE InpStyle=STYLE_DOT; // Stil der zyklischen Linien
```

© 2000-2025, MetaQuotes Ltd.

```
input int              InpWidth=1;          // Breite der zyklischen Linien
input bool             InpBack=false;       // Objekt im Hintergrund
input bool             InpSelection=true;   // Wählen um zu bewegen
input bool             InpHidden=true;      // Ausgeblendet in der Objektliste
input long             InpZOrder=0;         // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Zyklische Linien                                        |
//+------------------------------------------------------------------+
bool CyclesCreate(const long            chart_ID=0,        // ID des Charts
                  const string          name="Cycles",     // Objektname
                  const int             sub_window=0,      // Nummer des Unterfensters
                  datetime              time1=0,           // Zeit des ersten Punktes
                  double                price1=0,          // Preis des ersten Punktes
                  datetime              time2=0,           // Zeit des zweiten Punktes
                  double                price2=0,          // Preis des zweiten Punkte
                  const color           clr=clrRed,        // Farbe der zyklischen Lin
                  const ENUM_LINE_STYLE style=STYLE_SOLID, // Stil der zyklischen Lini
                  const int             width=1,           // Breite der zyklischen Li
                  const bool            back=false,        // Im Hintergrund
                  const bool            selection=true,    // Wählen um zu bewegen
                  const bool            hidden=true,       // Ausgeblendet in der List
                  const long            z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeCyclesEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- zyklische Linien auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_CYCLES,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": Zyklische Linien konnten nicht erstellt werden! Fehlercode = ",GetLastE
      return(false);
     }
//--- Linienfarbe angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung der Linien mit dem Maus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool CyclesPointChange(const long   chart_ID=0,    // ID des Charts
                       const string name="Cycles", // Objektname
                       const int    point_index=0, // Nummer des Punktes
                       datetime     time=0,        // Zeitkoordinate des Punktes
                       double       price=0)       // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Löscht Zyklische Linien                                          |
//+------------------------------------------------------------------+
bool CyclesDelete(const long   chart_ID=0,    // ID des Charts
                  const string name="Cycles") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zyklische Linien löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Zyklische Linien konnten nicht gelöscht werden! Fehlercode = ",GetLastE
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von zyklische Linien        |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeCyclesEmptyPoints(datetime &time1,double &price1,
                             datetime &time2,double &price2)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time2)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time1,10,temp);
      //--- den zweiten Punkt 9 Bars nach links vom ersten Punkt setzen
      time2=temp[0];
     }
//--- Wenn der Preis des zweiten Punktes nicht angegeben wird, dann wird es der Preis
   if(!price2)
      price2=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten der zyklische Linien verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
```

© 2000-2025, MetaQuotes Ltd.

```
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
    ;}
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um die zyklische Linien zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- eine Trendlinie erstellen
   if(!CyclesCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int h_steps=bars/5;
//--- den zweiten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2<bars-1)
         d2+=1;
      //--- den Punkt bewegen
      if(!CyclesPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   h_steps=bars/4;
//--- den ersten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d1<bars-1)
         d1+=1;
      //--- den Punkt bewegen
      if(!CyclesPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   CyclesDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROWED_LINE

Linie mit einem Pfeil.



Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart eine Linie mit einem Pfeil. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt das graphische Objekt \"Linie mit einem Pfe
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="ArrowedLine"; // Name der Linie
input int             InpDate1=35;           // Datum des ersten Punktes in %
input int             InpPrice1=60;          // Preis des ersten Punktes in %
input int             InpDate2=65;           // Datum des zweiten Punktes in %
input int             InpPrice2=40;          // Preis des zweiten Punktes in %
input color           InpColor=clrRed;       // Linienfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASH;   // Linienstil
input int             InpWidth=2;            // Linienbreite
input bool            InpBack=false;         // Linie im Hintergrund
input bool            InpSelection=true;     // Wählen zu bewegen
input bool            InpHidden=true;        // in der Liste der Objekten ausgeblendet
input long            InpZOrder=0;           // Priorität auf Mausklick
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Erstellt eine Linie mit Pfeil auf den angegebenen Koordinaten  |
//+------------------------------------------------------------------+
bool ArrowedLineCreate(const long            chart_ID=0,         // ID des Charts
                       const string          name="ArrowedLine", // Linienname
                       const int             sub_window=0,       // Nummer des Unterfe
                       datetime              time1=0,            // Zeit des ersten Pu
                       double                price1=0,           // Preis des ersten P
                       datetime              time2=0,            // Zeit des zweiten P
                       double                price2=0,           // Preis des zweiten
                       const color           clr=clrRed,         // Linienfarbe
                       const ENUM_LINE_STYLE style=STYLE_SOLID,  // Linienstil
                       const int             width=1,            // Linienbreite
                       const bool            back=false,         // Im Hintergrund
                       const bool            selection=true,     // Wählen zu bewegen
                       const bool            hidden=true,        // Verborgen in der I
                       const long            z_order=0)          // Priorität für Maus
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeArrowedLineEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Wir erstellen eine Linie mit einem Pfeil an den angegebenen Koordinaten
   if(!ObjectCreate(chart_ID,name,OBJ_ARROWED_LINE,sub_window,time1,price1,time2,price
     {
      Print(__FUNCTION__,
            ": Eine Linie mit dem Pfeil konnte nicht erstellt werden! Fehlercode = ",G
      return(false);
     }
//--- Farbe der Linie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt der Linie mit einem Pfeil                 |
//+------------------------------------------------------------------+
bool ArrowedLinePointChange(const long   chart_ID=0,          // ID des Charts
                            const string name="ArrowedLine", // Linienname
                            const int    point_index=0,      // Nummer des Bezugspunkt
                            datetime     time=0,             // Zeitkoordinate des Bez
                            double       price=0)            // Preiskoordinate des Be
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt der Linie
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion löscht die Linie mit dem Pfeil aus dem Chart        |
//+------------------------------------------------------------------+
bool ArrowedLineDelete(const long   chart_ID=0,          // ID des Charts
                       const string name="ArrowedLine") // Linienname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Die Linie mit einem Pfeil löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Eine Linie mit dem Pfeil konnte nicht gelöscht werden! Fehlercode = ",G
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Bezugspunkte der Linie für lehre Werte   |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void ChangeArrowedLineEmptyPoints(datetime &time1,double &price1,
                                  datetime &time2,double &price2)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Wenn Zeit des zweiten Punktes nicht angegeben ist, wird er 9 Bars nach links vor
   if(!time2)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time1,10,temp);
      //--- den zweiten Punkt 9 Bars nach links vom ersten Punkt setzen
      time2=temp[0];
     }
//--- Wenn der Preis des zweiten Punktes nicht angegeben wird, dann wird es der Preis
   if(!price2)
      price2=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Bezugspunkte der Linie verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Wir definieren Punkte, um die Linie zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- Die Linie mit einem Pfeil erstellen
   if(!ArrowedLineCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir die Bezugspunkte der Linie bewegen
//--- Schleifenzähler
   int v_steps=accuracy/5;
//--- Wir bewegen den zweiten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p2<accuracy-1)
         p2+=1;
      //--- den Punkt bewegen
      if(!ArrowedLinePointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- Wir bewegen den ersten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
         p1-=1;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den Punkt bewegen
      if(!ArrowedLinePointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Schleifenzähler
   int h_steps=bars/2;
//--- Wir bewegen die beiden Bezugspunkte horizontal gleichzeitig
   for(int i=0;i<h_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(d1<bars-1)
         d1+=1;
      if(d2>1)
         d2-=1;
      //--- Die Punkte bewegen
      if(!ArrowedLinePointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!ArrowedLinePointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.03 Sekunden
      Sleep(30);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Die Linie mit einem Pfeil löschen
   ArrowedLineDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_CHANNEL

Äquidistanter Kanal.



Hinweis

Für einen äquidistanten Kanal können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben. Sie können auch die Farbfüllung des Kanals angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart einen äquidistanten Kanal. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Äquidistanter Kana
#property description "Koordinaten der Ankerpunkten werden in Pixeln angegeben"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Channel";   // Kanalname
input int             InpDate1=25;         // Datum des ersten Punktes in %
input int             InpPrice1=60;        // Preis des ersten Punktes in %
input int             InpDate2=65;         // Datum des zweiten Punktes in %
input int             InpPrice2=80;        // Preis des zweiten Punktes in %
input int             InpDate3=30;         // Datum des dritten Punktes in %
```

© 2000-2025, MetaQuotes Ltd.

```
input int              InpPrice3=40;         // Preis des dritten Punktes in %
input color            InpColor=clrRed;      // Farbe des Kanals
input ENUM_LINE_STYLE  InpStyle=STYLE_DASH;  // Stil der Kanallinie
input int              InpWidth=2;           // Linienbreite
input bool             InpBack=false;        // Kanal im Hintergrund
input bool             InpFill=false;        // Farbfüllung des Kanals
input bool             InpSelection=true;    // Wählen zu bewegen
input bool             InpRayLeft=false;     // Fortsetzung des Kanals nach links
input bool             InpRayRight=false;    // Fortsetzung des Kanals nach rechts
input bool             InpHidden=true;       // Ausgeblendet in der Objektliste
input long             InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt einen äquidistanten Kanal auf angegebenen Koordinaten   |
//+------------------------------------------------------------------+
bool ChannelCreate(const long            chart_ID=0,        // ID des Charts
                   const string          name="Channel",    // Kanalname
                   const int             sub_window=0,      // Nummer des Unterfenster
                   datetime              time1=0,           // Zeit des ersten Punktes
                   double                price1=0,          // Preis des ersten Punkte
                   datetime              time2=0,           // Zeit des zweiten Punkte
                   double                price2=0,          // Preis des zweiten Punkt
                   datetime              time3=0,           // Zeit des dritten Punkte
                   double                price3=0,          // Preis des dritten Punkt
                   const color           clr=clrRed,        // Kanalfarbe
                   const ENUM_LINE_STYLE style=STYLE_SOLID, // Stil der Kanallinien
                   const int             width=1,           // Breite der Kanallinien
                   const bool            fill=false,        // Farbfüllung des Kanals
                   const bool            back=false,        // Im Hintergrund
                   const bool            selection=true,    // Auswählen um zu bewegen
                   const bool            ray_left=false,    // Fortsetzung des Kanals
                   const bool            ray_right=false,   // Fortsetzung des Kanals
                   const bool            hidden=true,       // Ausgeblendet in der Obj
                   const long            z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeChannelEmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Kanal auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_CHANNEL,sub_window,time1,price1,time2,price2,tim
     {
      Print(__FUNCTION__,
            ": Kanal konnte nicht erstellt werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Kanalfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Breite der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- aktivieren (true) oder deaktivieren (false) Kanalfüllung
   ObjectSetInteger(chart_ID,name,OBJPROP_FILL,fill);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen des Kanals für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt die Kanalbindung                                      |
//+------------------------------------------------------------------+
bool ChannelPointChange(const long   chart_ID=0,     // ID des Charts
                        const string name="Channel", // Kanalname
                        const int    point_index=0,  // Nummer des Punktes
                        datetime     time=0,         // Zeitkoordinate des Punktes
                        double       price=0)        // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Löscht den Kanal                                                 |
//+------------------------------------------------------------------+
bool ChannelDelete(const long   chart_ID=0,     // ID des Charts
                   const string name="Channel") // Kanalname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Kanal löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": kann nicht den Kanal löschen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Kanalbindungspunkten für leere Werte     |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeChannelEmptyPoints(datetime &time1,double &price1,datetime &time2,
                              double &price2,datetime &time3,double &price3)
  {
//--- wenn die Zeit des zweiten (rechten) Punktes nicht angegeben ist, wird sie auf de
   if(!time2)
      time2=TimeCurrent();
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price2)
      price2=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten (linken) Punktes nicht angegeben ist, wird sie auf 9 Ba
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, bewegen wir ihn 300 Punkt
   if(!price1)
      price1=price2+300*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird sie der Zeit des zwe
   if(!time3)
      time3=time1;
//--- wenn der Preis des dritten Punktes nicht angegeben ist, wird er dem Preis des zw
   if(!price3)
      price3=price2;
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Kanalbindungspunkten verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um den Kanal zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Erstellen einen äquidistanten Kanal
   if(!ChannelCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],price
```

© 2000-2025, MetaQuotes Ltd.

```
      InpStyle,InpWidth,InpFill,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Kanalbindungspunkte
//--- Schleifenzähler
   int h_steps=bars/6;
//--- den zweiten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2<bars-1)
         d2+=1;
      //--- den Punkt bewegen
      if(!ChannelPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den ersten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d1>1)
         d1-=1;
      //--- den Punkt bewegen
      if(!ChannelPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
```

© 2000-2025, MetaQuotes Ltd.

```
   int v_steps=accuracy/10;
//--- den dritten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p3>1)
         p3-=1;
      //--- den Punkt bewegen
      if(!ChannelPointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Kanal aus dem Chart löschen
   ChannelDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_STDDEVCHANNEL

Kanal der Standardabweichung.



Hinweis

Für das Objekt "Kanal der Standardabweichung" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben. Sie können auch die Farbfüllung des Kanals angeben.

Um die Werte des Kanals der Standardabweichung zu ändern, wird Eigenschaft OBJPROP_DEVIATION verwendet.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Kanal der Standardabweichung". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Kanal der Standard
#property description "Koordinaten der Ankerpunkten werden in Pixeln angegeben"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="StdDevChannel";    // Kanalname
input int             InpDate1=10;                // Datum des ersten Punktes in %
input int             InpDate2=40;                // Datum des zweiten Punktes in %
input double          InpDeviation=1.0;           // Abweichung
```

© 2000-2025, MetaQuotes Ltd.

```
input color           InpColor=clrRed;           // Kanalfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Stil der Kanallinien
input int             InpWidth=2;                // Kanallinienbreite
input bool            InpFill=false;         // Farbfüllung des Kanals
input bool            InpBack=false;             // Im Hintergrund
input bool            InpSelection=true;         // Wählen um zu bewegen
input bool            InpRayLeft=false;          // Fortsetzung des Kanals nach links
input bool            InpRayRight=false;         // Fortsetzung des Kanals nach rechts
input bool            InpHidden=true;            // Ausgeblendet in der Objektliste
input long            InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Kanal der Standardabweichung mit angegebenen Koordinaten|
//+------------------------------------------------------------------+
bool StdDevChannelCreate(const long             chart_ID=0,        // ID des Charts
                         const string           name="Channel",    // Kanalname
                         const int              sub_window=0,      // Nummer des Unter
                         datetime               time1=0,           // Zeit des ersten P
                         datetime               time2=0,           // Zeit des zweiten
                         const double           deviation=1.0,     // Abweichung
                         const color            clr=clrRed,        // Kanalfarbe
                         const ENUM_LINE_STYLE  style=STYLE_SOLID, // Stil der Kanallin
                         const int              width=1,           // Breite der Kanall
                         const bool             fill=false,        // Farbfüllung des P
                         const bool             back=false,        // Im Hintergrund
                         const bool             selection=true,    // Wählen um zu bewe
                   const bool             ray_left=false,    // Fortsetzung des Kanals
                   const bool             ray_right=false,   // Fortsetzung des Kanals
                         const bool             hidden=true,       // Ausgeblendet in d
                         const long             z_order=0)         // Priorität auf Mau
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeChannelEmptyPoints(time1,time2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Kanal auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_STDDEVCHANNEL,sub_window,time1,0,time2,0))
     {
      Print(__FUNCTION__,
            ": Kanal der Standardabweichung konnte nicht erstellt werden! Fehlercode =
      return(false);
     }
//--- Wert der Abweichung setzen, Kanalbreite abhängt von ihm
   ObjectSetDouble(chart_ID,name,OBJPROP_DEVIATION,deviation);
//--- Kanalfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- aktivieren (true) oder deaktivieren (false) Kanalfüllung
   ObjectSetInteger(chart_ID,name,OBJPROP_FILL,fill);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen des Kanals für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt die Kanalbindung                                      |
//+------------------------------------------------------------------+
bool StdDevChannelPointChange(const long   chart_ID=0,     // ID des Charts
                              const string name="Channel", // Kanalname
                              const int    point_index=0,  // Nummer des Ankerpunktes
                              datetime     time=0)         // Zeitkoordinate des Anker
  {
//--- wenn die Zeit des Punktes nicht angegeben ist, bewegen wir ihn auf den letzten B
   if(!time)
      time=TimeCurrent();
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,0))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert Abweichung des Kanals                                     |
//+------------------------------------------------------------------+
bool StdDevChannelDeviationChange(const long   chart_ID=0,     // ID des Charts
                                  const string name="Channel", // Kanalname
```

© 2000-2025, MetaQuotes Ltd.

```
                                    const double deviation=1.0)  // Abweichung
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Winkel der Trendlinie ändern
   if(!ObjectSetDouble(chart_ID,name,OBJPROP_DEVIATION,deviation))
     {
      Print(__FUNCTION__,
            ": Konnte nicht Kanalabweichung ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht den Kanal                                                 |
//+------------------------------------------------------------------+
bool StdDevChannelDelete(const long   chart_ID=0,     // ID des Charts
                         const string name="Channel") // Kanalname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Kanal löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": kann nicht den Kanal löschen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Überprüft die Werte der Kanalbindungspunkten für leere Werte     |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeChannelEmptyPoints(datetime &time1,datetime &time2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den zweiten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
   {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 ||
      InpDate2<0 || InpDate2>100)
      {
       Print("Fehler! Ungültige Eingabeparameter! ");
       return;
      }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Kanalbindungspunkten verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um den Kanal zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
//--- wir erstellen den Kanal der Standardabweichung
   if(!StdDevChannelCreate(0,InpName,0,date[d1],date[d2],InpDeviation,InpColor,InpStyl
      InpWidth,InpFill,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,InpZOrder
     {
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir den Kanal horizontal nach rechts und erweitern ihn
//--- Schleifenzähler
   int h_steps=bars/2;
//--- Kanal bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(d1<bars-1)
         d1+=1;
      if(d2<bars-1)
         d2+=1;
      //--- Ankerpunkte bewegen
      if(!StdDevChannelPointChange(0,InpName,0,date[d1]))
         return;
      if(!StdDevChannelPointChange(0,InpName,1,date[d2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   double v_steps=InpDeviation*2;
//--- den Kanal erweitern
   for(double i=InpDeviation;i<v_steps;i+=10.0/accuracy)
     {
      if(!StdDevChannelDeviationChange(0,InpName,i))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Kanal aus dem Chart löschen
   StdDevChannelDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
```

© 2000-2025, MetaQuotes Ltd.

```
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_REGRESSION

Der Kanal auf der linearen Regression.



Hinweis

Für das Objekt "Kanal auf der linearen Regression" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben. Sie können auch die Farbfüllung des Kanals angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart den Kanal auf der linearen Regression. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Kanal auf der line
#property description "Koordinaten der Ankerpunkten werden in Pixeln angegeben"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Regression"; // Kanalname
input int             InpDate1=10;          // Datum des ersten Punktes in %
input int             InpDate2=40;          // Datum des zweiten Punktes in %
input color           InpColor=clrRed;      // Kanalfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASH;  // Stil der Kanallinien
input int             InpWidth=2;           // Breite der Kanallinien
```

© 2000-2025, MetaQuotes Ltd.

```
input bool              InpFill=false;         // Farbfüllung des Kanals
input bool              InpBack=false;         // Kanal im Hintergrund
input bool              InpSelection=true;     // Wählen um zu bewegen
input bool              InpRayLeft=false;      // Fortsetzung des Kanals nach links
input bool              InpRayRight=false;     // Fortsetzung des Kanals nach rechts
input bool              InpHidden=true;        // Ausgeblendet in der Objektliste
input long              InpZOrder=0;           // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Kanal auf der linearen Regression mit Koordinaten       |
//+------------------------------------------------------------------+
bool RegressionCreate(const long            chart_ID=0,        // ID des Charts
                      const string          name="Regression", // Kanalname
                      const int             sub_window=0,      // Nummer des Unterfens
                      datetime              time1=0,           // Zeit des ersten Punk
                      datetime              time2=0,           // Zeit des zweiten Pur
                      const color           clr=clrRed,        // Kanalfarbe
                      const ENUM_LINE_STYLE style=STYLE_SOLID, // Stil der Kanallinier
                      const int             width=1,           // Breite der Kanallini
                      const bool            fill=false,        // Farbfüllung des Kana
                      const bool            back=false,        // Im Hintergrund
                      const bool            selection=true,    // Wählen um zu beweger
                      const bool            ray_left=false,    // Fortsetzung des Kana
                      const bool            ray_right=false,   // Fortsetzung des Kana
                      const bool            hidden=true,       // Ausgeblendet in der
                      const long            z_order=0)         // Priorität auf Mausk
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeRegressionEmptyPoints(time1,time2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Kanal auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_REGRESSION,sub_window,time1,0,time2,0))
     {
      Print(__FUNCTION__,
            ": Kanal auf der linearen Regression konnte nicht erstellt werden! Fehlerc
      return(false);
     }
//--- Kanalfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- aktivieren (true) oder deaktivieren (false) Kanalfüllung
   ObjectSetInteger(chart_ID,name,OBJPROP_FILL,fill);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen des Kanals für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
```

© 2000-2025, MetaQuotes Ltd.

```
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt die Kanalbindung                                      |
//+------------------------------------------------------------------+
bool RegressionPointChange(const long   chart_ID=0,     // ID des Charts
                           const string name="Channel", // Kanalname
                           const int    point_index=0,  // Nummer des Ankerpunktes
                           datetime     time=0)         // Zeitkoordinate des Ankerpun
  {
//--- wenn die Zeit des Punktes nicht angegeben ist, bewegen wir ihn auf den letzten B
   if(!time)
      time=TimeCurrent();
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,0))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht den Kanal                                                 |
//+------------------------------------------------------------------+
bool RegressionDelete(const long   chart_ID=0,     // ID des Charts
                      const string name="Channel") // Kanalname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Kanal löschen
   if(!ObjectDelete(chart_ID,name))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__,
            ": kann nicht den Kanal löschen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Kanalbindungspunkten für leere Werte     |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeRegressionEmptyPoints(datetime &time1,datetime &time2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 ||
      InpDate2<0 || InpDate2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Kanalbindungspunkten verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um den Kanal zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
//--- Einen Kanal auf der linearen Regression erstellen
   if(!RegressionCreate(0,InpName,0,date[d1],date[d2],InpColor,InpStyle,InpWidth,
      InpFill,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir den Kanal auf der linearen Regression horizontal nach rechts
//--- Schleifenzähler
   int h_steps=bars/2;
//--- Kanal bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(d1<bars-1)
         d1+=1;
      if(d2<bars-1)
         d2+=1;
      //--- Ankerpunkte bewegen
      if(!RegressionPointChange(0,InpName,0,date[d1]))
         return;
      if(!RegressionPointChange(0,InpName,1,date[d2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
```

© 2000-2025, MetaQuotes Ltd.

```
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Kanal aus dem Chart löschen
   RegressionDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_PITCHFORK

Andrew's Pitchfork.



Hinweis

Für das Objekt "Andrew's Pitchfork" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Sie können auch die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Andrew's Pitchfork". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Andrew's Pitchfor
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Pitchfork";  // Name
input int             InpDate1=14;          // Datum des ersten Punktes in %
input int             InpPrice1=40;         // Preis des ersten Punktes in %
input int             InpDate2=18;          // Datum des zweiten Punktes in %
input int             InpPrice2=50;         // Preis des zweiten Punktes in %
input int             InpDate3=18;          // Datum des dritten Punktes in %
input int             InpPrice3=30;         // Preis des dritten Punktes in %
```

© 2000-2025, MetaQuotes Ltd.

```
input color           InpColor=clrRed;      // Farbe
input ENUM_LINE_STYLE InpStyle=STYLE_SOLID; // Linienstil
input int             InpWidth=1;           // Linienbreite
input bool            InpBack=false;        // Pitchfork im Hintergrund
input bool            InpSelection=true;    // Wählen um zu bewegen
input bool            InpRayLeft=false;     // Fortsetzung von Pitchfork nach links
input bool            InpRayRight=false;    // Fortsetzung von Pitchfork nach rechts
input bool            InpHidden=true;       // Ausgeblendet in der Objektliste
input long            InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Andrew's Pitchfork auf angegebenen Koordinaten          |
//+------------------------------------------------------------------+
bool PitchforkCreate(const long            chart_ID=0,        // ID des Charts
                     const string          name="Pitchfork",  // Name
                     const int             sub_window=0,      // Nummer des Unterfenst
                     datetime              time1=0,           // Zeit des ersten Punkt
                     double                price1=0,          // Preis des ersten Punk
                     datetime              time2=0,           // Zeit des zweiten Punk
                     double                price2=0,          // Preis des zweiten Pun
                     datetime              time3=0,           // Zeit des dritten Punk
                     double                price3=0,          // Preis des dritten Pun
                     const color           clr=clrRed,        // Linienfarbe
                     const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil
                     const int             width=1,           // Linienbreite
                  const bool            back=false,        // In der Hintergrund
                     const bool            selection=true,    // Wählen um zu bewegen
                     const bool            ray_left=false,    // Fortsetzung von Pitch
                     const bool            ray_right=false,   // Fortsetzung von Pitch
                 const bool            hidden=true,       // Verborgen in der Liste der
                    const long            z_order=0)         // Priorität für Mausklic
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeChannelEmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Andrew's Pitchfork auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_PITCHFORK,sub_window,time1,price1,time2,price2,t
     {
      Print(__FUNCTION__,
            ": Objekt \" Andrew's Pitchfork\" konnte nicht erstellt werden! Fehlercode
      return(false);
     }
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen von Andrew's Pitchfork für Be
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung von Andrew's Pitchfork r
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung von Andrew's Pitchfork r
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt Anzahl der Niveaus von "Andrew's Pitchfork" und Parameter an|
//+------------------------------------------------------------------+
bool PitchforkLevelsSet(int              levels,           // Anzahl der Niveaulinien
                        double           &values[],        // Werte der Linien
                        color            &colors[],        // Farbe der Linien
                        ENUM_LINE_STYLE  &styles[],        // Linienstil
                        int              &widths[],        // Linienbreite
                        const long       chart_ID=0,       // ID des Charts
                        const string     name="Pitchfork") // Name
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) ||
      levels!=ArraySize(widths) || levels!=ArraySize(widths))
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Niveaus,
      return(false);
     }
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,DoubleToString(100*values[i],1
     ;}
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Andrew's Pitchfork                 |
//+------------------------------------------------------------------+
bool PitchforkPointChange(const long   chart_ID=0,       // ID des Charts
                          const string name="Pitchfork", // Kanalname
                          const int    point_index=0,    // Nummer des Ankerpunktes
                          datetime     time=0,           // Zeitkoordinate des Punktes
                          double       price=0)          // Preiskoordinate des Punkte
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Andrew's Pitchfork"                           |
//+------------------------------------------------------------------+
bool PitchforkDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="Pitchfork") // Kanalname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Kanal löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Andrew's Pitchfork\" konnte nicht gelöscht werden! Fehlercode
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
 ;}
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Andrew's Pitchfork      |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeChannelEmptyPoints(datetime &time1,double &price1,datetime &time2,
                              double &price2,datetime &time3,double &price3)
  {
//--- wenn die Zeit des zweiten (rechten oberen) Punktes nicht angegeben ist, wird sie
   if(!time2)
      time2=TimeCurrent();
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price2)
      price2=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten (linken) Punktes nicht angegeben ist, wird sie auf 9 Ba
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, bewegen wir ihn 200 Punkt
   if(!price1)
      price1=price2-200*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird sie der Zeit des zwe
   if(!time3)
      time3=time2;
//--- wenn der Preis des dritten Punktes nicht angegeben ist, bewegen wir ihn 200 Punk
   if(!price3)
      price3=price1-200*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
```

© 2000-2025, MetaQuotes Ltd.

```
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten von Andrew's Pitchfork verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Andrew's Pitchfork zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Andrew's Pitchfork erstellen
   if(!PitchforkCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],pric
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy/10;
//--- den ersten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
         p1-=1;
      //--- den Punkt bewegen
```

© 2000-2025, MetaQuotes Ltd.

```
      if(!PitchforkPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int h_steps=bars/8;
//--- den dritten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d3<bars-1)
         d3+=1;
      //--- den Punkt bewegen
      if(!PitchforkPointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy/10;
//--- den zweiten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p2>1)
         p2-=1;
      //--- den Punkt bewegen
      if(!PitchforkPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Andrew's Pitchfork aus dem Chart löschen
   PitchforkDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_GANNLINE

Gann Linie.



Hinweis

Für das Objekt "Gann Linie" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Um die Linienneigung anzugeben, können Sie Gann Winkel mit dem Maßstab und Koordinaten des zweiten Ankerpunktes verwenden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Gann Linie". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Gann Linie\"."
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="GannLine";        // Linienname
input int              InpDate1=20;               // Datum des ersten Punktes in %
input int              InpPrice1=75;              // Preis des ersten Punktes in %
input int              InpDate2=80;               // Datum des zweiten Punktes in %
input double           InpAngle=0.0;              // Gann Winkel
```

© 2000-2025, MetaQuotes Ltd.

```
input double           InpScale=1.0;              // Maßstab
input color            InpColor=clrRed;           // Linienfarbe
input ENUM_LINE_STYLE  InpStyle=STYLE_DASHDOTDOT; // Linienstil
input int              InpWidth=2;                // Linienbreite
input bool             InpBack=false;             // Die Linie im Hintergrund
input bool             InpSelection=true;         // Wählen um zu bewegen
input bool             InpRayLeft=false;          // Fortsetzung der Linie nach links
input bool             InpRayRight=true;          // Fortsetzung der Linie nach rechts
input bool             InpHidden=true;            // Ausgeblendet in der Objektliste
input long             InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Gann Linie auf Koordinaten, Winkel und Maßstab          |
//+------------------------------------------------------------------+
bool GannLineCreate(const long             chart_ID=0,        // ID des Charts
                    const string           name="GannLine",   // Linienname
                    const int              sub_window=0,      // Nummer des Fensters
                    datetime               time1=0,           // Zeit des ersten Punkte
                    double                 price1=0,          // Preis des ersten Punkt
                    datetime               time2=0,           // Zeit des zweiten Punkt
                    const double           angle=1.0,         // Gann Winkel
                    const double           scale=1.0,         // Maßstab
                    const color            clr=clrRed,        // Linienfarbe
                    const ENUM_LINE_STYLE  style=STYLE_SOLID, // Linienstil
                    const int              width=1,           // Linienbreite
                    const bool             back=false,        // In der Hintergrund
                    const bool             selection=true,    // Wählen um zu bewegen
                    const bool             ray_left=false,    // Fortsetzung der Linie
                    const bool             ray_right=true,    // Fortsetzung der Linie
                    const bool             hidden=true,       // Verborgen in der Liste
                    const long             z_order=0)         // Priorität für Mausklic
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeGannLineEmptyPoints(time1,price1,time2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Gann Linie" auf angegebenen Koordinaten erstellen
//--- Richtige Preiskoordinate des zweiten Ankerpunktes wird automatisch
//--- neu gesetzt nachdem Gann Winkel und (oder) Maßstab verändert ist,
   if(!ObjectCreate(chart_ID,name,OBJ_GANNLINE,sub_window,time1,price1,time2,0))
     {
      Print(__FUNCTION__,
            ": Objekt \" Gann Linie\" konnte nicht erstellt werden! Fehlercode = ",Get
      return(false);
     }
//--- Gann Winkel ändern
   ObjectSetDouble(chart_ID,name,OBJPROP_ANGLE,angle);
//--- Ändern wir den Maßstab (Anzahl der Pips pro Bar)
   ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale);
//--- Farbe der Linie setzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen der Linie für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung der Linie nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung der Linie nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Gann Linien                        |
//+------------------------------------------------------------------+
bool GannLinePointChange(const long   chart_ID=0,      // ID des Charts
                         const string name="GannLine", // Linienname
                         const int    point_index=0,   // Nummer des Ankerpunktes
                         datetime     time=0,          // Zeitkoordinate des Punktes
                         double       price=0)         // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt der Linie
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Ändert Gann Winkel                                               |
//+------------------------------------------------------------------+
bool GannLineAngleChange(const long   chart_ID=0,      // ID des Charts
                         const string name="GannLine", // Linienname
                         const double angle=1.0)       // Gann Winkel
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Gann Winkel ändern
   if(!ObjectSetDouble(chart_ID,name,OBJPROP_ANGLE,angle))
     {
      Print(__FUNCTION__,
            ": Kann den Gann Winkel nicht ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Maßstab von "Gann Linie"                              |
//+------------------------------------------------------------------+
bool GannLineScaleChange(const long   chart_ID=0,      // ID des Charts
                         const string name="GannLine", // Linienname
                         const double scale=1.0)       // Maßstab
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern wir den Maßstab (Anzahl der Pips pro Bar)
   if(!ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Maßstab ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Die Funktion löscht das Objekt "Gann Linie"                      |
//+------------------------------------------------------------------+
bool GannLineDelete(const long   chart_ID=0,      // ID des Charts
                    const string name="GannLine") // Linienname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Gann Linie löschen
   if(!ObjectDelete(chart_ID,name))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print(__FUNCTION__,
             ": Objekt \" Gann Linie\" konnte nicht gelöscht werden! Fehlercode = ",Get
       return(false);
      }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Gann Linie              |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeGannLineEmptyPoints(datetime &time1,double &price1,datetime &time2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Bezugspunkte der Linie verwendet werden
   datetime date[];
```

© 2000-2025, MetaQuotes Ltd.

```
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Gann Linie zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
//--- Gann Linie erstellen
   if(!GannLineCreate(0,InpName,0,date[d1],price[p1],date[d2],InpAngle,InpScale,InpCol
      InpStyle,InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,InpZOrde
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir den Ankerpunkt von der Linie und ändern den Winkel
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- Wir bewegen den ersten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
         p1-=1;
      //--- den Punkt bewegen
      if(!GannLinePointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Definieren wir den aktuellen wert des Gann-Winkels (hat
//--- nach der Bewegung des ersten Punktes geändert)
   double curr_angle;
   if(!ObjectGetDouble(0,InpName,OBJPROP_ANGLE,0,curr_angle))
      return;
//--- Schleifenzähler
   v_steps=accuracy/8;
//--- Gann Winkel ändern
   for(int i=0;i<v_steps;i++)
     {
      if(!GannLineAngleChange(0,InpName,curr_angle-0.05*i))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- die Linie aus dem Chart löschen
   GannLineDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_GANNFAN

Gann Fan.



Hinweis

Für "Gann Fan" können Sie den Trendtyp aus der Enumeration ENUM_GANN_DIRECTION angeben. Durch die Anpassung des Maßstabs (OBJPROP_SCALE), können Sie den Winkel der Fanlinien ändern.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Gann Fan". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Gann Fan\"."
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="GannFan";          // Name
input int             InpDate1=15;                // Datum des ersten Punktes in %
input int             InpPrice1=25;               // Preis des ersten Punktes in %
input int             InpDate2=85;                // Datum des zweiten Punktes in %
input double          InpScale=2.0;               // Maßstab
input bool            InpDirection=false;         // Trendrichtung
input color           InpColor=clrRed;            // Farbe von Fan
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Stil der Fanlinien
input int             InpWidth=1;                // Breite der Fanlinien
input bool            InpBack=false;             // Fan im Hintergrund
input bool            InpSelection=true;         // Wählen um zu bewegen
input bool            InpHidden=true;            // Ausgeblendet in der Objektliste
input long            InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt "Gann Fan"                                              |
//+------------------------------------------------------------------+
bool GannFanCreate(const long            chart_ID=0,        // ID des Charts
                   const string          name="GannFan",    // Name des Fans
                   const int             sub_window=0,      // Nummer des Unterfenster
                   datetime              time1=0,           // Zeit des ersten Punktes
                   double                price1=0,          // Preis des ersten Punkte
                   datetime              time2=0,           // Zeit des zweiten Punkte
                   const double          scale=1.0,         // Maßstab
                   const bool            direction=true,    // Trendrichtung
                   const color           clr=clrRed,        // Fanfarbe
                   const ENUM_LINE_STYLE style=STYLE_SOLID, // Stil der Fanlinien
                   const int             width=1,           // Breite der Fanlinien
                   const bool            back=false,        // Im Hintergrund
                   const bool            selection=true,    // Auswählen um zu beweger
                   const bool            hidden=true,       // Ausgeblendet in der Obj
                   const long            z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeGannFanEmptyPoints(time1,price1,time2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Gann Fan" auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_GANNFAN,sub_window,time1,price1,time2,0))
     {
      Print(__FUNCTION__,
            ": Objekt \" Gann Fan\" konnte nicht erstellt werden! Fehlercode = ",GetLa
      return(false);
     }
//--- Ändern wir den Maßstab (Anzahl der Pips pro Bar)
   ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale);
//--- Ändern wir die Richtung des Trends von "Gann Fan" (true - absteigend, falsch - a
   ObjectSetInteger(chart_ID,name,OBJPROP_DIRECTION,direction);
//--- Fanfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Fanlinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen von Fan für Bewegung
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Gann Fan                           |
//+------------------------------------------------------------------+
bool GannFanPointChange(const long   chart_ID=0,     // ID des Charts
                        const string name="GannFan", // Name
                        const int    point_index=0,  // Nummer des Punktes
                        datetime     time=0,         // Zeitkoordinate des Punktes
                        double       price=0)        // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ankerpunkt von Fan bewegen
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     ;}
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Maßstab von "Gann Fan"                                |
//+------------------------------------------------------------------+
bool GannFanScaleChange(const long   chart_ID=0,     // ID des Charts
                        const string name="GannFan", // Name
                        const double scale=1.0)      // Maßstab
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern wir den Maßstab (Anzahl der Pips pro Bar)
   if(!ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale))
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Maßstab ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Trendrichtung von "Gann Fan"                          |
//+------------------------------------------------------------------+
bool GannFanDirectionChange(const long   chart_ID=0,      // ID des Charts
                            const string name="GannFan", // Name
                            const bool   direction=true) // Trendrichtung
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern wir die Trendrichtung von "Gann Fan"
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_DIRECTION,direction))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Trendrichtung ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion löscht das Objekt "Gann Fan"                        |
//+------------------------------------------------------------------+
bool GannFanDelete(const long   chart_ID=0,      // ID des Charts
                   const string name="GannFan") // Name
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Gann Fan löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Gann Fan\" konnte nicht gelöscht werden! Fehlercode = ",GetLa
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Gann Fan                |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void ChangeGannFanEmptyPoints(datetime &time1,double &price1,datetime &time2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
    ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten von Fan verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
    ;}
//--- Füllen den Array der Preise
```

© 2000-2025, MetaQuotes Ltd.

```
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Gann Fan zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
//--- Gann Fan erstellen
   if(!GannFanCreate(0,InpName,0,date[d1],price[p1],date[d2],InpScale,InpDirection,
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir den Ankerpunkt von Fan
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- Wir bewegen den ersten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1<accuracy-1)
         p1+=1;
      //--- den Punkt bewegen
      if(!GannFanPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Ändern wir die Trendrichtung von "Gann Fan" zu absteigend
   GannFanDirectionChange(0,InpName,true);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Fan aus dem Chart löschen
   GannFanDelete(0,InpName);
   ChartRedraw();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
 }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_GANNGRID

Gann Gitter.



Hinweis

Für "Gann Gitter" können Sie den Trendtyp aus der Enumeration ENUM_GANN_DIRECTION angeben. Durch die Anpassung des Maßstabs (OBJPROP_SCALE), können Sie den Winkel der Gitterlinien ändern.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Gann Gitter". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Gann Gitter\"."
#property description "Koordinaten der Gitterankerpunkten werden in Prozente"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="GannGrid";        // Gittername
input int             InpDate1=15;               // Datum des ersten Punktes in %
input int             InpPrice1=25;              // Preis des ersten Punktes in %
input int             InpDate2=35;               // Datum des zweiten Punktes in %
input double          InpScale=3.0;              // Maßstab
input bool            InpDirection=false;        // Trendrichtung
```

© 2000-2025, MetaQuotes Ltd.

```
input color           InpColor=clrRed;           // Gitterfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Gitterlinienstil
input int             InpWidth=1;                // Breite der Fanlinien
input bool            InpBack=false;             // Das Gitter im Hintergrund
input bool            InpSelection=true;         // Wählen um zu bewegen
input bool            InpHidden=true;            // Ausgeblendet in der Objektliste
input long            InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt "Gann Gitter"                                           |
//+------------------------------------------------------------------+
bool GannGridCreate(const long            chart_ID=0,        // ID des Charts
                    const string          name="GannGrid",   // Gittername
                    const int             sub_window=0,      // Nummer des Fensters
                    datetime              time1=0,           // Zeit des ersten Punkte
                    double                price1=0,          // Preis des ersten Punkt
                    datetime              time2=0,           // Zeit des zweiten Punkt
                    const double          scale=1.0,         // Maßstab
                    const bool            direction=true,    // Trendrichtung
                    const color           clr=clrRed,        // Gitterfarbe
                    const ENUM_LINE_STYLE style=STYLE_SOLID, // Gitterlinienstil
                    const int             width=1,           // Gitterlinienbreite
                    const bool            back=false,        // In der Hintergrund
                    const bool            selection=true,    // Wählen um zu bewegen
                    const bool            hidden=true,       // Verborgen in der Liste
                    const long            z_order=0)         // Priorität für Mausklic
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeGannGridEmptyPoints(time1,price1,time2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Gann Gitter" auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_GANNGRID,sub_window,time1,price1,time2,0))
     {
      Print(__FUNCTION__,
            ": Objekt \" Gann Gitter\" konnte nicht erstellt werden! Fehlercode = ",Ge
      return(false);
     }
//--- Ändern wir den Maßstab (Anzahl der Pips pro Bar)
   ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale);
//--- Ändern wir die Richtung des Trends von "Gann Gitter" (true - absteigend, falsch
   ObjectSetInteger(chart_ID,name,OBJPROP_DIRECTION,direction);
//--- Gitterfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Gitterlinienstil angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Gitterlinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- aktivieren (true) oder deaktivieren (false) Wählen des Gitters für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Gann Gitter                        |
//+------------------------------------------------------------------+
bool GannGridPointChange(const long   chart_ID=0,      // ID des Charts
                         const string name="GannGrid", // Gittername
                         const int    point_index=0,   // Nummer des Ankerpunktes
                         datetime     time=0,          // Zeitkoordinate des Punktes
                         double       price=0)         // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Gitterankerpunkt bewegen
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Maßstab von "Gann Gitter"                             |
//+------------------------------------------------------------------+
bool GannGridScaleChange(const long   chart_ID=0,      // ID des Charts
                         const string name="GannGrid", // Gitter
                         const double scale=1.0)       // Maßstab
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern wir den Maßstab (Anzahl der Pips pro Bar)
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Maßstab ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Trendrichtung von "Gann Gitter"                       |
//+------------------------------------------------------------------+
bool GannGridDirectionChange(const long   chart_ID=0,      // ID des Charts
                             const string name="GannGrid", // Gittername
                             const bool   direction=true)  // Trendrichtung
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern wir die Trendrichtung von "Gann Gitter"
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_DIRECTION,direction))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Trendrichtung ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion löscht das Objekt "Gann Gitter"                     |
//+------------------------------------------------------------------+
bool GannGridDelete(const long   chart_ID=0,      // ID des Charts
                    const string name="GannGrid") // Gittername
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Gann Gitter löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Gann Gitter\" konnte nicht gelöscht werden! Fehlercode = ",Ge
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Gann Gitter             |
//| und setzt Standardwerte für leere Werte                          |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void ChangeGannGridEmptyPoints(datetime &time1,double &price1,datetime &time2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
    ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten des Gitters verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Gann Gitter zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
//--- Gann Gitter erstellen
   if(!GannGridCreate(0,InpName,0,date[d1],price[p1],date[d2],InpScale,InpDirection,
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Gitterankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy/4;
//--- Wir bewegen den ersten Bezugspunkt vertikal
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1<accuracy-1)
         p1+=1;
      if(!GannGridPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int h_steps=bars/4;
//--- den zweiten Punkt horizontal bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2<bars-1)
         d2+=1;
      if(!GannGridPointChange(0,InpName,1,date[d2],0))
```

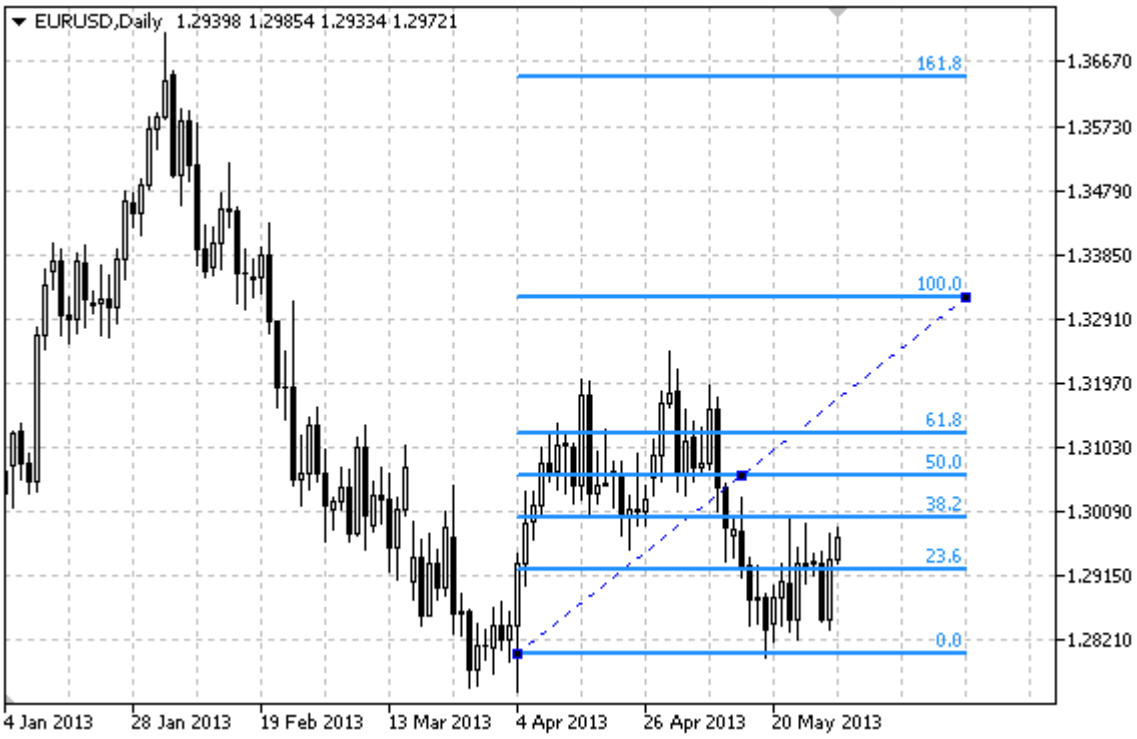
© 2000-2025, MetaQuotes Ltd.

```
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Ändern wir die Trendrichtung des Gitters zu absteigend
   GannGridDirectionChange(0,InpName,true);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Gitter aus dem Chart löschen
   GannGridDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_FIBO

Fibonacci Levels.



Hinweis

Für das Objekt "Fibonacci Levels" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Sie können auch die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Fibonacci Levels". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Fibonacci Levels\"
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="FiboLevels";      // Objektname
input int             InpDate1=10;               // Datum des ersten Punktes in %
input int             InpPrice1=65;              // Preis des ersten Punktes in %
input int             InpDate2=90;               // Datum des zweiten Punktes in %
input int             InpPrice2=85;              // Preis des zweiten Punktes in %
input color           InpColor=clrRed;           // Objektfarbe
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Linienstil
input int             InpWidth=2;                // Linienbreite
input bool            InpBack=false;             // Objekt im Hintergrund
input bool            InpSelection=true;         // Wählen um zu bewegen
input bool            InpRayLeft=false;          // Fortsetzung des Objekts nach links
input bool            InpRayRight=false;         // Fortsetzung des Objekts nach recht
input bool            InpHidden=true;            // Ausgeblendet in der Objektliste
input long            InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Objekt Fibonacci Levels auf angegebenen Koordinaten     |
//+------------------------------------------------------------------+
bool FiboLevelsCreate(const long            chart_ID=0,        // ID des Charts
                      const string          name="FiboLevels", // Objektname
                      const int             sub_window=0,      // Nummer des Unterfens
                      datetime              time1=0,           // Zeit des ersten Punk
                      double                price1=0,          // Preis des ersten Pun
                      datetime              time2=0,           // Zeit des zweiten Pun
                      double                price2=0,          // Preis des zweiten Pu
                      const color           clr=clrRed,        // Objektfarbe
                      const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil des Objek
                      const int             width=1,           // Breite der Objektlin
                      const bool            back=false,        // Im Hintergrund
                      const bool            selection=true,    // Wählen um zu bewegen
                      const bool            ray_left=false,    // Fortsetzung des Obje
                      const bool            ray_right=false,   // Fortsetzung des Obje
                      const bool            hidden=true,       // Ausgeblendet in der
                      const long            z_order=0)         // Priorität auf Mausk
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeFiboLevelsEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Fibonacci Levels auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_FIBO,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Levels\" konnte nicht erstellt werden! Fehlercode =
      return(false);
     }
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Objekts für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
```

© 2000-2025, MetaQuotes Ltd.

```
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Objekts nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Objekts nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Ebenen und ihren Parameter an                |
//+------------------------------------------------------------------+
bool FiboLevelsSet(int              levels,            // Anzahl der Ebenelinien
                   double           &values[],         // Werte der Linien
                   color            &colors[],         // Farbe der Linien
                   ENUM_LINE_STYLE  &styles[],         // Linienstil
                   int              &widths[],         // Breite der Linien
                   const long       chart_ID=0,        // ID des Charts
                   const string     name="FiboLevels") // Objektname
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) || levels!=ArraySize(widt
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Ebenen, F
      return(false);
     }
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,DoubleToString(100*values[i],1
     }
//--- die erfolgreiche Umsetzung
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Fibonacci Levels                   |
//+------------------------------------------------------------------+
bool FiboLevelsPointChange(const long   chart_ID=0,        // ID des Charts
                           const string name="FiboLevels", // Objektname
                           const int    point_index=0,     // Nummer des Punktes
                           datetime     time=0,            // Zeitkoordinate des Punkt
                           double       price=0)           // Preiskoordinate des Punk
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Fibonacci Levels"                             |
//+------------------------------------------------------------------+
bool FiboLevelsDelete(const long   chart_ID=0,        // ID des Charts
                      const string name="FiboLevels") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Levels\" konnte nicht gelöscht werden! Fehlercode =
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Fibonacci Levels und    |
//| setzt Standardwerte für leere Werte                              |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void ChangeFiboLevelsEmptyPoints(datetime &time1,double &price1,
                                 datetime &time2,double &price2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price2)
      price2=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, bewegen wir ihn 200 Punkt
   if(!price1)
      price1=price2-200*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten von Fibonacci Levels verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
       return;
      }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Fibonacci Levels zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- Objekt erstellen
   if(!FiboLevelsCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden,InpZOrde
      {
       return;
      }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy*2/5;
//--- den ersten Punkt bewegen
   for(int i=0;i<v_steps;i++)
      {
       //--- den nächsten Wert nehmen
       if(p1>1)
          p1-=1;
       //--- den Punkt bewegen
       if(!FiboLevelsPointChange(0,InpName,0,date[d1],price[p1]))
          return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
       if(IsStopped())
          return;
       //--- den Chart neu zeichnen
       ChartRedraw();
      }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy*4/5;
//--- den zweiten Punkt bewegen
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p2>1)
         p2-=1;
      //--- den Punkt bewegen
      if(!FiboLevelsPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   FiboLevelsDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_FIBOTIMES

Fibonacci Zeitzonen.



Hinweis

Für "Fibonacci Zeitzonen" können Sie die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Fibonacci Zeitzonen". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Fibonacci Zeitzone
#property description "Koordinaten der Ankerpunkten werden in Pixeln angegeben"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="FiboTimes";       // Objektname
input int             InpDate1=10;               // Datum des ersten Punktes in %
input int             InpPrice1=45;              // Preis des ersten Punktes in %
input int             InpDate2=20;               // Datum des zweiten Punktes in %
input int             InpPrice2=55;              // Preis des zweiten Punktes in %
input color           InpColor=clrRed;           // Objektfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Linienstil
input int             InpWidth=2;                // Linienbreite
input bool            InpBack=false;             // Objekt im Hintergrund
```

© 2000-2025, MetaQuotes Ltd.

```
input bool             InpSelection=true;          // Wählen um zu bewegen
input bool             InpHidden=true;             // Ausgeblendet in der Objektliste
input long             InpZOrder=0;                // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Objekt Fibonacci Zeitzonen auf angegebenen Koordinaten  |
//+------------------------------------------------------------------+
bool FiboTimesCreate(const long             chart_ID=0,         // ID des Charts
                     const string           name="FiboTimes",   // Objektname
                     const int              sub_window=0,       // Nummer des Unterfenst
                     datetime               time1=0,            // Zeit des ersten Punkt
                     double                 price1=0,           // Preis des ersten Punk
                     datetime               time2=0,            // Zeit des zweiten Punk
                     double                 price2=0,           // Preis des zweiten Pur
                     const color            clr=clrRed,         // Objektfarbe
                     const ENUM_LINE_STYLE  style=STYLE_SOLID,  // Linienstil des Objekt
                     const int              width=1,            // Breite der Objektlini
                     const bool             back=false,         // In der Hintergrund
                     const bool             selection=true,     // Wählen um zu bewegen
                     const bool             hidden=true,        // Verborgen in der List
                     const long             z_order=0)          // Priorität für Mauskli
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeFiboTimesEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Fibonacci Zeitzonen" auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_FIBOTIMES,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Zeitzonen\" konnte nicht erstellt werden! Fehlercoc
      return(false);
     }
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Objekts für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Ebenen und ihren Parameter an                |
//+------------------------------------------------------------------+
bool FiboTimesLevelsSet(int             levels,           // Anzahl der Ebenelinien
                        double          &values[],        // Werte der Linien
                        color           &colors[],        // Farbe der Linien
                        ENUM_LINE_STYLE &styles[],        // Linienstil
                        int             &widths[],        // Linienbreite
                        const long      chart_ID=0,       // ID des Charts
                        const string    name="FiboTimes") // Objektname
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) ||
      levels!=ArraySize(widths) || levels!=ArraySize(widths))
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Ebenen, F
      return(false);
     }
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,DoubleToString(values[i],1));
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Fibonacci Zeitzonen                |
//+------------------------------------------------------------------+
bool FiboTimesPointChange(const long   chart_ID=0,       // ID des Charts
                          const string name="FiboTimes", // Objektname
                          const int    point_index=0,    // Nummer des Ankerpunktes
                          datetime     time=0,           // Zeitkoordinate des Punktes
                          double       price=0)          // Preiskoordinate des Punkte
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Fibonacci Zeitzonen"                          |
//+------------------------------------------------------------------+
bool FiboTimesDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="FiboTimes") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Zeitzonen\" konnte nicht gelöscht werden! Fehlercod
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Prüft die Werte der Ankerpunkten von "Fibonacci Zeitzonen"       |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeFiboTimesEmptyPoints(datetime &time1,double &price1,
                                datetime &time2,double &price2)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 2 Balken nac
   if(!time2)
     {
      //--- Array um die Zeit der Öffnung der letzten 3 Bars zu empfangen
      datetime temp[3];
      CopyTime(Symbol(),Period(),time1,3,temp);
      //--- stellen wir den ersten Punkt 2 Baren links auf dem zweiten Punkt
      time2=temp[0];
     }
//--- Wenn der Preis des zweiten Punktes nicht angegeben wird, dann wird es der Preis
   if(!price2)
      price2=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten von Fibonacci Zeitzonen verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Fibonacci Zeitzonen zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- Objekt erstellen
   if(!FiboTimesCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int h_steps=bars*2/5;
//--- den zweiten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2<bars-1)
         d2+=1;
      //--- den Punkt bewegen
      if(!FiboTimesPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   h_steps=bars*3/5;
//--- den ersten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d1<bars-1)
         d1+=1;
      //--- den Punkt bewegen
      if(!FiboTimesPointChange(0,InpName,0,date[d1],price[p1]))
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   FiboTimesDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_FIBOFAN

Fibonacci-Fan.



Hinweis

Für "Fibonacci Fan" können Sie die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Fibonacci-Fan". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Fibonacci-Fan\"."
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="FiboFan";          // Name
input int             InpDate1=10;                // Datum des ersten Punktes in %
input int             InpPrice1=25;               // Preis des ersten Punktes in %
input int             InpDate2=30;                // Datum des zweiten Punktes in %
input int             InpPrice2=50;               // Preis des zweiten Punktes in %
input color           InpColor=clrRed;            // Linienfarbe
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Linienstil
input int             InpWidth=2;                 // Linienbreite
```

© 2000-2025, MetaQuotes Ltd.

```
input bool             InpBack=false;              // Objekt im Hintergrund
input bool             InpSelection=true;          // Wählen um zu bewegen
input bool             InpHidden=true;             // Ausgeblendet in der Objektliste
input long             InpZOrder=0;                // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Objekt Fibonacci-Fan auf angegebenen Koordinaten        |
//+------------------------------------------------------------------+
bool FiboFanCreate(const long             chart_ID=0,        // ID des Charts
                   const string           name="FiboFan",    // Name von Fan
                   const int              sub_window=0,      // Nummer des Unterfenster
                   datetime               time1=0,           // Zeit des ersten Punktes
                   double                 price1=0,          // Preis des ersten Punkte
                   datetime               time2=0,           // Zeit des zweiten Punkte
                   double                 price2=0,          // Preis des zweiten Punkt
                   const color            clr=clrRed,        // Fan Linienfarbe
                   const ENUM_LINE_STYLE  style=STYLE_SOLID, // Fan Linienstil
                   const int              width=1,           // Fan Linienbreite
                   const bool             back=false,        // Im Hintergrund
                   const bool             selection=true,    // Auswählen um zu beweger
                   const bool             hidden=true,       // Ausgeblendet in der Obj
                   const long             z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeFiboFanEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Fibonacci Fan" auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_FIBOFAN,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Fan\" konnte nicht erstellt werden! Fehlercode = ",
      return(false);
     }
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen von Fan für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Ebenen und ihren Parameter an                |
//+------------------------------------------------------------------+
bool FiboFanLevelsSet(int             levels,         // Anzahl der Ebenelinien
                      double          &values[],      // Werte der Linien
                      color           &colors[],      // Farbe der Ebenelinien
                      ENUM_LINE_STYLE &styles[],      // Stil der Ebenelinien
                      int             &widths[],      // Breite der Linien
                      const long      chart_ID=0,     // ID des Charts
                      const string    name="FiboFan") // Name von Fan
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) ||
      levels!=ArraySize(widths) || levels!=ArraySize(widths))
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Ebenen, F
      return(false);
     }
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,DoubleToString(100*values[i],1
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Fibonacci Fan                      |
//+------------------------------------------------------------------+
bool FiboFanPointChange(const long   chart_ID=0,     // ID des Charts
                        const string name="FiboFan", // Name von Fan
                        const int    point_index=0,  // Nummer des Punktes
                        datetime     time=0,         // Zeitkoordinate des Punktes
```

© 2000-2025, MetaQuotes Ltd.

```
                        double       price=0)        // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Löscht das Objekt "Fibonacci Fan"                                |
//+------------------------------------------------------------------+
bool FiboFanDelete(const long   chart_ID=0,     // ID des Charts
                   const string name="FiboFan") // Name von Fan
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Fan löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Fan\" konnte nicht gelöscht werden! Fehlercode = ",
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Fibonacci-Fan und       |
//| setzt Standardwerte für leere Werte                              |
//+------------------------------------------------------------------+
void ChangeFiboFanEmptyPoints(datetime &time1,double &price1,
                              datetime &time2,double &price2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price2)
```

© 2000-2025, MetaQuotes Ltd.

```
      price2=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, bewegen wir ihn 200 Punkt
   if(!price1)
      price1=price2-200*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
    ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten von Fibonacci-Fan verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
```

© 2000-2025, MetaQuotes Ltd.

```
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Fibonacci-Fan zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- Objekt erstellen
   if(!FiboFanCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte von Fan
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- den ersten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1<accuracy-1)
         p1+=1;
      //--- den Punkt bewegen
      if(!FiboFanPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int h_steps=bars/4;
//--- den zweiten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2<bars-1)
         d2+=1;
      //--- den Punkt bewegen
      if(!FiboFanPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
```

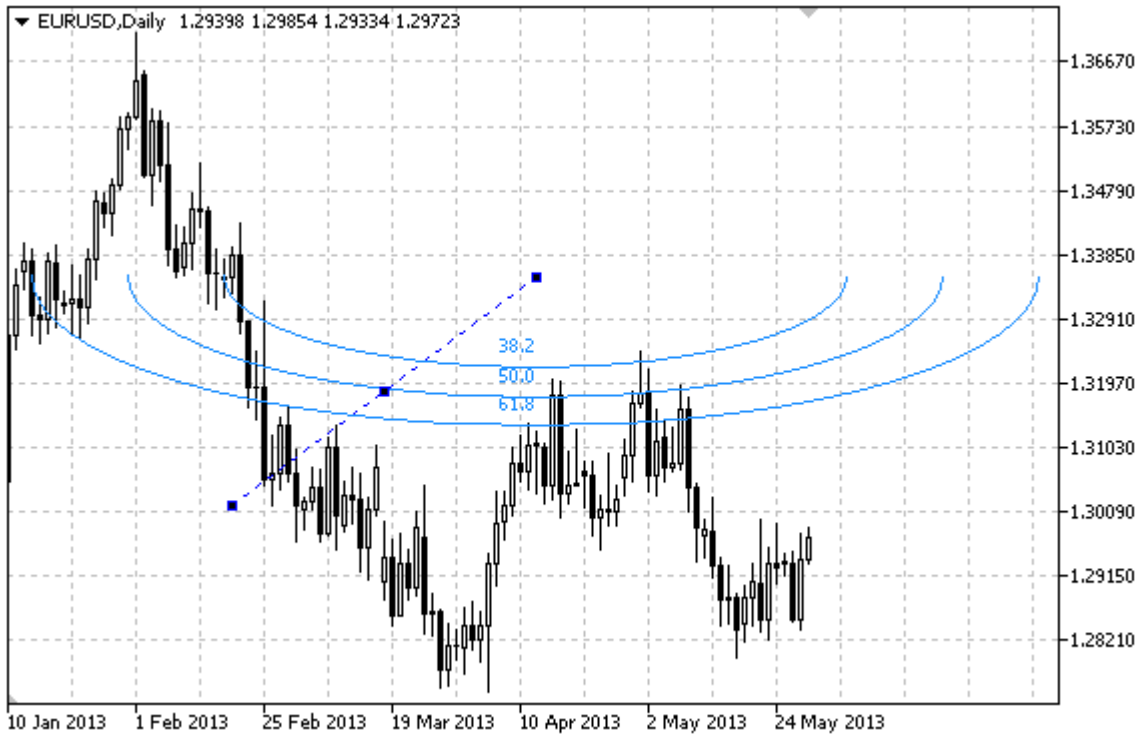
© 2000-2025, MetaQuotes Ltd.

```
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   FiboFanDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
 }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_FIBOARC

Fibonacci Arcs.



Hinweis

Für "Fibonacci Arcs" können Sie Anzeigemodus der gesamten Ellipse angeben. Der Krümmungsradius der Linien kann durch Änderung des Maßstabs und Koordinaten der Ankerpunkten gesetzt werden.

Sie können auch die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Fibonacci Arcs". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Fibonacci Arcs\".'
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="FiboArc";         // Objektname
input int             InpDate1=25;               // Datum des ersten Punktes in %
input int             InpPrice1=25;              // Preis des ersten Punktes in %
input int             InpDate2=35;               // Datum des zweiten Punktes in %
input int             InpPrice2=55;              // Preis des zweiten Punktes in %
input double          InpScale=3.0;              // Maßstab
```

© 2000-2025, MetaQuotes Ltd.

```
input bool             InpFullEllipse=true;       // Form von Bögen
input color            InpColor=clrRed;           // Linienfarbe
input ENUM_LINE_STYLE  InpStyle=STYLE_DASHDOTDOT; // Linienstil
input int              InpWidth=2;                // Linienbreite
input bool             InpBack=false;             // Objekt im Hintergrund
input bool             InpSelection=true;         // Wählen um zu bewegen
input bool             InpHidden=true;            // Ausgeblendet in der Objektliste
input long             InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Objekt Fibonacci Arcs auf angegebenen Koordinaten       |
//+------------------------------------------------------------------+
bool FiboArcCreate(const long            chart_ID=0,          // ID des Charts
                   const string          name="FiboArc",      // Objektname
                   const int             sub_window=0,        // Nummer des Unterfenste
                   datetime              time1=0,             // Zeit des ersten Punkte
                   double                price1=0,            // Preis des ersten Punkt
                   datetime              time2=0,             // Zeit des zweiten Punkt
                   double                price2=0,            // Preis des zweiten Punk
                   const double          scale=1.0,           // Maßstab
                   const bool            full_ellipse=false,  // Form der Bögen
                   const color           clr=clrRed,          // Linienfarbe
                   const ENUM_LINE_STYLE style=STYLE_SOLID,   // Linienstil
                   const int             width=1,             // Linienbreite
                   const bool            back=false,          // Im Hintergrund
                   const bool            selection=true,      // Wählen um zu bewegen
                   const bool            hidden=true,         // Ausgeblendet in der Ob
                   const long            z_order=0)           // Priorität auf Mausklic
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeFiboArcEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Fibonacci Arcs" auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_FIBOARC,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Arcs\" konnte nicht erstellt werden! Fehlercode = "
      return(false);
     ;}
//--- die Skala setzen
   ObjectSetDouble(chart_ID,name,OBJPROP_SCALE,scale);
//--- Anzeige der Bögen als volle Ellipse (true) oder Hälfte (false)
   ObjectSetInteger(chart_ID,name,OBJPROP_ELLIPSE,full_ellipse);
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen der Bögen für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
 }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Ebenen und ihren Parameter an                |
//+------------------------------------------------------------------+
bool FiboArcLevelsSet(int             levels,         // Anzahl der Ebenelinien
                      double          &values[],      // Werte der Linien
                      color           &colors[],      // Farbe der Ebenelinien
                      ENUM_LINE_STYLE &styles[],      // Stil der Ebenelinien
                      int             &widths[],      // Breite der Linien
                      const long      chart_ID=0,     // ID des Charts
                      const string    name="FiboArc") // Objektname
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) ||
      levels!=ArraySize(widths) || levels!=ArraySize(widths))
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Ebenen, F
      return(false);
    ;}
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,DoubleToString(100*values[i],1
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Fibonacci Arcs                     |
//+------------------------------------------------------------------+
bool FiboArcPointChange(const long   chart_ID=0,     // ID des Charts
                        const string name="FiboArc", // Objektname
                        const int    point_index=0,  // Nummer des Punktes
                        datetime     time=0,         // Zeitkoordinate des Punktes
                        double       price=0)        // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Fibonacci Arcs"                               |
//+------------------------------------------------------------------+
bool FiboArcDelete(const long   chart_ID=0,     // ID des Charts
                   const string name="FiboArc") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Arcs\" konnte nicht gelöscht werden! Fehlercode = "
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Fibonacci Arcs          |
```

© 2000-2025, MetaQuotes Ltd.

```
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeFiboArcEmptyPoints(datetime &time1,double &price1,
                              datetime &time2,double &price2)
  {
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time2)
      time2=TimeCurrent();
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price2)
      price2=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, bewegen wir ihn 300 Punkt
   if(!price1)
      price1=price2-300*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten von Fibonacci Arcs verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Fibonacci Arcs zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- Objekt erstellen
   if(!FiboArcCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],InpScale,
      InpFullEllipse,InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrd
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy/5;
//--- den ersten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1<accuracy-1)
         p1+=1;
      //--- den Punkt bewegen
      if(!FiboArcPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int h_steps=bars/5;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- den zweiten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2<bars-1)
         d2+=1;
      //--- den Punkt bewegen
      if(!FiboArcPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   FiboArcDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_FIBOCHANNEL

Fibonacci-Kanal.



Hinweis

Für das Objekt "Fibonacci-Kanal" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Sie können auch die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Fibonacci-Kanal". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Fibonacci-Kanal\"."
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="FiboChannel";     // Kanalname
input int             InpDate1=20;               // Datum des ersten Punktes in %
input int             InpPrice1=10;              // Preis des ersten Punktes in %
input int             InpDate2=60;               // Datum des zweiten Punktes in %
input int             InpPrice2=30;              // Preis des zweiten Punktes in %
input int             InpDate3=20;               // Datum des dritten Punktes in %
```

© 2000-2025, MetaQuotes Ltd.

```
input int              InpPrice3=25;              // Preis des dritten Punktes in %
input color            InpColor=clrRed;           // Kanalfarbe
input ENUM_LINE_STYLE  InpStyle=STYLE_DASHDOTDOT; // Stil der Kanallinien
input int              InpWidth=2;                // Kanallinienbreite
input bool             InpBack=false;             // Im Hintergrund
input bool             InpSelection=true;         // Wählen um zu bewegen
input bool             InpRayLeft=false;          // Fortsetzung des Kanals nach links
input bool             InpRayRight=false;         // Fortsetzung des Kanals nach rechts
input bool             InpHidden=true;            // Ausgeblendet in der Objektliste
input long             InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Objekt Fibonacci-Kanal auf angegebenen Koordinaten      |
//+------------------------------------------------------------------+
bool FiboChannelCreate(const long             chart_ID=0,         // ID des Charts
                       const string           name="FiboChannel", // Kanalname
                       const int              sub_window=0,       // Nummer des Unterfe
                       datetime               time1=0,            // Zeit des ersten Pu
                       double                 price1=0,           // Preis des ersten P
                       datetime               time2=0,            // Zeit des zweiten P
                       double                 price2=0,           // Preis des zweiten
                       datetime               time3=0,            // Zeit des dritten P
                       double                 price3=0,           // Preis des dritten
                       const color            clr=clrRed,         // Kanalfarbe
                       const ENUM_LINE_STYLE  style=STYLE_SOLID,  // Stil der Kanallini
                       const int              width=1,            // Breite der Kanalli
                       const bool             back=false,         // Im Hintergrund
                       const bool             selection=true,     // Wählen zu bewegen
                       const bool             ray_left=false,     // Fortsetzung des Ka
                       const bool             ray_right=false,    // Fortsetzung des Ka
                       const bool             hidden=true,        // Verborgen in der I
                       const long             z_order=0)          // Priorität für Maus
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeFiboChannelEmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Kanal auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_FIBOCHANNEL,sub_window,time1,price1,time2,price2
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci-Kanal\" konnte nicht erstellt werden! Fehlercode =
      return(false);
     }
//--- Kanalfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Kanallinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen des Kanals für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Kanals nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Ebenen und ihren Parameter an                |
//+------------------------------------------------------------------+
bool FiboChannelLevelsSet(int              levels,              // Anzahl der Ebenelinie
                          double           &values[],           // Werte der Linien
                          color            &colors[],           // Farbe der Linien
                          ENUM_LINE_STYLE  &styles[],           // Stil der Linien
                          int              &widths[],           // Linienbreite
                          const long       chart_ID=0,          // ID des Charts
                          const string     name="FiboChannel") // Objektname
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) ||
      levels!=ArraySize(widths) || levels!=ArraySize(widths))
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Ebenen, F
      return(false);
     }
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
```

© 2000-2025, MetaQuotes Ltd.

```
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,DoubleToString(100*values[i],1
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Fibonacci-Kanal                    |
//+------------------------------------------------------------------+
bool FiboChannelPointChange(const long   chart_ID=0,          // ID des Charts
                            const string name="FiboChannel", // Kanalname
                            const int    point_index=0,      // Nummer des Bezugspunkt
                            datetime     time=0,             // Zeitkoordinate des Bez
                            double       price=0)            // Preiskoordinate des Be
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht den Kanal                                                 |
//+------------------------------------------------------------------+
bool FiboChannelDelete(const long   chart_ID=0,          // ID des Charts
                       const string name="FiboChannel") // Kanalname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- den Kanal löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci-Kanal\" konnte nicht gelöscht werden! Fehlercode =
      return(false);
     }
//--- die erfolgreiche Umsetzung
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Fibonacci-Kanal und     |
//| setzt Standardwerte für leere Werte                              |
//+------------------------------------------------------------------+
void ChangeFiboChannelEmptyPoints(datetime &time1,double &price1,datetime &time2,
                                  double &price2,datetime &time3,double &price3)
  {
//--- wenn die Zeit des zweiten (rechten) Punktes nicht angegeben ist, wird sie auf de
   if(!time2)
      time2=TimeCurrent();
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price2)
      price2=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten (linken) Punktes nicht angegeben ist, wird sie auf 9 Ba
   if(!time1)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time2,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, bewegen wir ihn 300 Punkt
   if(!price1)
      price1=price2+300*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird sie der Zeit des zwe
   if(!time3)
      time3=time1;
//--- wenn der Preis des dritten Punktes nicht angegeben ist, wird er dem Preis des zw
   if(!price3)
      price3=price2;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Kanalbindungspunkten verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um den Kanal zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Fibonacci-Kanal erstellen
   if(!FiboChannelCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],pr
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidden
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Kanalbindungspunkte
//--- Schleifenzähler
   int h_steps=bars/10;
//--- den ersten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d1>1)
         d1-=1;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den Punkt bewegen
      if(!FiboChannelPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int v_steps=accuracy/10;
//--- den zweiten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p2>1)
         p2-=1;
      //--- den Punkt bewegen
      if(!FiboChannelPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy/15;
//--- den dritten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p3<accuracy-1)
         p3+=1;
      //--- den Punkt bewegen
      if(!FiboChannelPointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
```

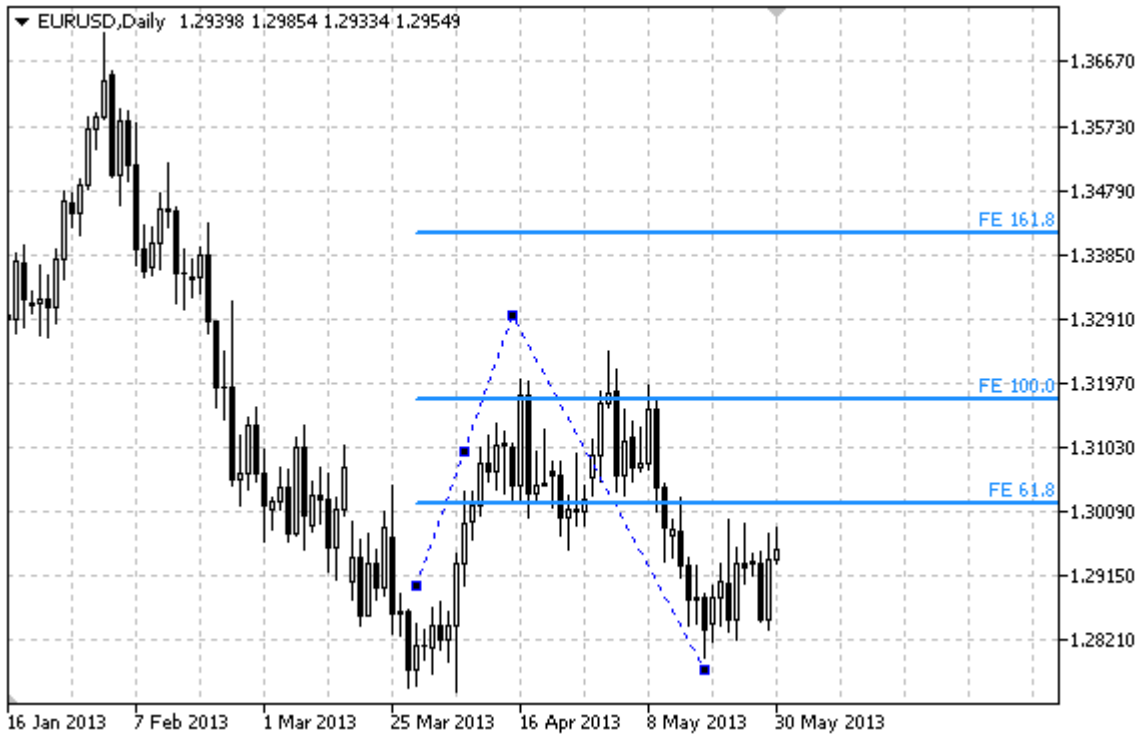
© 2000-2025, MetaQuotes Ltd.

```
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Kanal aus dem Chart löschen
   FiboChannelDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_EXPANSION

Fibonacci Expansion.



Hinweis

Für das Objekt "Fibonacci Expansion" können Sie die weitere Anzeige nach rechts und/oder nach links (Eigenschaften OBJPROP_RAY_RIGHT und OBJPROP_RAY_LEFT jeweils) angeben.

Sie können auch die Anzahl der Linien-Ebenen, ihre Werte und Farben angeben.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Fibonacci Expansion". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Fibonacci Expansio
#property description "Koordinaten des Bezugspunktes werden in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="FiboExpansion";   // Objektname
input int             InpDate1=10;               // Datum des ersten Punktes in %
input int             InpPrice1=55;              // Preis des ersten Punktes in %
input int             InpDate2=30;               // Datum des zweiten Punktes in %
input int             InpPrice2=10;              // Preis des zweiten Punktes in %
input int             InpDate3=80;               // Datum des dritten Punktes in %
```

© 2000-2025, MetaQuotes Ltd.

```
input int              InpPrice3=75;              // Preis des dritten Punktes in %
input color            InpColor=clrRed;           // Objektfarbe
input ENUM_LINE_STYLE  InpStyle=STYLE_DASHDOTDOT; // Linienstil
input int              InpWidth=2;                // Linienbreite
input bool             InpBack=false;             // Objekt im Hintergrund
input bool             InpSelection=true;         // Wählen um zu bewegen
input bool             InpRayLeft=false;          // Fortsetzung des Objekts nach links
input bool             InpRayRight=false;         // Fortsetzung des Objekts nach recht
input bool             InpHidden=true;            // Ausgeblendet in der Objektliste
input long             InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Objekt Fibonacci Expansion auf angegebenen Koordinaten  |
//+------------------------------------------------------------------+
bool FiboExpansionCreate(const long             chart_ID=0,           // ID des Charts
                         const string           name="FiboExpansion", // Name
                         const int              sub_window=0,         // Nummer des Unt
                         datetime               time1=0,              // Zeit des erste
                         double                 price1=0,             // Preis des erst
                         datetime               time2=0,              // Zeit des zweit
                         double                 price2=0,             // Preis des zwei
                         datetime               time3=0,              // Zeit des dritt
                         double                 price3=0,             // Preis des drit
                         const color            clr=clrRed,           // Objektfarbe
                         const ENUM_LINE_STYLE  style=STYLE_SOLID,    // Linienstil
                         const int              width=1,              // Linienbreite
                         const bool             back=false,           // Im Hintergrund
                         const bool             selection=true,       // Wählen um zu b
                         const bool             ray_left=false,       // Fortsetzung de
                         const bool             ray_right=false,      // Fortsetzung de
                         const bool             hidden=true,          // Ausgeblendet i
                         const long             z_order=0)            // Priorität auf
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeFiboExpansionEmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- "Fibonacci Expansion" auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_EXPANSION,sub_window,time1,price1,time2,price2,t
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Expansion\" konnte nicht erstellt werden! Fehlercod
      return(false);
     }
//--- Objektfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Objekts für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Objekts nach links
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_LEFT,ray_left);
//--- aktivieren (true) oder deaktivieren (false) Fortsetzung des Objekts nach rechts
   ObjectSetInteger(chart_ID,name,OBJPROP_RAY_RIGHT,ray_right);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Ebenen und ihren Parameter an                |
//+------------------------------------------------------------------+
bool FiboExpansionLevelsSet(int              levels,                // Anzahl der Ebenel
                            double           &values[],             // Werte der Linien
                            color            &colors[],             // Farbe der Linien
                            ENUM_LINE_STYLE  &styles[],             // Linienstil
                            int              &widths[],             // Breite der Linier
                            const long       chart_ID=0,            // ID des Charts
                            const string     name="FiboExpansion") // Objektname
  {
//--- Überprüfen wir die Größe des Arrays
   if(levels!=ArraySize(colors) || levels!=ArraySize(styles) ||
      levels!=ArraySize(widths) || levels!=ArraySize(widths))
     {
      Print(__FUNCTION__,": Länge des Arrays entspricht nicht der Anzahl der Ebenen, F
      return(false);
     }
//--- Die Anzahl der Ebenen angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_LEVELS,levels);
//--- die Eigenschaften der Ebenen im Zyklus angeben
   for(int i=0;i<levels;i++)
     {
      //--- Wert der Ebene
      ObjectSetDouble(chart_ID,name,OBJPROP_LEVELVALUE,i,values[i]);
      //--- Farbe der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELCOLOR,i,colors[i]);
      //--- Stil der Ebene
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELSTYLE,i,styles[i]);
      //--- Stärke der Ebene
```

© 2000-2025, MetaQuotes Ltd.

```
      ObjectSetInteger(chart_ID,name,OBJPROP_LEVELWIDTH,i,widths[i]);
      //--- Beschreibung der Ebene
      ObjectSetString(chart_ID,name,OBJPROP_LEVELTEXT,i,"FE "+DoubleToString(100*value
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von Fibonacci Expansion                |
//+------------------------------------------------------------------+
bool FiboExpansionPointChange(const long   chart_ID=0,           // ID des Charts
                              const string name="FiboExpansion", // Objektname
                              const int    point_index=0,        // Nummer des Punktes
                              datetime     time=0,               // Zeitkoordinate des
                              double       price=0)              // Preiskoordinate de
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Fibonacci Expansion"                          |
//+------------------------------------------------------------------+
bool FiboExpansionDelete(const long   chart_ID=0,           // ID des Charts
                         const string name="FiboExpansion") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \" Fibonacci Expansion\" konnte nicht gelöscht werden! Fehlercod
      return(false);
     ;}
//--- die erfolgreiche Umsetzung
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Überprüft die Werte der Ankerpunkten von Fibonacci Expansion und |
//| setzt Standardwerte für leere Werte                              |
//+------------------------------------------------------------------+
void ChangeFiboExpansionEmptyPoints(datetime &time1,double &price1,datetime &time2,
                                    double &price2,datetime &time3,double &price3)
  {
//--- wenn die Zeit des dritten (rechten) Punktes nicht angegeben ist, wird sie auf de
   if(!time3)
      time3=TimeCurrent();
//--- wenn der Preis des dritten Punktes nicht angegeben ist, wird er dem Bid-Wert gle
   if(!price3)
      price3=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des ersten (linken) Punktes nicht angegeben ist, wird sie auf 9 Ba
//--- Array um die Zeit der Öffnung der letzten 10 Bars zu empfangen
   datetime temp[];
   ArrayResize(temp,10);
   if(!time1)
     {
      CopyTime(Symbol(),Period(),time3,10,temp);
      //--- stellen wir den ersten Punkt 9 Baren links dem zweiten Punkt
      time1=temp[0];
     }
//--- wenn der Preis des ersten Punktes nicht angegeben ist, wird er dem Preis des dri
   if(!price1)
      price1=price3;
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 7 Baren von
   if(!time2)
      time2=temp[2];
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, bewegen wir ihn 250 Punk
   if(!price2)
      price2=price1-250*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
```

© 2000-2025, MetaQuotes Ltd.

```
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten des Objekts verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um Fibonacci Expansion zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Fibonacci Expansion erstellen
   if(!FiboExpansionCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],
      InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpRayLeft,InpRayRight,InpHidder
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy/10;
//--- den ersten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
```

© 2000-2025, MetaQuotes Ltd.

```
         p1-=1;
      //--- den Punkt bewegen
      if(!FiboExpansionPointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy/2;
//--- den dritten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p3>1)
         p3-=1;
      //--- den Punkt bewegen
      if(!FiboExpansionPointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy*4/5;
//--- den zweiten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p2<accuracy-1)
         p2+=1;
      //--- den Punkt bewegen
      if(!FiboExpansionPointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
```

© 2000-2025, MetaQuotes Ltd.

```
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   FiboExpansionDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ELLIOTWAVE5

Elliott Imoulswelle.



Hinweis

Für "Elliott Impulswelle" können Sie Zeichnung der Linien zwischen den Punkten (Eigenschaft OBJPROP_DRAWLINES) aktivieren/deaktivieren, und auch die Höhe der Welle Kennzeichnung (Enumeration ENUM_ELLIOT_WAVE_DEGREE) setzen.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Elliott Impulswelle". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Elliott Impulswell
#property description "Koordinaten der Ankerpunkten werden in Pixeln angegeben"
#property description "Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string                InpName="ElliotWave5";   // Objektname
input int                   InpDate1=10;             // Datum des ersten Punktes in
input int                   InpPrice1=90;            // Preis des ersten Punktes in
input int                   InpDate2=20;             // Datum des zweiten Punktes ir
input int                   InpPrice2=40;            // Preis des zweiten Punktes ir
input int                   InpDate3=30;             // Datum des dritten Punktes ir
```

© 2000-2025, MetaQuotes Ltd.

```
input int                      InpPrice3=60;            // Preis des dritten Punktes ir
input int                      InpDate4=40;             // Datum des vierten Punktes ir
input int                      InpPrice4=10;            // Preis des vierten Punktes ir
input int                      InpDate5=60;             // Datum des fünften Punktes ir
input int                      InpPrice5=40;            // Preis des fünften Punktes ir
input ENUM_ELLIOT_WAVE_DEGREE InpDegree=ELLIOTT_MINOR; // Niveau
input bool                     InpDrawLines=true;       // Anzeige der Linien
input color                    InpColor=clrRed;         // Linienfarbe
input ENUM_LINE_STYLE          InpStyle=STYLE_DASH;     // Linienstil
input int                      InpWidth=2;              // Linienbreite
input bool                     InpBack=false;           // Objekt im Hintergrund
input bool                     InpSelection=true;       // Wählen um zu bewegen
input bool                     InpHidden=true;          // Ausgeblendet in der Objektli
input long                     InpZOrder=0;             // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Elliot Impulswelle auf angegebenen Koordinaten          |
//+------------------------------------------------------------------+
bool ElliotWave5Create(const long                  chart_ID=0,              // ID de
                       const string                name="ElliotWave5",      // Name
                       const int                   sub_window=0,            // Numme
                       datetime                    time1=0,                 // Zeit
                       double                      price1=0,                // Preis
                       datetime                    time2=0,                 // Zeit
                       double                      price2=0,                // Preis
                       datetime                    time3=0,                 // Zeit
                       double                      price3=0,                // Preis
                       datetime                    time4=0,                 // Zeit
                       double                      price4=0,                // Preis
                       datetime                    time5=0,                 // Zeit
                       double                      price5=0,                // Preis
                       const ENUM_ELLIOT_WAVE_DEGREE degree=ELLIOTT_MINUETTE, // Grad
                       const bool                  draw_lines=true,         // Anzei
                       const color                 clr=clrRed,              // Objek
                       const ENUM_LINE_STYLE       style=STYLE_SOLID,       // Linie
                       const int                   width=1,                 // Linie
                       const bool                  back=false,              // Im Hi
                       const bool                  selection=true,          // Wähle
                       const bool                  hidden=true,             // Ausge
                       const long                  z_order=0)               // Prior
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeElliotWave5EmptyPoints(time1,price1,time2,price2,time3,price3,time4,price4,ti
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erstellt Elliott Impulswelle auf angegebenen Koordinaten
   if(!ObjectCreate(chart_ID,name,OBJ_ELLIOTWAVE5,sub_window,time1,price1,time2,price2
      price3,time4,price4,time5,price5))
     {
      Print(__FUNCTION__,
```

© 2000-2025, MetaQuotes Ltd.

```
            ": Objekt \"Elliott Impulswelle\" konnte nicht erstellt werden! Fehlercode
      return(false);
     }
//--- Grad (Wellengröße) setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_DEGREE,degree);
//--- aktivieren (true) oder deaktivieren (false) Anzeige der Linien
   ObjectSetInteger(chart_ID,name,OBJPROP_DRAWLINES,draw_lines);
//--- Objektfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Objekts für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von "Elliott Impulswelle"              |
//+------------------------------------------------------------------+
bool ElliotWave5PointChange(const long   chart_ID=0,          // ID des Charts
                            const string name="ElliotWave5", // Objektname
                            const int    point_index=0,      // Nummer des Bezugspunkt
                            datetime     time=0,             // Zeitkoordinate des Bez
                            double       price=0)            // Preiskoordinate des Be
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht "Elliott Impulswelle"                                     |
//+------------------------------------------------------------------+
bool ElliotWave5Delete(const long   chart_ID=0,         // ID des Charts
                       const string name="ElliotWave5") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \"Elliott Impulswelle\" konnte nicht gelöscht werden! Fehlercode
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Prüft die Werte der Ankerpunkten von "Elliott Impulswelle"       |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeElliotWave5EmptyPoints(datetime &time1,double &price1,
                                  datetime &time2,double &price2,
                                  datetime &time3,double &price3,
                                  datetime &time4,double &price4,
                                  datetime &time5,double &price5)
  {
//--- Array um die Zeit der Öffnung der letzten 10 Bars zu empfangen
   datetime temp[];
   ArrayResize(temp,10);
//--- Daten erhalten
   CopyTime(Symbol(),Period(),TimeCurrent(),10,temp);
//--- den Wert von einem Pip auf dem aktuellen Chart erhalten
   double point=SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird er auf 9 Balken auf
   if(!time1)
      time1=temp[0];
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird er auf 7 Balken auf
   if(!time2)
      time2=temp[2];
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, bewegen wir ihn 300 Punk
   if(!price2)
      price2=price1-300*point;
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird er auf 5 Balken auf
   if(!time3)
      time3=temp[4];
//--- wenn der Preis des dritten Punktes nicht angegeben ist, bewegen wir ihn 250 Punk
   if(!price3)
      price3=price1-250*point;
//--- wenn die Zeit des vierten Punktes nicht angegeben ist, wird er auf 3 Balken auf
   if(!time4)
      time4=temp[6];
//--- wenn der Preis des vierten Punktes nicht angegeben ist, bewegen wir ihn 550 Punk
   if(!price4)
      price4=price1-550*point;
//--- wenn die Zeit des fünften Punktes nicht angegeben ist, wird sie auf dem aktuelle
   if(!time5)
      time5=temp[9];
//--- wenn der Preis des fünften Punktes nicht angegeben ist, bewegen wir ihn 450 Punk
   if(!price5)
      price5=price1-450*point;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100 ||
      InpDate4<0 || InpDate4>100 || InpPrice4<0 || InpPrice4>100 ||
      InpDate5<0 || InpDate5>100 || InpPrice5<0 || InpPrice5>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten des Objekts verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um "Elliott Impulswelle" zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int d4=InpDate4*(bars-1)/100;
   int d5=InpDate5*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
   int p4=InpPrice4*(accuracy-1)/100;
   int p5=InpPrice5*(accuracy-1)/100;
//--- Elliott Impulswelle erstellen
   if(!ElliotWave5Create(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],pr
      date[d4],price[p4],date[d5],price[p5],InpDegree,InpDrawLines,InpColor,InpStyle,I
      InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy/5;
//--- den fünften Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p5<accuracy-1)
         p5+=1;
      //--- den Punkt bewegen
      if(!ElliotWave5PointChange(0,InpName,4,date[d5],price[p5]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
```

© 2000-2025, MetaQuotes Ltd.

```
         return;
       //--- den Chart neu zeichnen
       ChartRedraw();
      }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy/5;
//--- den zweiten und dritten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(p2<accuracy-1)
         p2+=1;
      if(p3>1)
         p3-=1;
      //--- Die Punkte bewegen
      if(!ElliotWave5PointChange(0,InpName,1,date[d2],price[p2]))
         return;
      if(!ElliotWave5PointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy*4/5;
//--- den ersten und vierten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(p1>1)
         p1-=1;
      if(p4<accuracy-1)
         p4+=1;
      //--- Die Punkte bewegen
      if(!ElliotWave5PointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!ElliotWave5PointChange(0,InpName,3,date[d4],price[p4]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
   ElliotWave5Delete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ELLIOTWAVE3

Elliott Korrekturwelle.



Hinweis

Für "Elliott Korrekturwelle" können Sie Zeichnung der Linien zwischen den Punkten (Eigenschaft OBJPROP_DRAWLINES) aktivieren/deaktivieren, und auch die Höhe der Welle Kennzeichnung (Enumeration ENUM_ELLIOT_WAVE_DEGREE) setzen.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Elliott Korrekturwelle". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Elliott Korrekturw
#property description "Koordinaten der Ankerpunkten werden als Prozentsatz der Größe d
#property description "des Charts angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string                InpName="ElliotWave3";   // Objektname
input int                   InpDate1=10;             // Datum des ersten Punktes in
input int                   InpPrice1=90;            // Preis des ersten Punktes in
input int                   InpDate2=30;             // Datum des zweiten Punktes in
input int                   InpPrice2=10;            // Preis des zweiten Punktes in
input int                   InpDate3=50;             // Datum des dritten Punktes in
```

© 2000-2025, MetaQuotes Ltd.

```
input int                      InpPrice3=40;            // Preis des dritten Punktes ir
input ENUM_ELLIOT_WAVE_DEGREE  InpDegree=ELLIOTT_MINOR; // Niveau
input bool                     InpDrawLines=true;       // Anzeige der Linien
input color                    InpColor=clrRed;         // Linienfarbe
input ENUM_LINE_STYLE          InpStyle=STYLE_DASH;     // Linienstil
input int                      InpWidth=2;              // Linienbreite
input bool                     InpBack=false;           // Objekt im Hintergrund
input bool                     InpSelection=true;       // Wählen um zu bewegen
input bool                     InpHidden=true;          // Ausgeblendet in der Objektli
input long                     InpZOrder=0;             // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Elliot Korrekturwelle auf angegebenen Koordinaten       |
//+------------------------------------------------------------------+
bool ElliotWave3Create(const long                   chart_ID=0,              // ID de
                       const string                 name="ElliotWave3",      // Name
                       const int                    sub_window=0,            // Numme
                       datetime                     time1=0,                 // Zeit
                       double                       price1=0,                // Preis
                       datetime                     time2=0,                 // Zeit
                       double                       price2=0,                // Preis
                       datetime                     time3=0,                 // Zeit
                       double                       price3=0,                // Preis
                       const ENUM_ELLIOT_WAVE_DEGREE degree=ELLIOTT_MINUETTE, // Grad
                       const bool                   draw_lines=true,         // Anzei
                       const color                  clr=clrRed,              // Objek
                       const ENUM_LINE_STYLE        style=STYLE_SOLID,       // Linie
                       const int                    width=1,                 // Linie
                       const bool                   back=false,              // Im Hi
                       const bool                   selection=true,          // Wähle
                       const bool                   hidden=true,             // Ausge
                       const long                   z_order=0)               // Prio
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeElliotWave3EmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erstellt Elliott Korrekturwelle auf angegebenen Koordinaten
   if(!ObjectCreate(chart_ID,name,OBJ_ELLIOTWAVE3,sub_window,time1,price1,time2,price2
     {
      Print(__FUNCTION__,
            ": Objekt \"Elliott Korrekturwelle\" konnte nicht erstellt werden! Fehlerc
      return(false);
     }
//--- Grad (Wellengröße) setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_DEGREE,degree);
//--- aktivieren (true) oder deaktivieren (false) Anzeige der Linien
   ObjectSetInteger(chart_ID,name,OBJPROP_DRAWLINES,draw_lines);
//--- Objektfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Linienstil setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Objekts für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt von "Elliott Korrekturwelle"           |
//+------------------------------------------------------------------+
bool ElliotWave3PointChange(const long   chart_ID=0,          // ID des Charts
                            const string name="ElliotWave3", // Objektname
                            const int    point_index=0,      // Nummer des Bezugspunkt
                            datetime     time=0,             // Zeitkoordinate des Bez
                            double       price=0)            // Preiskoordinate des Be
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht "Elliott Korrekturwelle"                                  |
//+------------------------------------------------------------------+
bool ElliotWave3Delete(const long   chart_ID=0,         // ID des Charts
```

© 2000-2025, MetaQuotes Ltd.

```
                        const string name="ElliotWave3") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \"Elliott Korrekturwelle\" konnte nicht gelöscht werden! Fehlerc
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Prüft die Werte der Ankerpunkten von "Elliott Korrekturwelle"    |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeElliotWave3EmptyPoints(datetime &time1,double &price1,
                                  datetime &time2,double &price2,
                                  datetime &time3,double &price3)
  {
//--- Array um die Zeit der Öffnung der letzten 10 Bars zu empfangen
   datetime temp[];
   ArrayResize(temp,10);
//--- Daten erhalten
   CopyTime(Symbol(),Period(),TimeCurrent(),10,temp);
//--- den Wert von einem Pip auf dem aktuellen Chart erhalten
   double point=SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des ersten Punktes nicht angegeben ist, wird er auf 9 Balken auf c
   if(!time1)
      time1=temp[0];
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird er auf 5 Balken auf
   if(!time2)
      time2=temp[4];
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, bewegen wir ihn 300 Punk
   if(!price2)
      price2=price1-300*point;
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird er auf 1 Balken auf
   if(!time3)
      time3=temp[8];
//--- wenn der Preis des dritten Punktes nicht angegeben ist, bewegen wir ihn 200 Punk
   if(!price3)
      price3=price1-200*point;
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
    ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten des Objekts verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um "Elliott Korrekturwelle" zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Elliott Korrekturwelle erstellen
   if(!ElliotWave3Create(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],pr
      InpDegree,InpDrawLines,InpColor,InpStyle,InpWidth,InpBack,InpSelection,InpHidder
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte
//--- Schleifenzähler
   int v_steps=accuracy/5;
//--- den dritten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p3<accuracy-1)
         p3+=1;
      //--- den Punkt bewegen
      if(!ElliotWave3PointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy*4/5;
//--- den ersten und zweiten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(p1>1)
         p1-=1;
      if(p2<accuracy-1)
         p2+=1;
      //--- Die Punkte bewegen
      if(!ElliotWave3PointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!ElliotWave3PointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Objekt aus dem Chart löschen
```

© 2000-2025, MetaQuotes Ltd.

```
   ElliotWave3Delete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_RECTANGLE

Rechteck.



Hinweis

Für das Rechteck, können Sie die Farbfüllung durch den Eigenschaften OBJPROP_FILL einstellen.

Beispiel

Das folgende Skript erstellt und bewegt eine Rechteck auf dem Chart. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt Rechteck auf dem Chart."
#property description "Koordinaten der Ankerpunkten werden als"
#property description "Prozentsatz der Größe des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Rectangle"; // Name des Rechtecks
input int             InpDate1=40;         // Datum des ersten Punktes in %
input int             InpPrice1=40;        // Preis des ersten Punktes in %
input int             InpDate2=60;         // Datum des zweiten Punktes in %
input int             InpPrice2=60;        // Preis des zweiten Punktes in %
input color           InpColor=clrRed;     // Farbe des Rechtecks
input ENUM_LINE_STYLE InpStyle=STYLE_DASH; // Linienstil des Rechtecks
input int             InpWidth=2;          // Linienbreite des Rechtecks
```

© 2000-2025, MetaQuotes Ltd.

```
input bool             InpFill=true;        // Farbfüllung des Rechtecks
input bool             InpBack=false;       // Rechteck im Hintergrund
input bool             InpSelection=true;   // Wählen zu bewegen
input bool             InpHidden=true;      // Ausgeblendet in der Objektliste
input long             InpZOrder=0;         // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt ein Rechteck auf angegebenen Koordinaten                |
//+------------------------------------------------------------------+
bool RectangleCreate(const long            chart_ID=0,        // ID des Charts
                     const string          name="Rectangle",  // Name des Rechtecks
                     const int             sub_window=0,      // Nummer des Unterfenst
                     datetime              time1=0,           // Zeit des ersten Punkt
                     double                price1=0,          // Preis des ersten Punk
                     datetime              time2=0,           // Zeit des zweiten Punk
                     double                price2=0,          // Preis des zweiten Pur
                     const color           clr=clrRed,        // Farbe des Rechtecks
                     const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil des Rechte
                     const int             width=1,           // Linienbreite des Rech
                     const bool            fill=false,        // Farbfüllung des Recht
                     const bool            back=false,        // In der Hintergrund
                     const bool            selection=true,    // Wählen um zu bewegen
                     const bool            hidden=true,       // Verborgen in der List
                     const long            z_order=0)         // Priorität für Mauskli
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeRectangleEmptyPoints(time1,price1,time2,price2);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Rechteck auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_RECTANGLE,sub_window,time1,price1,time2,price2))
     {
      Print(__FUNCTION__,
            ": konnte nicht Rechteck erstellen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Farbe des Rechtecks setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Rechtecklinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Linienbreite des Rechtecks setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- aktivieren (true) oder deaktivieren (false) Farbfüllung des Rechtecks
   ObjectSetInteger(chart_ID,name,OBJPROP_FILL,fill);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen des Rechtecks für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt des Rechtecks                          |
//+------------------------------------------------------------------+
bool RectanglePointChange(const long   chart_ID=0,       // ID des Charts
                          const string name="Rectangle", // Name des Rechtecks
                          const int    point_index=0,    // Nummer des Ankerpunktes
                          datetime     time=0,           // Zeitkoordinate des Punktes
                          double       price=0)          // Preiskoordinate des Punkte
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Rechteck                                              |
//+------------------------------------------------------------------+
bool RectangleDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="Rectangle") // Name des Rechtecks
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Rechteck löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": konnte nicht Rechteck löschen! Fehlercode = ",GetLastError());
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft Werte der Ankerpunkten des Rechtecks                   |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeRectangleEmptyPoints(datetime &time1,double &price1,
                                datetime &time2,double &price2)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time2)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time1,10,temp);
      //--- den zweiten Punkt 9 Bars nach links vom ersten Punkt setzen
      time2=temp[0];
     }
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, bewegen wir ihn 300 Punk
   if(!price2)
      price2=price1-300*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten des Rechtecks verwendet werden
   datetime date[];
```

© 2000-2025, MetaQuotes Ltd.

```
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um ein Rechteck zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
//--- Rechteck erstellen
   if(!RectangleCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],InpColor,
      InpStyle,InpWidth,InpFill,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte des Rechtecks
//--- Schleifenzähler
   int h_steps=bars/2;
//--- die Punkte bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(d1<bars-1)
         d1+=1;
      if(d2>1)
         d2-=1;
      //--- Die Punkte bewegen
      if(!RectanglePointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!RectanglePointChange(0,InpName,1,date[d2],price[p2]))
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- die Punkte bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(p1<accuracy-1)
         p1+=1;
      if(p2>1)
         p2-=1;
      //--- Die Punkte bewegen
      if(!RectanglePointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!RectanglePointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Rechteck aus dem Chart löschen
   RectangleDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_TRIANGLE

Dreieck.



Hinweis

Für das Dreieck, können Sie die Farbfüllung durch den Eigenschaften OBJPROP_FILL einstellen.

Beispiel

Das folgende Skript erstellt und bewegt ein Dreieck auf dem Chart. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt Dreieck auf dem Chart."
#property description "Koordinaten der Ankerpunkten werden als"
#property description "Prozentsatz der Größe des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Triangle";        // Name des Dreiecks
input int             InpDate1=25;               // Datum des ersten Punktes in %
input int             InpPrice1=50;              // Preis des ersten Punktes in %
input int             InpDate2=70;               // Datum des zweiten Punktes in %
input int             InpPrice2=70;              // Preis des zweiten Punktes in %
input int             InpDate3=65;               // Datum des dritten Punktes in %
input int             InpPrice3=20;              // Preis des dritten Punktes in %
input color           InpColor=clrRed;           // Farbe des Dreiecks
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Linienstil des Dreiecks
input int             InpWidth=2;                // Linienbreite des Dreiecks
input bool            InpFill=false;             // Farbfüllung des Dreiecks
input bool            InpBack=false;             // Dreieck im Hintergrund
input bool            InpSelection=true;         // Wählen um zu bewegen
input bool            InpHidden=true;            // Ausgeblendet in der Objektliste
input long            InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt ein Dreieck auf angegebenen Koordinaten                 |
//+------------------------------------------------------------------+
bool TriangleCreate(const long            chart_ID=0,        // ID des Charts
                    const string          name="Triangle",   // Name des Dreiecks
                    const int             sub_window=0,      // Nummer des Unterfenste
                    datetime              time1=0,           // Zeit des ersten Punkte
                    double                price1=0,          // Preis des ersten Punkt
                    datetime              time2=0,           // Zeit des zweiten Punkt
                    double                price2=0,          // Preis des zweiten Punk
                    datetime              time3=0,           // Zeit des dritten Punkt
                    double                price3=0,          // Preis des dritten Punk
                    const color           clr=clrRed,        // Farbe des Dreiecks
                    const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil des Dreieck
                    const int             width=1,           // Dreiecklinienbreite
                    const bool            fill=false,        // Farbfüllung des Dreiec
                    const bool            back=false,        // In der Hintergrund
                    const bool            selection=true,    // Wählen um zu bewegen
                    const bool            hidden=true,       // Verborgen in der Liste
                    const long            z_order=0)         // Priorität für Mausklic
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeTriangleEmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Dreieck auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_TRIANGLE,sub_window,time1,price1,time2,price2,t
     {
      Print(__FUNCTION__,
            ": konnte nicht Dreieck erstellen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Farbe des Dreiecks setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Dreiecklinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Dreiecklinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- aktivieren (true) oder deaktivieren (false) Dreieckfüllung
   ObjectSetInteger(chart_ID,name,OBJPROP_FILL,fill);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- aktivieren (true) oder deaktivieren (false) Wählen von Dreieck für Bewegung
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt des Dreiecks                           |
//+------------------------------------------------------------------+
bool TrianglePointChange(const long   chart_ID=0,      // ID des Charts
                         const string name="Triangle", // Name des Dreiecks
                         const int    point_index=0,   // Nummer des Ankerpunktes
                         datetime     time=0,          // Zeitkoordinate des Punktes
                         double       price=0)         // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Dreieck                                               |
//+------------------------------------------------------------------+
bool TriangleDelete(const long   chart_ID=0,      // ID des Charts
                    const string name="Triangle") // Name des Dreiecks
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Dreieck löschen
   if(!ObjectDelete(chart_ID,name))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print(__FUNCTION__,
             ": Könnte nicht das Dreieck löschen! Fehlercode = ",GetLastError());
       return(false);
      }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft Werte der Ankerpunkten des Dreiecks                    |
//| und setzt Standardwerte für leere Werte                          |
//+------------------------------------------------------------------+
void ChangeTriangleEmptyPoints(datetime &time1,double &price1,
                               datetime &time2,double &price2,
                               datetime &time3,double &price3)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time2)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time1,10,temp);
      //--- den zweiten Punkt 9 Bars nach links vom ersten Punkt setzen
      time2=temp[0];
     }
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, bewegen wir ihn 300 Punk
   if(!price2)
      price2=price1-300*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird sie der Zeit des zwe
   if(!time3)
      time3=time2;
//--- wenn der Preis des ersten Punktes nicht angegeben ist, wird er dem Preis des ers
   if(!price3)
      price3=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
```

© 2000-2025, MetaQuotes Ltd.

```
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten des Dreiecks verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um ein Dreieck zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Dreieck erstellen
   if(!TriangleCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],price
      InpColor,InpStyle,InpWidth,InpFill,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte des Dreiecks
//--- Schleifenzähler
```

© 2000-2025, MetaQuotes Ltd.

```
   int v_steps=accuracy*3/10;
//--- den ersten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p1>1)
         p1-=1;
      //--- den Punkt bewegen
      if(!TrianglePointChange(0,InpName,0,date[d1],price[p1]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int h_steps=bars*9/20-1;
//--- den zweiten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d2>1)
         d2-=1;
      //--- den Punkt bewegen
      if(!TrianglePointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   v_steps=accuracy/4;
//--- den dritten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p3<accuracy-1)
         p3+=1;
      //--- den Punkt bewegen
      if(!TrianglePointChange(0,InpName,2,date[d3],price[p3]))
```

© 2000-2025, MetaQuotes Ltd.

```
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Dreieck aus dem Chart löschen
   TriangleDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ELLIPSE

Ellipse.



Hinweis

Für die Ellipse, können Sie die Farbfüllung durch den Eigenschaften OBJPROP_FILL einstellen.

Beispiel

Das folgende Skript erstellt und bewegt eine Ellipse auf dem Chart. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt Ellipse auf dem Chart."
#property description "Koordinaten der Ankerpunkten werden"
#property description "als Prozentsatz der Größe des Fensters angegeben"
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpName="Ellipse";         // Name der Ellipse
input int             InpDate1=30;               // Datum des ersten Punktes in %
input int             InpPrice1=20;              // Preis des ersten Punktes in %
input int             InpDate2=70;               // Datum des zweiten Punktes in %
input int             InpPrice2=80;              // Preis des zweiten Punktes in %
input int             InpDate3=50;               // Datum des dritten Punktes in %
input int             InpPrice3=60;              // Preis des dritten Punktes in %
input color           InpColor=clrRed;           // Farbe der Ellipse
input ENUM_LINE_STYLE InpStyle=STYLE_DASHDOTDOT; // Ellipselinienstil
```

© 2000-2025, MetaQuotes Ltd.

```
input int              InpWidth=2;                // Linienbrete der Ellipse
input bool             InpFill=false;             // Farbfüllung der Ellipse
input bool             InpBack=false;             // Ellipse Im Hintergrund
input bool             InpSelection=true;         // Wählen um zu bewegen
input bool             InpHidden=true;            // Ausgeblendet in der Objektliste
input long             InpZOrder=0;               // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt eine Ellipse auf angegebenen Koordinaten                |
//+------------------------------------------------------------------+
bool EllipseCreate(const long            chart_ID=0,        // ID des Charts
                   const string          name="Ellipse",    // Name der Ellipse
                   const int             sub_window=0,      // Nummer des Unterfenster
                   datetime              time1=0,           // Zeit des ersten Punktes
                   double                price1=0,          // Preis des ersten Punkte
                   datetime              time2=0,           // Zeit des zweiten Punkte
                   double                price2=0,          // Preis des zweiten Punkt
                   datetime              time3=0,           // Zeit des dritten Punkte
                   double                price3=0,          // Preis des dritten Punkt
                   const color           clr=clrRed,        // Farbe der Ellipse
                   const ENUM_LINE_STYLE style=STYLE_SOLID, // Ellipselinienstil
                   const int             width=1,           // Linienbreite der Ellips
                   const bool            fill=false,        // Farbfüllung der Ellipse
                   const bool            back=false,        // Im Hintergrund
                   const bool            selection=true,    // Auswählen um zu beweger
                   const bool            hidden=true,       // Ausgeblendet in der Obj
                   const long            z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten der Bezugspunkte angeben, wenn sie nicht gesetzt sind
   ChangeEllipseEmptyPoints(time1,price1,time2,price2,time3,price3);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ellipse auf angegebenen Koordinaten erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ELLIPSE,sub_window,time1,price1,time2,price2,tir
     {
      Print(__FUNCTION__,
            ": konnte nicht Ellipse erstellen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Ellipsefarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Ellipselinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite der Ellipselinien einstellen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- aktivieren (true) oder deaktivieren (false) Ellipsefüllung
   ObjectSetInteger(chart_ID,name,OBJPROP_FILL,fill);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Wählen der Ellipse für Bewegung
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt den Ankerpunkt der Ellipse                            |
//+------------------------------------------------------------------+
bool EllipsePointChange(const long   chart_ID=0,     // ID des Charts
                        const string name="Ellipse", // Name der Ellipse
                        const int    point_index=0,  // Nummer des Punktes
                        datetime     time=0,         // Zeitkoordinate des Punktes
                        double       price=0)        // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,point_index,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht die Ellipse                                               |
//+------------------------------------------------------------------+
bool EllipseDelete(const long   chart_ID=0,     // ID des Charts
                   const string name="Ellipse") // Name der Ellipse
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ellipse löschen
   if(!ObjectDelete(chart_ID,name))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__,
            ": Könnte nicht sie Ellipse löschen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft Werte der Ankerpunkten der Ellipse und für Leerwerte   |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeEllipseEmptyPoints(datetime &time1,double &price1,
                              datetime &time2,double &price2,
                              datetime &time3,double &price3)
  {
//--- Wenn Zeit des ersten Punktes nicht angegeben ist, wird es auf dem aktuellen Bar
   if(!time1)
      time1=TimeCurrent();
//--- Wenn der Preis des ersten Punktes nicht angegeben wird, dann wird es einen Wert
   if(!price1)
      price1=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- wenn die Zeit des zweiten Punktes nicht angegeben ist, wird sie auf 9 Baren nach
   if(!time2)
     {
      //--- Array um Eröffnungszeit der letzten 10 Bars aufzunehmen
      datetime temp[10];
      CopyTime(Symbol(),Period(),time1,10,temp);
      //--- den zweiten Punkt 9 Bars nach links vom ersten Punkt setzen
      time2=temp[0];
     }
//--- wenn der Preis des zweiten Punktes nicht angegeben ist, bewegen wir ihn 300 Punk
   if(!price2)
      price2=price1-300*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//--- wenn die Zeit des dritten Punktes nicht angegeben ist, wird sie der Zeit des zwe
   if(!time3)
      time3=time2;
//--- wenn der Preis des dritten Punktes nicht angegeben ist, wird er dem Preis des zw
   if(!price3)
      price3=price1;
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate1<0 || InpDate1>100 || InpPrice1<0 || InpPrice1>100 ||
      InpDate2<0 || InpDate2>100 || InpPrice2<0 || InpPrice2>100 ||
      InpDate3<0 || InpDate3>100 || InpPrice3<0 || InpPrice3>100)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("Fehler! Ungültige Eingabeparameter! ");
       return;
     ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten der Ankerpunkten der Ellipse verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Definieren die Punkte um die Ellipse zu zeichnen
   int d1=InpDate1*(bars-1)/100;
   int d2=InpDate2*(bars-1)/100;
   int d3=InpDate3*(bars-1)/100;
   int p1=InpPrice1*(accuracy-1)/100;
   int p2=InpPrice2*(accuracy-1)/100;
   int p3=InpPrice3*(accuracy-1)/100;
//--- Ellipse erstellen
   if(!EllipseCreate(0,InpName,0,date[d1],price[p1],date[d2],price[p2],date[d3],price
      InpColor,InpStyle,InpWidth,InpFill,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir die Ankerpunkte der Ellipse
//--- Schleifenzähler
   int v_steps=accuracy/5;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- den ersten und zweiten Punkt bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- die nächsten Werte nehmen
      if(p1<accuracy-1)
         p1+=1;
      if(p2>1)
         p2-=1;
      //--- Die Punkte bewegen
      if(!EllipsePointChange(0,InpName,0,date[d1],price[p1]))
         return;
      if(!EllipsePointChange(0,InpName,1,date[d2],price[p2]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int h_steps=bars/5;
//--- den dritten Punkt bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d3>1)
         d3-=1;
      //--- den Punkt bewegen
      if(!EllipsePointChange(0,InpName,2,date[d3],price[p3]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- die Ellipse aus dem Chart löschen
   EllipseDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_THUMB_UP

Zeichen "Gut".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Zeichen (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Icons "Gut". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Gut\" (Daumen nach oben)."
#property description "Koordinate des Bezugspunktes wird in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string             InpName="ThumbUp";     // Name des Zeichens
input int                InpDate=75;            // Datum des Bezugspunktes in %
input int                InpPrice=25;           // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR  InpAnchor=ANCHOR_TOP;  // Bezugsmethode
input color              InpColor=clrRed;       // Farbe des Zeichens
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE   InpStyle=STYLE_DOT;   // Stil der Einfassungslinie
input int               InpWidth=5;           // Größe des Zeichens
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=true;    // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Zeichen "Gut"                                           |
//+------------------------------------------------------------------+
bool ArrowThumbUpCreate(const long              chart_ID=0,           // ID des Charts
                        const string            name="ThumbUp",       // Name des Zeic
                      const int               sub_window=0,         // Nummer des Unter
                       datetime               time=0,               // Zeit des Bezugs
                       double                 price=0,              // Preis des Bezug
                       const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Bezugsmethode
                       const color            clr=clrRed,           // Farbe des Zeich
                       const ENUM_LINE_STYLE  style=STYLE_SOLID,    // Stil der Einfas
                       const int              width=3,              // Größe des Zeich
                       const bool             back=false,           // Im Hintergrund
                       const bool             selection=true,       // Wählen zu beweg
                       const bool             hidden=true,          // Verborgen in de
                       const long             z_order=0)            // Priorität auf M
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_THUMB_UP,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Gut\ erstellen"! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowThumbUpMove(const long   chart_ID=0,      // ID des Charts
                      const string name="ThumbUp", // Name des Objekts
                      datetime     time=0,         // Zeitkoordinate des Bezugspunkts
                      double       price=0)        // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Weise der Bindung von "Gut"                           |
//+------------------------------------------------------------------+
bool ArrowThumbUpAnchorChange(const long              chart_ID=0,        // ID des Cha
                              const string            name="ThumbUp",    // Objektname
                              const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Bindungsme
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bindungsmethode ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Gut"                                         |
//+------------------------------------------------------------------+
bool ArrowThumbUpDelete(const long   chart_ID=0,     // ID des Charts
                        const string name="ThumbUp") // Zeichenname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Gut\ löschen"! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen der Zeichen definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das Zeichen "Gut" auf dem Chart
   if(!ArrowThumbUpCreate(0,InpName,0,date[d],price[p],InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen und seine Position relativ zu dem Zeich
//--- Schleifenzähler
   int h_steps=bars/4;
//--- bewegen den Bezugspunkt
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d>1)
         d-=1;
      //--- den Punkt bewegen
      if(!ArrowThumbUpMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int v_steps=accuracy/4;
//--- bewegen den Bezugspunkt
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p<accuracy-1)
         p+=1;
      //--- den Punkt bewegen
      if(!ArrowThumbUpMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- Die Position des Bezugspunktes relativ zum Zeichen ändern
   ArrowThumbUpAnchorChange(0,InpName,ANCHOR_BOTTOM);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Zeichen aus dem Chart löschen
   ArrowThumbUpDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_THUMB_DOWN

Zeichen "Schlecht".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Zeichen (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Icons "Schlecht". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Schlecht\" (Daumen nach unten)."
#property description "Koordinate des Bezugspunktes wird in Prozent von"
#property description "der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="ThumbDown";      // Name des Zeichens
input int               InpDate=25;               // Datum des Bezugspunktes in %
input int               InpPrice=75;              // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR InpAnchor=ANCHOR_BOTTOM; // Bindungsmethode
```

© 2000-2025, MetaQuotes Ltd.

```
input color             InpColor=clrRed;      // Farbe des Zeichens
input ENUM_LINE_STYLE   InpStyle=STYLE_DOT;   // Stil der Einfassungslinie
input int               InpWidth=5;           // Größe des Zeichens
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=true;    // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Zeichen "Schlecht"                                      |
//+------------------------------------------------------------------+
bool ArrowThumbDownCreate(const long              chart_ID=0,           // ID des Char
                          const string            name="ThumbDown",     // Name des Ze
                      const int               sub_window=0,         // Nummer des Unter
                       datetime                time=0,               // Zeit des Bezugs
                       double                  price=0,              // Preis des Bezug
                       const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Bezugsmethode
                       const color             clr=clrRed,           // Farbe des Zeich
                       const ENUM_LINE_STYLE   style=STYLE_SOLID,    // Stil der Einfas
                       const int               width=3,              // Größe des Zeich
                       const bool              back=false,           // Im Hintergrund
                       const bool              selection=true,       // Wählen zu beweg
                       const bool              hidden=true,          // Verborgen in de
                       const long              z_order=0)            // Priorität auf M
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_THUMB_DOWN,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Schlecht\ erstellen"! Fehlercode = ",GetLastError
      return(false);
     }
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowThumbDownMove(const long   chart_ID=0,       // ID des Charts
                        const string name="ThumbDown", // Name des Objekts
                  datetime     time=0,           // Zeitkoordinate des Bezugspunkts
                  double       price=0)          // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Weise der Bindung von "Schlecht"                      |
//+------------------------------------------------------------------+
bool ArrowThumbDownAnchorChange(const long              chart_ID=0,        // ID des C
                                const string            name="ThumbDown",  // Name des
                          const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Bindungsmetho
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bindungsmethode ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht Zeichen "Schlecht"                                        |
//+------------------------------------------------------------------+
bool ArrowThumbDownDelete(const long   chart_ID=0,       // ID des Charts
                          const string name="ThumbDown") // Name des Zichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Schlecht\ löschen"! Fehlercode = ",GetLastError())
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
```

© 2000-2025, MetaQuotes Ltd.

```
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen der Zeichen definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das Zeichen "Schlecht" auf dem Chart
   if(!ArrowThumbDownCreate(0,InpName,0,date[d],price[p],InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen und seine Position relativ zu dem Zeich
//--- Schleifenzähler
   int h_steps=bars/4;
//--- bewegen den Bezugspunkt
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!ArrowThumbDownMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
```

© 2000-2025, MetaQuotes Ltd.

```
         return;
       //--- den Chart neu zeichnen
       ChartRedraw();
       // Verzögerung 0.05 Sekunden
       Sleep(50);
      }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schleifenzähler
   int v_steps=accuracy/4;
//--- bewegen den Bezugspunkt
   for(int i=0;i<v_steps;i++)
     {
       //--- den nächsten Wert nehmen
       if(p>1)
         p-=1;
       //--- den Punkt bewegen
       if(!ArrowThumbDownMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
       if(IsStopped())
         return;
       //--- den Chart neu zeichnen
       ChartRedraw();
      }
//--- Die Position des Bezugspunktes relativ zum Zeichen ändern
   ArrowThumbDownAnchorChange(0,InpName,ANCHOR_TOP);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Zeichen aus dem Chart löschen
   ArrowThumbDownDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_UP

Zeichen "Pfeil nach oben".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Zeichen (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Icons "Pfeil nach oben". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Pfeil nach oben\"."
#property description "Koordinate des Bezugspunktes wird als"
#property description "Prozentsatz der Größe des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string             InpName="ArrowUp";     // Name es Zeichens
input int                InpDate=25;            // Datum des Bezugspunktes in %
input int                InpPrice=25;           // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR  InpAnchor=ANCHOR_TOP;  // Bezugsmethode
input color              InpColor=clrRed;       // Farbe des Zeichens
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE   InpStyle=STYLE_DOT;   // Stil der Einfassungslinie
input int               InpWidth=5;           // Größe des Zeichens
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=false;   // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Zeichen "Pfeil nach oben"                               |
//+------------------------------------------------------------------+
bool ArrowUpCreate(const long              chart_ID=0,           // ID des Charts
                   const string            name="ArrowUp",       // Name des Zeichens
                   const int               sub_window=0,         // Nummer des Unterfe
                   datetime                time=0,               // Zeit des Bezugspur
                   double                  price=0,              // Preis des Bezugspu
                   const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Bezugsmethode
                   const color             clr=clrRed,           // Farbe des Zeichens
                   const ENUM_LINE_STYLE   style=STYLE_SOLID,    // Stil der Einfassur
                   const int               width=3,              // Größe des Zeichens
                   const bool              back=false,           // Im Hintergrund
                   const bool              selection=true,       // Wählen zu bewegen
                   const bool              hidden=true,          // Verborgen in der I
                   const long              z_order=0)            // Priorität auf Maus
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_UP,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Pfeil nach oben\ erstellen"! Fehlercode = ",GetLas
      return(false);
     }
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowUpMove(const long   chart_ID=0,      // ID des Charts
                 const string name="ArrowUp", // Name des Objekts
                 datetime     time=0,         // Zeitkoordinate des Bezugspunkts
                 double       price=0)        // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Weise der Bindung von "Pfeil nach oben"               |
//+------------------------------------------------------------------+
bool ArrowUpAnchorChange(const long              chart_ID=0,        // ID des Charts
                         const string            name="ArrowUp",    // Name des Objekt
                         const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Bindungsmethode
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern die Position des Bezugspunkts
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Pfeil nach oben"                             |
//+------------------------------------------------------------------+
bool ArrowUpDelete(const long   chart_ID=0,     // ID des Charts
                   const string name="ArrowUp") // Name des Zeichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Pfeil nach oben\ löschen"! Fehlercode = ",GetLastE
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen der Zeichen definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das Zeichen "Pfeil nach oben" auf dem Chart
   if(!ArrowUpCreate(0,InpName,0,date[d],price[p],InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen und seine Position relativ zu dem Zeich
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- bewegen den Bezugspunkt
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p<accuracy-1)
         p+=1;
      //--- den Punkt bewegen
      if(!ArrowUpMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Die Position des Bezugspunktes relativ zum Zeichen ändern
   ArrowUpAnchorChange(0,InpName,ANCHOR_BOTTOM);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Zeichen aus dem Chart löschen
   ArrowUpDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_DOWN

Zeichen "Pfeil nach unten".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Zeichen (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Icons "Pfeil nach unten". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Pfeil nach unten\"."
#property description "Koordinate des Bezugspunktes wird als"
#property description "Prozentsatz der Größe des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string             InpName="ArrowDown";      // Zeichenname
input int                InpDate=75;               // Datum des Bezugspunktes in %
input int                InpPrice=75;              // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR  InpAnchor=ANCHOR_BOTTOM;  // Bindungsmethode
input color              InpColor=clrRed;       // Farbe des Zeichens
input ENUM_LINE_STYLE    InpStyle=STYLE_DOT;    // Stil der Einfassungslinie
```

© 2000-2025, MetaQuotes Ltd.

```
input int                InpWidth=5;           // Größe des Zeichens
input bool               InpBack=false;        // Das Zeichen im Hintergrund
input bool               InpSelection=false;   // Wählen zu bewegen
input bool               InpHidden=true;       // Ausblenden in der Liste der Objekten
input long               InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Zeichen "Pfeil nach unten"                              |
//+------------------------------------------------------------------+
bool ArrowDownCreate(const long              chart_ID=0,           // ID des Charts
                     const string            name="ArrowDown",     // Name des Zeicher
                     const int               sub_window=0,         // Nummer des Unter
                     datetime                time=0,               // Zeit des Bezugsp
                     double                  price=0,              // Preis des Bezugs
                     const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Bezugsmethode
                     const color             clr=clrRed,           // Farbe des Zeiche
                     const ENUM_LINE_STYLE   style=STYLE_SOLID,    // Stil der Einfass
                     const int               width=3,              // Größe des Zeiche
                     const bool              back=false,           // Im Hintergrund
                     const bool              selection=true,       // Wählen zu bewege
                     const bool              hidden=true,          // Verborgen in der
                     const long              z_order=0)            // Priorität auf Ma
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_DOWN,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Pfeil nach unten\ erstellen"! Fehlercode = ",GetLa
      return(false);
     }
//--- Bindungsmethode
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowDownMove(const long   chart_ID=0,        // ID des Charts
                   const string name="ArrowDown", // Name des Objekts
                   datetime     time=0,           // Zeitkoordinate des Bezugspunkts
                   double       price=0)          // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Weise der Bindung von "Pfeil nach unten"              |
//+------------------------------------------------------------------+
bool ArrowDownAnchorChange(const long              chart_ID=0,        // ID des Charts
                   const string name="ArrowDown", // Name des Objekts
                           const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Bindungsmetho
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ändern die Position des Bezugspunkts
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Pfeil nach unten"                            |
//+------------------------------------------------------------------+
bool ArrowDownDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="ArrowDown") // Name des Zeichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Pfeil nach unten\ löschen"! Fehlercode = ",GetLast
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
```

© 2000-2025, MetaQuotes Ltd.

```
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen der Zeichen definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das Zeichen "Pfeil nach unten" auf dem Chart
   if(!ArrowDownCreate(0,InpName,0,date[d],price[p],InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen und seine Position relativ zu dem Zeich
//--- Schleifenzähler
   int v_steps=accuracy/2;
//--- bewegen den Bezugspunkt
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p>1)
         p-=1;
      //--- den Punkt bewegen
      if(!ArrowDownMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
```

© 2000-2025, MetaQuotes Ltd.

```
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Die Position des Bezugspunktes relativ zum Zeichen ändern
   ArrowDownAnchorChange(0,InpName,ANCHOR_TOP);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Zeichen aus dem Chart löschen
   ArrowDownDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_STOP

Zeichen "Stop".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Zeichen (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Icons "Stop". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Stop\"."
#property description "Koordinate des Bezugspunktes wird als"
#property description "Prozentsatz der Größe des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string             InpName="ArrowStop";      // Name des Zeichens
input int                InpDate=10;               // Datum des Bezugspunktes in %
input int                InpPrice=50;              // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR InpAnchor=ANCHOR_BOTTOM; // Bindungsmethode
input color              InpColor=clrRed;       // Farbe des Zeichens
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE   InpStyle=STYLE_DOT;   // Stil der Einfassungslinie
input int               InpWidth=5;           // Größe des Zeichens
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=false;   // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt das Zeichen "Stop"                                      |
//+------------------------------------------------------------------+
bool ArrowStopCreate(const long              chart_ID=0,           // ID des Charts
                     const string            name="ArrowStop",     // Name des Zeicher
                     const int               sub_window=0,         // Nummer des Unter
                     datetime                time=0,               // Zeit des Bezugsp
                     double                  price=0,              // Preis des Bezugs
                     const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Bezugsmethode
                     const color             clr=clrRed,           // Farbe des Zeiche
                     const ENUM_LINE_STYLE   style=STYLE_SOLID,    // Stil der Einfass
                     const int               width=3,              // Größe des Zeiche
                     const bool              back=false,           // Im Hintergrund
                     const bool              selection=true,       // Wählen zu bewege
                     const bool              hidden=true,          // Verborgen in der
                     const long              z_order=0)            // Priorität auf Ma
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_STOP,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Stop\ erstellen"! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowStopMove(const long   chart_ID=0,       // ID des Charts
                   const string name="ArrowStop", // Name des Objekts
                   datetime     time=0,           // Zeitkoordinate des Bezugspunkts
                   double       price=0)          // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Weise der Bindung von "Stop"                          |
//+------------------------------------------------------------------+
bool ArrowStopAnchorChange(const long              chart_ID=0,        // ID des Charts
                   const string name="ArrowStop", // Name des Objekts
                           const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Position des
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bindungsmethode ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Stop"                                        |
//+------------------------------------------------------------------+
bool ArrowStopDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="ArrowStop") // Name des Zeichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Stop\ löschen"! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen der Zeichen definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das Zeichen "Stop" auf dem Chart
   if(!ArrowStopCreate(0,InpName,0,date[d],price[p],InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen und seine Position relativ zu dem Zeich
//--- Schleifenzähler
   int h_steps=bars*2/5;
//--- bewegen den Bezugspunkt
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!ArrowStopMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.025 Sekunden
      Sleep(25);
     }
//--- Die Position des Bezugspunktes relativ zum Zeichen ändern
   ArrowStopAnchorChange(0,InpName,ANCHOR_TOP);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- Schleifenzähler
   h_steps=bars*2/5;
//--- bewegen den Bezugspunkt
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!ArrowStopMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.025 Sekunden
      Sleep(25);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Zeichen aus dem Chart löschen
   ArrowStopDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_CHECK

Zeichen "Checkmark".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Zeichen (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Icons "Checkmark". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Checkmark\"."
#property description "Koordinate des Bezugspunktes wird als"
#property description "Prozentsatz der Größe des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string             InpName="ArrowCheck"; // Name des Zeichens
input int                InpDate=10;           // Datum des Bezugspunktes in %
input int                InpPrice=50;          // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR InpAnchor=ANCHOR_TOP; // Bezugsmethode
input color              InpColor=clrRed;      // Farbe des Zeichens
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_LINE_STYLE   InpStyle=STYLE_DOT;   // Stil der Einfassungslinie
input int               InpWidth=5;           // Größe des Zeichens
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=false;   // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt Zeichen "Checkmark"                                     |
//+------------------------------------------------------------------+
bool ArrowCheckCreate(const long              chart_ID=0,           // ID des Charts
                      const string            name="ArrowCheck",    // Name des Zeiche
                      const int               sub_window=0,         // Nummer des Unte
                      datetime                time=0,               // Zeit des Bezugs
                      double                  price=0,              // Preis des Bezug
                      const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Bezugsmethode
                      const color             clr=clrRed,           // Farbe des Zeich
                      const ENUM_LINE_STYLE   style=STYLE_SOLID,    // Stil der Einfas
                      const int               width=3,              // Größe des Zeich
                      const bool              back=false,           // Im Hintergrund
                      const bool              selection=true,       // Wählen zu beweg
                      const bool              hidden=true,          // Verborgen in de
                      const long              z_order=0)            // Priorität auf M
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_CHECK,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Checkmark\ erstellen"! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowCheckMove(const long   chart_ID=0,        // ID des Charts
                    const string name="ArrowCheck", // Name des Objekts
                    datetime     time=0,            // Zeitkoordinate des Bezugspunkts
                    double       price=0)           // Preiskoordinate des Bezugspunkt
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Weise der Bindung von "Checkmark"                     |
//+------------------------------------------------------------------+
bool ArrowCheckAnchorChange(const long              chart_ID=0,        // ID des Chart
                            const string            name="ArrowCheck", // Name des Obj
                            const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Bindungsmeth
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bindungsmethode ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Checkmark"                                   |
//+------------------------------------------------------------------+
bool ArrowCheckDelete(const long   chart_ID=0,        // ID des Charts
                      const string name="ArrowCheck") // Name des Zeichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Checkmark\" löschen! Fehlercode = ",GetLastError()
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen der Zeichen definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das Zeichen "Checkmark" auf dem Chart
   if(!ArrowCheckCreate(0,InpName,0,date[d],price[p],InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen und seine Position relativ zu dem Zeich
//--- Schleifenzähler
   int h_steps=bars*2/5;
//--- bewegen den Bezugspunkt
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!ArrowCheckMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.025 Sekunden
      Sleep(25);
     }
//--- Die Position des Bezugspunktes relativ zum Zeichen ändern
   ArrowCheckAnchorChange(0,InpName,ANCHOR_BOTTOM);
//--- den Chart neu zeichnen
   ChartRedraw();
//--- Schleifenzähler
   h_steps=bars*2/5;
//--- bewegen den Bezugspunkt
   for(int i=0;i<h_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!ArrowCheckMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.025 Sekunden
      Sleep(25);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Zeichen aus dem Chart löschen
   ArrowCheckDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_LEFT_PRICE

Linkes Preisschild.



Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart ein linkes Preisschild. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript erstellt ein linkes Preisschild auf dem Chart."
#property description "Koordinate des Bezugspunktes wird in Prozent"
#property description "von der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="LeftPrice";  // Name des Preisschilds
input int               InpDate=100;          // Datum des Bezugspunktes in %
input int               InpPrice=10;          // Preis des Bezugspunktes in %
input color             InpColor=clrRed;      // Farbe des Schilds
input ENUM_LINE_STYLE   InpStyle=STYLE_SOLID; // Stil der Einfassungslinie
input int               InpWidth=2;           // Größe der Preisschilder
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=true;    // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt ein linkes Preisschild                                  |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ArrowLeftPriceCreate(const long            chart_ID=0,        // ID des Charts
                          const string          name="LeftPrice",  // Name des Preiss
                          const int             sub_window=0,      // Nummer des Unte
                          datetime              time=0,            // Zeit des Bezugsp
                          double                price=0,           // Preis des Bezugs
                          const color           clr=clrRed,        // Farbe des Schil
                          const ENUM_LINE_STYLE style=STYLE_SOLID, // Stil der Einfass
                          const int             width=1,           // Größe des Preis
                          const bool            back=false,        // In der Hintergru
                          const bool            selection=true,    // Wählen zu bewege
                          const bool            hidden=true,       // Verborgen in de
                          const long            z_order=0)         // Priorität für Ma
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ein Preisschild erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_LEFT_PRICE,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht das linke Preisschild erstellen! Fehlercode = ",GetLastError
      return(false);
     }
//--- Die Farbe des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in de
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool ArrowLeftPriceMove(const long   chart_ID=0,       // ID des Charts
                        const string name="LeftPrice", // Name des Schilds
                     datetime     time=0,           // Zeitkoordinate des Bezugspunkts
                     double       price=0)          // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das linke Preisschild aus dem Chart                       |
//+------------------------------------------------------------------+
bool ArrowLeftPriceDelete(const long   chart_ID=0,       // ID des Charts
                          const string name="LeftPrice") // Name des Schilds
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht das linke Preisschild löschen! Fehlercode = ",GetLastError()
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Bezugspunkte des Preisschilds verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
    ;}
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen des Schilds definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das linke Preisschild auf dem Chart
   if(!ArrowLeftPriceCreate(0,InpName,0,date[d],price[p],InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen
//--- Schleifenzähler
   int v_steps=accuracy*4/5;
//--- bewegen den Bezugspunkt
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p<accuracy-1)
         p+=1;
      //--- den Punkt bewegen
      if(!ArrowLeftPriceMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schild aus dem Chart löschen
   ArrowLeftPriceDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_RIGHT_PRICE

Rechtes Preisschild.



Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart ein rechtes Preisschild. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript erstellt ein rechtes Preisschild auf dem Chart."
#property description "Koordinate des Bezugspunktes wird in Prozent"
#property description "von der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="RightPrice"; // Name des Preisschilds
input int               InpDate=0;            // Datum des Bezugspunktes in %
input int               InpPrice=90;          // Preis des Bezugspunktes in %
input color             InpColor=clrRed;      // Farbe des Schilds
input ENUM_LINE_STYLE   InpStyle=STYLE_SOLID; // Stil der Einfassungslinie
input int               InpWidth=2;           // Größe der Preisschilder
input bool              InpBack=false;        // Das Zeichen im Hintergrund
input bool              InpSelection=true;    // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt ein rechtes Preisschild                                 |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool ArrowRightPriceCreate(const long            chart_ID=0,        // ID des Charts
                           const string          name="RightPrice", // Name des Preiss
                           const int             sub_window=0,      // Nummer des Unte
                           datetime              time=0,            // Zeit des Bezugs
                           double                price=0,           // Preis des Bezug
                           const color           clr=clrRed,        // Farbe des Schil
                           const ENUM_LINE_STYLE style=STYLE_SOLID, // Stil der Einfas
                           const int             width=1,           // Größe des Preis
                           const bool            back=false,        // In der Hintergr
                           const bool            selection=true,    // Wählen zu beweg
                           const bool            hidden=true,       // Verborgen in de
                           const long            z_order=0)         // Priorität für N
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Ein Preisschild erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_RIGHT_PRICE,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht das rechte Preisschild erstellen! Fehlercode = ",GetLastErro
      return(false);
     }
//--- Die Farbe des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Die Größe des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool ArrowRightPriceMove(const long   chart_ID=0,        // ID des Charts
                         const string name="RightPrice", // Name des Schilds
                   datetime     time=0,            // Zeitkoordinate des Bezugspunkts
                   double       price=0)           // Preiskoordinate des Bezugspunkt
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das rechte Preisschild aus dem Chart                      |
//+------------------------------------------------------------------+
bool ArrowRightPriceDelete(const long   chart_ID=0,        // ID des Charts
                           const string name="RightPrice") // Name des Schilds
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht das rechte Preisschild löschen! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Bezugspunkte des Preisschilds verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen des Schilds definieren
   int d=InpDate*(bars-1)/100;
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen das rechte Preisschild auf dem Chart
   if(!ArrowRightPriceCreate(0,InpName,0,date[d],price[p],InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
```

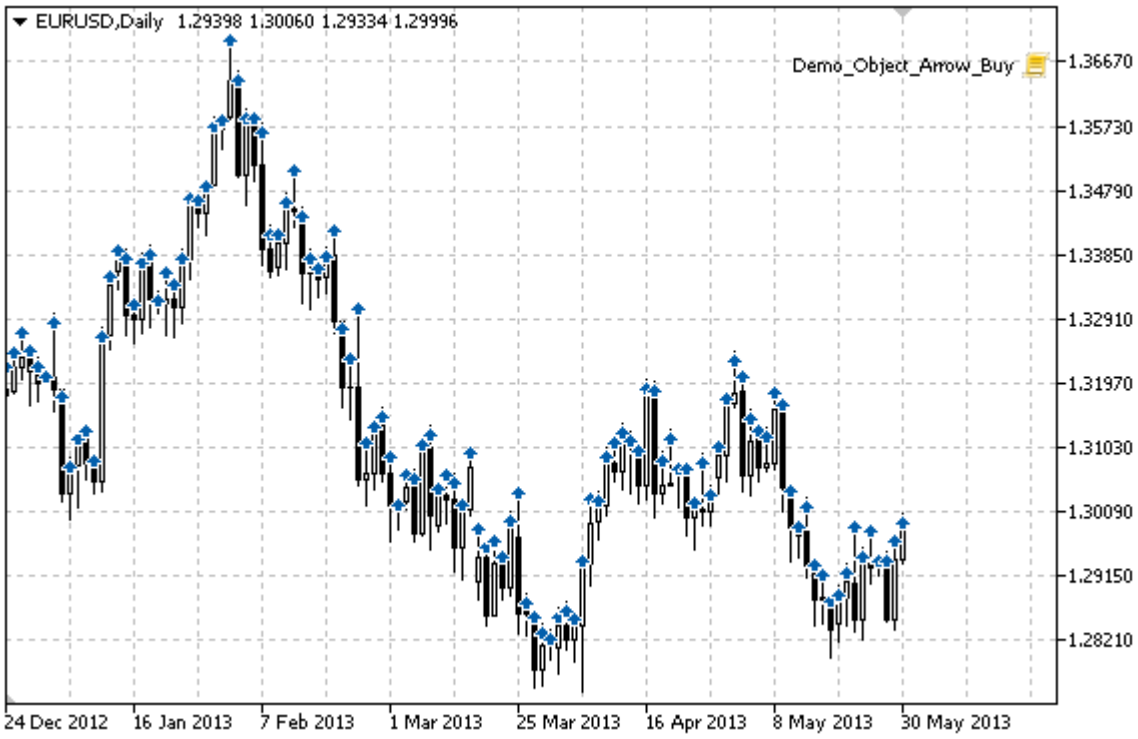
© 2000-2025, MetaQuotes Ltd.

```
      }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Jetzt werden wir den Bezugspunkt bewegen
//--- Schleifenzähler
   int v_steps=accuracy*4/5;
//--- bewegen den Bezugspunkt
   for(int i=0;i<v_steps;i++)
     {
      //--- den nächsten Wert nehmen
      if(p>1)
         p-=1;
      //--- den Punkt bewegen
      if(!ArrowRightPriceMove(0,InpName,date[d],price[p]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Schild aus dem Chart löschen
   ArrowRightPriceDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_BUY

Zeichen "Buy".



Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Zeichen "Buy". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Buy\" im Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input color InpColor=C'3,95,172'; // Farbe des Symbols
//+------------------------------------------------------------------+
//| Erstellt das Zeichen "Buy"                                       |
//+------------------------------------------------------------------+
bool ArrowBuyCreate(const long            chart_ID=0,        // ID des Charts
                    const string          name="ArrowBuy",   // Name des Zeichens
                    const int             sub_window=0,      // Nummer des Fensters
                    datetime              time=0,            // Zeit des Bezugspunkts
                    double                price=0,           // Preis des Bezugspunkts
                    const color           clr=C'3,95,172',   // Farbe des Zeichens
                    const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil (für Markie
                    const int             width=1,           // Liniengröße (für Mark
                    const bool            back=false,        // In der Hintergrund
                    const bool            selection=false,   // Markieren für Bewegung
```

© 2000-2025, MetaQuotes Ltd.

```
                    const bool            hidden=true,       // Verborgen in der Liste
                    const long            z_order=0)         // Priorität für Mausklic
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_BUY,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Icon \"Buy\" konnte nicht erstellt werden! Fehlercode = ",GetLastError
      return(false);
     ;}
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil für Markierung setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Liniengröße für Markierung setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowBuyMove(const long   chart_ID=0,       // ID des Charts
                  const string name="ArrowBuy", // Objektname
                  datetime     time=0,          // Zeitkoordinate des Bezugspunkts
                  double       price=0)         // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print(__FUNCTION__,
             ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
       return(false);
      }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Buy"                                         |
//+------------------------------------------------------------------+
bool ArrowBuyDelete(const long   chart_ID=0,       // ID des Charts
                    const string name="ArrowBuy") // Name des Zeichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Zeichen \"Buy\" konnte nicht gelöscht werden! Fehlercode = ",GetLastErr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date[]; // Array zu speichern Daten von sichtbaren Bars
   double   low[];  // Array zu speichern Low-Priese von sichtbaren Bars
   double   high[]; // Array zu speichern High-Preise von sichtbaren Bars
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(low,bars);
   ArrayResize(high,bars);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen die Anordnung von Low-Preise
   if(CopyLow(Symbol(),Period(),0,bars,low)==-1)
     {
      Print("Kann nicht den Wert der Low-Preise kopieren! Fehlercode = ",GetLastError
      return;
     }
//--- Füllen die Anordnung von High-Preise
   if(CopyHigh(Symbol(),Period(),0,bars,high)==-1)
     {
      Print("Kann nicht den Wert der High-Preise kopieren! Fehlercode = ",GetLastErro
      return;
     }
//--- Erstellen Zeichen "Buy" auf Low für jedes sichtbare Bar
   for(int i=0;i<bars;i++)
     {
      if(!ArrowBuyCreate(0,"ArrowBuy_"+(string)i,0,date[i],low[i],InpColor))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- Bewegen Zeichens "Buy" auf High für jedes sichtbare Bar
   for(int i=0;i<bars;i++)
     {
      if(!ArrowBuyMove(0,"ArrowBuy_"+(string)i,date[i],high[i]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Zeichen "Buy" löschen
   for(int i=0;i<bars;i++)
     {
      if(!ArrowBuyDelete(0,"ArrowBuy_"+(string)i))
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW_SELL

Zeichen "Sell".



Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Zeichen "Sell". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Der Skript zieht Zeichen \"Sell\" im Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input color InpColor=C'225,68,29'; // Farbe der Zeichen
//+------------------------------------------------------------------+
//| Erstellt das Zeichen "Sell"                                      |
//+------------------------------------------------------------------+
bool ArrowSellCreate(const long            chart_ID=0,        // ID des Charts
                     const string          name="ArrowSell",  // Name der Zeichen
                     const int             sub_window=0,      // Nummer des Unterfenst
                     datetime              time=0,            // Zeit des Bezugspunkts
                     double                price=0,           // Preis des Bezugspunkt
                     const color           clr=C'225,68,29',  // Farbe des Zeichens
                     const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil (zu wähler
                     const int             width=1,           // Liniengröße (zu wähle
                     const bool            back=false,        // In der Hintergrund
                     const bool            selection=false,   // Wählen für Bewegung
```

© 2000-2025, MetaQuotes Ltd.

```
                      const bool            hidden=true,       // Verborgen in der List
                      const long            z_order=0)         // Priorität für Mausklі
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW_SELL,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Sell\ erstellen"! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Die Farbe des Zeichens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil für Markierung setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Liniengröße für Markierung setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowSellMove(const long   chart_ID=0,       // ID des Charts
                   const string name="ArrowSell", // Name des Objekts
                   datetime     time=0,           // Zeitkoordinate des Bezugspunkts
                   double       price=0)          // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Zeichen "Sell"                                        |
//+------------------------------------------------------------------+
bool ArrowSellDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="ArrowSell") // Name des Zeichens
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Zeichen löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Kann nicht Zeichen \"Sell\ löschen"! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bi
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date[]; // Array zu speichern Daten von sichtbaren Bars
   double   low[];  // Array zu speichern Low-Priese von sichtbaren Bars
   double   high[]; // Array zu speichern High-Preise von sichtbaren Bars
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(low,bars);
   ArrayResize(high,bars);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen die Anordnung von Low-Preise
   if(CopyLow(Symbol(),Period(),0,bars,low)==-1)
     {
      Print("Kann nicht den Wert der Low-Preise kopieren! Fehlercode = ",GetLastError
      return;
     }
//--- Füllen die Anordnung von High-Preise
   if(CopyHigh(Symbol(),Period(),0,bars,high)==-1)
     {
      Print("Kann nicht den Wert der High-Preise kopieren! Fehlercode = ",GetLastErro
      return;
     }
//--- Erstellen Icons "Sell" auf High für jedes sichtbare Bar
   for(int i=0;i<bars;i++)
     {
      if(!ArrowSellCreate(0,"ArrowSell_"+(string)i,0,date[i],high[i],InpColor))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- Bewegen Icons "Sell" auf Low für jedes sichtbare Bar
   for(int i=0;i<bars;i++)
     {
      if(!ArrowSellMove(0,"ArrowSell_"+(string)i,date[i],low[i]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Zeichen "Sell" löschen
   for(int i=0;i<bars;i++)
     {
      if(!ArrowSellDelete(0,"ArrowSell_"+(string)i))
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_ARROW

Zeichen "Pfeil".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ARROW_ANCHOR gewählt werden.

Größe Pfeile (mehr als 5) können nur durch Setzen der entsprechenden Eigenschaft OBJPROP_WIDTH beim Schreiben von Code in MetaEditor erstellt werden.

Die gewünschte Art der Pfeile kann durch Einstellung einer der Codes von Symbole von Font Wingdings ausgewählt werden.

Beispiel

Das folgende Skript erstellt Objekt "Pfeil" auf dem Chart und ändert seinen Typ. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt einen zufälligen Pfeil im Chartfenster."
#property description "Koordinate des Bezugspunktes wird in Prozent"
#property description "von der Größe des Chartfensters eingegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="Arrow";        // Name des Pfeils
input int               InpDate=50;             // Datum des Bezugspunktes in %
```

© 2000-2025, MetaQuotes Ltd.

```
input int                InpPrice=50;             // Preis des Bezugspunktes in %
input ENUM_ARROW_ANCHOR  InpAnchor=ANCHOR_TOP;    // Bezugsmethode
input color              InpColor=clrDodgerBlue; // Farbe des Pfeils
input ENUM_LINE_STYLE    InpStyle=STYLE_SOLID;    // Stil der Einfassungslinie
input int                InpWidth=10;             // Größe des Pfeils
input bool               InpBack=false;           // Pfeil im Hintergrund
input bool               InpSelection=false;      // Wählen zu bewegen
input bool               InpHidden=true;          // Ausgeblendet in der Liste der Objek
input long               InpZOrder=0;             // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt einen Pfeil                                             |
//+------------------------------------------------------------------+
bool ArrowCreate(const long              chart_ID=0,           // ID des Charts
                 const string            name="Arrow",         // Name des Pfeils
                 const int               sub_window=0,         // Nummer des Unterfens
                 datetime                time=0,               // Zeit des Bezugspunkt
                 double                  price=0,              // Preis des Bezugspunk
                 const uchar             arrow_code=252,       // Code des Pfeils
                 const ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM, // Position des Bezugsp
                 const color             clr=clrRed,           // Farbe des Pfeils
                 const ENUM_LINE_STYLE   style=STYLE_SOLID,    // Stil der Einfassungs
                 const int               width=3,              // Größe des Pfeils
                 const bool              back=false,           // Im Hintergrund
                 const bool              selection=true,       // Wählen zu bewegen
                 const bool              hidden=true,          // Verborgen in der Lis
                 const long              z_order=0)            // Priorität auf Mausk
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeArrowEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Pfeil erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_ARROW,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Pfeil konnte nicht erstellt werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Code des Pfeils setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ARROWCODE,arrow_code);
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Farbe des Pfeils setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Stil der Einfassungslinie setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Größe des Pfeils setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
//--- Wenn Sie ein grafisches Objekt durch Funktion ObjectCreate erstellen, das Objekt
//--- kann nicht ausgewählt und verschoben werden. Innerhalb dieser Methode ist der St
//--- true, was Sie das Objekt auswählen und verschieben erlaubt
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool ArrowMove(const long   chart_ID=0,   // ID des Charts
               const string name="Arrow", // Name des Objekts
              datetime     time=0,         // Zeitkoordinate des Bezugspunkts
               double       price=0)           // Preiskoordinate des Bezugspunkts
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Ändert Code des Pfeils                                           |
//+------------------------------------------------------------------+
bool ArrowCodeChange(const long   chart_ID=0,   // ID des Charts
                     const string name="Arrow", // Name des Objekts
                     const uchar  code=252)     // Code des Pfeils
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Code des Pfeils ändern
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ARROWCODE,code))
     {
      Print(__FUNCTION__,
            ": Code des Pfeils konnte nicht geändert werden! Fehlercode = ",GetLastErr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert Bindungsmethode                                           |
//+------------------------------------------------------------------+
bool ArrowAnchorChange(const long              chart_ID=0,        // ID des Charts
                       const string            name="Arrow",      // Name des Objekts
                       const ENUM_ARROW_ANCHOR anchor=ANCHOR_TOP) // Bindungsmethode
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bindungsmethode ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor))
     {
      Print(__FUNCTION__,
            ": Bindungsmethode konnte nicht geändert werden! Fehlercode = ",GetLastErr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht den Pfeil                                                 |
//+------------------------------------------------------------------+
bool ArrowDelete(const long   chart_ID=0,   // ID des Charts
                 const string name="Arrow") // Name des Pfeils
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Pfeil löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Pfeil konnte nicht gelöscht werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte |
//| setzt Standardwerte |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void ChangeArrowEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100 || InpPrice<0 || InpPrice>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
    ;}
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Größe des Arrays price
   int accuracy=1000;
//--- Arrays für Werte von Daten und Preise, die für Setzung und Änderung
//--- der Koordinaten von Ankerpunkte verwendet werden
   datetime date[];
   double   price[];
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(price,accuracy);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Füllen den Array der Preise
//--- den Minimal- und Maximalwert der Charts finden
   double max_price=ChartGetDouble(0,CHART_PRICE_MAX);
   double min_price=ChartGetDouble(0,CHART_PRICE_MIN);
//--- Schritte der Preisänderung finden und das Array füllen
   double step=(max_price-min_price)/accuracy;
   for(int i=0;i<accuracy;i++)
      price[i]=min_price+i*step;
//--- Punkte für das Zeichnen des Pfeils definieren
   int d=InpDate*(bars-1)/100;
```

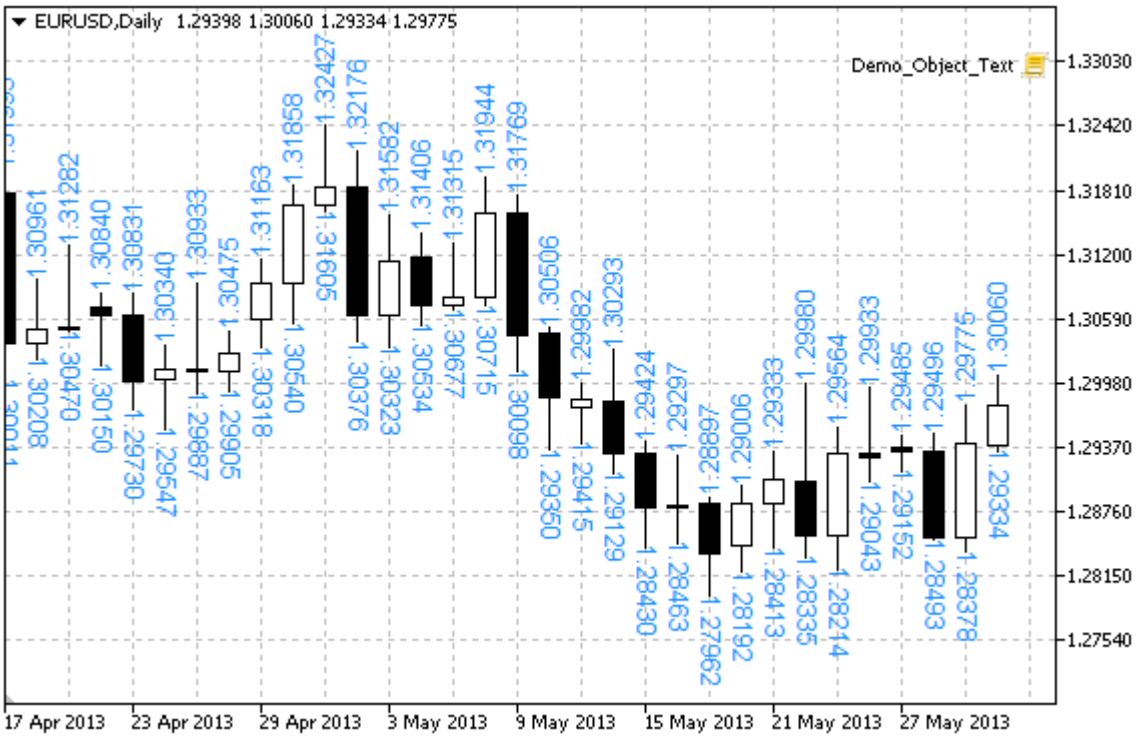
© 2000-2025, MetaQuotes Ltd.

```
   int p=InpPrice*(accuracy-1)/100;
//--- erstellen einen Pfeil auf dem Chart
   if(!ArrowCreate(0,InpName,0,date[d],price[p],32,InpAnchor,InpColor,
      InpStyle,InpWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen
   ChartRedraw();
//--- Alle Varianten von Erstellung der Pfeile in Zyklus betrachten
   for(int i=33;i<256;i++)
     {
      if(!ArrowCodeChange(0,InpName,(uchar)i))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung in einer halben Sekunde
      Sleep(500);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Pfeil aus dem Chart löschen
   ArrowDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_TEXT

Objekt "Text".



Hinweis

Sie können die Position des Ankerpunkts relativ zum Text aus der Enumeration ENUM_ANCHOR_POINT wählen. Sie können auch den Textwinkel durch die Eigenschaft OBJPROP_ANGLE geändert werden.

Beispiel

Das folgende Skript erstellt auf dem Chart einige Objekte "Text". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Text\"."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string             InpFont="Arial";          // Schrift
input int                InpFontSize=10;           // Schriftgröße
input color              InpColor=clrRed;          // Farbe
input double             InpAngle=90.0;            // Der Winkel in Grad
input ENUM_ANCHOR_POINT InpAnchor=ANCHOR_LEFT;    // Bindungsmethode
input bool               InpBack=false;            // Objekt im Hintergrund
input bool               InpSelection=false;       // Wählen zu bewegen
input bool               InpHidden=true;           // Ausblenden in der Liste der Objekt
```

© 2000-2025, MetaQuotes Ltd.

```
input long              InpZOrder=0;              // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt das Objekt "Text"                                       |
//+------------------------------------------------------------------+
bool TextCreate(const long              chart_ID=0,               // ID des Charts
                const string            name="Text",              // Objektname
                const int               sub_window=0,             // Nummer des Unteri
                datetime                time=0,                   // Zeit des Ankerpur
                double                  price=0,                  // Preis des Ankerpu
                const string            text="Text",              // Der Text
                const string            font="Arial",             // Schrift
                const int               font_size=10,             // Schriftgröße
                const color             clr=clrRed,               // Farbe
                const double            angle=0.0,                // Text Winkel
                const ENUM_ANCHOR_POINT anchor=ANCHOR_LEFT_UPPER, // Bindungsmethode
                const bool              back=false,               // Im Hintergrund
                const bool              selection=false,          // Wählen um zu bewe
                const bool              hidden=true,              // Ausgeblendet in d
                const long              z_order=0)                // Priorität auf Mau
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeTextEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erstellen wir das Objekt "Text"
   if(!ObjectCreate(chart_ID,name,OBJ_TEXT,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Objekt \"Text\" konnte nicht erstellt werden! Fehlercode = ",GetLastErr
      return(false);
     }
//--- den Text setzen
   ObjectSetString(chart_ID,name,OBJPROP_TEXT,text);
//--- Textschrift setzen
   ObjectSetString(chart_ID,name,OBJPROP_FONT,font);
//--- Schriftgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_FONTSIZE,font_size);
//--- Text-Winkel angeben
   ObjectSetDouble(chart_ID,name,OBJPROP_ANGLE,angle);
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Objekts mit dem Maus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt den Bezugspunkt                                           |
//+------------------------------------------------------------------+
bool TextMove(const long   chart_ID=0,  // ID des Charts
              const string name="Text", // Objektname
              datetime     time=0,      // Zeitkoordinate des Punktes
              double       price=0)     // Preiskoordinate des Punktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastErro
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Text des Objektes                                     |
//+------------------------------------------------------------------+
bool TextChange(const long   chart_ID=0,  // ID des Charts
                const string name="Text", // Objektname
                const string text="Text") // Text
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ändern wir den Text des Objektes
   if(!ObjectSetString(chart_ID,name,OBJPROP_TEXT,text))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Text ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
 ;}
//+------------------------------------------------------------------+
//| Löscht das Objekt "Text"                                         |
//+------------------------------------------------------------------+
bool TextDelete(const long   chart_ID=0,  // ID des Charts
                const string name="Text") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \"Text\" konnte nicht gelöscht werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte              |
//| setzt Standardwerte                                              |
//+------------------------------------------------------------------+
void ChangeTextEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bi
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date[]; // Array zu speichern Daten von sichtbaren Bars
   double   low[];  // Array zu speichern Low-Priese von sichtbaren Bars
   double   high[]; // Array zu speichern High-Preise von sichtbaren Bars
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(low,bars);
   ArrayResize(high,bars);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
       return;
      }
//--- Füllen die Anordnung von Low-Preise
   if(CopyLow(Symbol(),Period(),0,bars,low)==-1)
      {
       Print("Kann nicht den Wert der Low-Preise kopieren! Fehlercode = ",GetLastError
       return;
      }
//--- Füllen die Anordnung von High-Preise
   if(CopyHigh(Symbol(),Period(),0,bars,high)==-1)
      {
       Print("Kann nicht den Wert der High-Preise kopieren! Fehlercode = ",GetLastErro
       return;
      }
//--- Definieren wir, wie oft die Inschriften zu Tun
   int scale=(int)ChartGetInteger(0,CHART_SCALE);
//--- Schritt definieren
   int step=1;
   switch(scale)
      {
       case 0:
          step=12;
          break;
       case 1:
          step=6;
          break;
       case 2:
          step=4;
          break;
       case 3:
          step=2;
          break;
      }
//--- Text für die Werte von High und Low der Baren erstellen (mit Zwischenräume)
   for(int i=0;i<bars;i+=step)
      {
       //--- Texte erstellen
       if(!TextCreate(0,"TextHigh_"+(string)i,0,date[i],high[i],DoubleToString(high[i],
          InpColor,InpAngle,InpAnchor,InpBack,InpSelection,InpHidden,InpZOrder))
         {
          return;
         }
       if(!TextCreate(0,"TextLow_"+(string)i,0,date[i],low[i],DoubleToString(low[i],5),
          InpColor,-InpAngle,InpAnchor,InpBack,InpSelection,InpHidden,InpZOrder))
         {
          return;
         }
```

© 2000-2025, MetaQuotes Ltd.

```
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Texte löschen
   for(int i=0;i<bars;i+=step)
     {
      if(!TextDelete(0,"TextHigh_"+(string)i))
         return;
      if(!TextDelete(0,"TextLow_"+(string)i))
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_LABEL

Objekt "Text-Label."



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ANCHOR_POINT gewählt werden. Koordinaten des Bezugspunktes werden in Pixel angegeben.

Sie können den Winkel des Anbindepunktes des Labels durch Enumeration ENUM_BASE_CORNER wählen.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Text-Label". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Text-Label\"."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="Label";         // Name des Labels
input int               InpX=150;                // X Abstand
input int               InpY=150;                // Y Abstand
input string            InpFont="Arial";         // Schrift
input int               InpFontSize=14;          // Schriftgröße
input color             InpColor=clrRed;         // Farbe
input double            InpAngle=0.0;            // Der Winkel in Grad
```

© 2000-2025, MetaQuotes Ltd.

```
input ENUM_ANCHOR_POINT InpAnchor=ANCHOR_CENTER; // Bindungsmethode
input bool              InpBack=false;           // Objekt im Hintergrund
input bool              InpSelection=true;    // Wählen zu bewegen
input bool              InpHidden=true;       // Ausblenden in der Liste der Objekten
input long              InpZOrder=0;          // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt ein Text-Label                                          |
//+------------------------------------------------------------------+
bool LabelCreate(const long              chart_ID=0,               // ID des Charts
                 const string            name="Label",             // Name des Labels
                 const int               sub_window=0,             // Nummer des Unter
                 const int               x=0,                      // X-Koordinate
                 const int               y=0,                      // Y-Koordinate
                 const ENUM_BASE_CORNER  corner=CORNER_LEFT_UPPER, // Winkel des Chart
                 const string            text="Label",             // Text
                 const string            font="Arial",             // Schrift
                 const int               font_size=10,             // Schriftgröße
                 const color             clr=clrRed,               // Farbe
                 const double            angle=0.0,                // Text Winkel
                 const ENUM_ANCHOR_POINT anchor=ANCHOR_LEFT_UPPER, // Bindungsmethode
                 const bool              back=false,               // Im Hintergrund
                 const bool              selection=false,          // Wählen um zu bew
                 const bool              hidden=true,              // Ausgeblendet in
                 const long              z_order=0)                // Priorität auf Ma
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ein Text-Label erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_LABEL,sub_window,0,0))
     {
      Print(__FUNCTION__,
            ": Text-Label konnte nicht erstellt werden! Fehlercode = ",GetLastError())
      return(false);
     }
//--- Die Koordinaten des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y);
//--- wählen die Ecke des Charts, relativ zu der die Punktkoordinaten eingegeben werde
   ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner);
//--- den Text setzen
   ObjectSetString(chart_ID,name,OBJPROP_TEXT,text);
//--- Textschrift setzen
   ObjectSetString(chart_ID,name,OBJPROP_FONT,font);
//--- Schriftgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_FONTSIZE,font_size);
//--- Text-Winkel angeben
   ObjectSetDouble(chart_ID,name,OBJPROP_ANGLE,angle);
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt ein Text-Label                                            |
//+------------------------------------------------------------------+
bool LabelMove(const long   chart_ID=0,   // ID des Charts
               const string name="Label", // Name des Labels
               const int    x=0,          // X-Koordinate
               const int    y=0)          // Y-Koordinate
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ein Text-Label bewegen
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x))
     {
      Print(__FUNCTION__,
            ": Kann nicht X-Koordinate des Labels bewegen! Fehlercode = ",GetLastError
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y))
     {
      Print(__FUNCTION__,
            ": Kann nicht Y-Koordinate des Labels bewegen! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert Winkel des Charts das Label zu binden                     |
//+------------------------------------------------------------------+
bool LabelChangeCorner(const long             chart_ID=0,               // ID des Char
                       const string           name="Label",             // Name des La
                       const ENUM_BASE_CORNER corner=CORNER_LEFT_UPPER) // Winkel des
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Winkel der Bindung ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner))
     {
      Print(__FUNCTION__,
            ": Kann den Winkel der Bindung nicht ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Ändert den Text des Objektes                                     |
//+------------------------------------------------------------------+
bool LabelTextChange(const long   chart_ID=0,   // ID des Charts
                     const string name="Label", // Objektname
                     const string text="Text")  // Text
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ändern wir den Text des Objektes
   if(!ObjectSetString(chart_ID,name,OBJPROP_TEXT,text))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Text ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht ein Text-Label                                            |
//+------------------------------------------------------------------+
bool LabelDelete(const long   chart_ID=0,   // ID des Charts
                 const string name="Label") // Name des Labels
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Text-Label konnte nicht gelöscht werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Koordinaten des Labels in lokalen Variablen speichern
   int x=InpX;
   int y=InpY;
//--- Größe des Chartfensters
   long x_distance;
   long y_distance;
//--- Größe des Chartfensters definieren
   if(!ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0,x_distance))
     {
      Print("Konnte nicht Breite des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
   if(!ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0,y_distance))
     {
      Print("Konnte nicht Höhe des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpX<0 || InpX>x_distance-1 || InpY<0 || InpY>y_distance-1)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- den ersten Text für das Label vorbereiten
   string text;
   StringConcatenate(text,"Linke obere Ecke: ",x,",",y);
//--- ein Text-Label auf dem Chart erstellen
   if(!LabelCreate(0,InpName,0,InpX,InpY,CORNER_LEFT_UPPER,text,InpFont,InpFontSize,
      InpColor,InpAngle,InpAnchor,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- Den Chart neu zeichnen und halbe Sekunden warten
   ChartRedraw();
   Sleep(500);
//--- bewegen das Label und ändern seinen Text
//--- Anzahl der Iterationen für Achsen
   int h_steps=(int)(x_distance/2-InpX);
   int v_steps=(int)(y_distance/2-InpY);
//--- Text-Label nach unten bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- Koordinate ändern
      y+=2;
      //--- bewegen das Label und ändern seinen Text
      MoveAndTextChange(x,y,"Linke obere Ecke: ");
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Text-Label nach rechts bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- Koordinate ändern
      x+=2;
      //--- bewegen das Label und ändern seinen Text
      MoveAndTextChange(x,y,"Linke obere Ecke: ");
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Text-Label nach oben bewegen
   for(int i=0;i<v_steps;i++)
     {
      //--- Koordinate ändern
      y-=2;
      //--- bewegen das Label und ändern seinen Text
      MoveAndTextChange(x,y,"Linke obere Ecke: ");
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Text-Label nach links bewegen
   for(int i=0;i<h_steps;i++)
     {
      //--- Koordinate ändern
      x-=2;
      //--- bewegen das Label und ändern seinen Text
      MoveAndTextChange(x,y,"Linke obere Ecke: ");
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- nun bewegen wir den Punkt (Winkel der Bindung ändern)
//--- in den unteren linken Winkel bewegen
   if(!LabelChangeCorner(0,InpName,CORNER_LEFT_LOWER))
      return;
//--- ändern wir den Text des Labels
   StringConcatenate(text,"Linke untere Ecke: ",x,",",y);
   if(!LabelTextChange(0,InpName,text))
      return;
//--- Den Chart neu zeichnen und zwei Sekunden warten
   ChartRedraw();
   Sleep(2000);
//--- in den unteren rechten Winkel bewegen
   if(!LabelChangeCorner(0,InpName,CORNER_RIGHT_LOWER))
      return;
//--- ändern wir den Text des Labels
   StringConcatenate(text,"Rechte untere Ecke: ",x,",",y);
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!LabelTextChange(0,InpName,text))
      return;
//--- Den Chart neu zeichnen und zwei Sekunden warten
   ChartRedraw();
   Sleep(2000);
//--- in den oberen rechten Winkel bewegen
   if(!LabelChangeCorner(0,InpName,CORNER_RIGHT_UPPER))
      return;
//--- ändern wir den Text des Labels
   StringConcatenate(text,"Rechte obere Ecke: ",x,",",y);
   if(!LabelTextChange(0,InpName,text))
      return;
//--- Den Chart neu zeichnen und zwei Sekunden warten
   ChartRedraw();
   Sleep(2000);
//--- in den oberen linken Winkel bewegen
   if(!LabelChangeCorner(0,InpName,CORNER_LEFT_UPPER))
      return;
//--- ändern wir den Text des Labels
   StringConcatenate(text,"Linke obere Ecke: ",x,",",y);
   if(!LabelTextChange(0,InpName,text))
      return;
//--- Den Chart neu zeichnen und zwei Sekunden warten
   ChartRedraw();
   Sleep(2000);
//--- das Schild löschen
   LabelDelete(0,InpName);
//--- Den Chart neu zeichnen und halbe Sekunden warten
   ChartRedraw();
   Sleep(500);
//---
  }
//+------------------------------------------------------------------+
//| Die Funktion bewegt das Objekt und ändert seinen Text            |
//+------------------------------------------------------------------+
bool MoveAndTextChange(const int x,const int y,string text)
  {
//--- Label bewegen
   if(!LabelMove(0,InpName,x,y))
      return(false);
//--- ändern wir den Text des Labels
   StringConcatenate(text,text,x,",",y);
   if(!LabelTextChange(0,InpName,text))
      return(false);
//--- Überprüfen die Fakten der Zwangsabschaltung von Skript
   if(IsStopped())
      return(false);
//--- den Chart neu zeichnen
   ChartRedraw();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Verzögerung von 0.01 Sekunden
   Sleep(10);
//--- Funktion beendet
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_BUTTON

Objekt "Schaltfläche".



Hinweis

Koordinaten des Ankerpunkts werden in Pixeln angegeben. Sie können den Winkel des Ankerpunktes der Schaltfläche durch Enumeration ENUM_BASE_CORNER auswählen.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Schaltfläche". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt eine Schaltfläche auf dem Chart."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="Button";            // Schaltflächenname
input ENUM_BASE_CORNER InpCorner=CORNER_LEFT_UPPER; // Winkel des Charts zu binden
input string           InpFont="Arial";             // Schrift
input int              InpFontSize=14;              // Schriftgröße
input color            InpColor=clrBlack;           // Textfarbe
input color            InpBackColor=C'236,233,216'; // Hintergrundfarbe
input color            InpBorderColor=clrNONE;      // Rahmenfarbe
input bool             InpState=false;              // Gedrückt/Ungedrückt
input bool             InpBack=false;               // Objekt im Hintergrund
input bool             InpSelection=false;          // Wählen zu bewegen
```

© 2000-2025, MetaQuotes Ltd.

```
input bool              InpHidden=true;              // Ausgeblendet in der Liste der O
input long              InpZOrder=0;                 // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt eine Taste                                              |
//+------------------------------------------------------------------+
bool ButtonCreate(const long              chart_ID=0,               // ID des Charts
                  const string            name="Button",            // Schaltflächenna
                  const int               sub_window=0,             // Nummer des Unte
                  const int               x=0,                      // X-Koordinate
                  const int               y=0,                      // Y-Koordinate
                  const int               width=50,                 // Breite der Scha
                  const int               height=18,                // Höhe der Schalt
                  const ENUM_BASE_CORNER  corner=CORNER_LEFT_UPPER, // Winkel des Cha
                  const string            text="Button",            // Text
                  const string            font="Arial",             // Schrift
                  const int               font_size=10,             // Schriftgröße
                  const color             clr=clrBlack,             // Textfarbe
                  const color             back_clr=C'236,233,216',  // Hintergrundfarb
                  const color             border_clr=clrNONE,       // Rahmenfarbe
                  const bool              state=false,              // Gedrückt/Ungedr
                  const bool              back=false,               // Im Hintergrund
                  const bool              selection=false,          // Auswählen um zu
                  const bool              hidden=true,              // Ausgeblendet in
                  const long              z_order=0)                // Priorität auf M
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- eine Schaltfläche erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_BUTTON,sub_window,0,0))
     {
      Print(__FUNCTION__,
            ": Konnte nicht Schaltfläche erstellen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Schaltflächenkoordinaten setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y);
//--- Schaltflächengröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width);
   ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height);
//--- wählen die Ecke des Charts, relativ zu der die Punktkoordinaten eingegeben werde
   ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner);
//--- den Text setzen
   ObjectSetString(chart_ID,name,OBJPROP_TEXT,text);
//--- Textschrift setzen
   ObjectSetString(chart_ID,name,OBJPROP_FONT,font);
//--- Schriftgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_FONTSIZE,font_size);
//--- Textfarbe setzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Hintergrundfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_BGCOLOR,back_clr);
//--- Rahmenfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_BORDER_COLOR,border_clr);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- set button state
   ObjectSetInteger(chart_ID,name,OBJPROP_STATE,state);
//--- aktivieren (true) oder deaktivieren (false) Bewegung der Schaltfläche mit dem Ma
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Bewegt die Schaltfläche                                          |
//+------------------------------------------------------------------+
bool ButtonMove(const long   chart_ID=0,    // ID des Charts
                const string name="Button", // Name der Schaltfläche
                const int    x=0,           // X Koordinate
                const int    y=0)           // Y Koordinate
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Schaltfläche bewegen
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x))
     {
      Print(__FUNCTION__,
            ": Kann nicht X-Koordinate der Schaltfläche bewegen! Fehlercode = ",GetLas
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y))
     {
      Print(__FUNCTION__,
            ": Kann nicht Y-Koordinate der Schaltfläche bewegen! Fehlercode = ",GetLas
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Größe einer Schaltfläche                              |
//+------------------------------------------------------------------+
bool ButtonChangeSize(const long   chart_ID=0,    // ID des Charts
```

© 2000-2025, MetaQuotes Ltd.

```
                      const string name="Button", // Name der Schaltfläche
                      const int    width=50,      // Breite der Schaltfläche
                      const int    height=18)     // Höhe der Schaltfläche
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Größe der Schaltfläche ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Breite der Schaltfläche ändern! Fehlercode = ",GetLast
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Höhe der Schaltfläche ändern! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert Winkel des Charts die Taste zu binden                     |
//+------------------------------------------------------------------+
bool ButtonChangeCorner(const long             chart_ID=0,               // ID des Cha
                        const string           name="Button",            // Name der S
                        const ENUM_BASE_CORNER corner=CORNER_LEFT_UPPER) // Winkel des
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Winkel der Bindung ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner))
     {
      Print(__FUNCTION__,
            ": Kann den Winkel der Bindung nicht ändern! Fehlercode = ",GetLastError()
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert Text der Schaltfläche                                     |
//+------------------------------------------------------------------+
bool ButtonTextChange(const long   chart_ID=0,     // ID des Chart
                      const string name="Button", // Name der Schaltfläche
                      const string text="Text")   // Text
  {
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- ändern wir den Text des Objektes
   if(!ObjectSetString(chart_ID,name,OBJPROP_TEXT,text))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Text ändern! Fehlercode = ",GetLastError());
      return(false);
    ;}
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht die Schaltfläche                                          |
//+------------------------------------------------------------------+
bool ButtonDelete(const long   chart_ID=0,    // ID des Charts
                  const string name="Button") // Name der Schaltfläche
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- die Schaltfläche löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Könnte nicht die Schaltfläche löschen! Fehlercode = ",GetLastError());
      return(false);
    ;}
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Größe des Chartfensters
   long x_distance;
   long y_distance;
//--- Größe des Chartfensters definieren
   if(!ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0,x_distance))
     {
      Print("Konnte nicht Breite des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
   if(!ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0,y_distance))
     {
      Print("Konnte nicht Höhe des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
//--- Schritt für Änderung der Schaltflächengröße definieren
```

© 2000-2025, MetaQuotes Ltd.

```
   int x_step=(int)x_distance/32;
   int y_step=(int)y_distance/32;
//--- Schaltflächenkoordinaten und ihre Größe setzen
   int x=(int)x_distance/32;
   int y=(int)y_distance/32;
   int x_size=(int)x_distance*15/16;
   int y_size=(int)y_distance*15/16;
//--- eine Schaltfläche erstellen
   if(!ButtonCreate(0,InpName,0,x,y,x_size,y_size,InpCorner,"Press",InpFont,InpFontSiz
      InpColor,InpBackColor,InpBorderColor,InpState,InpBack,InpSelection,InpHidden,Inp
     {
      return;
    ;}
//--- den Chart neu zeichnen
   ChartRedraw();
//--- Verkleinern die Schaltfläche im Zyklus
   int i=0;
   while(i<13)
     {
      //--- Halb-Sekunden-Verzögerung
      Sleep(500);
      //--- Die Schaltfläche gedrückt setzen
      ObjectSetInteger(0,InpName,OBJPROP_STATE,true);
      //--- Die Grafik neu zeichnen und 0,2 Sekunden warten
      ChartRedraw();
      Sleep(200);
      //--- die Position und Größe der Schaltfläche neu definieren
      x+=x_step;
      y+=y_step;
      x_size-=x_step*2;
      y_size-=y_step*2;
      //--- Schaltfläche verkleinern
      ButtonMove(0,InpName,x,y);
      ButtonChangeSize(0,InpName,x_size,y_size);
      //--- Schaltfläche im ungedrückt Zustand setzen
      ObjectSetInteger(0,InpName,OBJPROP_STATE,false);
      //--- den Chart neu zeichnen
      ChartRedraw();
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- Zykluszähler erhöhen
      i++;
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- die Schaltfläche löschen
   ButtonDelete(0,InpName);
   ChartRedraw();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- 1 Sekunde warten
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_CHART

Objekt "Chart".



Hinweis

Koordinaten des Ankerpunkts werden in Pixeln angegeben. Sie können den Winkel des Bindungspunktes durch Enumeration ENUM_BASE_CORNER auswählen.

Für das Objekt "Chart" können Sie ein Symbol, Ausmaß und Periode Zeit, als auch aktivieren / deaktivieren Sie den Anzeigemodus der Skalen Preise und Termine auszuwählen.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Chart". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt das Objekt \"Chart\"."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="Chart";              // Objektname
input string           InpSymbol="EURUSD";           // Symbol
input ENUM_TIMEFRAMES  InpPeriod=PERIOD_H1;          // Periode
input ENUM_BASE_CORNER InpCorner=CORNER_LEFT_UPPER; // Winkel des Charts um Objekt zu
input int              InpScale=2;                   // Skala
input bool             InpDateScale=true;            // Anzeige der Zeitskala
input bool             InpPriceScale=true;           // Anzeige der Preisskala
```

© 2000-2025, MetaQuotes Ltd.

```
input color            InpColor=clrRed;             // Rahmenfarbe bei dem Wählen
input ENUM_LINE_STYLE  InpStyle=STYLE_DASHDOTDOT;   // Linienstil bei dem Wählen
input int              InpPointWidth=1;             // Größe des Punktes für Bewegung
input bool             InpBack=false;               // Objekt im Hintergrund
input bool             InpSelection=true;           // Wählen um zu bewegen
input bool             InpHidden=true;              // Ausgeblendet in der Liste der O
input long             InpZOrder=0;                 // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt das Objekt "Chart"                                      |
//+------------------------------------------------------------------+
bool ObjectChartCreate(const long              chart_ID=0,               // ID des Cha
                       const string            name="Chart",             // Objektname
                       const int               sub_window=0,             // Nummer des
                       const string            symbol="EURUSD",          // Symbol
                       const ENUM_TIMEFRAMES   period=PERIOD_H1,         // Periode
                       const int               x=0,                      // X-Koordina
                       const int               y=0,                      // Y-Koordina
                       const int               width=300,                // Breite
                       const int               height=200,               // Höhe
                       const ENUM_BASE_CORNER  corner=CORNER_LEFT_UPPER, // Winkel des
                       const int               scale=2,                  // Skala
                       const bool              date_scale=true,          // Anzeige de
                       const bool              price_scale=true,         // Anzeige de
                       const color             clr=clrRed,               // Rahmenfarb
                       const ENUM_LINE_STYLE   style=STYLE_SOLID,        // Linienstil
                       const int               point_width=1,            // Größe des
                       const bool              back=false,               // Im Hinterg
                       const bool              selection=false,          // Markieren
                       const bool              hidden=true,              // Ausgeblend
                       const long              z_order=0)                // Priorität
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erstellen wir das Objekt "Chart"
   if(!ObjectCreate(chart_ID,name,OBJ_CHART,sub_window,0,0))
     {
      Print(__FUNCTION__,
            ": Objekt \"Chart\" konnte nicht erstellt werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- Objektkoordinaten angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y);
//--- Objektgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width);
   ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height);
//--- wählen die Ecke des Charts, relativ zu der die Punktkoordinaten eingegeben werde
   ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner);
//--- den Symbol setzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetString(chart_ID,name,OBJPROP_SYMBOL,symbol);
//--- den Zeitraum setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_PERIOD,period);
//--- die Skala setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_CHART_SCALE,scale);
//--- die Zeitskala anzeigen (true) oder verbergen (false)
   ObjectSetInteger(chart_ID,name,OBJPROP_DATE_SCALE,date_scale);
//--- die Preisskala anzeigen (true) oder verbergen (false)
   ObjectSetInteger(chart_ID,name,OBJPROP_PRICE_SCALE,price_scale);
//--- Farbe des Rahmens bei der Markierung des Objektes
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil des Rahmens bei der Markierung des Objektes
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- die Größe des Punktes, den benutzt werden kann, um ein Objekt zu bewegen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,point_width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Setzt den Symbol und den Zeitraum des Objekts "Chart"            |
//+------------------------------------------------------------------+
bool ObjectChartSetSymbolAndPeriod(const long            chart_ID=0,        // ID des C
                                   const string          name="Chart",      // Objektna
                                   const string          symbol="EURUSD",   // Symbol
                                   const ENUM_TIMEFRAMES period=PERIOD_H1) // Zeitrahm
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Jetzt setzen wir den Symbol und den Zeitraum des Objekts "Chart"
   if(!ObjectSetString(chart_ID,name,OBJPROP_SYMBOL,symbol))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Symbol für Objekt \"Chart\" erstellen! Fehlercode = ",
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_PERIOD,period))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Zeitraum für Objekt \"Chart\" erstellen! Fehlercode =
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt das Objekt "Chart"                                        |
//+------------------------------------------------------------------+
bool ObjectChartMove(const long   chart_ID=0,   // ID des Charts (nicht des Objektes)
                     const string name="Chart", // Objektname
                     const int    x=0,          // X-Koordinate
                     const int    y=0)          // Y-Koordinate
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objekt bewegen
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x))
     {
      Print(__FUNCTION__,
            ": Kann nicht X-Koordinate des Objektes \"Chart\" bewegen! Fehlercode = ",
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y))
     {
      Print(__FUNCTION__,
            ": Kann nicht Y-Koordinate des Objektes \"Chart\" bewegen! Fehlercode = ",
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Größe des Objektes "Chart"                            |
//+------------------------------------------------------------------+
bool ObjectChartChangeSize(const long   chart_ID=0,   // ID des Charts (nicht des Obje
                           const string name="Chart", // Objektname
                           const int    width=300,    // Breite
                           const int    height=200)   // Höhe
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objektgröße ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Breite des Objekts \"Chart\" ändern! Fehlercode = ",Ge
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__,
            ": Konnte nicht die Höhe des Objekts \"Chart\" ändern! Fehlercode = ",GetL
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt ID des Objektes "Chart" zurück                              |
//+------------------------------------------------------------------+
long ObjectChartGetID(const long   chart_ID=0,   // ID des Charts (nicht des Objekts)
                      const string name="Chart") // Objektname
  {
//--- eine Variable vorbereiten um ID des Objekts "Chart" zu erhalten
   long id=-1;
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ID erhalten
   if(!ObjectGetInteger(chart_ID,name,OBJPROP_CHART_ID,0,id))
     {
      Print(__FUNCTION__,
            ": Konnte nicht ID des Objekts \"Chart\" erhalten! Fehlercode = ",GetLastE
     }
//--- Ergebnis zurückgeben
   return(id);
 ;}
//+------------------------------------------------------------------+
//| Löscht das Objekt "Chart"                                        |
//+------------------------------------------------------------------+
bool ObjectChartDelete(const long   chart_ID=0,   // ID des Charts (nicht des Objektes
                       const string name="Chart") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \"Chart\" konnte nicht gelöscht werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Die Anzahl der Symbolen in "Market Watch" erhalten
   int  symbols=SymbolsTotal(true);
//--- Bestimmen wir, ob es ein Symbol mit dem angegebenen Namen in der Liste der Symbo
   bool exist=false;
   for(int i=0;i<symbols;i++)
      if(InpSymbol==SymbolName(i,true))
        {
         exist=true;
         break;
        }
   if(!exist)
     {
      Print("Fehler! Dieses Symbol ",InpSymbol," ist nicht in \"Market Watch\" verfügb
      return;
    ;}
//--- Prüfung der Richtigkeit der Eingabeparameter
   if(InpScale<0 || InpScale>5)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }

//--- Größe des Chartfensters
   long x_distance;
   long y_distance;
//--- Größe des Chartfensters definieren
   if(!ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0,x_distance))
     {
      Print("Konnte nicht Breite des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
   if(!ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0,y_distance))
     {
      Print("Konnte nicht Höhe des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
//--- Objektkoordinaten und seine Größe setzen
   int x=(int)x_distance/16;
   int y=(int)y_distance/16;
   int x_size=(int)x_distance*7/16;
   int y_size=(int)y_distance*7/16;
//--- Erstellen wir das Objekt "Chart"
   if(!ObjectChartCreate(0,InpName,0,InpSymbol,InpPeriod,x,y,x_size,y_size,InpCorner,I
      InpPriceScale,InpColor,InpStyle,InpPointWidth,InpBack,InpSelection,InpHidden,Inp
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
```

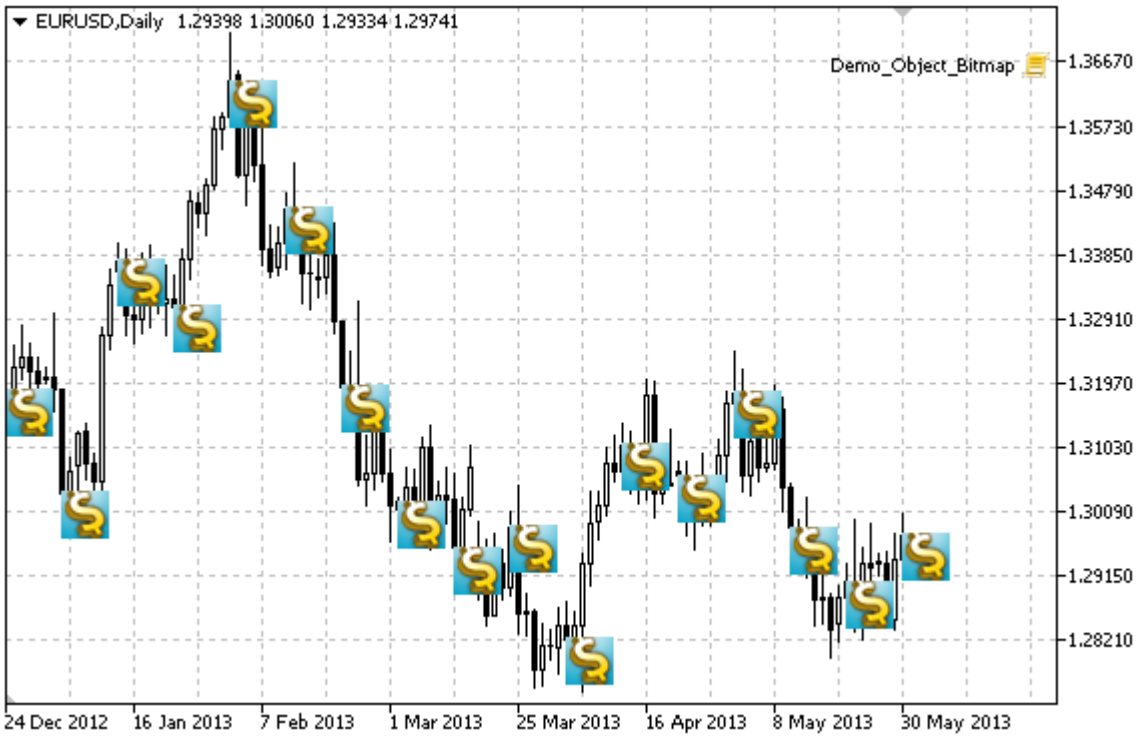
© 2000-2025, MetaQuotes Ltd.

```
   Sleep(1000);
//--- Dehnen wir das Objekt "Chart"
   int steps=(int)MathMin(x_distance*7/16,y_distance*7/16);
   for(int i=0;i<steps;i++)
     {
      //--- Größe ändern
      x_size+=1;
      y_size+=1;
      if(!ObjectChartChangeSize(0,InpName,x_size,y_size))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- Die Grafik neu zeichnen und 0,01 Sekunden warten
      ChartRedraw();
      Sleep(10);
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Zeitrahmen des Charts ändern
   if(!ObjectChartSetSymbolAndPeriod(0,InpName,InpSymbol,PERIOD_M1))
      return;
   ChartRedraw();
//--- Verzögerung von 3 Sekunden
   Sleep(3000);
//--- das Objekt löschen
   ObjectChartDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde warten
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_BITMAP

Objekt "Bild".



Hinweis

Für Objekt "Bild" können Sie Scope der Zeichnung.

Beispiel

Das folgende Skript erstellt auf dem Chart einige Bilder. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein Bild im Chartfenster."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string          InpFile="\\Images\\dollar.bmp"; // Dateiname mit dem Bild
input int             InpWidth=24;                    // X-Koordinate des Rahmens
input int             InpHeight=24;                   // Y-Koordinate des Rahmens
input int             InpXOffset=4;                   // X-Offset des Rahmens
input int             InpYOffset=4;                   // X-Offset des Rahmens
input color           InpColor=clrRed;                // Farbe des Rahmens bei der Zut
input ENUM_LINE_STYLE InpStyle=STYLE_SOLID;           // Linienstil bei der Zuteilung
input int             InpPointWidth=1;                // Größe des Punktes für Bewegur
input bool            InpBack=false;                  // Objekt im Hintergrund
input bool            InpSelection=false;             // Wählen zu bewegen
```

© 2000-2025, MetaQuotes Ltd.

```
input bool            InpHidden=true;                  // Ausgeblendet in der Liste der
input long            InpZOrder=0;                     // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erzeugt ein Bild im Chartfenster                                 |
//+------------------------------------------------------------------+
bool BitmapCreate(const long            chart_ID=0,        // ID des Charts
                  const string          name="Bitmap",     // Bildname
                  const int             sub_window=0,      // Nummer des Unterfensters
                  datetime              time=0,            // Zeit des Ankerpunktes
                  double                price=0,           // Preis des Ankerpunktes
                  const string          file="",           // Name der Bilddatei
                  const int             width=10,          // X-Koordinate des Gültigk
                  const int             height=10,         // Y-Koordinate des Gültigk
                  const int             x_offset=0,        // X-Verschiebung des Gülti
                  const int             y_offset=0,        // Y-Verschiebung des Gülti
                  const color           clr=clrRed,        //Rahmenfarbe bei dem Wähle
                  const ENUM_LINE_STYLE style=STYLE_SOLID, // Linienstil bei dem Wähle
                  const int             point_width=1,     // Größe des Ankerpunktes
                  const bool            back=false,        // Im Hintergrund
                  const bool            selection=false,   // Auswählen um zu bewegen
                  const bool            hidden=true,       // Ausgeblendet in der List
                  const long            z_order=0)         // Priorität auf Mausklick
  {
//--- Die Koordinaten des Bezugspunkts angeben, wenn sie nicht gesetzt sind
   ChangeBitmapEmptyPoint(time,price);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erstellen wir ein Bild
   if(!ObjectCreate(chart_ID,name,OBJ_BITMAP,sub_window,time,price))
     {
      Print(__FUNCTION__,
            ": Konnte nicht das Bild im Chartfenster erstellen! Fehlercode = ",GetLast
      return(false);
     }
//--- Pfad zur Datei mit dem Bild setzen
   if(!ObjectSetString(chart_ID,name,OBJPROP_BMPFILE,file))
     {
      Print(__FUNCTION__,
            ": Bild konnte nicht geladen werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- Bildrahmen eingebenö wenn die Werte von Breite oder Höhe
//--- größer als die Breite und Höhe des ursprünglichen Bildes sind, dann
//--- wird es nicht gezeichnet; wenn die Werte von Breite und Höhe sind kleiner als di
//--- dann sein Teil, der diesen Größen entspricht, gezeichnet wird
   ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width);
   ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height);
//--- setzen den Teil des Bildes, der in den Rahmen angezeigt werden soll
//--- Standardmäßig ist es der linke obere Bereich des Bildes; durch den Wert
```

© 2000-2025, MetaQuotes Ltd.

```
//--- kann es von diesem Bild bewegt werden
   ObjectSetInteger(chart_ID,name,OBJPROP_XOFFSET,x_offset);
   ObjectSetInteger(chart_ID,name,OBJPROP_YOFFSET,y_offset);
//--- Farbe des Rahmens bei der Markierung des Objektes
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil des Rahmens bei der Markierung des Objektes
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- die Größe des Punktes, den benutzt werden kann, um ein Objekt zu bewegen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,point_width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Setzt ein neues Bild für die Zeichnung                           |
//+------------------------------------------------------------------+
bool BitmapSetImage(const long   chart_ID=0,    // ID des Charts
                    const string name="Bitmap", // Name der Zeichnung
                    const string file="")       // Pfad zur Datei
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Pfad zur Datei mit dem Bild setzen
   if(!ObjectSetString(chart_ID,name,OBJPROP_BMPFILE,file))
     {
      Print(__FUNCTION__,
            ": Bild konnte nicht geladen werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt ein Bild im Chartfenster                                  |
//+------------------------------------------------------------------+
bool BitmapMove(const long   chart_ID=0,    // ID des Charts
                const string name="Bitmap", // Bildname
                datetime     time=0,        // Zeit des Ankerpunktes
                double       price=0)       // Preis des Ankerpunktes
  {
//--- Wenn die Koordinaten des Punktes nicht angegeben sind, dann verschieben wir es a
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!time)
      time=TimeCurrent();
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- bewegen den Bezugspunkt
   if(!ObjectMove(chart_ID,name,0,time,price))
     {
      Print(__FUNCTION__,
            ": der Bezugspunkt konnte nicht bewegt werden! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Größe von Gültigkeitsbereich (Bildgröße)              |
//+------------------------------------------------------------------+
bool BitmapChangeSize(const long   chart_ID=0,     // ID des Charts
                      const string name="Bitmap", // Bildname
                      const int    width=0,       // Breite des Bildes
                      const int    height=0)      // Höhe des Bildes
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bildgröße ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Breite des Bildes ändern! Fehlercode = ",GetLastError
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Höhe des Bildes ändern! Fehlercode = ",GetLastError())
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Koordinate der oberen linken Ecke von Bildrahmen      |
//+------------------------------------------------------------------+
bool BitmapMoveVisibleArea(const long   chart_ID=0,     // ID des Charts
                           const string name="Bitmap", // Bildname
                           const int    x_offset=0,    // X-Koordinate des Gültigkeits
                           const int    y_offset=0)    // Y-Koordinate des Gültigkeits
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Koordinaten des Gültigkeitsbereichs des Bildes ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XOFFSET,x_offset))
     {
      Print(__FUNCTION__,
            ": X-Koordinate des Objektrahmens konnte nicht geändert werden! Fehlercode
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YOFFSET,y_offset))
     {
      Print(__FUNCTION__,
            ": Y-Koordinate des Objektrahmens konnte nicht geändert werden! Fehlercode
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Bild                                                  |
//+------------------------------------------------------------------+
bool BitmapDelete(const long   chart_ID=0,    // ID des Charts
                  const string name="Bitmap") // Bildname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Könnte nicht ein Bild löschen! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Überprüft den Wert des Bezugspunkts für lehre Werte |
//| setzt Standardwerte |
//+------------------------------------------------------------------+
void ChangeBitmapEmptyPoint(datetime &time,double &price)
  {
//--- Wenn Punktzeit nicht angegeben ist, wird es auf dem aktuellen Bar sein
   if(!time)
      time=TimeCurrent();
//--- Wenn der Preis des Punktes nicht angegeben wird, dann wird es einen Wert von Bid
   if(!price)
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date[];  // Array speichert Daten von sichtbaren Bars
   double   close[]; // Array speichert Close-Preise
//--- Name der Bilddatei
   string   file="\\Images\\dollar.bmp";
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Speicher reservieren
   ArrayResize(date,bars);
   ArrayResize(close,bars);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Array mit Close-Preise Füllen
   if(CopyClose(Symbol(),Period(),0,bars,close)==-1)
     {
      Print("Kann nicht die Werte der Close-Preise kopieren! Fehlercode = ",GetLastErr
      return;
     }
//--- Definieren wir, wie oft das Bild anzuzeigen
   int scale=(int)ChartGetInteger(0,CHART_SCALE);
//--- Schritt definieren
   int step=1;
   switch(scale)
     {
      case 0:
         step=27;
         break;
      case 1:
         step=14;
         break;
      case 2:
         step=7;
         break;
      case 3:
         step=4;
         break;
      case 4:
         step=2;
         break;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Bilder für die Werte von High und Low der Baren erstellen (mit Zwischenräume)
   for(int i=0;i<bars;i+=step)
     {
      //--- Bilder erstellen
      if(!BitmapCreate(0,"Bitmap_"+(string)i,0,date[i],close[i],InpFile,InpWidth,InpHe
         InpYOffset,InpColor,InpStyle,InpPointWidth,InpBack,InpSelection,InpHidden,Inp
        {
         return;
       ;}
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Zeichen "Sell" löschen
   for(int i=0;i<bars;i+=step)
     {
      if(!BitmapDelete(0,"Bitmap_"+(string)i))
         return;
      if(!BitmapDelete(0,"Bitmap_"+(string)i))
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_BITMAP_LABEL

Objekt "Bitmap Label".



Hinweis

Die Position des Bezugspunktes relativ zum Zeichen kann aus der Enumeration ENUM_ANCHOR_POINT gewählt werden. Koordinaten des Bezugspunktes werden in Pixel angegeben.

Sie können auch den Bezugswinkel von Bitmap Label aus der Enumeration ENUM_BASE_CORNER wählen.

Für Objekt "Bitmap Label" können Sie Scope der Zeichnung.

Beispiel

Das folgende Skript erstellt auf dem Chart einige Bilder. Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt Objekt \"Bitmap Label\"."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="BmpLabel";              // Name von Label
input string            InpFileOn="\\Images\\dollar.bmp"; // Der Dateiname für Modus C
input string            InpFileOff="\\Images\\euro.bmp";  // Der Dateiname für Modus C
input bool              InpState=false;                   // Label gedrückt/ungedrückt
input ENUM_BASE_CORNER  InpCorner=CORNER_LEFT_UPPER;      // Der Winkel des Charts für
input ENUM_ANCHOR_POINT InpAnchor=ANCHOR_CENTER;          // Bezugsmethode
```

© 2000-2025, MetaQuotes Ltd.

```
input color             InpColor=clrRed;                  // Farbe des Rahmens bei der
input ENUM_LINE_STYLE   InpStyle=STYLE_SOLID;             // Linienstil bei der Markie
input int               InpPointWidth=1;                  // Größe des Punktes zu bewe
input bool              InpBack=false;                    // Objekt im Hintergrund
input bool              InpSelection=false;               // Wählen zu bewegen
input bool              InpHidden=true;                   // Ausgeblendet in der Liste
input long              InpZOrder=0;                      // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt das Objekt "Bitmap Label"                               |
//+------------------------------------------------------------------+
bool BitmapLabelCreate(const long              chart_ID=0,               // ID des Cha
                       const string            name="BmpLabel",          // Name des O
                       const int               sub_window=0,             // Nummer des
                       const int               x=0,                      // X-Koordina
                       const int               y=0,                      // Y-Koordina
                       const string            file_on="",               // Bild im Mo
                       const string            file_off="",              // Bild im Mo
                       const int               width=0,                  // X-Koordina
                       const int               height=0,                 // Y-Koordina
                       const int               x_offset=10,              // X-Offset d
                       const int               y_offset=10,              // Y-Offset d
                       const bool              state=false,              // gedruckt/u
                       const ENUM_BASE_CORNER  corner=CORNER_LEFT_UPPER, // Ecke des O
                       const ENUM_ANCHOR_POINT anchor=ANCHOR_LEFT_UPPER, // Bindungsme
                       const color             clr=clrRed,               // Rahmenfarb
                       const ENUM_LINE_STYLE   style=STYLE_SOLID,        // Linienstil
                       const int               point_width=1,            // Größe des
                       const bool              back=false,               // Im Hinterg
                       const bool              selection=false,          // Markieren
                       const bool              hidden=true,              // Ausgeblend
                       const long              z_order=0)                // Priorität
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Bitmap Label erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_BITMAP_LABEL,sub_window,0,0))
     {
      Print(__FUNCTION__,
            ": Objekt \"Bitmap Label\" konnte nicht erstellt werden! Fehlercode = ",Ge
      return(false);
     }
//--- Wählen Bilder für Modi On und Off
   if(!ObjectSetString(chart_ID,name,OBJPROP_BMPFILE,0,file_on))
     {
      Print(__FUNCTION__,
            ": Bild für Modus On konnte nicht erstellt werden! Fehlercode = ",GetLastE
      return(false);
     }
   if(!ObjectSetString(chart_ID,name,OBJPROP_BMPFILE,1,file_off))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print(__FUNCTION__,
             ": Bild für Modus Off konnte nicht erstellt werden! Fehlercode = ",GetLast
       return(false);
      }
//--- Die Koordinaten des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y);
//--- Bildrahmen eingebenö wenn die Werte von Breite oder Höhe
//--- größer als die Breite und Höhe des ursprünglichen Bildes sind, dann
//--- wird es nicht gezeichnet; wenn die Werte von Breite und Höhe sind kleiner als di
//--- dann sein Teil, der diesen Größen entspricht, gezeichnet wird
   ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width);
   ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height);
//--- setzen den Teil des Bildes, der in den Rahmen angezeigt werden soll
//--- Standardmäßig ist es der linke obere Bereich des Bildes; durch den Wert
//--- kann es von diesem Bild bewegt werden
   ObjectSetInteger(chart_ID,name,OBJPROP_XOFFSET,x_offset);
   ObjectSetInteger(chart_ID,name,OBJPROP_YOFFSET,y_offset);
//--- Zustand von Label (gedrückt oder nicht)
   ObjectSetInteger(chart_ID,name,OBJPROP_STATE,state);
//--- wählen die Ecke des Charts, relativ zu der die Punktkoordinaten eingegeben werde
   ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner);
//--- die Bindungsmethode setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ANCHOR,anchor);
//--- Farbe des Rahmens bei der Markierung des Objektes
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil des Rahmens bei der Markierung des Objektes
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- die Größe des Punktes, den benutzt werden kann, um ein Objekt zu bewegen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,point_width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ein neues Bild für "Bitmap Label" setzen                         |
//+------------------------------------------------------------------+
bool BitmapLabelSetImage(const long   chart_ID=0,      // ID des Charts
                         const string name="BmpLabel", // Name des Objekts
                         const int    on_off=0,        // Modifier (On oder Off)
```

© 2000-2025, MetaQuotes Ltd.

```
                          const string file="")         // Pfad zur Datei
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Pfad zur Datei mit dem Bild setzen
   if(!ObjectSetString(chart_ID,name,OBJPROP_BMPFILE,on_off,file))
     {
      Print(__FUNCTION__,
            ": Bild konnte nicht geladen werden! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Verschiebt das Objekt "Bitmap Label"                             |
//+------------------------------------------------------------------+
bool BitmapLabelMove(const long   chart_ID=0,      // ID des Charts
                     const string name="BmpLabel", // Name des Labels
                     const int    x=0,             // X-Koordinate
                     const int    y=0)             // Y-Koordinate
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objekt bewegen
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x))
     {
      Print(__FUNCTION__,
            ": X-Koordinate des Objekts konnte nicht bewegt werden! Fehlercode = ",Get
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y))
     {
      Print(__FUNCTION__,
            ": Y-Koordinate des Objekts konnte nicht bewegt werden! Fehlercode = ",Get
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert Rahmen des Objekts (Objektgröße)                          |
//+------------------------------------------------------------------+
bool BitmapLabelChangeSize(const long   chart_ID=0,      // ID des Charts
                     const string name="BmpLabel", // Name des Labels
                           const int    width=0,         // Breite von Label
                           const int    height=0)        // Höhe von Label
  {
//--- Setzen den Wert des Fehlers zurück
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Objektgröße ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width))
     {
      Print(__FUNCTION__,
            ": Breite des Objekts konnte nicht geändert werden! Fehlercode = ",GetLast
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height))
     {
      Print(__FUNCTION__,
            ": Höhe des Objekts konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Koordinate der oberen linken Ecke von Bildrahmen      |
//+------------------------------------------------------------------+
bool BitmapLabelMoveVisibleArea(const long   chart_ID=0,      // ID des Charts
                                const string name="BmpLabel", // Name des Objekts
                                const int    x_offset=0,      // X-Koordinate des Rahm
                                const int    y_offset=0)      // Y-Koordinate des Rahm
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Koordinaten des Objektrahmens ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XOFFSET,x_offset))
     {
      Print(__FUNCTION__,
            ": X-Koordinate des Objektrahmens konnte nicht geändert werden! Fehlercode
      return(false);
     ;}
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YOFFSET,y_offset))
     {
      Print(__FUNCTION__,
            ": Y-Koordinate des Objektrahmens konnte nicht geändert werden! Fehlercode
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Bitmap Label"                                 |
//+------------------------------------------------------------------+
bool BitmapLabelDelete(const long   chart_ID=0,      // ID des Charts
                       const string name="BmpLabel") // Name des Objekts
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \"Bitmap Label\" konnte nicht gelöscht werden! Fehlercode = ",Ge
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Größe des Chartfensters
   long x_distance;
   long y_distance;
//--- Größe des Chartfensters definieren
   if(!ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0,x_distance))
     {
      Print("Konnte nicht Breite des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
   if(!ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0,y_distance))
     {
      Print("Konnte nicht Höhe des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
//--- Die Koordinaten des Objekts definieren
   int x=(int)x_distance/2;
   int y=(int)y_distance/2;
//--- Objektgröße und Koordinate des Rahmens setzen
   int width=32;
   int height=32;
   int x_offset=0;
   int y_offset=0;
//--- Bitmap Label auf der Mitte des Fensters platzieren
   if(!BitmapLabelCreate(0,InpName,0,x,y,InpFileOn,InpFileOff,width,height,x_offset,y_
      InpCorner,InpAnchor,InpColor,InpStyle,InpPointWidth,InpBack,InpSelection,InpHidd
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Größe des Objektrahmens im Loop ändern
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0;i<6;i++)
     {
      //--- Größe des Objektrahmens ändern
      width--;
      height--;
      if(!BitmapLabelChangeSize(0,InpName,width,height))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.3 Sekunden
      Sleep(300);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Koordinaten des Objektrahmens im Loop ändern
   for(int i=0;i<2;i++)
     {
      //--- Koordinaten des Objektrahmens ändern
      x_offset++;
      y_offset++;
      if(!BitmapLabelMoveVisibleArea(0,InpName,x_offset,y_offset))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.3 Sekunden
      Sleep(300);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- das Schild löschen
   BitmapLabelDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_EDIT

Object "Eingabefeld".



Hinweis

Koordinaten des Ankerpunkts werden in Pixeln angegeben. Sie können den Winkel des Ankerpunktes des Objekts "Eingabefeld" durch Enumeration ENUM_BASE_CORNER auswählen.

Sie können auch eine der Ausrichtung von Text innerhalb der "Eingabefelds" in Enumeration ENUM_ALIGN_MODE wählen.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Eingabefeld". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt das Objekt \"Eingabefeld\"."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string            InpName="Edit";              // Objektname
input string            InpText="Text";              // Objekttext
input string            InpFont="Arial";             // Schrift
input int               InpFontSize=14;              // Schriftgröße
input ENUM_ALIGN_MODE   InpAlign=ALIGN_CENTER;       // Textausrichtung
input bool              InpReadOnly=false;           // Bearbeitung
input ENUM_BASE_CORNER  InpCorner=CORNER_LEFT_UPPER; // Winkel des Charts zu binden
input color             InpColor=clrBlack;           // Textfarbe
```

© 2000-2025, MetaQuotes Ltd.

```
input color             InpBackColor=clrWhite;       // Hintergrundfarbe
input color             InpBorderColor=clrBlack;     // Rahmenfarbe
input bool              InpBack=false;               // Objekt im Hintergrund
input bool              InpSelection=false;          // Wählen zu bewegen
input bool              InpHidden=true;              // Ausgeblendet in der Liste der O
input long              InpZOrder=0;                 // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt das Objekt "Eingabefeld"                                |
//+------------------------------------------------------------------+
bool EditCreate(const long             chart_ID=0,               // ID des Charts
                const string           name="Edit",              // Objektname
                const int              sub_window=0,             // Nummer des Unterfe
                const int              x=0,                      // X-Koordinate
                const int              y=0,                      // Y-Koordinate
                const int              width=50,                 // Breite
                const int              height=18,                // Höhe
                const string           text="Text",              // Text
                const string           font="Arial",             // Schrift
                const int              font_size=10,             // Schriftgröße
                const ENUM_ALIGN_MODE  align=ALIGN_CENTER,       // Textausrichtung
                const bool             read_only=false,          // Bearbeitung
                const ENUM_BASE_CORNER corner=CORNER_LEFT_UPPER, // Winkel des Charts
                const color            clr=clrBlack,             // Textfarbe
                const color            back_clr=clrWhite,        // Hintergrundfarbe
                const color            border_clr=clrNONE,       // Rahmenfarbe
                const bool             back=false,               // Im Hintergrund
                const bool             selection=false,          // Wählen um zu beweg
                const bool             hidden=true,              // Ausgeblendet in de
                const long             z_order=0)                // Priorität auf Maus
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Eingabefeld erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_EDIT,sub_window,0,0))
     {
      Print(__FUNCTION__,
            ": Objekt \"Eingabefeld\" konnte nicht erstellt werden! Fehlercode = ",Get
      return(false);
     }
//--- Objektkoordinaten angeben
   ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y);
//--- Objektgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width);
   ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height);
//--- den Text setzen
   ObjectSetString(chart_ID,name,OBJPROP_TEXT,text);
//--- Textschrift setzen
   ObjectSetString(chart_ID,name,OBJPROP_FONT,font);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Schriftgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_FONTSIZE,font_size);
//--- Modus von Textausrichtung im Objekt setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_ALIGN,align);
//--- aktivieren (true) oder deaktivieren (false) den schreibgeschützten Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_READONLY,read_only);
//--- setzen die Ecke des Diagramms, in Bezug auf die die Koordinaten des Objekts best
   ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner);
//--- Textfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Hintergrundfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_BGCOLOR,back_clr);
//--- Rahmenfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_BORDER_COLOR,border_clr);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegt das Objekt "Eingabefeld"                                  |
//+------------------------------------------------------------------+
bool EditMove(const long   chart_ID=0,  // ID des Charts
              const string name="Edit", // Objektname
              const int    x=0,         // X-Koordinate
              const int    y=0)         // Y-Koordinate
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objekt bewegen
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x))
     {
      Print(__FUNCTION__,
            ": X-Koordinate des Objekts konnte nicht bewegt werden! Fehlercode = ",Get
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y))
     {
      Print(__FUNCTION__,
            ": Y-Koordinate des Objekts konnte nicht bewegt werden! Fehlercode = ",Get
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Größe des Objektes "Eingabefeld"                      |
//+------------------------------------------------------------------+
bool EditChangeSize(const long   chart_ID=0,  // ID des Charts
                    const string name="Edit", // Objektname
                    const int    width=0,     // Breite
                    const int    height=0)    // Höhe
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objektgröße ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width))
     {
      Print(__FUNCTION__,
            ": Breite des Objekts konnte nicht geändert werden! Fehlercode = ",GetLast
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height))
     {
      Print(__FUNCTION__,
            ": Höhe des Objekts konnte nicht geändert werden! Fehlercode = ",GetLastEr
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Text des Objektes "Eingabefeld"                       |
//+------------------------------------------------------------------+
bool EditTextChange(const long   chart_ID=0,  // ID des Charts
                    const string name="Edit", // Objektname
                    const string text="Text") // Text
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ändern wir den Text des Objektes
   if(!ObjectSetString(chart_ID,name,OBJPROP_TEXT,text))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Text ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Gibt den Text des Objektes "Eingabefeld" zurück                  |
//+------------------------------------------------------------------+
bool EditTextGet(string      &text,         // Text
                 const long   chart_ID=0,  // ID des Charts
                 const string name="Edit") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- erhalten wir den Text des Objektes
   if(!ObjectGetString(chart_ID,name,OBJPROP_TEXT,0,text))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Text erhalten! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Eingabefeld"                                  |
//+------------------------------------------------------------------+
bool EditDelete(const long   chart_ID=0,  // ID des Charts
                const string name="Edit") // Objektname
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
            ": Objekt \"Eingabefeld\" konnte nicht gelöscht werden! Fehlercode = ",Get
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Größe des Chartfensters
   long x_distance;
   long y_distance;
//--- Größe des Chartfensters definieren
   if(!ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0,x_distance))
     {
      Print("Konnte nicht Breite des Charts bekommen! Fehlercode = ",GetLastError());
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
    ;}
   if(!ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0,y_distance))
     {
      Print("Konnte nicht Höhe des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
//--- Schritt für Änderung der Eingabefeldgröße definieren
   int x_step=(int)x_distance/64;
//--- Eingabefeldkoordinaten und seine Größe setzen
   int x=(int)x_distance/8;
   int y=(int)y_distance/2;
   int x_size=(int)x_distance/8;
   int y_size=InpFontSize*2;
//--- den Text in einer lokalen Variable speichern
   string text=InpText;
//--- Eingabefeld erstellen
   if(!EditCreate(0,InpName,0,x,y,x_size,y_size,InpText,InpFont,InpFontSize,InpAlign,I
      InpCorner,InpColor,InpBackColor,InpBorderColor,InpBack,InpSelection,InpHidden,In
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Eingabefeld strecken
   while(x_size-x<x_distance*5/8)
     {
      //--- Breite der Eingabefeld erhöhen
      x_size+=x_step;
      if(!EditChangeSize(0,InpName,x_size,y_size))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- Die Grafik neu zeichnen und 0,05 Sekunden warten
      ChartRedraw();
      Sleep(50);
     }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Text ändern
   for(int i=0;i<20;i++)
     {
      //--- "+" am Anfang und am Ende hinzufügen
      text="+"+text+"+";
      if(!EditTextChange(0,InpName,text))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
```

© 2000-2025, MetaQuotes Ltd.

```
         return;
       //--- Die Grafik neu zeichnen und 0,1 Sekunden warten
       ChartRedraw();
       Sleep(100);
      }
//--- Halb-Sekunden-Verzögerung
   Sleep(500);
//--- Eingabefeld löschen
   EditDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde warten
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_EVENT

Objekt "Ereignis".



Hinweis

Wenn Sie den Mauszeiger auf das Ereignis schweben, erscheint sein Text.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Ereignis". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Ereignis\"."
#property description "Datum des Ankerpunkts wird als Prozentsatz"
#property description "des Breite des Chartfensters angegeben."
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="Event";     // Name des Ereignisses
input int              InpDate=25;          // Datum des Ereignisses in %
input string           InpText="Text";      // Text des Ereignisses
input color            InpColor=clrRed;     // Farbe des Ereignisses
input int              InpWidth=1;          // Größe des Punktes für Bewegung
input bool             InpBack=false;       //Ereignis Im Hintergrund
input bool             InpSelection=false; // Wählen um zu bewegen
input bool             InpHidden=true;      // Ausgeblendet in der Objektliste
```

© 2000-2025, MetaQuotes Ltd.

```
input long            InpZOrder=0;        // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt das Objekt "Ereignis" auf dem Chart                     |
//+------------------------------------------------------------------+
bool EventCreate(const long            chart_ID=0,       // ID des Charts
                 const string          name="Event",     // Name des Ereignisses
                 const int             sub_window=0,     // Nummer des Unterfensters
                 const string          text="Text",      // Text des Ereignisses
                 datetime              time=0,           // Zeit
                 const color           clr=clrRed,       // Farbe
                 const int             width=1,          // Punktdicke bei Wählen
                 const bool            back=false,       // Im Hintergrund
                 const bool            selection=false,  // Wählen um zu bewegen
                 const bool            hidden=true,      // Ausgeblendet in der Objektl
                 const long            z_order=0)        // Priorität auf Mausklick
  {
//--- wenn die Zeit nicht angegeben ist, wird Objekt auf dem letzten Balken erstellt w
   if(!time)
      time=TimeCurrent();
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Erstellen wir das Objekt "Ereignis"
   if(!ObjectCreate(chart_ID,name,OBJ_EVENT,sub_window,time,0))
     {
      Print(__FUNCTION__,
            ": Objekt \"Ereignis\" konnte nicht erstellt werden! Fehlercode = ",GetLas
      return(false);
     }
//--- den Text des Ereignisses setzen
   ObjectSetString(chart_ID,name,OBJPROP_TEXT,text);
//--- Farbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Dicke des Ankerpunktes, wenn ein Objekt ausgewählt ist, setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- aktivieren (true) oder deaktivieren (false) Bewegung des Ereignisses mit dem Mau
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Text des Objektes "Ereignis"                          |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool EventTextChange(const long   chart_ID=0,   // ID des Charts
                     const string name="Event", // Name des Ereignisses
                     const string text="Text")  // Text
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ändern wir den Text des Objektes
   if(!ObjectSetString(chart_ID,name,OBJPROP_TEXT,text))
     {
      Print(__FUNCTION__,
            ": Konnte nicht den Text ändern! Fehlercode = ",GetLastError());
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Bewegung des Objekts "Ereignis"                                  |
//+------------------------------------------------------------------+
bool EventMove(const long   chart_ID=0,   // ID des Charts
               const string name="Event", // Name des Ereignisses
               datetime     time=0)       // Zeit
  {
//--- wenn die Zeit nicht angegeben ist, bewegen wir das Ereignis auf den letzten Bal
   if(!time)
      time=TimeCurrent();
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Objekt bewegen
   if(!ObjectMove(chart_ID,name,0,time,0))
     {
      Print(__FUNCTION__,
            ": Objekt \"Ereignis\" konnte nicht bewegt werden! Fehlercode = ",GetLastE
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht das Objekt "Ereignis"                                     |
//+------------------------------------------------------------------+
bool EventDelete(const long   chart_ID=0,   // ID des Charts
                 const string name="Event") // Name des Ereignisses
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Objekt löschen
   if(!ObjectDelete(chart_ID,name))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print(__FUNCTION__,
            ": Objekt \"Ereignis\" konnte nicht gelöscht werden! Fehlercode = ",GetLas
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Überprüfen die Richtigkeit der Eingabeparameter
   if(InpDate<0 || InpDate>100)
     {
      Print("Fehler! Ungültige Eingabeparameter! ");
      return;
     }
//--- Die Anzahl der sichtbaren Bars im Chart-Fenster
   int bars=(int)ChartGetInteger(0,CHART_VISIBLE_BARS);
//--- Array speichert Datumswerte, die verwendet werden,
//--- um die Koordinaten der Ankerpunkten des Objekts "Ereignis" zu setzen und zu bewe
   datetime date[];
//--- Speicher reservieren
   ArrayResize(date,bars);
//--- Füllen die Anordnung von Daten
   ResetLastError();
   if(CopyTime(Symbol(),Period(),0,bars,date)==-1)
     {
      Print("Kann nicht den Wert der Zeit kopieren! Fehlercode = ",GetLastError());
      return;
     }
//--- Definieren die Punkte um das Objekt zu erstellen
   int d=InpDate*(bars-1)/100;
//--- Erstellen wir das Objekt "Ereignis"
   if(!EventCreate(0,InpName,0,InpText,date[d],InpColor,InpWidth,
      InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- jetzt bewegen wir das Objekt
//--- Schleifenzähler
   int h_steps=bars/2;
//--- Objekt bewegen
   for(int i=0;i<h_steps;i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- den nächsten Wert nehmen
      if(d<bars-1)
         d+=1;
      //--- den Punkt bewegen
      if(!EventMove(0,InpName,date[d]))
         return;
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- den Chart neu zeichnen
      ChartRedraw();
      // Verzögerung 0.05 Sekunden
      Sleep(50);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- den Kanal aus dem Chart löschen
   EventDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

OBJ_RECTANGLE_LABEL

Objekt "Rechteckiges Label."



Hinweis

Koordinaten des Ankerpunkts werden in Pixeln angegeben. Sie können den Winkel des Anbindepunktes des rechteckigen Labels durch Enumeration ENUM_BASE_CORNER wählen. Rahmentyp für ein rechteckiges Label kann durch ENUM_BORDER_TYPE gesetzt werden.

Es wird für Erstellung und Gestaltung der grafischen Benutzerschnittstelle verwendet.

Beispiel

Das folgende Skript erstellt und bewegt auf dem Chart Objekt "Rechteckiges Label". Um die Eigenschaften eines grafischen Objekts zu erstellen und ändern, gibt es speziale Funktionen, die Sie "wie es ist" für Ihre eigenen Programme verwenden können.

```
//--- Beschreibung
#property description "Das Skript erstellt ein graphisches Objekt \"Rechteckiges Label
//--- Zeigen den Fenster von Eingabeparametern in der Skript-Startup
#property script_show_inputs
//--- Eingangsparameter von Skript
input string           InpName="RectLabel";          // Label Name
input color            InpBackColor=clrSkyBlue;      // Hintergrundfarbe
input ENUM_BORDER_TYPE InpBorder=BORDER_FLAT;        // Rahmentyp
input ENUM_BASE_CORNER InpCorner=CORNER_LEFT_UPPER; // Winkel des Charts zu binden
input color            InpColor=clrDarkBlue;         // Die Farbe des flachen Randes (F
input ENUM_LINE_STYLE  InpStyle=STYLE_SOLID;         // Stil des flachen Randes (Flat)
input int              InpLineWidth=3;               // Dicke des flachen Randes (Flat)
```

© 2000-2025, MetaQuotes Ltd.

```
input bool              InpBack=false;               // Objekt im Hintergrund
input bool              InpSelection=true;           // Wählen um zu bewegen
input bool              InpHidden=true;              // Ausgeblendet in der Liste der O
input long              InpZOrder=0;                 // Priorität auf Mausklick
//+------------------------------------------------------------------+
//| Erstellt ein rechteckiges Label                                  |
//+------------------------------------------------------------------+
bool RectLabelCreate(const long             chart_ID=0,               // ID des Charts
                     const string           name="RectLabel",         // Label Name
                     const int              sub_window=0,             // Nummer des Un
                     const int              x=0,                      // X-Koordinate
                     const int              y=0,                      // Y-Koordinate
                     const int              width=50,                 // Breite
                     const int              height=18,                // Höhe
                     const color            back_clr=C'236,233,216',  // Hintergrundfa
                     const ENUM_BORDER_TYPE border=BORDER_SUNKEN,     // Rahmentyp
                     const ENUM_BASE_CORNER corner=CORNER_LEFT_UPPER, // Winkel des Ch
                     const color            clr=clrRed,               // Die Farbe des
                     const ENUM_LINE_STYLE  style=STYLE_SOLID,        // Stil des flac
                     const int              line_width=1,             // Breite des fl
                     const bool             back=false,               // Im Hintergrun
                     const bool             selection=false,          // Wählen um zu
                     const bool             hidden=true,              // Ausgeblendet
                     const long             z_order=0)                // Priorität auf
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ein rechteckiges Label erstellen
   if(!ObjectCreate(chart_ID,name,OBJ_RECTANGLE_LABEL,sub_window,0,0))
     {
      Print(__FUNCTION__,
            ": rechteckiges Label konnte nicht erstellt werden! Fehlercode = ",GetLast
      return(false);
     }
//--- Die Koordinaten des Schilds setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y);
//--- Labelgröße setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width);
   ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height);
//--- Hintergrundfarbe setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_BGCOLOR,back_clr);
//--- Rahmentyp setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_BORDER_TYPE,border);
//--- wählen die Ecke des Charts, relativ zu der die Punktkoordinaten eingegeben werde
   ObjectSetInteger(chart_ID,name,OBJPROP_CORNER,corner);
//--- Die Farbe des flachen Randes (in Modus Flat) setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_COLOR,clr);
//--- Linienstil des flachen Rahmens setzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(chart_ID,name,OBJPROP_STYLE,style);
//--- Breite des flachen Rahmens setzen
   ObjectSetInteger(chart_ID,name,OBJPROP_WIDTH,line_width);
//--- Im Vordergrund (false) oder Hintergrund (true) anzeigen
   ObjectSetInteger(chart_ID,name,OBJPROP_BACK,back);
//--- Aktivieren (true) oder deaktivieren (false) Mausbewegung Modus
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTABLE,selection);
   ObjectSetInteger(chart_ID,name,OBJPROP_SELECTED,selection);
//--- Verbergen (true) oder Anzeigen (false) den Namen des graphischen Objektes in der
   ObjectSetInteger(chart_ID,name,OBJPROP_HIDDEN,hidden);
//--- setzen die Priorität für eine Mausklick-Ereignisse auf dem Chart
   ObjectSetInteger(chart_ID,name,OBJPROP_ZORDER,z_order);
//--- die erfolgreiche Umsetzung
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Bewegt ein rechteckiges Label                                    |
//+------------------------------------------------------------------+
bool RectLabelMove(const long   chart_ID=0,       // ID des Charts
                   const string name="RectLabel", // Label Name
                   const int    x=0,              // X-Koordinate
                   const int    y=0)              // Y-Koordinate
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- ein rechteckiges Label bewegen
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XDISTANCE,x))
     {
      Print(__FUNCTION__,
            ": Kann nicht X-Koordinate des Labels bewegen! Fehlercode = ",GetLastError
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YDISTANCE,y))
     {
      Print(__FUNCTION__,
            ": Kann nicht Y-Koordinate des Labels bewegen! Fehlercode = ",GetLastError
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert die Größe eines rechteckigen Labels                       |
//+------------------------------------------------------------------+
bool RectLabelChangeSize(const long   chart_ID=0,       // ID des Charts
                         const string name="RectLabel", // Name des Labels
                         const int    width=50,         // Breite des Labels
                         const int    height=18)        // Höhe des Labels
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Größe des Labels ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_XSIZE,width))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Breite des Labels ändern! Fehlercode = ",GetLastError
      return(false);
     }
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_YSIZE,height))
     {
      Print(__FUNCTION__,
            ": Konnte nicht die Höhe des Labels ändern! Fehlercode = ",GetLastError())
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Ändert den Rahmentyp des rechteckigen Labels                     |
//+------------------------------------------------------------------+
bool RectLabelChangeBorderType(const long             chart_ID=0,           // ID des
                               const string           name="RectLabel",     // Name de
                               const ENUM_BORDER_TYPE border=BORDER_SUNKEN) // Rahment
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- Rahmentyp ändern
   if(!ObjectSetInteger(chart_ID,name,OBJPROP_BORDER_TYPE,border))
     {
      Print(__FUNCTION__,
            ": Konnte nicht Rahmentyp ändern! Fehlercode = ",GetLastError());
      return(false);
     ;}
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Löscht ein rechteckiges Label                                    |
//+------------------------------------------------------------------+
bool RectLabelDelete(const long   chart_ID=0,       // ID des Charts
                     const string name="RectLabel") // Name des Labels
  {
//--- Setzen den Wert des Fehlers zurück
   ResetLastError();
//--- das Schild löschen
   if(!ObjectDelete(chart_ID,name))
     {
      Print(__FUNCTION__,
```

© 2000-2025, MetaQuotes Ltd.

```
            ": rechteckiges Label konnte nicht gelöscht werden! Fehlercode = ",GetLast
      return(false);
     }
//--- die erfolgreiche Umsetzung
   return(true);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Größe des Chartfensters
   long x_distance;
   long y_distance;
//--- Größe des Chartfensters definieren
   if(!ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0,x_distance))
     {
      Print("Konnte nicht Breite des Charts bekommen! Fehlercode = ",GetLastError());
      return;
    ;}
   if(!ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0,y_distance))
     {
      Print("Konnte nicht Höhe des Charts bekommen! Fehlercode = ",GetLastError());
      return;
     }
//--- Koordinaten eines rechteckigen Labels definieren
   int x=(int)x_distance/4;
   int y=(int)y_distance/4;
//--- Labelgröße setzen
   int width=(int)x_distance/4;
   int height=(int)y_distance/4;
//--- ein rechteckiges Label erstellen
   if(!RectLabelCreate(0,InpName,0,x,y,width,height,InpBackColor,InpBorder,InpCorner,
      InpColor,InpStyle,InpLineWidth,InpBack,InpSelection,InpHidden,InpZOrder))
     {
      return;
     }
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Größe des rechteckigen Labels ändern
   int steps=(int)MathMin(x_distance/4,y_distance/4);
   for(int i=0;i<steps;i++)
     {
      //--- Größe ändern
      width+=1;
      height+=1;
      if(!RectLabelChangeSize(0,InpName,width,height))
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
 //--- Überprüfen die Fakten von Zwangsabschaltung der Skript
      if(IsStopped())
         return;
      //--- Die Grafik neu zeichnen und 0,01 Sekunden warten
      ChartRedraw();
      Sleep(10);
     }
//--- 1 Sekunde Verzögerung
   Sleep(1000);
//--- Rahmentyp ändern
   if(!RectLabelChangeBorderType(0,InpName,BORDER_RAISED))
      return;
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- Rahmentyp ändern
   if(!RectLabelChangeBorderType(0,InpName,BORDER_SUNKEN))
      return;
//--- den Chart neu zeichnen und 1 Sekunde warten
   ChartRedraw();
   Sleep(1000);
//--- das Schild löschen
   RectLabelDelete(0,InpName);
   ChartRedraw();
//--- 1 Sekunde warten
   Sleep(1000);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

Objekteigenschaften

Grafische Objekte können je nach dem Objekttyp zahlreiche Eigenschaften haben. Entsprechend den Funktionen zu grafischen Objekten werden Werte von Objekteigenschaften vorgegeben und erhalten.

Alle Objekte, die bei der technischen Analyse verwendet werden, werden an Grafiken nach den Koordinaten der Zeit und des Preises gebunden. Dazu gehören Trendlinie, Kanäle, Fibonacci usw. Es gibt aber eine Reihe von Hilfsobjekten, die zur Verbesserung des Interfaces dienen und an den immer sichtbaren Teil des Charts gebunden werden (Hauptfenster oder Subfester der Indikatoren):

Objekt	Identifikator	X/Y	Width/ Height	Date/Price	OBJPROP_CORNER	OBJPROP_ANCHOR	OBJPROP_ANGLE
Text	OBJ_TEXT	–	–	Ja	–	Ja	Ja
Label	OBJ_LABEL	Ja	Ja (read-only)	–	Ja	Ja	Ja
Button	OBJ_BUTTON	Ja	Ja	–	Ja	–	–
Bitmap	OBJ_BITMAP	–	Ja (read-only)	Ja	–	Ja	–
Bitmap Label	OBJ_BITMAP_LABEL	Ja	Ja (read-only)	–	Ja	Ja	–
Edit	OBJ_EDIT	Ja	Ja	–	Ja	–	–
Rectangle Label	OBJ_RECTANGLE_LABEL	Ja	Ja	–	Ja	–	–

In der Tabelle werden folgende Bezeichnungen verwendet:

- **X/Y** - die Koordinaten des Bindepunkts werden in Pixel hinsichtlich einer der Ecken der Grafik angegeben;
- **Width/Height** - Objekte haben Breite und Höhe. Wenn "read-only" angegeben ist, heißt das, dass die Höhe und die Breite erst nach dem Zeichnen des Objekts auf der Grafik berechnet werden;
- **Date/Price** - das Paar Datum/Preis gibt die Koordinaten des Bindepunktes an;
- **OBJPROP_CORNER** - gibt eine Ecke der Grafik an, in Hinsicht auf welche Koordinaten des Bindepunkts angegeben werden. Kann einen der vier Aufzählungswerte ENUM_BASE_CORNER annehmen;
- **OBJPROP_ANCHOR** - gibt den Bindepunkt im Objekt selbst an und kann einen der 9 Aufzählungswerte ENUM_ANCHOR_POINT annehmen. Gerade von diesem Punkt werden Koordinaten in Pixel zur ausgewählten Ecke der Grafik angegeben;
- **OBJPROP_ANGLE** - gibt den Drehungswinkel entgegen dem Uhrzeigersinn an.

© 2000-2025, MetaQuotes Ltd.

Die Funktionen, die die Eigenschaften von grafischen Objekten eingeben, und die Operationen der Erstellung ObjectCreate() und Bewegen ObjectMove() von Objekte auf dem Chart sind tatsächliche verwendet, um Befehle dem Chart zu senden. Der erfolgreiche Erfüllung dieser Funktionen wird der Befehl in die allgemeine Warteschlange der Chartereignisse platziert. Visuelle Veränderung der Eigenschaften von grafischen Objekten wird während der Verarbeitung der Ereigniswarteschlange von diesem Chart geändert.

Aus diesem Grund sollte man nicht erwarten, dass die graphische Objekten sofort nach dem Aufruf dieser Funktionen visuell aktualisiert wird. Im Allgemeinen werden die graphische Objekten durch das Terminal automatisch nach Änderung-Ereignisse aktualisiert - die Ankunft der neuen Quote, Änderung von Chartfenstergröße, etc.

Um graphischen Objekten zu aktualisieren, verwenden Sie ChartRedraw().

für die Funktionen ObjectSetInteger() und ObjectGetInteger()

ENUM_OBJECT_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
OBJPROP_COLOR	Farbe	color
OBJPROP_STYLE	Stil	ENUM_LINE_STYLE
OBJPROP_WIDTH	Weite von Linie	int
OBJPROP_BACK	Objekt auf dem Hintergrund	bool
OBJPROP_ZORDER	Priorität eines graphischen Obj	long

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ects für Empfang des Ereignisses sein en Mausklick auf dem Chart (CHARTEVENT_CLICK). Standardmäßig ist der Wert bei Erstellen auf Null gesetz	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	t, aber Sie können die Priorität erhöhen. Bei der Anwendung der Objekte auf jeder anderen, nur ein Objekt mit der höchsten Priorität bekommt das Ere	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ignis CHARTEVENT_CLICK.	
OBJPROP_FILL	Füllen ein Objekt mit Farbe (für OBJ_RECTANGLE, OBJ_TRIANGLE, OBJ_ELLIPSE, OBJ_CHANNEL, OBJ_STDDEVCHANNEL, OBJ	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	_REGRESSION)	
OBJPROP_HIDDEN	Anzeige des Namens von graphischem Objekt in der Objektliste aus dem Terminal-Menü "Charts" - "Objekte" - "Objektliste". Wert tru	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ermöglicht es, ein Objekt aus der Liste zu verstecken. Standardmäßig wird true für die Objekte, die Kalendereignisse und Handelsgeschichte	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	zeigen, und auch für die Objekte, die aus MQL5-Programmen erstellt werden, angegeben. Um solche graphischen Objekte graphischen Objekte	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	zu sehen und Zugang zu ihren Eigenschaften zu haben, klicken Sie auf "Alle" im Fenster "Objektliste".	
OBJPROP_SELECTED	Hervorheben des Objektes	bool
OBJPROP_READONLY	Möglichkeit von	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Texteditieren im Objekt Edit	
OBJPROP_TYPE	Objekttyp	ENUM_OBJECT r/o
OBJPROP_TIME	Koordinate der Zeit	datetime Modifikator=Nummer des Bezugspunktes
OBJPROP_SELECTABLE	Zugänglichkeit des Objekts	bool
OBJPROP_CREATETIME	Erzeugungszeit des Objekts	datetime r/o
OBJPROP_LEVELS	Levelszahl	int
OBJPROP_LEVELCOLOR	Farbe von	color Modifikator=Levelnummer

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Linie-Level	
OBJPROP_LEVELSTYLE	Stil der Linie-Level	ENUM_LINE_STYLE Modifikator=Levelnummer
OBJPROP_LEVELWIDTH	Dicke der Linie-Level	int Modifikator=Levelnummer
OBJPROP_ALIGN	Horizontale Textausrichtung in der "Edit"-Objekt (OBJ_EDIT)	ENUM_ALIGN_MODE
OBJPROP_FONTSIZE	Schriftgröße	int
OBJPROP_RAY_LEFT	Strahlset	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	zt sich nach links fort	
OBJPROP_RAY_RIGHT	Strahl setzt sich nach rechts fort	bool
OBJPROP_RAY	Eine vertikale Linie erstreckt sich auf alle der Chart-Fenster	bool
OBJPROP_ELLIPSE	Anzeigen	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	die ganzen Ellipse für das Objekt "Fibonacci Arcs" (OBJ_FIBOARC)	
OBJPROP_ARROWCODE	Weichenkode für Objekt "Weiche"	uchar
OBJPROP_TIMEFRAMES	Sichtbarkeit des Objekts auf Timeframes	Flaggenvorrat flags

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
OBJPROP_ANCHOR	Stellung des Bezugspunktes des grafischen Objekts	ENUM_ARROW_ANCHOR (für OBJ_ARROW), ENUM_ANCHOR_POINT (für OBJ_LABEL, OBJ_BITMAP_LABEL und OBJ_TEXT)
OBJPROP_XDISTANCE	Distanz in Pixel X-Achse vom Bezugswinkel (s. Hinweis)	int
OBJPROP_YDISTANCE	Distanz in Pixel Y-Achse vom	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Bezugswinkel (s. Hinweis)	
OBJPROP_DIRECTION	Trend des Gann Objekts	ENUM_GANN_DIRECTION
OBJPROP_DEGREE	Level von Elliott Wave Markierung	ENUM_ELLIOT_WAVE_DEGREE
OBJPROP_DRAWLINES	Linienderstellung für Elliott Wave Markierung	bool
OBJPROP_STATE	Schaltfläc	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	henstellung (Gedrückt/nichtgedrückt)	
OBJPROP_CHART_ID	ID von Objekt "Chart" (OBJ_CHART). Es erlaubt mit den Eigenschaften dieses Objektes als einer regelmäßigen Diagra	long r/o

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	mit den Funktionen der Sektion Operationen mit Charts arbeiten, aber es gibt einige Ausnahmen.	
OBJPROP_XSIZE	Die Breite des Objekts entlang der X-Achse in Pix	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	eln. Wird für OBJ_LABEL (read only), OBJ_BUTTON, OBJ_CHART, OBJ_BITMAP, OBJ_BITMAP_LABEL, OBJ_EDIT, OBJ_RECTANGLE_LABEL angegeben.	
OBJPROP_YSIZE	Die Hö	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	he des Objekts entlang der Y-Achse in Pixeln. Wird für OBJ_LABEL (read only), OBJ_BUTTON, OBJ_CHART, OBJ_BITMAP, OBJ_BITMAP_LABEL, OBJ_EDIT, OBJ	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	_RECTANGLE_LABEL angegeben.	
OBJPROP_XOFFSET	Die X Koordinate von der linken oberen Ecke des rechteckigen sichtbaren Bereich in der grafischen Objekte "Bit	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	map Label" und "Bitmap" (OBJ_BITMAP_LABEL und OBJ_BITMAP). Der Wert wird in Pixel relativ zur oberen linken Ecke des ursprünglichen Bildes ges	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	etzt.	
OBJPROP_YOFFSET	Die Y Koordinate von der linken oberen Ecke des rechteckigen sichtbaren Bereich in der grafischen Objekte "Bitmap Label" und "Bitmap" (OBJ	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	_BITMAP_LABEL und OBJ_BITMAP). Der Wert wird in Pixel relativ zur oberen linken Ecke des ursprünglichen Bildes gesetzt.	
OBJPROP_PERIOD	Periode für Objekt "C	ENUM_TIMEFRAMES

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	hart"	
OBJPROP_DATE_SCALE	Merkmal der Darstellung von Preisskala für Objekt "Chart"	bool
OBJPROP_PRICE_SCALE	Merkmal der Darstellung von Preisskala für Objekt "Chart"	bool
OBJPROP_CHART_SCALE	Massstab für Objekt	int Größe im Bereich 0-5

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	"Chart"	
OBJPROP_BGCOLOR	Hintergrundfarbe für OBJ_EDIT, OBJ_BUTTON, OBJ_RECTANGLE_LABEL	color
OBJPROP_CORNER	Winkel des Charts für Snap des Grafischen Objektes	ENUM_BASE_CORNER
OBJPROP_BORDER_TYPE	Frame	ENUM_BORDER_TYPE

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	- Typ für Object "Rechteckige Marke"	
OBJPROP_BORDER_COLOR	Die Rahmenfarbe für die Objekte OBJ_EDIT und OBJ_BUTTON	color

Wenn Sie benutzen Operationen mit Charts für Objekt "Chart" (OBJ_CHART), hat es die folgenden Einschränkungen:

- Es kann nicht durch ChartClose() geschlossen werden;
- Symbol/Periode kann nicht durch ChartSetSymbolPeriod() geädert werden;
- Die folgenden Eigenschaften funktionieren nicht: CHART_SCALE, CHART_BRING_TO_TOP, CHART_SHOW_DATE_SCALE und CHART_SHOW_PRICE_SCALE (ENUM_CHART_PROPERTY_INTEGER).

Für Objekte OBJ_BITMAP_LABEL und OBJ_BITMAP, kann ein besonderer Modus der Bilddarstellung festgelegt werden. In diesem Modus nur ein Teil des Bildes, bei dem eine rechteckige sichtbaren Bereich angewendet wird, während der Rest des Bildes wird unsichtbar. Die Größe dieses Bereiches sollten mit den Eigenschaften OBJPROP_XSIZE und OBJPROP_YSIZE gestellt werden. Der sichtbare Bereich kann nur innerhalb des ursprünglichen Bildes mit den Eigenschaften OBJPROP_XOFFSET und OBJPROP_YOFFSET "bewegt" werden.

© 2000-2025, MetaQuotes Ltd.

Für Objekte mit festgelegten Eigenschaften: OBJ_BUTTON, OBJ_RECTANGLE_LABEL, OBJ_EDIT und OBJ_CHART geben die OBJPROP_XDISTANCE und OBJPROP_YDISTANCE Eigenschaften die Lage des linken oberen Punkts hinsichtlich der Ecke der Grafik (OBJPROP_CORNER), von der die Koordinaten X und Y in Pixel berechnet werden.

für die Funktionen ObjectSetDouble() und ObjectGetDouble()

ENUM_OBJECT_PROPERTY_DOUBLE

Identifikator	Beschreibung	Typ der Eigenschaft
OBJPROP_PRICE	Koordinate des Preises	double Modifikator=Nummer des Bezugspunktes
OBJPROP_LEVELVALUE	Levelwert	double Modifikator=Levelnummer
OBJPROP_SCALE	Massstab (Eigenschaft der Objekte des Objektes "Fibinacci Arcs")	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
OBJPROP_ANGLE	Winkel. Für die aus einem Programmers tellten Objekte, für die kein Winkel angegeben ist, ist der Wert gleich EMPTY_VALUE	double
OBJPROP_DEVIATION	Abweichung für Kanal der	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Standardabweichung	

für die Funktionen ObjectSetString() und ObjectGetString()

ENUM_OBJECT_PROPERTY_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
OBJPROP_NAME	Objektname	string
OBJPROP_TEXT	Objektbeschreibung (Text, der im Objekt gibt)	string
OBJPROP_TOOLTIP	Der Text eines Tooltip. We	string

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	nn die Eigenschaft nicht angegeben ist, wird der Tooltip automatisch vom Terminal generiert. Ein Tooltip kann durch Setzen des "\n" (Zeilenvorschub) We	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	rtes deaktiviert werden	
OBJPROP_LEVELTEXT	Levelbeschreibung	string Modifikator=Levelnummer
OBJPROP_FONT	Schriftart	string
OBJPROP_BMPFILE	Name der BMP-Datei für Objekt "Graphische Marke". Sehen Sie auch Ressourcen	string Modifikator: 0-Stellung ON, 1-Stellung OFF

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
OBJPROP_SYMBOL	Symbol für Objekt "Chart"	string

Für OBJ_RECTANGLE_LABEL ("rechteckige Marke") kann Mann eine von drei Arten, die auf Werte im ENUM_BORDER_TYPE entsprechen, angeben.

ENUM_BORDER_TYPE

Identifikator	Beschreibung
BORDER_FLAT	Flache Form
BORDER_RAISED	Konvexe Form
BORDER_SUNKEN	Konkave Form

Für das Onjekt OBJ_EDIT ("Eingabefeld") und die Funktion ChartScreenShot() können Sie den Typ der horizontalen Ausrichtung mit der Enumerationswerte ENUM_ALIGN_MODE angeben.

ENUM_ALIGN_MODE

Identifier	Beschreibung
ALIGN_LEFT	Linksbündige Ausrichtung
ALIGN_CENTER	Zentrierte Ausrichtung (für"Eingabefeld")
ALIGN_RIGHT	Rechtsbündige Ausrichtung

Beispiel:

```
#define  UP          "\x0431"
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   string label_name="my_OBJ_LABEL_object";
```

© 2000-2025, MetaQuotes Ltd.

```
  if(ObjectFind(0,label_name)<0)
    {
     Print("Object",label_name,"not found. Error code = ",GetLastError());
     //--- erzeugen wir das Objekt Label
     ObjectCreate(0,label_name,OBJ_LABEL,0,0,0);
     //--- stellen wir die Koordinate X ein
     ObjectSetInteger(0,label_name,OBJPROP_XDISTANCE,200);
     //--- stellen wir die Koordinate Y ein
     ObjectSetInteger(0,label_name,OBJPROP_YDISTANCE,300);
     //--- Geben wir Textfarbe vor
     ObjectSetInteger(0,label_name,OBJPROP_COLOR,clrWhite);
     //--- stellen wir text für Objekt Label ein
     ObjectSetString(0,label_name,OBJPROP_TEXT,UP);
     //--- stellen wir Schrift der Unterschrift ein
     ObjectSetString(0,label_name,OBJPROP_FONT,"Wingdings");
     //--- Stellen wir Schriftgroesse ein
     ObjectSetInteger(0,label_name,OBJPROP_FONTSIZE,10);
     //--- drehen wir um 45 Grad mit dem Uhrzeigerlauf herum
     ObjectSetDouble(0,label_name,OBJPROP_ANGLE,-45);
     //--- verbieten wir Hervorhebung mit der Maus
     ObjectSetInteger(0,label_name,OBJPROP_SELECTABLE,false);
     //--- zeichnen wir auf dem Chart
     ChartRedraw(0);
    }
}
```

© 2000-2025, MetaQuotes Ltd.

Methode der Objektbindung

Die grafischen Objekte Text, Label, Bitmap und Bitmap Label (OBJ_TEXT, OBJ_LABEL, OBJ_BITMAP und OBJ_BITMAP_LABEL) verfügen über eine der neun Arten der Bindung, die durch die OBJPROP_ANCHOR Eigenschaft gesetzt werden.

Objekt	Identifikator	X/Y	Width/Height	Date/Price	OBJPROP_CORNER	OBJPROP_ANCHOR	OBJPROP_ANGLE
Text	OBJ_TEXT	—	—	Ja	—	**Ja**	Ja
Label	OBJ_LABEL	Ja	Ja (read-only)	—	Ja	**Ja**	Ja
Button	OBJ_BUTTON	Ja	Ja	—	Ja	—	—
Bitmap	OBJ_BITMAP	—	Ja (read-only)	Ja	—	**Ja**	—
Bitmap Label	OBJ_BITMAP_LABEL	Ja	Ja (read-only)	—	Ja	**Ja**	—
Edit	OBJ_EDIT	Ja	Ja	—	Ja	—	—
Rectangle Label	OBJ_RECTANGLE_LABEL	Ja	Ja	—	Ja	—	—

In der Tabelle werden folgende Bezeichnungen verwendet:

- **X/Y** - die Koordinaten des Bindepunkts werden in Pixel hinsichtlich einer der Ecken der Grafik angegeben;
- **Width/Height** - Objekte haben Breite und Höhe. Wenn "read-only" angegeben ist, heißt das, dass die Höhe und die Breite erst nach dem Zeichnen des Objekts auf der Grafik berechnet werden;
- **Date/Price** - das Paar Datum/Preis gibt die Koordinaten des Bindepunktes an;
- **OBJPROP_CORNER** - gibt eine Ecke der Grafik an, in Hinsicht auf welche Koordinaten des Bindepunkts angegeben werden. Kann einen der vier Aufzählungswerte ENUM_BASE_CORNER annehmen;
- **OBJPROP_ANCHOR** - gibt den Bindepunkt im Objekt selbst an und kann einen der 9 Aufzählungswerte ENUM_ANCHOR_POINT annehmen. Gerade von diesem Punkt werden Koordinaten in Pixel zur ausgewählten Ecke der Grafik angegeben;
- **OBJPROP_ANGLE** - gibt den Drehungswinkel entgegen dem Uhrzeigersinn an.

Die notwendige Variante kann durch die Funktion ObjectSetInteger(handle_Chart, Name_Objekt, OBJPROP_ANCHOR, Verbindungs_Weg) angegeben werden, wo Verbindungs_Weg - einer der Werte von Enumeration ENUM_ANCHOR_POINT ist.

ENUM_ANCHOR_POINT

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
ANCHOR_LEFT_UPPER	Bindepunkt in der linken oberen Ecke
ANCHOR_LEFT	Bindepunkt links zentriert
ANCHOR_LEFT_LOWER	Bindepunkt in der linken unteren Ecke
ANCHOR_LOWER	Bindepunkt unten zentriert
ANCHOR_RIGHT_LOWER	Bindepunkt in der rechten unteren Ecke
ANCHOR_RIGHT	Bindepunkt rechts zentriert
ANCHOR_RIGHT_UPPER	Bindepunkt rechts zentriert
ANCHOR_UPPER	Bindepunkt oben zentriert
ANCHOR_CENTER	Bindepunkt im Objektzentrum

Die Objekte OBJ_BUTTON, OBJ_RECTANGLE_LABEL und OBJ_EDIT haben einen festen Bindepunkt in der linken oberen Ecke (ANCHOR_LEFT_UPPER).

Beispiel:

```
string text_name="my_OBJ_TEXT_object";
if(ObjectFind(0,text_name)<0)
  {
   Print("Object",text_name,"not found. Error code = ",GetLastError());
   //--- wir bekommen maximalen Preis des Charts
   double chart_max_price=ChartGetDouble(0,CHART_PRICE_MAX,0);
   //--- erzeugen wir das Objekt Label
   ObjectCreate(0,text_name,OBJ_TEXT,0,TimeCurrent(),chart_max_price);
   //--- stellen wir Textfarbe fest
   ObjectSetInteger(0,text_name,OBJPROP_COLOR,clrWhite);
   //--- stellen wir Hintergrundfarbe fest
   ObjectSetInteger(0,text_name,OBJPROP_BGCOLOR,clrGreen);
   //--- stellen wir Text für Objekt Label fest
   ObjectSetString(0,text_name,OBJPROP_TEXT,TimeToString(TimeCurrent()));
   //--- stellen wir Schriftgroesse der überschrift fest
   ObjectSetString(0,text_name,OBJPROP_FONT,"Trebuchet MS");
   //--- stellen wir Schriftgroesse fest
   ObjectSetInteger(0,text_name,OBJPROP_FONTSIZE,10);
   //--- stellen wir snap zum oberen rechten Winkel fest
   ObjectSetInteger(0,text_name,OBJPROP_ANCHOR,ANCHOR_RIGHT_UPPER);
   //--- schwenken wir um 90 Grad gegen dem Uhrzeigerlauf
   ObjectSetDouble(0,text_name,OBJPROP_ANGLE,90);
   //--- verbieten wir Objekthervorheben mit der Maus
   ObjectSetInteger(0,text_name,OBJPROP_SELECTABLE,false);
   //--- zeichnen wir auf dem Chart
   ChartRedraw(0);
  }
```

© 2000-2025, MetaQuotes Ltd.

Graphische Objekte Arrow (OBJ_ARROW) haben nur 2 Snapverfahren ihrer Koordinaten. Identifikatoren werden in ENUM_ARROW_ANCHOR aufgezählt.

ENUM_ARROW_ANCHOR

Identifikator	Beschreibung
ANCHOR_TOP	Bindepunkt für Weiche befindet sich oben
ANCHOR_BOTTOM	Bindepunkt für Weiche befindet sich unten

Beispiel:

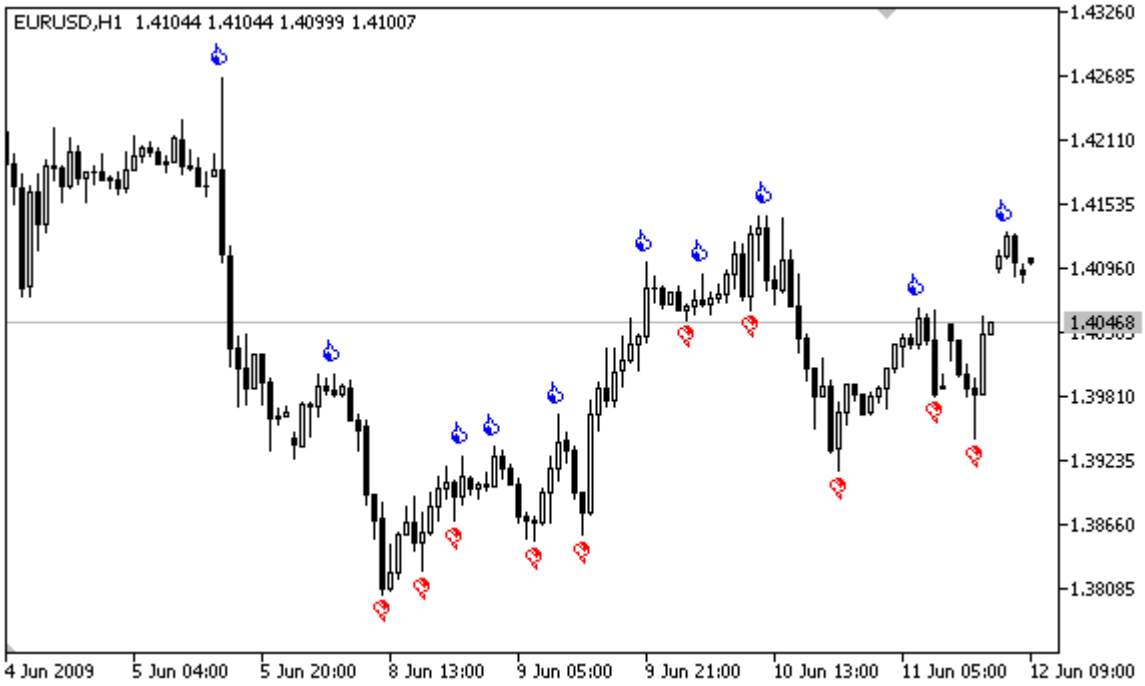
```
void OnStart()
  {
//--- Dienstfelder
   double Ups[],Downs[];
   datetime Time[];
//--- stellen wir für Felder Timeserienmerkmal fest
   ArraySetAsSeries(Ups,true);
   ArraySetAsSeries(Downs,true);
   ArraySetAsSeries(Time,true);
//--- erzeugen wir handle des Indikators Fractals
   int FractalsHandle=iFractals(NULL,0);
   Print("FractalsHandle =",FractalsHandle);
//--- stellen wir Fehlerkode auf Null
   ResetLastError();
//--- versuchen wir, Indikatorwert zu kopieren
   int copied=CopyBuffer(FractalsHandle,0,0,1000,Ups);
   if(copied<=0)
     {
      Print("Obere Fractals Kopieren Fehler. Error = ",GetLastError());
      return;
     }

   ResetLastError();
//--- versuchen wir Indikatorwerte zu kopieren
   copied=CopyBuffer(FractalsHandle,1,0,1000,Downs);
   if(copied<=0)
     {
      Print("Untere Fractals Kopieren Fehler. Error = ",GetLastError());
      return;
     }

   ResetLastError();
//--- kopieren wir Zeitreihe mit der Eroeffnungszeit der letzten 1000 Bars
   copied=CopyTime(NULL,0,0,1000,Time);
   if(copied<=0)
     {
      Print("Nicht gelungen, Eroeffnungszeit der letzten 1000 Bars zu kopieren");
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
     }

  int upcounter=0,downcounter=0; // zählen wir die Weichen
  bool created;// bekommen wir das Ergebnis des Versuches, ein Objekt zu erzeugen
  for(int i=2;i<copied;i++)// sehen wir die Werte des Idikators iFractals durch
    {
     if(Ups[i]!=EMPTY_VALUE)// oberer Fraktal gefunden
       {
        if(upcounter<10)// erzeugen nicht mehr als 10 Objekte "oben"
          {
           //--- versuchen wir das Objekt "oben" zu erzeugen
           created=ObjectCreate(0,string(Time[i]),OBJ_ARROW_THUMB_UP,0,Time[i],Ups[i]
           if(created)// wenn erzeugt - machen wir tuning dafür
             {
              //--- Ankerpunkt ist unten, um die Bar nicht zu bedecken
              ObjectSetInteger(0,string(Time[i]),OBJPROP_ANCHOR,ANCHOR_BOTTOM);
              //--- letzter Strich - zeichnen
              ObjectSetInteger(0,string(Time[i]),OBJPROP_COLOR,clrBlue);
              upcounter++;
             }
          }
       }
     if(Downs[i]!=EMPTY_VALUE)// neuer Fraktal gefunden
       {
        if(downcounter<10)// erzeugen wir nicht mehr als 10 Objekte "unten"
          {
           //--- vesuchen wir das Objekt "unten" zu erzeugen
           created=ObjectCreate(0,string(Time[i]),OBJ_ARROW_THUMB_DOWN,0,Time[i],Down
           if(created)// wenn erzeugt  - machen wir tuning dafür
             {
              //--- Bundepunkt ist oben, um die Bar nicht zu bedecken
              ObjectSetInteger(0,string(Time[i]),OBJPROP_ANCHOR,ANCHOR_TOP);
              //--- letzter Strich - zeichnen
              ObjectSetInteger(0,string(Time[i]),OBJPROP_COLOR,clrRed);
              downcounter++;
             }
          }
       }
    }
  }
```

Nach der Durchführung von Script wird der Chart etwa so aussehen, wie auf diesem Bild.

© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

Die Ecke der Grafik, an den das Objekt gebunden ist

Es gibt eine Reihe grafischer Objekte, für welche man die Ecke der Grafik angeben kann, hinsichtlich dessen die Koordinaten in Pixel angegeben werden. Dazu gehören die folgenden Objekt-Typen (in Klammern sind Identifikatoren des Objekt-Typs):

- Label (OBJ_LABEL); Textzeichen
- Button (OBJ_BUTTON); Button
- Bitmap Label (OBJ_BITMAP_LABEL); Grafisches Zeichen
- Edit (OBJ_EDIT); Eingabefeld
- Rectangle Label (OBJ_RECTANGLE_LABEL). Winkelrechtes Zeichen

Objekt	Identifikator	X/Y	Width/Height	Date/Price	OBJPROP_CORNER	OBJPROP_ANCHOR	OBJPROP_ANGLE
Text	OBJ_TEXT	—	—	Ja	—	Ja	Ja
Label	OBJ_LABEL	Ja	Ja (read-only)	—	**Ja**	Ja	Ja
Button	OBJ_BUTTON	Ja	Ja	—	**Ja**	—	—
Bitmap	OBJ_BITMAP	—	Ja (read-only)	Ja	—	Ja	—
Bitmap Label	OBJ_BITMAP_LABEL	Ja	Ja (read-only)	—	**Ja**	Ja	—
Edit	OBJ_EDIT	Ja	Ja	—	**Ja**	—	—
Rectangle Label	OBJ_RECTANGLE_LABEL	Ja	Ja	—	**Ja**	—	—

In der Tabelle werden folgende Bezeichnungen verwendet:

- **X/Y** - die Koordinaten des Bindepunkts werden in Pixel hinsichtlich einer der Ecken der Grafik angegeben;
- **Width/Height** - Objekte haben Breite und Höhe. Wenn "read-only" angegeben ist, heißt das, dass die Höhe und die Breite erst nach dem Zeichnen des Objekts auf der Grafik berechnet werden;
- **Date/Price** - das Paar Datum/Preis gibt die Koordinaten des Bindepunktes an;
- **OBJPROP_CORNER** - gibt eine Ecke der Grafik an, in Hinsicht auf welche Koordinaten des Bindepunkts angegeben werden. Kann einen der vier Aufzählungswerte ENUM_BASE_CORNER annehmen;
- **OBJPROP_ANCHOR** - gibt den Bindepunkt im Objekt selbst an und kann einen der 9 Aufzählungswerte ENUM_ANCHOR_POINT annehmen. Gerade von diesem Punkt werden Koordinaten in Pixel zur ausgewählten Ecke der Grafik angegeben;
- **OBJPROP_ANGLE** - gibt den Drehungswinkel entgegen dem Uhrzeigersinn an.

© 2000-2025, MetaQuotes Ltd.

Um den Chartwinkel anzugeben, von dem Koordinaten X und Y in Pixel abgezählt werden, muss man die Funktion ObjectSetInteger(chartID, name, OBJPROP_CORNER, **chart_corner**) verwenden, wo:

- chartID - Chartidentifikator;
- name – Name des graphischen Objekts;
- OBJPROP_CORNER – Identifikator der Eigenschaft für Festlegung des Snapwinkels;
- **chart_corner** – der notwendiger Chartwinkel kann einen der Werte von Enumeration ENUM_BASE_CORNER annehmen.

ENUM_BASE_CORNER

Identifikator	Beschreibung
CORNER_LEFT_UPPER	Mittelpunkt der Koordinaten im oberen linken Winkel des Charts
CORNER_LEFT_LOWER	Mittelpunkt der Koordinaten im unteren linken Winkel des Charts
CORNER_RIGHT_LOWER	Mittelpunkt der Koordinaten im unteren rechten Winkel des Charts
CORNER_RIGHT_UPPER	Mittelpunkt der Koordinaten im oberen rechten Winkel des Charts

Beispiel:

```
void CreateLabel(long   chart_id,
                 string name,
                 int    chart_corner,
                 int    anchor_point,
                 string text_label,
                 int    x_ord,
                 int    y_ord)
  {
//---
   if(ObjectCreate(chart_id,name,OBJ_LABEL,0,0,0))
     {
      ObjectSetInteger(chart_id,name,OBJPROP_CORNER,chart_corner);
      ObjectSetInteger(chart_id,name,OBJPROP_ANCHOR,anchor_point);
      ObjectSetInteger(chart_id,name,OBJPROP_XDISTANCE,x_ord);
      ObjectSetInteger(chart_id,name,OBJPROP_YDISTANCE,y_ord);
      ObjectSetString(chart_id,name,OBJPROP_TEXT,text_label);
     }
   else
      Print("Erstellung des OBJ_LABEL-Objekts fehlgeschlagen",name,", Fehlercode = ",
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//---
   int height=(int)ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0);
   int width=(int)ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0);
   string arrows[4]={"LEFT_UPPER","RIGHT_UPPER","RIGHT_LOWER","LEFT_LOWER"};
   CreateLabel(0,arrows[0],CORNER_LEFT_UPPER,ANCHOR_LEFT_UPPER,arrows[0],50,50);
   CreateLabel(0,arrows[1],CORNER_RIGHT_UPPER,ANCHOR_RIGHT_UPPER,arrows[1],50,50);
   CreateLabel(0,arrows[2],CORNER_RIGHT_LOWER,ANCHOR_RIGHT_LOWER,arrows[2],50,50);
   CreateLabel(0,arrows[3],CORNER_LEFT_LOWER,ANCHOR_LEFT_LOWER,arrows[3],50,50);
  }
```

© 2000-2025, MetaQuotes Ltd.

Objektsichtbarkeit

Kombination von Flaggen der Objektsichbarkeit destimmt Timeframes des Charts, auf denen das Objekt dargestellt wird. Für Einstellung/Erhaltung des Wertes der Eigenschaft OBJPROP_TIMEFRAMES können die Funktionen ObjectSetInteger()/ObjectGetInteger() verwendet werden.

Konstante	Wert	Beschreibung
OBJ_PERIOD_M1	0x00000001	Objekt wird auf 1-Minute Chart gezeichnet
OBJ_PERIOD_M2	0x00000002	Objekt wird auf 2-Minuten Chart gezeichnet
OBJ_PERIOD_M3	0x00000004	Objekt wird auf 3-Minuten Chart gezeichnet
OBJ_PERIOD_M4	0x00000008	Objekt wird auf 4-Minuten Chart gezeichnet
OBJ_PERIOD_M5	0x00000010	Objekt wird auf 5-Minuten Chart gezeichnet
OBJ_PERIOD_M6	0x00000020	Objekt wird auf 6-Minuten Chart gezeichnet
OBJ_PERIOD_M10	0x00000040	Objekt wird auf 10-Minuten Chart gezeichnet
OBJ_PERIOD_M12	0x00000080	Objekt wird auf 12-Minuten Chart gezeichnet
OBJ_PERIOD_M15	0x00000100	Objekt wird auf 15-Minuten Chart gezeichnet
OBJ_PERIOD_M20	0x00000200	Objekt wird auf 20-Minuten Chart gezeichnet
OBJ_PERIOD_M30	0x00000400	Objekt wird auf 30-Minuten Chart gezeichnet
OBJ_PERIOD_H1	0x00000800	Objekt wird auf 1-Stunde Chart gezeichnet
OBJ_PERIOD_H2	0x00001000	Objekt wird auf 2-Stunden Chart gezeichnet
OBJ_PERIOD_H3	0x00002000	Objekt wird auf 3-Stunden Chart gezeichnet
OBJ_PERIOD_H4	0x00004000	Objekt wird auf 4-Stunden Chart gezeichnet
OBJ_PERIOD_H6	0x00008000	Objekt wird auf 6-Stunden Chart gezeichnet

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
OBJ_PERIOD_H8	0x00010000	Objekt wird auf 8-Stunden Chart gezeichnet
OBJ_PERIOD_H12	0x00020000	Objekt wird auf 12-Stunden Chart gezeichnet
OBJ_PERIOD_D1	0x00040000	Objekt wird auf Tageschart gezeichnet
OBJ_PERIOD_W1	0x00080000	Objekt wird auf Wochenchart gezeichnet
OBJ_PERIOD_MN1	0x00100000	Objekt wird auf Monatschart gezeichnet
OBJ_ALL_PERIODS	0x001fffff	Objekt wird auf allen Timeframes gezeichnet

Flaggen der Sichbarkeit können durch das Symbol "|" kombiniert werden, zB Flaggenkombination OBJ_PERIOD_M10|OBJ_PERIOD_H4 bedeutet, dass Objekt auf dem 10-Minuten und 4-Minuten timeframes sichtbar sein wird.

Beispiel:

```
void OnStart()
  {
//---
   string highlevel="PreviousDayHigh";
   string lowlevel="PreviousDayLow";
   double prevHigh;           // High des Vortages
   double prevLow;            // Low des Vortages
   double highs[],lows[];     // Felder für Erhaltung von High und Low

//--- stellen wir den Wert des letzten Fehlers zurück
   ResetLastError();
//---  wir bekommen 2 letzte Werte High auf Tagstimeframe
   int highsgot=CopyHigh(Symbol(),PERIOD_D1,0,2,highs);
   if(highsgot>0) // wenn  Kopieren erfolgreich ist
     {
      Print("Preise High für letzte 2 Tage erfolgreich erhalten");
      prevHigh=highs[0]; // High des Vortages
      Print("prevHigh =",prevHigh);
      if(ObjectFind(0,highlevel)<0) // Objekt mit dem Namen highlevel nicht gefunden
        {
         ObjectCreate(0,highlevel,OBJ_HLINE,0,0,0); // schaffen wir das Objekt Hotline
        }
      //--- stellen wir Preislevel für die Linie highlevel fest
      ObjectSetDouble(0,highlevel,OBJPROP_PRICE,0,prevHigh);
      //--- Stellen wir Sichtbarkeit nur für PERIOD_M10 und PERIOD_H4
      ObjectSetInteger(0,highlevel,OBJPROP_TIMEFRAMES,OBJ_PERIOD_M10|OBJ_PERIOD_H4);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   else
      {
       Print("Erhalten von Preisen High für letzte 2 Tage Fehler, Error = ",GetLastErro
      }

//---stellen wir den wert des letzen Fehlers zurück
   ResetLastError();
//--- bekommen wir 2 letzte Werte auf Tagstimeframe Low
   int lowsgot=CopyLow(Symbol(),PERIOD_D1,0,2,lows);
   if(lowsgot>0) // wenn Kopierenoperation erfolgreich ist
      Print("Preise Low für letzte 2 Tage erfolgreich erhalten");
      prevLow=lows[0]; // Low des Vortages
      Print("prevLow =",prevLow);
      if(ObjectFind(0,lowlevel)<0) // Objekt mit dem Namen lowlevel nicht gefunden
        {
         ObjectCreate(0,lowlevel,OBJ_HLINE,0,0,0); // erzeugen wir das Objekt "Hotline
        }
      //--- stellen wir Preislevel für Linie lowlevel fest
      ObjectSetDouble(0,lowlevel,OBJPROP_PRICE,0,prevLow);
      //--- stellen wir Sichbarkeit nur für  PERIOD_M10 und PERIOD_H4 fest
      ObjectSetInteger(0,lowlevel,OBJPROP_TIMEFRAMES,OBJ_PERIOD_M10|OBJ_PERIOD_H4);
     }
   else Print("Erhalten von Preisen für die letzten zwei Tage Fehler, Error = ",GetLas

   ChartRedraw(0); // zeichnen wir Chart zwangslaeufig neu
  }
```

Sehen Sie auch

PeriodSeconds, Period, Chartperioden, Datum und Zeit

© 2000-2025, MetaQuotes Ltd.

Levels von Elliott Wellen

Elliott Wellen werden durch zwei graphische Objekte der Typen OBJ_ELLIOTWAVE5 und OBJ_ELLIOTWAVE3 dargestellt. Für Festlegung der Wellengroesse (Verfahren der Wellenmarkierung) wird die Eigenschaft OBJPROP_DEGREE verwendet, der einen der Enumerationswerte ENUM_ELLIOT_WAVE_DEGREE festgestellt werden kann.

ENUM_ELLIOT_WAVE_DEGREE

Konstante	Beschreibung
ELLIOTT_GRAND_SUPERCYCLE	Hauptsuperzyklus (Grand Supercycle)
ELLIOTT_SUPERCYCLE	Superzyklus (Supercycle)
ELLIOTT_CYCLE	Zyklus (Cycle)
ELLIOTT_PRIMARY	primaeres Zyklus (Primary)
ELLIOTT_INTERMEDIATE	Mittelstelle (Intermediate)
ELLIOTT_MINOR	Sekundaeres Zyklus (Minor)
ELLIOTT_MINUTE	Minute (Minute)
ELLIOTT_MINUETTE	Sekunde (Minuette)
ELLIOTT_SUBMINUETTE	Subsekunde (Subminuette)

Beispiel:

```
   for(int i=0;i<ObjectsTotal(0);i++)
     {
      string currobj=ObjectName(0,i);
      if((ObjectGetInteger(0,currobj,OBJPROP_TYPE)==OBJ_ELLIOTWAVE3)  ||
         ((ObjectGetInteger(0,currobj,OBJPROP_TYPE)==OBJ_ELLIOTWAVE5)))
        {
         //--- stellen wir Einteilungsstufe in INTERMEDIATE
         ObjectSetInteger(0,currobj,OBJPROP_DEGREE,ELLIOTT_INTERMEDIATE);
         //--- Zeigen von Linien unter Wellenboegen schalten
         ObjectSetInteger(0,currobj,OBJPROP_DRAWLINES,true);
         //--- stellen wir Farbe der Linien ein
         ObjectSetInteger(0,currobj,OBJPROP_COLOR,clrBlue);
         //--- stellen wir Dicke der Linien ein
         ObjectSetInteger(0,currobj,OBJPROP_WIDTH,5);
         //--- stellen wir Beschreibung fest
         ObjectSetString(0,currobj,OBJPROP_TEXT,"test script");
        }
     }
```

© 2000-2025, MetaQuotes Ltd.

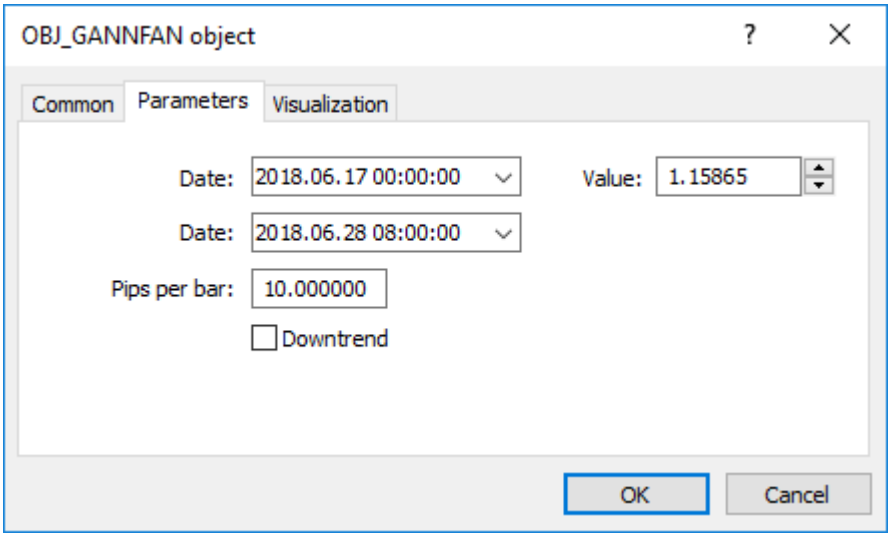
Gann Objekte

für Objekte Gannfan (OBJ_GANNFAN)und Ganngrid (OBJ_GANNGRID) kann einer der zwei Werte der Enumeration ENUM_GANN_DIRECTION angegeben werden, der Trendrichtung feststellt.

Konstante	Beschreibung
GANN_UP_TREND	Linie entspricht dem steidenden Trend
GANN_DOWN_TREND	Linie entspricht dem Fallenden Trend

für Einstellung des Massstabes der Grundlinie 1x1 wird die Funktion ObjectSetDouble(chart_handle, gann_object_name, OBJPROP_SCALE, scale) verwendet, wo:

- chart_handle - Chartfenster, in dem sich Objekt befindet;
- gann_object_name - Objektname;
- OBJPROP_SCALE - Identifikator der Eigenschaft "Scale";
- scale - der notwendige Massstab in Einheiten Pips/Bar.



Beispiel der Erzeugung von Gannfan:

```
void OnStart()
  {
//---
   string my_gann="OBJ_GANNFAN object";
   if(ObjectFind(0,my_gann)<0)// Objekt nicht gefunden
     {
      //--- melden wir Missefolg an
      Print("Object ",my_gann," not found. Error code = ",GetLastError());
      //--- wir erhalten maximalen Preis des Charts
      double chart_max_price=ChartGetDouble(0,CHART_PRICE_MAX,0);
      //--- wir erhalten minimalen Preis des Charts
      double chart_min_price=ChartGetDouble(0,CHART_PRICE_MIN,0);
      //--- wieviel Bars auf dem Chart gezeigt?
      int bars_on_chart=ChartGetInteger(0,CHART_VISIBLE_BARS);
      //--- erzeugen wir Feld, wo wir die Eroeffnungszeit jeder Bar schreiben
```

© 2000-2025, MetaQuotes Ltd.

```
      datetime Time[];
      //--- organisieren wir Zugang zum Feld wie in Timeserie
      ArraySetAsSeries(Time,true);
      //--- Jetzt kopieren wir darin Daten für die auf dem Chart sichtbaren Bars
      int times=CopyTime(NULL,0,0,bars_on_chart,Time);
      if(times<=0)
        {
         Print("Kopieren des Feldes mit Eroeffnungszeit fehler!");
         return;
        }
      //--- Vorbereitungen werden abgeschlossen

      //--- Index der Zentralbar auf dem Chart
      int center_bar=bars_on_chart/2;
      //--- Chartäquator - zwischen Maximum und Minimum
      double mean=(chart_max_price+chart_min_price)/2.0;
      //--- stellen wir Koordinaten des ersten Snappunktes ins Zentrum  fest
      ObjectCreate(0,my_gann,OBJ_GANNFAN,0,Time[center_bar],mean,
                   //--- der zweite Snappunkt ist recsht
                   Time[center_bar/2],(mean+chart_min_price)/2.0);
      Print("Time[center_bar]/="+(string)Time[center_bar]+"  Time[center_bar/2]="+(str
      //Print("Time[center_bar]/="+Time[center_bar]+"  Time[center_bar/2]="+Time[cente
      //--- stellen wir Massstab in Einheiten Pips/Bar
      ObjectSetDouble(0,my_gann,OBJPROP_SCALE,10);
      //--- stellen wir Liniendicke fest
      ObjectSetInteger(0,my_gann,OBJPROP_DIRECTION,GANN_UP_TREND);
      //--- stellen wir Liniendicke fest
      ObjectSetInteger(0,my_gann,OBJPROP_WIDTH,1);
      //--- Legen wir Linienstil
      ObjectSetInteger(0,my_gann,OBJPROP_STYLE,STYLE_DASHDOT);
      //--- und Linienfarbe fest
      ObjectSetInteger(0,my_gann,OBJPROP_COLOR,clrYellowGreen);
      //--- erlauben wir dem Benutzer, Objekt hervorzuheben
      ObjectSetInteger(0,my_gann,OBJPROP_SELECTABLE,true);
      //--- herben wir es selbst hervor
      ObjectSetInteger(0,my_gann,OBJPROP_SELECTED,true);
      //--- zeichnen wir auf dem Chart
      ChartRedraw(0);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Web-Farben

für Typ color werden folgende Preiskonstanten bestimmt:

clrBlack	clrDarkGreen	clrDarkSlateGray	clrOlive	clrGreen	clrTeal	clrNavy	clrPurple
clrMaroon	clrIndigo	clrMidnightBlue	clrDarkBlue	clrDarkOliveGreen	clrSaddleBrown	clrForestGreen	clrOliveDrab
clrSeaGreen	clrDarkGoldenrod	clrDarkSlateBlue	clrSienna	clrMediumBlue	clrBrown	clrDarkTurquoise	clrDimGray
clrLightSeaGreen	clrDarkViolet	clrFireBrick	clrMediumVioletRed	clrMediumSeaGreen	clrChocolate	clrCrimson	clrSteelBlue
clrGoldenrod	clrMediumSpringGreen	clrLawnGreen	clrCadetBlue	clrDarkOrchid	clrYellowGreen	clrLimeGreen	clrOrangeRed
clrDarkOrange	clrOrange	clrGold	clrYellow	clrChartreuse	clrLime	clrSpringGreen	clrAqua
clrDeepSkyBlue	clrBlue	clrMagenta	clrRed	clrGray	clrSlateGray	clrPeru	clrBlueViolet
clrLightSlateGray	clrDeepPink	clrMediumTurquoise	clrDodgerBlue	clrTurquoise	clrRoyalBlue	clrSlateBlue	clrDarkKhaki
clrIndianRed	clrMediumOrchid	clrGreenYellow	clrMediumAquamarine	clrDarkSeaGreen	clrTomato	clrRosyBrown	clrOrchid
clrMediumPurple	clrPaleVioletRed	clrCoral	clrCornflowerBlue	clrDarkGray	clrSandyBrown	clrMediumSlateBlue	clrTan
clrDarkSalmon	clrBurlyWood	clrHotPink	clrSalmon	clrViolet	clrLightCoral	clrSkyBlue	clrLightSalmon
clrPlum	clrKhaki	clrLightGreen	clrAquamarine	clrSilver	clrLightSkyBlue	clrLightSteelBlue	clrLightBlue
clrPaleGreen	clrThistle	clrPowderBlue	clrPaleGoldenrod	clrPaleTurquoise	clrLightGray	clrWheat	clrNavajoWhite
clrMoccasin	clrLightPink	clrGainsboro	clrPeachPuff	clrPink	clrBisque	clrLightGoldenrod	clrBlanchedAlmond
clrLemonChiffon	clrBeige	clrAntiqueWhite	clrPapayaWhip	clrCornsilk	clrLightYellow	clrLightCyan	clrLinen
clrLavender	clrMistyRose	clrOldLace	clrWhiteSmoke	clrSeashell	clrIvory	clrHoneydew	clrAliceBlue
clrLavenderBlush	clrMintCream	clrSnow	clrWhite				

Die Farbe für Objekte kann durch die Funktion ObjectSetInteger() und für Benutzerindikatoren durch die Funktion PlotIndexSetInteger() festgestellt werden. Für Erhalten des Farbwertes dienen Analogfunktionen ObjectGetInteger() und PlotIndexGetInteger().

© 2000-2025, MetaQuotes Ltd.

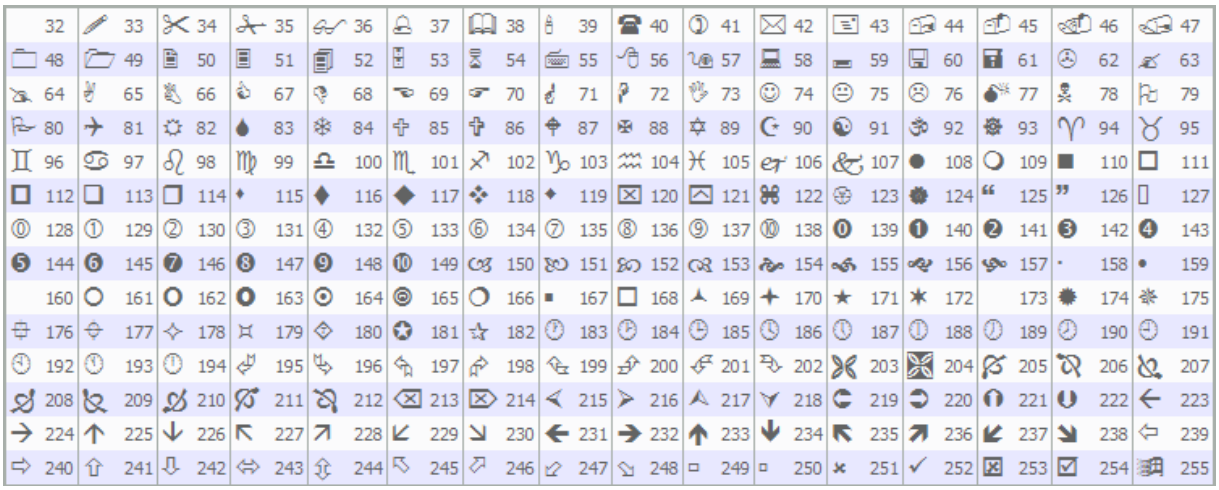
Beispiel:

```
//---- indicator settings
#property indicator_chart_window
#property indicator_buffers 3
#property indicator_plots   3
#property indicator_type1   DRAW_LINE
#property indicator_type2   DRAW_LINE
#property indicator_type3   DRAW_LINE
#property indicator_color1  clrBlue
#property indicator_color2  clrRed
#property indicator_color3  clrLime
```

© 2000-2025, MetaQuotes Ltd.

Wingdings

Symbolen der Schrift Wingdings, die mit dem Objekt OBJ_ARROW verwendet werden:



Feststellung des notwendigen Symbols erfolgt durch die Funktion ObjectSetInteger().

Beispiel:

```
void OnStart()
  {
//---
   string up_arrow="up_arrow";
   datetime time=TimeCurrent();
   double lastClose[1];
   int close=CopyClose(Symbol(),Period(),0,1,lastClose);        // wir bekommen Preis
//--- wenn der Preis erhalten
   if(close>0)
     {
      ObjectCreate(0,up_arrow,OBJ_ARROW,0,0,0,0,0);           // erzeugen wir Weiche
      ObjectSetInteger(0,up_arrow,OBJPROP_ARROWCODE,241);     // stellen wir Weichenkoc
      ObjectSetInteger(0,up_arrow,OBJPROP_TIME,time);         // Legen wir Zeit fest
      ObjectSetDouble(0,up_arrow,OBJPROP_PRICE,lastClose[0]);// legen wir Preis fest
      ChartRedraw(0);                                         // zeichnen wir Fenster r
     }
   else
      Print("Erhalten des letzten Preises Close Fehler! ");
  }
```

© 2000-2025, MetaQuotes Ltd.

Indikatorkonstanten

Vorbestimmt sind 37 technische Standardindikatoren, die in Programmen mit der Sprache MQL5 verwendet werden können. Außerdem kann man seine eigene Benutzerindikatoren durch die Funktion iCustom() erzeugen. Alle dafür notwendigen Konstanten werden in fuenf Gruppen aufgeteilt:

- Preiskonstanten – für Auswahl des Preis- oder Volumentypes, nach denen Indikator berechnet wird;
- Glaettungsmethoden – eingebaute Glaettungsmethoden, die in Indikatoren verwendet werden;
- Linien der Indikatoren – Identifikatoren der Indikatorpuffer beim Zugang zu Werten der Indikatoren durch die Funktion CopyBuffer();
- Zeichnungsstile – für Andeutung eines der 18 Typen der Zeichnung und Einstellung des Zeichnungsstils der Linie;
- Eigenschaften der Benutzindikatoren – werden in Funktionen für Arbeit mit Benutzerindikatoren verwendet;
- Typen der Indikatoren – werden für Andeutung des Typs des technischen Indikators bei der Erzeugung von handle durch die Funktion IndicatorCreate() verwendet;
- Indikatoren der Datentypen - werden für Festlegung des Datentyps verwendet, die durch das Feld des Typs MqlParam in die Funktion IndicatorCreate() übertragen werden.

© 2000-2025, MetaQuotes Ltd.

Preiskonstanten

Technische Indikatoren brauchen für ihre Berechnungen die Angaben von Preiswerten und/oder Volumenwerten, auf denen sie gerechnet werden. Es gibt 7 vorbestimmte Identifikatoren der Enumeration ENUM_APPLIED_PRICE, für Andeutung der notwendigen Preigrundlage der Berechnungen.

ENUM_APPLIED_PRICE

Identifikator	Beschreibung
PRICE_CLOSE	Eroeffnungspreis
PRICE_OPEN	Eroeffnungspreis
PRICE_HIGH	Maximaler Preis für die Periode
PRICE_LOW	Minimaler Preis für die Periode
PRICE_MEDIAN	Medianpreis, (high+low)/2
PRICE_TYPICAL	Typischer Preis, (high+low+close)/3
PRICE_WEIGHTED	Durchschnittlich gewogener Preis, (high+low+close+close)/4

Wenn das Volumen in Berechnungen verwendet wird, muss man einen der zwei Werte der Enumeration ENUM_APPLIED_VOLUME angeben .

ENUM_APPLIED_VOLUME

Identifikator	Beschreibung
VOLUME_TICK	Tickvolumen
VOLUME_REAL	Handelsvolumen

Der technische Indikator iStochastic() hat zwei Varianten für seine Berechnung :

- entweder nur die Preise Close;
- oder Preise High und Low.

für Auswahl der notwendigen Variante der Berechnung muss einen der Werte der Enumeration ENUM_STO_PRICE angegeben werden.

ENUM_STO_PRICE

Identifikator	Beschreibung
STO_LOWHIGH	Berechnung ist nach den Preisen Low/High
STO_CLOSECLOSE	Berechnung ist nach den Preisen Close/Close

Wenn ein technischer Indikator für seine Berechnungen Preisdaten verwendet, deren Typ durch die Enumeration ENUM_APPLIED_PRICE festgelegt wird, kann als Eingabepreisreihe handle jedes Indikators (der ins Terminal eingebaut wird oder vom Benutzer geschrieben wird) angegeben werden. In diesem Fall werden für Berechnungen Werte des Nullpuffers des Indikators verwendet werden. Das ermöglicht

© 2000-2025, MetaQuotes Ltd.

die Werte eines Indikators mittels der Werte eines anderen Indikators leicht zu erzeugen. Handle eines Benutzerindikators wird durch Aufruf der Funktion iCustom() erzeugt.

Beispiel:

```
#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   2
//--- input parameters
input int      RSIperiod=14;         // Periode für die Berechnung von RSI
input int      Smooth=8;             // Glaettungsperiode RSI
input ENUM_MA_METHOD meth=MODE_SMMA; // Glaettungsverfahren
//---- plot RSI
#property indicator_label1  "RSI"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//---- plot RSI_Smoothed
#property indicator_label2  "RSI_Smoothed"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrNavy
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- indicator buffers
double         RSIBuffer[];          // hier werden wir die Werte RSI aufbewahren
double         RSI_SmoothedBuffer[]; //hier haben wir Glaettungwerte RSI
int            RSIhandle;            // Handle für Indikator RSI
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,RSIBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,RSI_SmoothedBuffer,INDICATOR_DATA);
   IndicatorSetString(INDICATOR_SHORTNAME,"iRSI");
   IndicatorSetInteger(INDICATOR_DIGITS,2);
//---
   RSIhandle=iRSI(NULL,0,RSIperiod,PRICE_CLOSE);
//---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const int begin,
                const double &price[]
```

© 2000-2025, MetaQuotes Ltd.

```
                  )

  {
//--- stellen wir Wert des letzten Fehlers auf Null
   ResetLastError();
//--- bekommen wir Daten des Indikators RSI im Feld RSIBuffer[]
   int copied=CopyBuffer(RSIhandle,0,0,rates_total,RSIBuffer);
   if(copied<=0)
     {
      Print("Kopiren des Indikatorwertes Fehler RSI. Error = ",
            GetLastError(),",  copied =",copied);
      return(0);
     }
//---  using values of RSI erzeugen wir Indikator der Mittelwerte mit Verwendung der F
   int RSI_MA_handle=iMA(NULL,0,Smooth,0,meth,RSIhandle);
   copied=CopyBuffer(RSI_MA_handle,0,0,rates_total,RSI_SmoothedBuffer);
   if(copied<=0)
     {
      Print("Kopieren des gerichteten Feldes Fehler RSI. Error = ",
            GetLastError(),",  copied =",copied);
      return(0);
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

Glaettungsmethoden

Vielen technischen Indikatoren zugrunde liegt eine oder andere Methode der Mittelung von Preisserien. Einige technische Standardindikatoren fordern der Andeutung des Mittelungstyps als Parameter. Der Andeutung des notwendigen Typs der Mittelung dienen Identifikatoren, die in der Enumeration ENUM_MA_METHOD angegeben sind .

ENUM_MA_METHOD

Identifikator	Beschreibung
MODE_SMA	einfache Mittelung
MODE_EMA	exponentielle Mittelung
MODE_SMMA	geglaettete Mittelung
MODE_LWMA	lineal ausgewogene Mittelung

Beispiel:

```
double ExtJaws[];
double ExtTeeth[];
double ExtLips[];
//---- handles for moving averages
int    ExtJawsHandle;
int    ExtTeethHandle;
int    ExtLipsHandle;
//--- get MA's handles
ExtJawsHandle=iMA(NULL,0,JawsPeriod,0,MODE_SMMA,PRICE_MEDIAN);
ExtTeethHandle=iMA(NULL,0,TeethPeriod,0,MODE_SMMA,PRICE_MEDIAN);
ExtLipsHandle=iMA(NULL,0,LipsPeriod,0,MODE_SMMA,PRICE_MEDIAN);
```

© 2000-2025, MetaQuotes Ltd.

Linien der Indikatoren

Einige technische Indikatoren haben mehrere auf dem Chart gezeichnete Puffer. Numerierung der Indikatorpuffers faengt mit 0 an. Beim Kopieren der Indikatorwerte durch die Funktion CopyBuffer() ins Feld des Typs double kann man für einige Indikatoren nicht die Nummer des kopierten Puffers, sondern den Identifikator dieses Puffers angeben.

Identifikatoren der Indikatorlinien, zulaessige für Kopieren der Indikatorwerte iMACD(), iRVI() und iStochastic()

Konstante	Wert	Beschreibung
MAIN_LINE	0	Grundlinie
SIGNAL_LINE	1	Sygnallinie

Identifikatoren der Indikatorlinien, die beim Kopieren der Indikatorwerte ADX() und ADXW() zulaessig sind

Konstante	Wert	Beschreibung
MAIN_LINE	0	Grundlinie
PLUSDI_LINE	1	Linie +DI
MINUSDI_LINE	2	Linie -DI

Identifikatoren der Indikatorlinien, die beim Kopieren der Indikatorwerte iBands() zulaessig sind

Konstante	Wert	Beschreibung
BASE_LINE	0	Grundlinie
UPPER_BAND	1	Obere Grenze
LOWER_BAND	2	Untere Grenze

Identifikatoren der Indikatorlinien, die beim Kopieren der Indikatorwerte iEnvelopes() und iFractals() zulaessig sind

Konstante	Wert	Beschreibung
UPPER_LINE	0	Obere Linie
LOWER_LINE	1	Untere Linie

Identifikatiren der Indikatorlinien, die beim Kopieren der Werte des Indikators iGator() zulaessig sind

Konstante	Wert	Beschreibung
UPPER_HISTOGRAM	0	Oberes Histogramm
LOWER_HISTOGRAM	2	Unteres Histogramm

© 2000-2025, MetaQuotes Ltd.

Identifikatiren der Indikatorlinien, die beim Kopieren der Werte des Indikators iAlligator() zulaessig sind

Konstante	Wert	Beschreibung
GATORJAW_LINE	0	Linie der Kiefer
GATORTEETH_LINE	1	Linie der Zaehne
GATORLIPS_LINE	2	Linie der Lippen

Identifikatiren der Indikatorlinien, die beim Kopieren der Werte des Indikators iIchimoku() zulaessig sind

Konstante	Wert	Beschreibung
TENKANSEN_LINE	0	Linie Tenkan-sen
KIJUNSEN_LINE	1	Linie Kijun-sen
SENKOUSPANA_LINE	2	Linie Senkou Span A
SENKOUSPANB_LINE	3	Linie Senkou Span B
CHIKOUSPAN_LINE	4	Linie Chikou Span

© 2000-2025, MetaQuotes Ltd.

Zeichnungsstile

Bei Erzeugung der Benutzerindikatoren kann einer der 18 Typen der graphischen Konstruktion (Darstellungsart im Hauptfenster des Charts oder im Subfenster des Charts)angegeben werden, dessen Werte in der Enumeration ENUM_DRAW_TYPE spezifiziert werden.

Es ist zulaessig, in einem Benutzerindikator beliebige Arten der Konstruktion/Zeichnung der Indikatoren zu verwenden. Für jeden Konstruktionstyp muss von 1 bis 5 globale Felder für Datenspeicherung angegeben werden, die für die Zeichnung erforderlich sind. Diese Datenfelder muessen mit Indikatorpuffern durch die Funktion SetIndexBuffer() verbunden werden, und für jeden Puffer muss der Datentyp von der Enumeration ENUM_INDEXBUFFER_TYPE angegeben werden.

Abhängig vom Zeichnungsstil können Sie von 1 bis 4 Wertpuffer brauchen (vermerkt als INDICATOR_DATA), wenn der dynamische Farbenwechsel vom Stil zugelaeest wird (aale Stile mit dem Wort COLOR im Namen), braucht man auch einen Farbenpuffer (angegeben Typ INDICATOR_COLOR_INDEX). Farbenpuffer wird immer gebunden nach den dem Stil entsprechenden Wertpuffern.

ENUM_DRAW_TYPE

Identifikator	Beschreibung	Werte der Puffer	Farbe der Puffer
DRAW_NONE	Nicht gezeichnet	1	0
DRAW_LINE	Linie	1	0
DRAW_SECTION	Abschnitte	1	0
DRAW_HISTOGRAM	Histogramm von der Nulllinie	1	0
DRAW_HISTOGRAM2	Histogramm der 2 Indikatorpuffer	2	0
DRAW_ARROW	Weichenaufzeichnung	1	0
DRAW_ZIGZAG	Stil Zigzag laesst vertikale Abschnitte zu	2	0

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Werte der Puffer	Farbe der Puffer
DRAW_FILLING	Farbenfuellung zwischen zwei Levels	2	0
DRAW_BARS	Darstellung als Folge von Bars	4	0
DRAW_CANDLES	Darstellung als Folge von Kerzen	4	0
DRAW_COLOR_LINE	Mehrfarbige Linie	1	1
DRAW_COLOR_SECTION	Mehrfarbige Abschnitte	1	1
DRAW_COLOR_HISTOGRAM	Mehrfarbiges Histogramm von der Nulllinie	1	1
DRAW_COLOR_HISTOGRAM2	Mehrfarbiges Histogramm der zwei Indikatorpuffer	2	1
DRAW_COLOR_ARROW	Zeichnung mit mehrfarbigen Weichen	1	1

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Werte der Puffer	Farbe der Puffer
DRAW_COLOR_ZIGZAG	Mehrfarbiger ZigZag	2	1
DRAW_COLOR_BARS	Mehrfarbige Bars	4	1
DRAW_COLOR_CANDLES	Mehrfarbige Kerzen	4	1

für Praezisierung der Darstellung der gewaehlten Zeichnungsart werden Identifikatoren verwendet, die in Enumerationen ENUM_PLOT_PROPERTY aufgezählt werden.

für Funktionen PlotIndexSetInteger() und PlotIndexGetInteger()

ENUM_PLOT_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
PLOT_ARROW	Weichenkode für Stil DRAW_ARROW	uchar
PLOT_ARROW_SHIFT	Vertikale Verschiebung der Weichen für Stil DRAW_ARROW	int
PLOT_DRAW_BEGIN	Zahl der Anfangsbars ohne Zeichnung und Werte in DataWindow	int
PLOT_DRAW_TYPE	Typ der graphischen Konstruktion	ENUM_DRAW_TYPE
PLOT_SHOW_DATA	Merkmal der Darstellung von Konstruktionswerte im Fenster DataWindow	bool
PLOT_SHIFT	Verschiebung der graphischen Konstruktion des Indikators Zeit-Achse in Bars	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
PLOT_LINE_STYLE	Stil der zeichnerischen Linie	ENUM_LINE_STYLE
PLOT_LINE_WIDTH	Breite der zeichnerischen Linie	int
PLOT_COLOR_INDEXES	Anzahl der Farben	int
PLOT_LINE_COLOR	Pufferindex mit zeichnerischer Farbe	color Modifikator=Nummer der Indexfarbe

für die Funktion PlotIndexSetDouble()

ENUM_PLOT_PROPERTY_DOUBLE

Identifikator	Beschreibung	Typ der Eigenschaft
PLOT_EMPTY_VALUE	Leerwert für Konstruktion, für die es keine Zeichnung gibt	double

für die Funktion PlotIndexSetString()

ENUM_PLOT_PROPERTY_STRING

Identikator	Beschreibung	Typ der Eigenschaft
PLOT_LABEL	Name der graphischen Indikatorserie für Darstellung im Fenster DataWindow. Für komplexe grafische Stile, die mehrere Indikatorpuffer für Anzeige brauchen, können die Namen von jedem Puffer mit Trennzeichen ";" angegeben werden. Beispiel-Code wird in	string

© 2000-2025, MetaQuotes Ltd.

Identikator	Beschreibung	Typ der Eigenschaft
	DRAW_CANDLES gezeigt	

Es gibt 5 Stile mit denen die Linie im Benutzerindikator gezeichnet werden kann. Sie sind nur bei der Liniebreite 0 oder 1 anwendbar.

ENUM_LINE_STYLE

Identifikator	Beschreibung
STYLE_SOLID	Ausgezogene Linie
STYLE_DASH	diskontinierliche Linie
STYLE_DOT	Punktierte Linie
STYLE_DASHDOT	Strichpunktlinie
STYLE_DASHDOTDOT	Strich-zwei Punkte

für Einstellung des Stils der Liniezeichnung und Zeichnungsart wird die Funktion PlotIndexSetInteger() verwendet. Für Fibonacci-Erweiterungen können die Breite und Stil der Zeichnung durch die Funktion ObjectSetInteger() angegeben werden.

Beispiel:

```
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- indicator buffers
double         MABuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- Bindung des Feldes zum Indikatorpuffer mit Index 0
   SetIndexBuffer(0,MABuffer,INDICATOR_DATA);
//--- Einstellung der Liniezeichnung
   PlotIndexSetInteger(0,PLOT_DRAW_TYPE,DRAW_LINE);
//--- Einstellung des Stils für Liniezeichnung
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,STYLE_DOT);
//--- Einstellung der Linienfarbe
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,clrRed);
//--- Einstellung der Linienbreite
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,1);
//--- Einstellung der Marke für die Linie
   PlotIndexSetString(0,PLOT_LABEL,"Moving Average");
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---
   for(int i=prev_calculated;i<rates_total;i++)
     {
      MABuffer[i]=close[i];
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

Eigenschaften der Benutzeranzeiger

Zahl der Indikatorpuffer, die im Benutzerindikator verwendet werden können ist unbegrenzt. Aber es ist notwendig, für jedes Feld, das als Indikatorpuffer durch die Funktion SetIndexBuffer() zugeordnet wird, muss der Datentyp angegeben werden, den es aufbewahren wird. Das kann einer der Werte der Enumeration ENUM_INDEXBUFFER_TYPE sein.

ENUM_INDEXBUFFER_TYPE

Identifikator	Beschreibung
INDICATOR_DATA	Angaben für Zeichnen
INDICATOR_COLOR_INDEX	Farbe
INDICATOR_CALCULATIONS	Hilfspuffer für Zwischenberechnungen

Benutzerindikator hat viele Einstellungen für Darstellung und Erfassen. Diese Einstellungen werden durch Festlegung der entsprechenden Funktionen des Indikators durch die Funktionen IndicatorSetDouble(), IndicatorSetInteger() und IndicatorSetString() durchgeführt. Identifikatoren der Indikatoreigenschaften gibt es in Enumerationen ENUM_CUSTOMIND_PROPERTY.

ENUM_CUSTOMIND_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
INDICATOR_DIGITS	Genauigkeit der Darstellung von Indikatorwerten	int
INDICATOR_HEIGHT	Feste Höhe von Indikators	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	eigenen Fenster (Befehl von Preprozessor #property indicator_height)	
INDICATOR_LEVELS	Anzahl der Levels im Indikatorfenster	int
INDICATOR_LEVELCOLOR	Farbe der Levellinie	color Modifikator=Levelnummer
INDICATOR_LEVELSTYLE	Stil der Lev	ENUM_LINE_STYLE Modifiaktor=Levelnummer

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	elli nie	
INDICATOR_LEVELWIDTH	Bre ite der Lev elli nie	int Modifikator=Levelnummer
INDICATOR_FIXED_MINIMUM	Fes tes Min imu m für das Indi kat orf ens ter. Die Eig ens cha ft kan n nur von der Fun kti on Indi cat orS etI nte ger () fes tgel egt wer	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	den.	
INDICATOR_FIXED_MAXIMUM	Festgelegter Höchstwert für das Indikatorfenster. Die Eigenschaft kann nur von der Funktion IndicatorSetInteger() festgelegt werden.	bool

ENUM_CUSTOMIND_PROPERTY_DOUBLE

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
INDICATOR_MINIMUM	Minimum des Indikatorfensters	double
INDICATOR_MAXIMUM	Maximum des Indikatorfensters	double
INDICATOR_LEVELVALUE	Levelwert	double Modifikator=Levelnummer

ENUM_CUSTOMIND_PROPERTY_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
INDICATOR_SHORTNAME	Kurzer Indikatorname	string
INDICATOR_LEVELTEXT	Levelbeschreibung	string Modifikator=Levelnummer

© 2000-2025, MetaQuotes Ltd.

Beispiele:

```
//--- indicator settings
#property indicator_separate_window
#property indicator_buffers 4
#property indicator_plots   2
#property indicator_type1   DRAW_LINE
#property indicator_type2   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_color2  clrRed
//--- input parameters
extern int KPeriod=5;
extern int DPeriod=3;
extern int Slowing=3;
//--- indicator buffers
double MainBuffer[];
double SignalBuffer[];
double HighesBuffer[];
double LowesBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,MainBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,SignalBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,HighesBuffer,INDICATOR_CALCULATIONS);
   SetIndexBuffer(3,LowesBuffer,INDICATOR_CALCULATIONS);
//--- set accuracy
   IndicatorSetInteger(INDICATOR_DIGITS,2);
//--- set levels
   IndicatorSetInteger(INDICATOR_LEVELS,2);
   IndicatorSetDouble(INDICATOR_LEVELVALUE,0,20);
   IndicatorSetDouble(INDICATOR_LEVELVALUE,1,80);
//--- set maximum and minimum for subwindow
   IndicatorSetDouble(INDICATOR_MINIMUM,0);
   IndicatorSetDouble(INDICATOR_MAXIMUM,100);
//--- sets first bar from what index will be drawn
   PlotIndexSetInteger(0,PLOT_DRAW_BEGIN,KPeriod+Slowing-2);
   PlotIndexSetInteger(1,PLOT_DRAW_BEGIN,KPeriod+Slowing+DPeriod);
//--- set style STYLE_DOT for second line
   PlotIndexSetInteger(1,PLOT_LINE_STYLE,STYLE_DOT);
//--- name for DataWindow and indicator subwindow label
   IndicatorSetString(INDICATOR_SHORTNAME,"Stoch("+KPeriod+","+DPeriod+","+Slowing+")"
   PlotIndexSetString(0,PLOT_LABEL,"Main");
   PlotIndexSetString(1,PLOT_LABEL,"Signal");
//--- sets drawing line to empty value
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
   PlotIndexSetDouble(1,PLOT_EMPTY_VALUE,0.0);
//--- initialization done
  }
```

© 2000-2025, MetaQuotes Ltd.

Typen der technischen Indikatoren

Es gibt 2 Wege, handle eines Indikators für weiteren Zugang zu seinen Elementen programmgemaess zu erzeugen. Der erste Weg ist die direkte Andeutung des Funktionsnamens aus der Liste der technischen Indikatoren. Der zweite Weg ermöglicht durch die Funktion IndicatorCreate() handle jedes Indikators durch Zuordnung des Identifikatoren aus der Enumeration ENUM_INDICATOR einheitlich zu erzeugen. Beide Varianten der Erzeugung von handle sind gleichgestellt, Sie können diejenige verwenden, die bei der Schreibung des MQL5 Programms für Sie am meisten passt.

Bei der Erzeugung eines Indikators des Typs IND_CUSTOM, muss das Feld *type* des ersten Elements des Feldes der Eingabeparameter MqlParam den Wert TYPE_STRING aus der Enumeration ENUM_DATATYPE haben, und das Feld *string_value* des ersten Elements den Namen des Benutzerindikators enthalten.

ENUM_INDICATOR

Identifikator	Indikator
IND_AC	Accelerator Oscillator
IND_AD	Accumulation/Distribution
IND_ADX	Average Directional Index
IND_ADXW	ADX by Welles Wilder
IND_ALLIGATOR	Alligator
IND_AMA	Adaptive Moving Average
IND_AO	Awesome Oscillator
IND_ATR	Average True Range
IND_BANDS	Bollinger Bands®
IND_BEARS	Bears Power
IND_BULLS	Bulls Power
IND_BWMFI	Market Facilitation Index
IND_CCI	Commodity Channel Index
IND_CHAIKIN	Chaikin Oscillator
IND_CUSTOM	Custom indicator
IND_DEMA	Double Exponential Moving Average
IND_DEMARKER	DeMarker
IND_ENVELOPES	Envelopes
IND_FORCE	Force Index
IND_FRACTALS	Fractals
IND_FRAMA	Fractal Adaptive Moving Average
IND_GATOR	Gator Oscillator

© 2000-2025, MetaQuotes Ltd.

Identifikator	Indikator
IND_ICHIMOKU	Ichimoku Kinko Hyo
IND_MA	Moving Average
IND_MACD	MACD
IND_MFI	Money Flow Index
IND_MOMENTUM	Momentum
IND_OBV	On Balance Volume
IND_OSMA	OsMA
IND_RSI	Relative Strength Index
IND_RVI	Relative Vigor Index
IND_SAR	Parabolic SAR
IND_STDDEV	Standard Deviation
IND_STOCHASTIC	Stochastic Oscillator
IND_TEMA	Triple Exponential Moving Average
IND_TRIX	Triple Exponential Moving Averages Oscillator
IND_VIDYA	Variable Index Dynamic Average
IND_VOLUMES	Volumes
IND_WPR	Williams' Percent Range

© 2000-2025, MetaQuotes Ltd.

Identifikatoren der Datentypen

Bei der Erzeugung von handle des Indikators durch die Funktion IndicatorCreate() muss man das Feld des Typs MqlParam als letzter Parameter angeben. So hat die Struktur MqlParam, die Indikatorparameter beschreibt, ein spezielles Feld *type*. In diesem Feld gibt es Information über Datentyp (Realtyp, ganzzahliger Typ oder Zeilentyp), die vom entsprechenden Element dieses Feldes übertagen werden. Wert dieses Feldes der Struktur MqlParam kann einer der Enumerationswerte ENUM_DATATYPE sein.

ENUM_DATATYPE

Identifikator	Datentyp
TYPE_BOOL	bool
TYPE_CHAR	char
TYPE_UCHAR	uchar
TYPE_SHORT	short
TYPE_USHORT	ushort
TYPE_COLOR	color
TYPE_INT	int
TYPE_UINT	uint
TYPE_DATETIME	datetime
TYPE_LONG	long
TYPE_ULONG	ulong
TYPE_FLOAT	float
TYPE_DOUBLE	double
TYPE_STRING	string

Jedes Element dieses Feldes beschreibt den entsprechenden Eingabeparameter des erzeugten technischen Indikators, darum muessen der Typ und Reihenfolge der Elemente des Feldes entsprechend der Beschreibung strikt gehalten werden.

© 2000-2025, MetaQuotes Ltd.

Medium Zustand

Konstanten, die laufendes Medium von mql5-Programms beschreiben, werden in Gruppen aufgeteilt:

- Client-Terminal-Status – Information über das Client-Terminal;
- Information über das abgelaufene MQL5-Programm – Eigenschaften des mql5-Programms, die sein Verhalten zu steuern helfen;
- Symboleigenschaften – Erhalten von Handelsinformation über das Symbol;
- Kontoeigenschaften – Information über das laufende Handelskonto;
- Teststatistik – Testergebnisse eine Expert Advisors.

© 2000-2025, MetaQuotes Ltd.

Status des Client-Terminals

Identifikatoren zum Abrufen der Information über das Client-Terminal mit den Funktionen TerminalInfoInteger() und TerminalInfoString(). Als Parameter erwarten diese Funktionen Werte aus den Enumerationen ENUM_TERMINAL_INFO_INTEGER und ENUM_TERMINAL_INFO_STRING.

ENUM_TERMINAL_INFO_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
TERMINAL_BUILD	Die Build-Nummer des Client-Terminals	int
TERMINAL_COMMUNITY_ACCOUNT	Das Flag zeigt das Vorhandensein von Autorisierungsdaten der QL5.community im Terminal an	bool
TERMINAL_COMMUNITY_CONNECTION	Verbindung zur MQL5.community	bool
TERMINAL_CONNECTED	Verbindung mit dem Handelsserver	bool
TERMINAL_DLLS_ALLOWED	Berechtigung, DLL zu verwenden	bool
TERMINAL_TRADE_ALLOWED	Berechtigung zu handeln	bool
TERMINAL_EMAIL_ENABLED	Berechtigung zum Senden von E-Mails über den SMTP-Server und dem Login, der in den Terminaleinstellungen festgelegt sind	bool
TERMINAL_FTP_ENABLED	Berechtigung, Berichte via FTP zu senden an das angegebene Server für das in Terminaleinstellungen angegebene Handelskonto	bool
TERMINAL_NOTIFICATIONS_ENABLED	Berechtigung, Benachrichtigungen auf Smartphone zu senden	bool
TERMINAL_MAXBARS	Maximale Anzahl der Bars auf dem Chart	int
TERMINAL_MQID	Das Flag zeigt das Vorhandensein von Daten der MetaQuotes-ID für das Senden von Push-Benachrichtigungen	bool
TERMINAL_CODEPAGE	Nummer der Kodeseite der Sprache, die im Client-Terminal installiert ist	int
TERMINAL_CPU_CORES	Die Anzahl der CPU-Kerne im System	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
TERMINAL_DISK_SPACE	Freier Festplattenspeicher für den Ordner MQL5\Files des Terminals (des Agenten), in MB	int
TERMINAL_MEMORY_PHYSICAL	Physikalischer Speicher im System, in Mb	int
TERMINAL_MEMORY_TOTAL	Für das Terminal (des Agenten) verfügbarer Speicher, in MB	int
TERMINAL_MEMORY_AVAILABLE	Freier Speicher für das Terminal (Agenten), in MB	int
TERMINAL_MEMORY_USED	Verwendeter Speicher des Terminals (Agent), in MB	int
TERMINAL_X64	Angabe von "64-Bit-Terminal"	bool
TERMINAL_OPENCL_SUPPORT	Die Version von unterstützten OpenCL in Form von 0x00010002 = 1.2. "0" bedeutet, dass OpenCL nicht unterstützt wird	int
TERMINAL_SCREEN_DPI	Die Bildschirmauflösung wird in DPI (Dots per Inch) angegeben. Wenn Sie den Wert des Parameters wissen, können Sie die Größe der Grafikobjekte so angeben, dass sie gleich auf den Monitoren mit verschiedene Auflösungen aussehen.	int
TERMINAL_SCREEN_LEFT	Die linke Koordinate des virtuellen Bildschirms. Der virtuelle Bildschirm ist das Rechteck, das alle Bildschirme umfasst. Wenn es zwei Bildschirme im System gibt und ihre Reihenfolge von rechts nach links angegeben ist, kann die linke Koordinate des virtuellen Bildschirms an der Grenze der zwei Bildschirme liegen.	int
TERMINAL_SCREEN_TOP	Die obere Koordinate des virtuellen Bildschirms	int
TERMINAL_SCREEN_WIDTH	Terminalbreite	int
TERMINAL_SCREEN_HEIGHT	Terminalhöhe	int
TERMINAL_LEFT	Die linke Koordinate des Terminals hinsichtlich des virtuellen Bildschirms	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
TERMINAL_TOP	Die obere Koordinate des Terminals hinsichtlich des virtuellen Bildschirms	int
TERMINAL_RIGHT	Die rechte Koordinate des Terminals hinsichtlich des virtuellen Bildschirms	int
TERMINAL_BOTTOM	Die untere Koordinate des Terminals hinsichtlich des virtuellen Bildschirms	int
TERMINAL_PING_LAST	Der letzte bekannte Wert des Pings zum Handelsserver in Mikrosekunden. Eine Mikrosekunde ist eine millionstel Sekunde.	int
TERMINAL_VPS	Angabe, dass das Terminal auf dem MetaTrader Virtual Hosting Server (MetaTrader VPS) gestartet wird.	bool
Identifikator der Tasten	**Beschreibung**	
TERMINAL_KEYSTATE_LEFT	Status der Taste "Pfeil nach links"	int
TERMINAL_KEYSTATE_UP	Status der Taste "Pfeil nach oben"	int
TERMINAL_KEYSTATE_RIGHT	Status der Taste "Pfeil nach rechts"	int
TERMINAL_KEYSTATE_DOWN	Status der Taste "Pfeil nach unten"	int
TERMINAL_KEYSTATE_SHIFT	Status der Shift-Taste	int
TERMINAL_KEYSTATE_CONTROL	Status der Strg-Taste	int
TERMINAL_KEYSTATE_MENU	Status der Windows-Taste	int
TERMINAL_KEYSTATE_CAPSLOCK	Status der CapsLock-Taste	int
TERMINAL_KEYSTATE_NUMLOCK	Status der NumLock-Taste	int
TERMINAL_KEYSTATE_SCRLOCK	Status der ScrollLock-Taste	int
TERMINAL_KEYSTATE_ENTER	Status der Enter-Taste	int
TERMINAL_KEYSTATE_INSERT	Status der Insert-Taste	int
TERMINAL_KEYSTATE_DELETE	Status der Delete-Taste	int
TERMINAL_KEYSTATE_HOME	Status der Home-Taste	int
TERMINAL_KEYSTATE_END	Status der End-Taste	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
TERMINAL_KEYSTATE_TAB	Status der Tab-Taste	int
TERMINAL_KEYSTATE_PAGEUP	Status der PageUp-Taste	int
TERMINAL_KEYSTATE_PAGEDOWN	Status der PageDown-Taste	int
TERMINAL_KEYSTATE_ESCAPE	Status der Escape-Taste	int

Der Aufruf von TerminalInfoInteger(TERMINAL_KEYSTATE_XXX) gibt den gleichen Code des Status einer Taste zurück, wie die Funktion GetKeyState() in MSDN.

Beispiel für die Berechnung des Skalierungsfaktors:

```
//--- Erstellung einer Schaltfläche mit der Breite 1,5 Zoll
int screen_dpi = TerminalInfoInteger(TERMINAL_SCREEN_DPI); // DPI des Benutzerbildschi
int base_width = 144;                                      // Die Basisbreite in Bilds
int width      = (button_width * screen_dpi) / 96;         // Die Schaltflächenbreite
...

//--- Skalierungsfaktor in Prozent berechnen
int scale_factor=(TerminalInfoInteger(TERMINAL_SCREEN_DPI) * 100) / 96;
//--- Skalierungsfaktor verwenden
width=(base_width * scale_factor) / 100;
```

Mit dieser Verwendung, hat die Grafik-Ressource die gleiche Größe auf den Bildschirmen mit der unterschiedlichen Auflösung. Die Größe der Steuerelemente (Schaltflächen, Dialogfelder, etc.) entsprechen den Personalisierungsoptionen.

ENUM_TERMINAL_INFO_DOUBLE

Bezeichner	Beschreibung	Typ der Eigenschaft
TERMINAL_COMMUNITY_BALANCE	Kontostand des Nutzers in der MQL5.community	double
TERMINAL_RETRANSMISSION	Anteil von erneut gesendeten Netzwerkpaketen im TCP/IP Protokoll für alle laufenden Anwendungen und Dienste auf dem gegebenen PC. Auch im schnellsten und richtig konfigurierten Netzwerk kann es zu Paketverlusten kommen, als Ergebnis gibt es keine Bestätigungen über die Zustellung von Paketen zwischen dem	double

© 2000-2025, MetaQuotes Ltd.

Bezeichner	Beschreibung	Typ der Eigenschaft
	Empfänger und dem Sender. In solchen Fällen werden die "verloren gegangenen" Pakete erneut gesendet. Das ist kein Indikator der Verbindungsqualität eines konkreten Terminals zu einem konkreten Handelsserver, denn er wird für die ganze Netzwerkaktivität, einschließlich System- und Hintergrundsaktivität, berechnet. Der Wert TERMINAL_RETRANSMISSION wird einmal pro Minute aus dem Betriebssystem abgerufen. Das Terminal selbst berechnet diesen Wert nicht.	

Dateioperationen können nur in zwei Verzeichnissen durchgeführt werden, Pfade zu denen durch die Anforderung der Eigenschaften TERMINAL_DATA_PATH und TERMINAL_COMMONDATA_PATH erhalten werden können.

ENUM_TERMINAL_INFO_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
TERMINAL_LANGUAGE	Sprache des Terminals	string
TERMINAL_COMPANY	Name der Gesellschaft	string
TERMINAL_NAME	Terminalname	string
TERMINAL_PATH	Ordner aus dem das Terminal gestartet wird	string
TERMINAL_DATA_PATH	Ordner, in dem die Terminladaten gespeichert werden	string
TERMINAL_COMMONDATA_PATH	Gesamtordner aller Client-Terminals, die im Computer installiert werden	string
TERMINAL_CPU_NAME	CPU-Name	string
TERMINAL_CPU_ARCHITECTURE	CPU-Architektur	string

© 2000-2025, MetaQuotes Ltd.

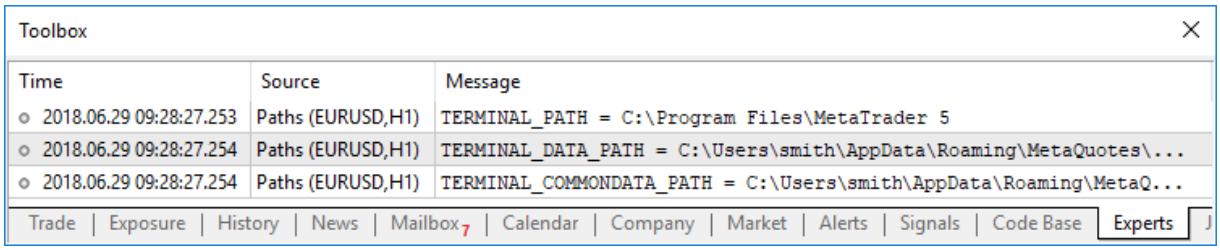
Identifikator	Beschreibung	Typ der Eigenschaft
TERMINAL_OS_VERSION	Name des Betriebssystems des Nutzers	string
TERMINAL_COLORTHEME_NAME	Farbschema des Terminals; mögliche Werte: Hell und Dunkel.	string

Zum besseren Verständnis der Pfade, die in Eigenschaften der Parameter TERMINAL_PATH, TERMINAL_DATA_PATH und TERMINAL_COMMONDATA_PATH gespeichert sind, ist es empfehlenswert, das Skript auszuführen, das diese Werte für die aktuelle Kopie des auf Ihrem Computer installierten Client-Terminals zurückgibt.

Beispiel: Das Skript gibt die Information über Terminalpfade zurück

```
//+------------------------------------------------------------------+
//|                                          Check_TerminalPaths.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
//---+
   Print("TERMINAL_PATH = ",TerminalInfoString(TERMINAL_PATH));
   Print("TERMINAL_DATA_PATH = ",TerminalInfoString(TERMINAL_DATA_PATH));
   Print("TERMINAL_COMMONDATA_PATH = ",TerminalInfoString(TERMINAL_COMMONDATA_PATH));
  }
```

Als Ergebnis der Ausführung des Skripts können Sie im Expert-Journal die Einträge sehen, wie sie unten angezeigt werden.



Informationen zum Farbschema des Terminals abrufen

© 2000-2025, MetaQuotes Ltd.

Das Terminal unterstützt zwei Farbschemata: Hell (Standard) und Dunkel. Bei der Entwicklung von nutzerdefinierten Anwendungen mit grafischen Nutzeroberflächen sollten die Programmierer das aktuelle Farbschema des Terminals berücksichtigen. Die in der Anwendung verwendeten visuellen Komponenten sollten dynamisch anpassbar sein, um die Nutzerfreundlichkeit zu verbessern und die visuelle Konsistenz zu wahren.

Um dies zu unterstützen, bietet die Sprache Funktionen an, mit denen das Farbschema des Terminals abgerufen werden kann:

- Der Wert TERMINAL_COLORTHEME_NAME aus der Enumeration ENUM_TERMINAL_INFO_STRING ermöglicht es, den Namen des aktuellen Farbschemas mit der Funktion TerminalInfoString abzurufen. Mögliche Werte: Light (hell) und Dark (dunkel).
- Verwenden Sie die Werte THEME_COLOR_* aus der Enumeration ENUM_TERMINAL_INFO_INTEGER, um die Farben bestimmter UI-Elemente über die Funktion TerminalInfoInteger abzurufen.

Identifikator	Beschreibung	Typ der Eigenschaft
THEME_COLOR_WINDOW	Hintergrundfarbe des Fensters	color
THEME_COLOR_WINDOWTEXT	Textfarbe im Fenster	color
THEME_COLOR_BTNTEXT	Textfarbe auf Schaltflächen	color
THEME_COLOR_GRAYTEXT	Inaktive (deaktivierte) Textfarbe	color
THEME_COLOR_INFOTEXT	Textfarbe eines Tooltips	color
THEME_COLOR_INFOBK	Die Farbe des Hintergrunds eines Tooltips	color
THEME_COLOR_3DFACE	Vorderseite von 3D-Elementen	color
THEME_COLOR_3DLIGHT	Helle Seite von 3D-Elementen	color
THEME_COLOR_3DSHADOW	Schattenseite von 3D-Elementen	color
THEME_COLOR_3DDKSHADOW	Dunkle Schatten von 3D-Elementen	color
THEME_COLOR_3DHILIGHT	Hervorhebung von 3D Elemente	color
THEME_COLOR_HIGHLIGHT	Hintergrund der ausgewählten Elemente	color
THEME_COLOR_HIGHLIGHTTEXT	Textfarbe der ausgewählten Elemente	color
THEME_COLOR_BTNFACE	Vorderseite von Schaltflächen	color
THEME_COLOR_BTNHILIGHT	Hervorgehobene Schaltflächen	color
THEME_COLOR_BTNSHADOW	Schatten von Schaltflächen	color
THEME_COLOR_MENU	Hintergrund des Menüs	color

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
THEME_COLOR_MENUBAR	Hintergrund des Menüs	color
THEME_COLOR_MENUTEXT	Menütext	color
THEME_COLOR_MENUHILIGHT	Hervorhebung des ausgewählten Menüelements	color
THEME_COLOR_ACTIVECAPTION	Aktiver Fenstertitel	color
THEME_COLOR_INACTIVECAPTION	Inaktiver Fenstertitel	color
THEME_COLOR_GRADIENTINACTIVECAPTION	Farbverlauf des inaktiven Fenstertitels	color
THEME_COLOR_CAPTIONTEXT	Text des Fenstertitels	color
THEME_COLOR_INACTIVECAPTIONTEXT	Text des Titels des inaktiven Fensters	color
THEME_COLOR_HOTTEXT	Hyperlinks oder aktive Elementen	color
THEME_COLOR_NONE	Keine Farbe ausgewählt	color
THEME_COLOR_SEPARATOR	Trennzeichen	color
THEME_COLOR_SCROLLBACK	Scrollbar	color
THEME_COLOR_LINE1	Hintergrundfarbe für ungerade Zeilen im Journal	color
THEME_COLOR_LINE2	Hintergrundfarbe für gerade Zeilen im Journal	color
THEME_COLOR_GRID	Die Farbe des Rasters im Journal	color
THEME_COLOR_SUMMARY	Hintergrundfarbe der Zusammenfassungszeile im Journal	color
THEME_COLOR_ERROR	Textfarbe der Fehlermeldung	color
THEME_COLOR_INVALID	Textfarbe für ungültige Werte	color
THEME_COLOR_NEGATIVE	Die Farbe für negative Werte	color
THEME_COLOR_POSITIVE	Die Farbe für positive Werte	color
THEME_COLOR_LINK	Die Farbe der Links	color
THEME_COLOR_LINKHOVER	Die Farbe für Link-Hover	color
THEME_COLOR_LINKTESTER	Die Farbe der Links der zwischengespeicherten Ergebnissen vorheriger Test-/Optimierungsläufe	color

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
THEME_COLOR_TEXTUP	Der Status "Schaltfläche losgelassen"	color
THEME_COLOR_TEXTDOWN	Der Status "Schaltfläche gedrückt"	color
THEME_COLOR_BACKUP	Die Farbe der Schaltflächen "BUY" und "SELL" im "One Click Trading" bei steigenden Kursen	color
THEME_COLOR_BACKDOWN	Die Farbe der Schaltflächen "BUY" und "SELL" im "One Click Trading" bei steigenden Kursen	color
THEME_COLOR_CLOSE	Die Farbe der Schaltfläche "Schließen"	color
THEME_COLOR_BUY	Die Farbe der Schaltfläche "BUY"	color
THEME_COLOR_SELL	Die Farbe der Schaltfläche "SELL"	color
THEME_COLOR_DEPOSIT	Die Farbe der Schaltfläche "Einzahlung"	color
THEME_COLOR_WITHDRAWAL	Die Farbe der Schaltfläche "Abheben"	color
THEME_COLOR_BID	Linienfarbe des Bid	color
THEME_COLOR_ASK	Linienfarbe des Ask	color
THEME_COLOR_STOPS	Linienfarbe der Stopps	color
THEME_COLOR_STOPS_RED	Hervorhebung der Stop-Loss-Werte bei negativem Gewinn (Registerkarte „Handel“)	color
THEME_COLOR_STOPS_GREEN	Hervorhebung der Stop-Loss-Werte bei positiven Gewinn (Registerkarte „Handel“)	color
THEME_COLOR_CONFIRM	Die Farbe der Schaltfläche "Accept" bm Auftragseingabefenster	color
THEME_COLOR_REQUOTE	Die Farbe der Schaltfläche "Requote" im Auftragseingabefenster	color
THEME_COLOR_REJECT	Die Farbe der Schaltfläche "Reject" im Auftragseingabefenster	color

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
THEME_COLOR_NOTIFICATION	Die Farbe einer Änderungsbenachrichtigung vom Server im Auftragseingabefenster	color
THEME_COLOR_RATING	Die Farbe der Bewertungsleiste in einem lernenden System	color
THEME_COLOR_BOOK_BUY	Hintergrundfarbe der Kauflevel in der Markttiefe	color
THEME_COLOR_BOOK_SELL	Hintergrundfarbe der Verkaufslevel in der Markttiefe	color
THEME_COLOR_BOOK_LAST	Die Farbe des letzten Handelsgeschäfts in der Markttiefe	color
THEME_COLOR_BOOK_STOP	Die Farbe des Stop-Loss-Levels in der Markttiefe	color
THEME_COLOR_BOOK_SPREAD	Hintergrundfarbe der Level innerhalb des Spreads in der Markttiefe	color
THEME_COLOR_TICKS_BID	Linienfarbe des Bid-Preises im Tick-Chart des Auftragseingabefensters	color
THEME_COLOR_TICKS_ASK	Linienfarbe des Ask-Preises im Tick-Chart des Auftragseingabefensters	color
THEME_COLOR_TICKS_LAST	Linienfarbe von "Last" im Tick-Chart des Auftragseingabefensters	color
THEME_COLOR_TICKS_CROSS	Die Farbe des Fadenkreuzes im Tick-Chart des Auftragseingabefensters	color
THEME_COLOR_TICKS_SL	Die Farbe der Stop-Loss-Linie im Tick-Chart des Auftragseingabefensters	color
THEME_COLOR_TICKS_TP	Die Farbe der Take-Profit-Linie im Tick-Chart des Auftragseingabefensters	color
THEME_COLOR_TESTER_START	Die Farbe der Schaltfläche "Start" im Test-/Optimierungsbereich	color

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
THEME_COLOR_TESTER_STOP	Die Farbe der Schaltfläche "Stop" im Test-/Optimierungsbereich	color
THEME_COLOR_TESTER_START_FRAME	Die Rahmenfarbe der Schaltfläche "Start" im Test-/Optimierungsbereich	color
THEME_COLOR_TESTER_STOP_FRAME	Die Rahmenfarbe der Schaltfläche "Stop" im Test-/Optimierungsbereich	color
THEME_COLOR_TESTER_PROGRESS	Die Farbe des Fortschrittsbalkens im Test/Optimierungsbereich	color
THEME_COLOR_TESTER_BALANCE	Die Farbe der Saldenlinie im Strategietester	color
THEME_COLOR_TESTER_EQUITY	Die Farbe der Kapitallinie im Strategietester	color
THEME_COLOR_TESTER_MARGIN	Diagrammfarbe der Einzahlungsbelastung im Strategietester	color
THEME_COLOR_PROFILER_CALL	Die Farbe der Codezeile bei einem Aufruf während der Profilerstellung	color
THEME_COLOR_PROFILER_CALLSEL	Die Farbe der ausgewählten Codezeile bei einem Aufruf während der Profilerstellung	color
THEME_COLOR_PROFILER_LINE	Die Farbe der Zeilen im Profiler-Journal	color
THEME_COLOR_PROFILER_LINESEL	Die Farbe der ausgewählten Zeilen im Profiler-Journal	color

© 2000-2025, MetaQuotes Ltd.

Eigenschaften des laufenden MQL5-Programms

Für das Erhalten von Informationen über ein laufendes mql5-Programm werden die in ENUM_MQL_INFO_INTEGER und ENUM_MQL_INFO_STRING aufgezählten Konstanten verwendet.

Für die Funktion MQLInfoInteger()

ENUM_MQL_INFO_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
MQL_HANDLES_USED	Die aktuelle Anzahl der aktiven Objekt-Handles. Dazu gehören sowohl dynamische (erstellt über new) als auch nicht-dynamische Objekte, globale/lokale Variablen oder Klassenmitglieder. Je mehr Handles ein Programm verwendet, desto mehr Ressourcen verbraucht es.	int
MQL_MEMORY_LIMIT	Die maximale Größe des dynamischen Speichers für ein MQL5-Programm in MB	int
MQL_MEMORY_USED	Die Größe des vom MQL5-Programm verwendeten Speichers in MB	int
MQL_PROGRAM_TYPE	Typ des MQL5-Programms	ENUM_PROGRAM_TYPE
MQL_DLLS_ALLOWED	Die Erlaubnis, DLL für das **vorgegebene laufende Programm** zu verwenden	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
MQL_TRADE_ALLOWED	Die Erlaubnis zum Handeln für das**vorgegebene laufende Programm**	bool
MQL_SIGNALS_ALLOWED	Die Erlaubnis für die Arbeit mit Signalen für das **vorgegebene laufende Programm**	bool
MQL_DEBUG	Hinweis, dass das Programm im Debugging-Modus läuft	bool
MQL_PROFILER	Hinweis, dass das Programm im Code Profiling-Modus läuft	bool
MQL_TESTER	Hinweis, dass das Programm im Tester läuft	bool
MQL_FORWARD	Hinweis, dass das Programm im Prozess des Vorwärts-Testens läuft	bool
MQL_OPTIMIZATION	Hinweis, dass das Programm im Optimierungsmodus läuft	bool
MQL_VISUAL_MODE	Hinweis, dass das Programm im Modus des visuellen Testens läuft	bool
MQL_FRAME_MODE	Hinweis, dass der Expert Advisor im Modus des Sammelns der Frames von	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Optimierungergebnissen läuft	
MQL_LICENSE_TYPE	Lizenztyp des EX5-Moduls. Die Lizenz bezieht sich auf das EX5-Modul, von dem aus eine Anfrage mit MQLInfoInteger (MQL_LICENSE_TYPE) gestellt wird.	ENUM_LICENSE_TYPE
MQL_STARTED_FROM_CONFIG	Gibt truezurück, wenn das Skript/Expert aus dem StartUp-Abschnitt der Konfigurationsdatei gestartet wurde. Das bedeutet, dass dieses Skript/Expert in der Konfigurationsdatei registriert wurde, mit der das Terminal gestartet wurde	bool

Für die Funktion MQLInfoString()

ENUM_MQL_INFO_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
MQL_PROGRAM_NAME	Name des laufenden MQL5-Programms	string
MQL_PROGRAM_PATH	Pfad für das **vorgegebene laufende Programm**	string

© 2000-2025, MetaQuotes Ltd.

Für die Abfrage der Information über den Typ des laufenden Programms werden die Werte der Aufzählung ENUM_PROGRAM_TYPE verwendet.

ENUM_PROGRAM_TYPE

Identifikator	Beschreibung
PROGRAM_SCRIPT	Skript
PROGRAM_EXPERT	Expert
PROGRAM_INDICATOR	Indikator
PROGRAM_SERVICE	Dienst (Service)

ENUM_LICENSE_TYPE

Identifikator	Beschreibung
LICENSE_FREE	Kostenlose Vollversion
LICENSE_DEMO	Demoversion eines kostenpflichtigen Produkts aus dem Market. Die Version funktioniert nur im Strategietester
LICENSE_FULL	Eine gekaufte lizenzierte Version erlaubt mindestens fünf Aktivierungen. Der Verkäufer kann die Anzahl der zulässigen Aktivierungen erhöhen
LICENSE_TIME	Version mit einer zeitlich begrenzten Lizenz

Beispiel:

```
ENUM_PROGRAM_TYPE mql_program=(ENUM_PROGRAM_TYPE)MQLInfoInteger(MQL_PROGRAM_TYPE);
switch(mql_program)
  {
   case PROGRAM_SCRIPT:
     {
      Print(__FILE__+" is script");
      break;
     }
   case PROGRAM_EXPERT:
     {
      Print(__FILE__+" is Expert Advisor");
      break;
     }
   case PROGRAM_INDICATOR:
     {
      Print(__FILE__+" is custom indicator");
      break;
     }
   default:Print("MQL5 type value is ",mql_program);
```

© 2000-2025, MetaQuotes Ltd.

```
   //---
   Print("MQLInfoInteger(MQL_MEMORY_LIMIT)=", MQLInfoInteger(MQL_MEMORY_LIMIT), " MB")
   Print("MQLInfoInteger(MQL_MEMORY_USED)=", MQLInfoInteger(MQL_MEMORY_USED), " MB");
   Print("MQLInfoInteger(MQL_HANDLES_USED)=", MQLInfoInteger(MQL_HANDLES_USED), " hand
```

© 2000-2025, MetaQuotes Ltd.

Information über das Symbol

für die Erhaltung der laufenden Marktinformation werden Funktionen SymbolInfoInteger(), SymbolInfoDouble() und SymbolInfoString() verwendet. Als zweiter Parameter dieser Funktionen ist es zulässig, einen der Identifikatoren von den Enumerationen ENUM_SYMBOL_INFO_INTEGER, ENUM_SYMBOL_INFO_DOUBLE und ENUM_SYMBOL_INFO_STRING zu übertragen.

für die Funktion SymbolInfoInteger()

ENUM_SYMBOL_INFO_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_SECTOR	Der Wirtschaftssektor, zu dem der Vermögenswert gehört.	ENUM_SYMBOL_SECTOR
SYMBOL_INDUSTRY	Die Industrie oder Wirtschaftszweig, zu dem das Sy	ENUM_SYMBOL_INDUSTRY

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	mbol gehört.	
SYMBOL_CUSTOM	Das Symbol ist benutzerdefiniert, mit anderen Worten wurde es basierend auf anderen Symbolen aus der Marktübersicht und/oder	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	externen Datenquellen erstellt	
SYMBOL_BACKGROUND_COLOR	Die Farbe des Hintergrunds, die für das Symbol in der Marktübersicht verwendet wird	color
SYMBOL_CHART_MODE	Preistyp für das Zeichn	ENUM_SYMBOL_CHART_MODE

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	en von Balken - Bid oder Last	
SYMBOL_EXIST	Ein Symbol mit diesem Namen existiert	bool
SYMBOL_SELECT	Symbol ist in MarketWatch gewählt	bool
SYMBOL_VISIBLE	Das ausgewählte Symbol	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ist sichtbar in MarketWatch. Einige Symbole (in der Regel sind das Kreuzkurse, die für die Berechnung von Marginanforderungen oder des Pro	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	fits in Kontowährung benötigt werden) werden automatisch ausgewählt, dabei können diese in der Marktübersicht nicht angezeigt werden. Damit solc	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	he Symbole angezeigt werden, muss man diese explizit auswählen.	
SYMBOL_SESSION_DEALS	Anzahl der Deals in der aktuellen Session	long
SYMBOL_SESSION_BUY_ORDERS	Die Gesamtzahl der Buy-Ordern	long

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	im Moment	
SYMBOL_SESSION_SELL_ORDERS	Die Gesamtzahl der Sell-Ordern im Moment	long
SYMBOL_VOLUME	Volume - Volumen im letzten Deal	long
SYMBOL_VOLUMEHIGH	Maximales Volumen pro Tag	long
SYMBOL_VOLUMELOW	Minimales Volum	long

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	en pro Tag	
SYMBOL_TIME	Zeit der letzten Quotation	datetime
SYMBOL_TIME_MSC	Zeitpunkt des letzten Kurses in Millisekunden seit 1970.01.01	long
SYMBOL_DIGITS	Anzahl der Dezimalzeichen	int
SYMBOL_SPREAD	Spreadgröße in	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Punkten	
SYMBOL_SPREAD_FLOAT	Symbol des fließenden Spreads	bool
SYMBOL_TICKS_BOOKDEPTH	Maximale Anzahl der in DOM gezeigten Anforderungen. Für Instrumente, die keine Quelle der Anforderu	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ngen haben, ist der Wert 0	
SYMBOL_TRADE_CALC_MODE	Mode der Berechnung von Vertragskosten	ENUM_SYMBOL_CALC_MODE
SYMBOL_TRADE_MODE	Typ der Orderdurchführung	ENUM_SYMBOL_TRADE_MODE
SYMBOL_START_TIME	Anfangsdatum der Versteigerung nach dem Inst	datetime

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	rument(gewöhnlich wird für Futures verwendet)	
SYMBOL_EXPIRATION_TIME	Abschlussdatum der Versteigerung nach dem Instrument (gewöhnlich wird für Futures ver	datetime

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	wendet)	
SYMBOL_TRADE_STOPS_LEVEL	Minimale Einrückung in Punkten vom laufenden Schlusspreis für Einstellung der Stop Order	int
SYMBOL_TRADE_FREEZE_LEVEL	Distanz der Einfrierung der Handelsoper	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ationen (in Punkten)	
SYMBOL_TRADE_EXEMODE	Modedes Dealsabschlusses	ENUM_SYMBOL_TRADE_EXECUTION
SYMBOL_SWAP_MODE	Modell der Swapberechnung	ENUM_SYMBOL_SWAP_MODE
SYMBOL_SWAP_ROLLOVER3DAYS	Wochentag für Anrechnung des dreifachen Swaps	ENUM_DAY_OF_WEEK
SYMBOL_MARGIN_HEDGED_USE_LEG	Berechnun	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	gsmodus der Hedging-Margin nach der größeren Seite (Buy oder Sell)	
SYMBOL_EXPIRATION_MODE	Flags der erlaubten Order-Ablaufmodi	int
SYMBOL_FILLING_MODE	Flags der erlaubten Order-Füll	int

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	modi	
SYMBOL_ORDER_MODE	Flags der erlaubten Ordertypen	int
SYMBOL_ORDER_GTC_MODE	Gültigkeitsdauer von StopLoss und TakeProfit Ordern, wenn SYMBOL_EXPIRATION_MODE=SYMBOL_EXPI	ENUM_SYMBOL_ORDER_GTC_MODE

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	RATION_GTC (Good till cancelled)	
SYMBOL_OPTION_MODE	Option styp	ENUM_SYMBOL_OPTION_MODE
SYMBOL_OPTION_RIGHT	Optionsrecht (Call/Put)	ENUM_SYMBOL_OPTION_RIGHT

für die Funktion SymbolInfoDouble()

ENUM_SYMBOL_INFO_DOUBLE

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_BID	Bid - das beste Verkaufsangebot	double
SYMBOL_BIDHIGH	maximales Bid pro Tag	double
SYMBOL_BIDLOW	Minimales Bid pro Tag	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_ASK	Ask - das beste Kaufsangebot	double
SYMBOL_ASKHIGH	Maximales Ask pro Tag	double
SYMBOL_ASKLOW	Minimales Ask pro Tag	double
SYMBOL_LAST	Preis des letzten Deals	double
SYMBOL_LASTHIGH	Maximales Last pro Tag	double
SYMBOL_LASTLOW	Minimales Last pro Tag	double
SYMBOL_VOLUME_REAL	Volume - Volumen im letzten Deal	double
SYMBOL_VOLUMEHIGH_REAL	Maximales Volumen pro Tag	double
SYMBOL_VOLUMELOW_REAL	Minimales Volumen pro Tag	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_OPTION_STRIKE	The strike price of an option. The price at which an option buyer can buy (in a Call option) or sell (in a Put option) the underlying asset, and the option seller is obliged to sell or buy the appropriate amount of the underlying asset.	double
SYMBOL_POINT	Wert eines Punktes	double
SYMBOL_TRADE_TICK_VALUE	Wert SYMBOL_TRADE_TIC	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	K_VALUE_PROFIT	
SYMBOL_TRADE_TICK_VALUE_PROFIT	Berechnete Kosten des Ticks für wirtschaftliche Position	double
SYMBOL_TRADE_TICK_VALUE_LOSS	Berechnete Kosten des Ticks für unwirtschaftliche Position	double
SYMBOL_TRADE_TICK_SIZE	Minimale Preisveränderung	double
SYMBOL_TRADE_CONTRACT_SIZE	Größe des Handelsvertrags	double
SYMBOL_TRADE_ACCRUED_INTEREST	Stückzins - Zinsbetrag, der proportional der Anzahl der Tage	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	berechnet wird, die seit dem Ausstellungsdatum der Anleihe oder seit dem letzten Zinstermin vergangen sind	
SYMBOL_TRADE_FACE_VALUE	Nennwert - der ursprüngliche Betrag einer Anleihe, der vom Emittent festgelegt wurde	double
SYMBOL_TRADE_LIQUIDITY_RATE	Liquiditätskoeffizient - der Anteil eines Vermögenswertes, den man als Marge	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	verwenden kann.	
SYMBOL_VOLUME_MIN	Minimale Volumen für Dealsabschluss	double
SYMBOL_VOLUME_MAX	Maximales Volumen für Dealsabschluss	double
SYMBOL_VOLUME_STEP	Minimaler Schritt der Volumenveränderung für Dealsabschluss	double
SYMBOL_VOLUME_LIMIT	Die maximale zulässige gesamte Volumen von einer offenen Position und schwebende Ordern in	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	einer Richtung (Kauf oder Verkauf) für das Symbol. Zum Beispiel mit der Begrenzung von 5 Lots, können Sie eine offene Kaufposition mit dem Volumen von 5 Lots haben und eine schwebende Order Sell Limit mit dem Volumen von 5 Lots stellen. Aber in diesem Fall können Sie nicht Buy	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Limit (seit dem Gesamtvolumen in eine Richtung wird die Begrenzung überschreiten) oder Sell Limit mit dem Volumen von mehr als 5 Lots stellen.	
SYMBOL_SWAP_LONG	Swapwert beim Kauf	double
SYMBOL_SWAP_SHORT	Swapwert beim Verkauf	double
SYMBOL_SWAP_SUNDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHO	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	RT) für Übernachtpositionen, die von SONNTAG auf den nächsten Tag gerollt werden. Die folgenden Werte werden unterstützt: • 0 - kein Swap wird berechnet • 1 - einf	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	achter Swap • 3-dreifacher Swap	
SYMBOL_SWAP_MONDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHORT) für Übernachtpositionen, die von Montag auf Dienstag gerollt	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	werden.	
SYMBOL_SWAP_TUESDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHORT) für Übernachtpositionen, die von Dienstag auf Mittwoch gerollt. werden.	double
SYMBOL_SWAP_WEDNESDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHORT) für Übernachtpositionen, die von Mittwoch auf Donnerstag	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	gerollt werden.	
SYMBOL_SWAP_THURSDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHORT) für Übernachtpositionen, die von Donnerstag auf Freitag gerollt werden.	double
SYMBOL_SWAP_FRIDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHORT) für Übernachtpositionen, die von Freitag auf Samsta	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	g gerollt werden.	
SYMBOL_SWAP_SATURDAY	Swap-Berechnungsquote (SYMBOL_SWAP_LONG oder SYMBOL_SWAP_SHORT) für Übernachtpositionen, die von Samstag auf Sonntag gerollt werden.	double
SYMBOL_MARGIN_INITIAL	Initiale Marge bezeichnet die Größe der erforderlichen Kautionssumme in Marge-Währung für Eröffnung einer Position im	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Volumen von einem Lot. Wird verwendet bei der Überprüfung von Geld eines des Kunden beim Eintritt in den Markt. Die Funktion bietet Informationen über die Höhe der erhobenen Marge, je nach Art und Richtung des Auftrags SymbolInfoMarginRate().	
SYMBOL_MARGIN_MAINTENANCE	Maintenance-Marge. Wenn sie	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	eingegeben ist, setzt sie die Marge in der Marge-Währung des Symbols, die von einem Lot abgezogen wird. Wird verwendet bei der Überprüfung von Geld eines des Kunden, wenn sein/ihr Kontozustand ändert. Wenn die Maintenance-Marge gleich 0 ist, wird die Initial Marge verwendet.	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Die Funktion bietet Informationen über die Höhe der erhobenen Marge, je nach Art und Richtung des Auftrags SymbolInfoMarginRate().	
SYMBOL_SESSION_VOLUME	Das Gesamtvolumen der Deals in der aktuellen Session	double
SYMBOL_SESSION_TURNOVER	Der Gesamtumsatz in der aktuellen Session	double
SYMBOL_SESSION_INTEREST	Der Gesamtumsatz der offenen	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	Positionen	
SYMBOL_SESSION_BUY_ORDERS_VOLUME	Das Gesamtvolumen der Buy-Ordern im Moment	double
SYMBOL_SESSION_SELL_ORDERS_VOLUME	Das Gesamtvolumen der Sell-Ordern im Moment	double
SYMBOL_SESSION_OPEN	Der Eröffnungskurs der Session	double
SYMBOL_SESSION_CLOSE	Der Schlusskurs der Session	double
SYMBOL_SESSION_AW	Der durchschnittliche Preis für die Session	double
SYMBOL_SESSION_PRICE_SETTLEMENT	Abrechnungspreis der aktuellen Session	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_SESSION_PRICE_LIMIT_MIN	Der minimale Preiswert für die Session	double
SYMBOL_SESSION_PRICE_LIMIT_MAX	Der maximale Preiswert für die Session	double
SYMBOL_MARGIN_HEDGED	Die Größe des Kontrakts oder der Marge für einen Lot gehedgter Positionen (gegensätzliche Positionen auf einem Symbol). Es gibt zwei Methoden zur Berechnung der Marge für gehedg	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	te Positionen. Die Berechnungsmethode wird vom Broker bestimmt. Basisberechnung: • Wenn die Anfangsmarge (SYMBOL_MARGIN_INITIAL) für das Symbol festgelegt ist, wird die abgesicherte Marge als absoluter Wert (in Geld) ange	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	gebe n. • Wen n die Anfa ngsm arge nicht festg elegt wurd e (= 0), dann wird die Größ e des Kontr akts in SYMB OL_ MAR GIN_ HED GED ange gebe n, die bei der Bere chnu ng der Marg e nach einer Form el je nach Symb ol (SYM BOL_	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	TRADE_CALC_MODE) verwendet wird. Berechnung nach der größeren Seite: • Der Wert SYMBOL_MARGIN_HEDGED wird nicht berechnet. • Das Volumen aller Short- und Long-Positionen auf dem Symbol wird berechnet.	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	• Für jede Seite wird ein gewichteter Eröffnungskurs und ein gewichteter Umrechnungskurs für die Währung des Depots berechnet. • Die Marge für die kleinere und größere Seite wird nach Formeln je nach Symboltyp	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	(SYMBOL_TRADE_CALC_MODE) berechnet. • Als Endwert der Marge gilt der größere Wert.	
SYMBOL_PRICE_CHANGE	Veränderung des aktuellen Kurses im Vergleich zum Ende des vorangegangenen Handelstages in %.	double
SYMBOL_PRICE_VOLATILITY	Preisvolatilität in %	double
SYMBOL_PRICE_THEORETICAL	Theoretischer Optionspreis	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_PRICE_DELTA	Das Delta von Optionen/Warrants zeigt den Wert, um den sich der Optionspreis ändert, wenn sich der Preis des zugrunde liegenden Vermögenswerts um 1 ändert.	double
SYMBOL_PRICE_THETA	Das Theta von Optionen/Warrants zeigt die Anzahl der Punkte, die der Optionspreis jeden Tag aufgrund einer	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	vorübergehenden Liquidation, d.h. bei Annäherung an das Verfallsdatum, verliert.	
SYMBOL_PRICE_GAMMA	Das Gamma von Optionen/Warrants zeigt die Änderungsrate des Deltas - wie schnell oder langsam sich die Optionsprämie ändert.	double
SYMBOL_PRICE_VEGA	Das Vega von Optionen/Warrants zeigt die Anzahl der Punkte	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	an, um die sich der Optionspreis bei einer Änderung der Volatilität um 1 % ändert.	
SYMBOL_PRICE_RHO	Das Rho von Optionen/Warrants spiegelt die Empfindlichkeit des theoretischen Optionspreises gegenüber einer Zinsänderung von 1% wider.	double
SYMBOL_PRICE_OMEGA	Das Omega von Optionen/Warrants. Die Optionselastizität	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	zeigt eine relative prozentuale Änderung des Optionspreises durch die prozentuale Änderung des Basiswertpreises.	
SYMBOL_PRICE_SENSITIVITY	Die Sensitivität von Optionen/Warrants zeigt an, um wie viele Punkte sich der Preis des Basiswertes der Option ändern sollte, so dass sich der Preis der Option um	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	einen Punkt ändert.	

für die Funktion SymbolInfoString()

ENUM_SYMBOL_INFO_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
SYMBOL_BASIS	Basiswert für ein Derivat	string
SYMBOL_CATEGORY	Der Name der Branche oder Kategorie, zu der das Handelssymbol gehört	string
SYMBOL_COUNTRY	Das Land, zu dem das Symbol gehört.	string
SYMBOL_SECTOR_NAME	Der Wirtschaftssektor, zu dem das Finanzsymbol gehört.	string
SYMBOL_INDUSTRY_NAME	Der Wirtschaftszweig oder Industrie, zu dem das Finanzsy	string

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	mbol gehört	
SYMBOL_CURRENCY_BASE	Basiswährung eines Symbols	string
SYMBOL_CURRENCY_PROFIT	Währung des Gewinns	string
SYMBOL_CURRENCY_MARGIN	Währung der Marge	string
SYMBOL_BANK	Herkunft der laufenden Kurse	string
SYMBOL_DESCRIPTION	Beschreibung des Symbols	string
SYMBOL_EXCHANGE	Der Name der Börse, an der das Finanzsymbol gehandelt wird	string
SYMBOL_FORMULA	Formel für das Erstellen eines benutzerdefinierten Symbols. Wenn der Name eines in der Formel verwendeten Finanzsymbols	string

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	mit einer Ziffer beginnt oder ein Sonderzeichen enthält (">" "...", "-", "&", "#" u.s.w.), sollten um diese Symbolnamen Anführungszeichen verwendet werden. • Synthetisches Symbol: "@ESU19"/	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	EURCAD • Kalenderangaben: "Si-9.13"-"Si-6.13" • Euro-In	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	dex:34.388057266*pow(EURUSD,0.31555)*pow(EURGBP,0.	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	3056)*pow(EURJPY,0.18911)*pow(EURCHF,0.11113)*pow(	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	EURSEK, 0.07855)	
SYMBOL_ISIN	Der Name eines Symbols im System ISIN (International Securities Identification Number). International Securities Identification Number ist ein 12-stelliger alphanumerischer Code zur eindeutigen Identifizierung eines Wertpapi	string

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	ers. Das Vorhandensein dieser Einstellung des Symbols wird auf der Seite eines Handelsservers ermittelt.	
SYMBOL_PAGE	Adresse der Webseite mit Informationen zum Symbol. Diese Adresse wird als Link bei Ansicht der Eigenschaften des Symbols im Terminal angezeigt	string
SYMBOL_PATH	Pfad im Symbolbaum	string

Der Preischart kann basierend auf Bid oder Last Preisen gezeichnet werden. Von der Auswahl des Preistyps hängt ab, wie die Balken im Terminal gebildet und angezeigt werden. Die möglichen Werte der Eigenschaft SYMBOL_CHART_MODE sind in der Aufzählung ENUM_SYMBOL_CHART_MODE angeführt

ENUM_SYMBOL_CHART_MODE

Bezeichnung	Beschreibung
SYMBOL_CHART_MODE_BID	Die Balken werden basierend auf Bid-Preisen gezeichnet

© 2000-2025, MetaQuotes Ltd.

Bezeichnung	Beschreibung
SYMBOL_CHART_MODE_LAST	Die Balken werden basierend auf Last-Preisen gezeichnet

für jedes finanzielles Instrument können mehrere Modi der Gültikeitsfrist (Ablauffrist) der wartenden Ordern angegeben werden. Für jeden Modus gibt es eine Flagge, Flaggen können miteinander durch die Operation des logischen **ODER** (|) kombiniert werden, ZB, SYMBOL_EXPIRATION_GTC| SYMBOL_EXPIRATION_SPECIFIED. Für Prüfung der Zulässigkeit des bestimmten Modus für Instrument muss man das Ergebnis der logischen **UND** (&) mit der Flagge des Modus vergleichen.

Wenn für Symbol die Flagge SYMBOL_EXPIRATION_SPECIFIED angegeben wird, kann beim Senden der wartenden Order ihre Gültigkeitsfrist explizit angegeben werden.

Identifikator	Wert	Beschreibung
SYMBOL_EXPIRATION_GTC	1	Die Order ist zeitlich uneingeschränkt gültig, bis zu ihrer expliziten Annullierung
SYMBOL_EXPIRATION_DAY	2	Die Order ist bis zum Tagesende gültig
SYMBOL_EXPIRATION_SPECIFIED	4	Ablauffrist wird in der Order angegeben werden
SYMBOL_EXPIRATION_SPECIFIED_DAY	8	Ablaufdatum wird in der Order angegeben werden

Beispiel:

```
//+------------------------------------------------------------------+
//| prüft Zulässigkeit des angegebenen Modus der Expiration          |
//+------------------------------------------------------------------+
bool IsExpirationTypeAllowed(string symbol,int exp_type)
  {
//--- erhalten wir Wert der Eigenschaft, die zulässige Modi der Ablauffrist beschreibt
   int expiration=(int)SymbolInfoInteger(symbol,SYMBOL_EXPIRATION_MODE);
//--- geben wir true zurück, wenn Modus exp_type zulässig ist
   return((expiration & exp_type)==exp_type);
  }
```

Wenn die Eigenschaft SYMBOL_EXPIRATION_MODE den Wert SYMBOL_EXPIRATION_GTC (die Order ist bis zum Löschen gültig) hat, wird die Zeit der Gültigkeit von Pending Orders und von StopLoss/TakeProfit Levels extra mithilfe der Aufzählung ENUM_SYMBOL_ORDER_GTC_MODE gesetzt.

ENUM_SYMBOL_ORDER_GTC_MODE

© 2000-2025, MetaQuotes Ltd.

Bezeichnung	Beschreibung
SYMBOL_ORDERS_GTC	Die Pending Orders und Stop Loss/Take Profit Levels sind unbegrenzt gültig, bis sie explizit abgebrochen werden
SYMBOL_ORDERS_DAILY	Die Orders sind nur innerhalb eines Handelstages gültig. An Ende des Tages werden alle StopLoss und TakeProfit Orders gelöscht, Pending Orders werden ebenfalls gelöscht
SYMBOL_ORDERS_DAILY_EXCLUDING_STOPS	Bei Wechsel eines Handelstages werden nur Pending Ordres gelöscht, StopLoss und TakeProfit Level bleben erhalten

Beim Senden eines Auftrags kann der Fülltyp des angegebenen Volumens in dem Auftrag angeben werden. Die möglichen volumenbasierten Ausführungsoptionen des Auftrags für jedes Symbol sind in der Tabelle angegeben. Es ist möglich, mehrere Modi für jedes Instrument über eine Kombination von Flags zu setzen. Die Kombination von Flags wird durch die Operation eines logischen **OR** ausgedrückt, zum Beispiel SYMBOL_FILLING_FOK|SYMBOL_FILLING_IOC. Um zu prüfen, ob ein bestimmter Modus für ein Handelsinstrument erlaubt ist, vergleichen Sie das Ergebnis des logischen **AND** (&) mit dem Modus-Flag - Beispiel.

Füllrichtlinie	ID	Wert	Beschreibung
Alles oder Nichts (Fill or Kill, FOK)	SYMBOL_FILLING_FOK	1	Ein Auftrag wird n

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			urmitdemangegebenenVolumenausgeführt.

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			Wenn die angeforderte Volumenein Finanzi

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			nstrumentsderzeitnichtaufdemMarktverfü

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			gbar ist, wird der Auftrag nicht ausgeführt.

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			Das gewünschte Volumen kann nicht ausmehrere

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			nverfügbaren Angeboten zusammensetzen.

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			Beim Senden einer Auftragsanfrage

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ülltyp ORDER_FILLING_FOK angegebenwerde

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			n. Die Möglichkeit der Verwendung von FOK wi

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			rddurchdenHandelsserverfestgelegt.
Jetzt oder Nie (Immediate or Cancel, IOC)	SYMBOL_FILLING_IOC	2	EinH

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ändlerakzeptiert, denAuftragmitdemmaxim

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			alam Marktverfügbaren Volumen in innerhalb der

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			sind em Auftrag angegeben en Volumens auszuf

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ühren. Wenn die Anfrage nicht vollständig ist

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			gesetzt werden kann, wird ein Auftrag mit dem

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			verfügbaren Volumen ausgeführt und das verb

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			leibendeVolumenwirdannulliert. BeimSen

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			denen ein es Auftrags soll ted af ür der Füll typ O

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			RDER_FILLING_IOC angegeben werden. Die Mö

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			gleichkeitderVerwendungvonIOC-Ordersw ir

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			dauf demHandelsserverfestgelegt.
Passiv	SYMBOL_FILLING_BOC	4	DieBOC-

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			Richtlinie (Book-or-Cancel) gehtdavonaus

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			, dass ein Auftrag nur in der Markttiefe platz

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ierte und nicht sofort ausgeführt werden kann

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			.Wenn der Auftrag bei der Platzierung sofort

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ausgeführt werden könnte, dann wird der erste

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			nie rt wer den . Die se Aus füh rungs rich tli ni i

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ekannnurdannangegebenenwerden,wennderP

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			reisdeserteiltenAuftragsschlechterista

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			Ist der aktuelle Markt. BOC-Aufträge werden z

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			urUmsetzungdespassivenHandelsverwendet

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			, damit der Auftrag nicht sofort bei der Platz

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ierungausgeführtunddieaktuelleLiquidit

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ätnichtverändertwird. NurLimit-undStop

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			- Limit-Orders werden nur unterstützt, d. h. das

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			Flag SYMBOL_ORDER_MODE soll die Werte SYM

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			BOL_ORDER_LIMIT und/oder SYMBOL_ORDER_ST

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			OP_LIMIT enthalten.
Rückkehren	Kein Identifikator		Im Fall eine Verteilweis

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			enAusführung (Füllung) wird eine Market-ode

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			rLimit-OrdermitRestvolumennichtstornie

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			rt, sonderrnweiterverarbeitet. Beim Sende

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			neines Auftrags sollte da für der Füllty p ORD

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			ER_FILLING_RETURN angegeben werden. Dies

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			eFormderRückkehrnachheinemgesendetenAuf

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			trägt sind im Modus der Marktausführung (Mark

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			tausführung-SYMBOL_TRADE_EXECUTION_MAR

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	ID	Wert	Beschreibung
			KET) nicht zulässig.

Beim Senden einer Handelsanfrage mit der Funktion OrderSend() kann im Feld *type_filling* die notwendige Volumenausführungspolitik eingestellt werden, und zwar in der speziellen Struktur MqlTradeRequest.Es stehen die Werte aus der Enumeration ENUM_ORDER_TYPE_FILLING zur Verfügung. Wenn kein Fülltyp angegeben wird, wird ORDER_FILLING_RETURN automatisch in der Handelsanfrage gesetzt. Der Fülltyp ORDER_FILLING_RETURN ist in jedem Ausführungsmodus außer bei der "Marktausführung" (SYMBOL_TRADE_EXECUTION_MARKET) aktiviert.

Wenn eine Handelsanfrage zur Ausführung **zur aktuellen Zeit** (time in force) gesendet wird, sollte nicht vergessen werden, dass Finanzmärkte keine Garantie dafür bieten, dass das gesamte angeforderte Volumen für ein bestimmtes Finanzinstrument zum gewünschten Preis verfügbar ist. Deshalb werden die Handelsoperationen in Echtzeit durch die Modi der Preis- und Volumenausführung geregelt. Die Modi, oder Ausführungsrichtlinien, definieren die Regeln für den Fall, dass der Preis sich geändert hat oder das angeforderte Volumen **momentan** nicht vollständig umgesetzt werden kann.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
Ausführungsmodus (Anfrage zur Ausführung)	Ausführung einer	SYMBOL_TRADE_EXECUTION_REQUEST

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	Markt-Order zu dem zuvor vom Broker erhaltenen Preis. Die Preise für eine bestimmte Markt-Order werden beim Broker angefordert, bevor die Order gesendet wird. Nach Erhalt der Preise kann die Auftragsausführung zu dem angegebenen Preis entweder bestätigt oder abgelehnt	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	werden.	
Sofortige Ausführung (Sofortige Ausführung)	Sofortige Ausführung einer Markt-Order zu dem angegebenen Preis. Wenn eine Handelsanfrage zur Ausführung gesendet wird, fügt die Plattform automatisch die aktuellen Preise dem Auftrag hinzu. • Wenn der Bro	SYMBOL_TRADE_EXECUTION_INSTANT

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	kein Preis akzeptiert, wird der Auftrag ausgeführt.	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	• Wenn der Broker den angefragten Preis nicht akzeptiert,	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	wird ein "Requote" gesendet – der Broker gibt Preise zurü	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	ck, zu dem ein der Auftrag ausgeführt werden kann.	
Marktausführung (Marktausführung)	Der Broker trifft	SYMBOL_TRADE_EXECUTION_MARKET

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	eine Entscheidung über den Ausführungspreis des Auftrags ohne zusätzliche Diskussion mit dem Händler. Das Absenden des Auftrags mit einem solchen Modus schließt die vorherige Zustimmung zu ihrer Ausführung zu diesem Preis ein.	
Börsliche Ausführung (Börsliche Ausführung)	Handelsgeschäfte werden zu den aktuellen Preisen des	SYMBOL_TRADE_EXECUTION_EXCHANGE

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	Marktes ausgeführt.	

Vor dem Senden eines Auftrags mit der aktuellen Ausführungszeit, für die korrekte Einstellung des Wertes von ORDER_TYPE_FILLING (Volumenausführungsart), können Sie die Funktion SymbolInfoInteger() mit jedem Finanzinstrument verwenden, um den Eigenschaftswert von SYMBOL_FILLING_MODE zu erhalten, der die Volumenausführungsarten, die für das Symbol erlaubt sind, als eine Kombination von Flags anzeigt. Der Fülltyp ORDER_FILLING_RETURN ist immer aktiviert, außer im Modus "Marktausführung" (SYMBOL_TRADE_EXECUTION_MARKET).

Die Verwendung der Fülltypen in Abhängigkeit vom Ausführungsmodus ist in der folgenden Tabelle dargestellt:

Ausführungsart	Alles oder Nichts (FOK, ORDER_FILLING_FOK)	Jetzt oder Nie (IOC, IOC ORDER_FILLING_IOC)	Rückkehren (Return, ORDER_FILLING_RETURN)
Sofortige Ausführung (SYMBOL_TRADE_EXECUTION_INSTANT)	+ (unabhängig von der Einstellung eines Symbols)	+ (unabhängig von der Einstellung eines Symbols)	+ (immer)
Ausführungsanfrage SYMBOL_TRADE_EXECUTION_REQUEST	+ (unabhängig von der Einstellung eines Symbols)	+ (unabhängig von der Einstellung eines Symbols)	+ (immer)
Marktausführung SYMBOL_TRADE_EXECUTION_MARKET	+ (definiert den Symboleinstellungen)	+ (definiert den Symboleinstellungen)	- (deaktiviert, unabhängig von den Symboleinstellungen)
Börsliche Ausführung SYMBOL_TRADE_EXECUTION_EXCHANGE	+ (definiert den Symboleinstellungen)	+ (definiert den Symboleinstellungen)	+ (immer)

Bei Pending-Orders (schwebenden Aufträgen) sollte unabhängig von der Ausführungsart (SYMBOL_TRADE_EXEMODE) die Ausführungsart ORDER_FILLING_RETURN verwendet werden, da solche Aufträge zum Zeitpunkt des Absendens nicht zur Ausführung vorgesehen sind. Bei der Verwendung von Pending-Orders (schwebenden Aufträgen) akzeptiert ein Händler im Voraus, dass der Broker, wenn die Bedingungen für ein Geschäft mit diesem Auftrag erfüllt sind, die von der Börse unterstützte Ausführungsart verwendet.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Prüfen, ob der angegebene Fülltyp erlaubt ist                    |
//+------------------------------------------------------------------+
bool IsFillingTypeAllowed(string symbol,int fill_type)
  {
//--- liefert den Wert der Eigenschaft, die den Fülltyp beschreibt
   int filling=(int)SymbolInfoInteger(symbol,SYMBOL_FILLING_MODE);
//--- Rückgabe von 'true', wenn der Fülltyp erlaubt ist
   return((filling&fill_type)==fill_type);
  }
```

Wenn die Funktion OrderSend() Handelsanfrage sendet, müssen Sie die Art der Bestellung für einige Operationen von ENUM_ORDER_TYPE eingeben. Nicht alle Arten von Aufträgen können für ein bestimmtes Finanzinstrument erlaubt sein, das Eigenschaft SYMBOL_ORDER_MODE beschreibt die Flags der zulässigen Arten von Aufträgen.

Bezeichnung	Wert	Beschreibung
SYMBOL_ORDER_MARKET	1	Markt-Orders sind erlaubt (Buy und Sell)
SYMBOL_ORDER_LIMIT	2	Limit-Orders sind erlaubt (Buy Limit und Sell Limit)
SYMBOL_ORDER_STOP	4	Stop-Orders sind erlaubt (Buy Stop und Sell Stop)
SYMBOL_ORDER_STOP_LIMIT	8	Stop-Limit-Orders sind erlaubt (Buy Stop Limit und Sell Stop Limit)
SYMBOL_ORDER_SL	16	Stop Loss ist erlaubt
SYMBOL_ORDER_TP	32	Take Profit ist erlaubt
SYMBOL_ORDER_CLOSEBY	64	Die Erlaubnis, eine Position durch eine Gegenposition, d.h. durch eine in die entgegengesetzte Richtung eröffnete Position auf demselben Symbol, zu schließen. Die Eigenschaft wird für Konten mit dem Hedging-Verrechnungssystem (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) gesetzt

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Die Funktion druckt Ordertypen, die für den Symbol erlaubt sind  |
//+------------------------------------------------------------------+
void Check_SYMBOL_ORDER_MODE(string symbol)
  {
//--- das Wert der Eigenschaft, die de erlaubten Ordertypen beschreibt
   int symbol_order_mode=(int)SymbolInfoInteger(symbol,SYMBOL_ORDER_MODE);
//--- Prüfung von Markt-Orders (Market Execution)
   if((SYMBOL_ORDER_MARKET&symbol_order_mode)==SYMBOL_ORDER_MARKET)
      Print(symbol+": Markt-Orders sind erlaubt (Buy und Sell)");
//--- Prüfung von Limit-Orders
   if((SYMBOL_ORDER_LIMIT&symbol_order_mode)==SYMBOL_ORDER_LIMIT)
      Print(symbol+": Orders Buy Limit und Sell Limit sind erlaubt");
//--- Prüfung von Stop-Orders
   if((SYMBOL_ORDER_STOP&symbol_order_mode)==SYMBOL_ORDER_STOP)
      Print(symbol+": Orders Buy Stop und Sell Stop sind erlaubt");
//--- Prüfung von Stop Limit Orders
   if((SYMBOL_ORDER_STOP_LIMIT&symbol_order_mode)==SYMBOL_ORDER_STOP_LIMIT)
      Print(symbol+": Orders Buy Stop Limit und Sell Stop Limit sind erlaubt");
//--- Prüfung ob Stop Loss Orders erlaubt sind
   if((SYMBOL_ORDER_SL&symbol_order_mode)==SYMBOL_ORDER_SL)
      Print(symbol+": Orders Stop Loss sind erlaubt");
//--- Prüfung ob Take Profit Orders erlaubt sind
   if((SYMBOL_ORDER_TP&symbol_order_mode)==SYMBOL_ORDER_TP)
      Print(symbol+": Orders Take Profit sind erlaubt");
//--- Prüfung, ob das Schließen einer Position durch eine Gegenposition erlaubt ist
   if((SYMBOL_ORDER_TP&symbol_order_mode)==SYMBOL_ORDER_CLOSEBY)
      Print(symbol+": Schließen durch Gegenposition ist erlaubt"
//---
  }
```

Für die Erhaltung der Information über Methode der Berechnung der Pfandgeldwerte für das Instrument (Größe der Margeanforderungen) ist die Enumeration ENUM_SYMBOL_CALC_MODE bestimmt.

ENUM_SYMBOL_CALC_MODE

Identifikator	Beschreibung	Formel
SYMBOL_CALC_MODE_FOREX	Forex mode - Berechnung von Gewinn und Marge für	Margin: Lots * Contract_Size / Leverage * Margin_Rate Profit: (close_price - open_price) * Contract_Size*Lots

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	Forex	
SYMBOL_CALC_MODE_FOREX_NO_LEVERAGE	Forex No Leverage mode - Berechnung des Gewinns und der Margin für Forex-Symbole ohne Berücksichtigung des Hebels	Margin: Lots * Contract_Size * Margin_Rate Profit: (close_price - open_price) * Contract_Size * Lots
SYMBOL_CALC_MODE_FUTURES	Futures mode - Berechnung von Marge und Gewinn für	Margin: Lots * InitialMargin * Margin_Rate Profit: (close_price - open_price) * TickPrice / TickSize*Lots

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	futures	
SYMBOL_CALC_MODE_CFD	CFD mode - Berechnung von Marge und Gewinn für CFD	Margin: Lots * ContractSize * MarketPrice * Margin_Rate Profit: (close_price - open_price) * Contract_Size * Lots
SYMBOL_CALC_MODE_CFDINDEX	CFD index mode - Berechnung von Marge und Gewinn für CFD durch Indexe	Margin: (Lots * ContractSize * MarketPrice) * TickPrice / TickSize * Margin_Rate Profit: (close_price - open_price) * Contract_Size * Lots
SYMBOL_CALC_MODE_CFDLEVERAGE	CFD Leverage mode - Berechnung von	Margin: (Lots * ContractSize * MarketPrice) / Leverage * Margin_Rate Profit: (close_price-open_price) * Contract_Size * Lots

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	Marge und Gewinn für CFD beim Leverage-Handel	
SYMBOL_CALC_MODE_EXCH_STOCKS	Exchange mode - Berechnung von Marge und Gewinn für den Handel von Wertpapieren an der Börse	Margin: Lots * ContractSize * LastPrice * Margin_Rate Profit: (close_price - open_price) * Contract_Size * Lots
SYMBOL_CALC_MODE_EXCH_FUTURES	Futures mode - Berechnung von Marg	Margin: Lots * InitialMargin * Margin_Rate oder Lots * MaintenanceMargin * Margin_Rate Profit: (close_price - open_price) * Lots * TickPrice / TickSize

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	e und Gewinn für den Handel von Futures-Kontrakte an der Börse	
SYMBOL_CALC_MODE_EXCH_FUTURES_FORTS	FORTS Futures mode - Berechnung von Marge und Gewinn für den Handel von Futures-Kontrakte an FORTS. Marge kann	Margin: Lots * InitialMargin * Margin_Rate oder Lots * MaintenanceMargin * Margin_Rate Profit: (close_price - open_price) * Lots * TickPrice / TickSize

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	auf die Menge der Abweichung MarginDiscount wie folgt reduziert werden: 1. Wenn der Preis für eine lange Position (Kauforder) weniger als der geschätzten Preis ist, MarginDiscount = Lots*((PriceSettle-PriceOrde	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	r) *TickPrice/TickSize) 2. Wenn der Preis für eine kurze Position (Verkaufsorder) weniger als der geschätzten Preis ist, dann MarginDiscount = Lots*((PriceOrder-PriceSettle)*TickPrice/TickSize) wo:	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	☐ PriceSettle-Abrechnungspreis (Clearing) vonderVor	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	herigen Session; □ Price Order-gewichtet durchschnit	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	tlichePreisderPositionoderderErröffnungspreis,de	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	rimOrder(Anfrage) angegeben ist; □TickPrice-Preis vo	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	nTick (Kostenvon Preisänderungendurcheinen Punkt)	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	☐TickSize-die Tick-Größe (minimaler Schritt der Preis	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	änderung)	
SYMBOL_CALC_MODE_EXCH_BONDS	Modus für Anleihen der Börse - Berechnung von Marge und Gewinn für den Handel von Anleihen an einer Börse.	Margin: Lots * ContractSize * FaceValue * open_price * /100 Profit: Lots * close_price * FaceValue * Contract_Size + AccruedInterest * Lots * ContractSize
SYMBOL_CALC_MODE_EXCH_STOCKS_MOEX	Modus für Aktien der Börse MOEX - Berechnu	Margin: Lots * ContractSize * LastPrice * Margin_Rate Profit: (close_price - open_price) * Contract_Size * Lots

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	ng von Marge und Gewinn für den Handel von Aktien an der Börse MOEX.	
SYMBOL_CALC_MODE_EXCH_BONDS_MOEX	Modus für Anleihen der Börse MOEX - Berechnung von Marge und Gewinn für den Handel von Anleihen an der Börs	Margin: Lots * ContractSize * FaceValue * open_price * /100 Profit: Lots * close_price * FaceValue * Contract_Size + AccruedInterest * Lots * ContractSize

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	e MOEX.	
SYMBOL_CALC_MODE_SERV_COLLATERAL	Collateral Modus - das Symbol wird als nicht-handelbares Vermögen auf einem Handelskonto verwendet. Der Marktwert der offenen Position ist aufgrund von Volumen, aktuellem Marktpreis,	Margin: kein Profit: kein Marktwert: Lots * ContractSize * MarketPrice * LiquidityRate

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	Kontraktgröße und Liquiditätskennzahl berechnet. Der Wert ist im Vermögen (Assets) aufgezeichnet, das zum Eigenkapital (Equity) summiert wird. So erhöhen die offenen Positionen auf ein solches Instrume	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Formel
	nt die Größe der verfügbaren Mittel (Free Margin) und dien en als Sicherheit für offene Positionen anderen gehandelten Instrumente	

Es gibt mehrere Handelsmodi für finanzielle Instrumente. Information über Handelsmodi für bestimmte Instrumente wird in Enumerationswerten ENUM_SYMBOL_TRADE_MODE dargestellt.

ENUM_SYMBOL_TRADE_MODE

Identifikator	Beschreibung
SYMBOL_TRADE_MODE_DISABLED	Symbolhandel ist verboten
SYMBOL_TRADE_MODE_LONGONLY	Erlaubt werden nur Käufe
SYMBOL_TRADE_MODE_SHORTONLY	Erlaubt werden nur Verkäufe

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
SYMBOL_TRADE_MODE_CLOSEONLY	Erlaubt werden die Operationen der Schliessung von Positionen
SYMBOL_TRADE_MODE_FULL	Keine Einschrankungen für Handelsoperationen

Mögliche Modi der Dealsabschlüsse für bestimmte Instrumente sind in der Enumeration ENUM_SYMBOL_TRADE_EXECUTION bestimmt.

ENUM_SYMBOL_TRADE_EXECUTION

Identifikator	Beschreibung
SYMBOL_TRADE_EXECUTION_REQUEST	Handel auf Anfrage
SYMBOL_TRADE_EXECUTION_INSTANT	Handel mit laufenden Preisen
SYMBOL_TRADE_EXECUTION_MARKET	Orderdurchführung auf dem Markt
SYMBOL_TRADE_EXECUTION_EXCHANGE	Börslicher Ausführung

Methoden der Swapsanrechnung bei Übertragung der Position werden in der Enumeration ENUM_SYMBOL_SWAP_MODE angegeben. Die Methode der Swapsanrechnung bestimmt die Maßeinheiten der Parameter SYMBOL_SWAP_LONG und SYMBOL_SWAP_SHORT. Zum Beispiel, wenn die Swaps in der Einzahlung-Währung von Kunden angerechnet werden, wird in den Parametern das Volumen des angerechneten Swaps gerade in der Einzahlung-Währung des Kunden angegeben.

ENUM_SYMBOL_SWAP_MODE

Identifikator	Beschreibung
SYMBOL_SWAP_MODE_DISABLED	Keine Swaps
SYMBOL_SWAP_MODE_POINTS	Swaps werden in Punkten angerechnet
SYMBOL_SWAP_MODE_CURRENCY_SYMBOL	Swaps werden in Geld in der Basiswährung des Symbols angerechnet
SYMBOL_SWAP_MODE_CURRENCY_MARGIN	Swaps werden in Geld in der Margin-Währung des Symbols angerechnet.
SYMBOL_SWAP_MODE_CURRENCY_DEPOSIT	Swaps werden in Geld in der Einzahlung-Währung des Kunden angerechnet
SYMBOL_SWAP_MODE_CURRENCY_PROFIT	Swaps werden in Geldwert der Währung für die Gewinnermittlung berechnet
SYMBOL_SWAP_MODE_INTEREST_CURRENT	Swaps werden in den jährlichen Zinsen vom Instrumentpreis zum Zeitpunkt der Berechnung des Swaps (Das Bankjahr wird mit 360 Tagen gerechnet) angerechnet

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
SYMBOL_SWAP_MODE_INTEREST_OPEN	Swaps werden in den jährlichen Zinsen vom Preis der Eröffnung der Position nach Symbol (Das Bankjahr wird mit 360 Tagen gerechnet) angerechnet
SYMBOL_SWAP_MODE_REOPEN_CURRENT	Swaps werden durch die Wiedereröffnung der Position angerechnet. Am Ende eines Handelstages wird die Position zwangsmäßig geschlossen. Am nächsten Tag wird die Position nach einem Schlusskurs +/- die angegebene Anzahl von Punkten (in den Parametern SYMBOL_SWAP_LONG und SYMBOL_SWAP_SHORT) wieder geöffnet
SYMBOL_SWAP_MODE_REOPEN_BID	Swaps werden durch die Wiedereröffnung der Position angerechnet. Am Ende eines Handelstages wird die Position zwangsmäßig geschlossen. Am nächsten Tag wird die Position nach dem aktuellen Geldkurs Bid +/- die angegebene Anzahl von Punkten (in den Parametern SYMBOL_SWAP_LONG und SYMBOL_SWAP_SHORT) wieder geöffnet

Für Andeutung des Wochentages sind Werte der Enumeration ENUM_DAY_OF_WEEK bestimmt.

ENUM_DAY_OF_WEEK

Identifikator	Beschreibung
SUNDAY	Sonntag
MONDAY	Montag
TUESDAY	Dienstag
WEDNESDAY	Mittwoch
THURSDAY	Donnerstag
FRIDAY	Freitag
SATURDAY	Samstag

Eine Option bezeichnet ein Recht, nicht jedoch die Pflicht, bestimmte Sache (Ware, Stocks, Futures, etc.) zu einem späteren Zeitpunkt zu einem vereinbarten Preis zu kaufen oder zu verkaufen. Die folgende Enumerationen beschreiben die Optionseigenschaften, wie Optionstyp und das Recht, den sie gewährt.

ENUM_SYMBOL_OPTION_RIGHT

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
SYMBOL_OPTION_RIGHT_CALL	Eine Call-Option bezeichnet ein Recht, eine Vermögenswert zu einem Festpreis zu kaufen
SYMBOL_OPTION_RIGHT_PUT	Eine Put-Option bezeichnet ein Recht, eine Vermögenswert zu einem Festpreis zu verkaufen

ENUM_SYMBOL_OPTION_MODE

Identifikator	Beschreibung
SYMBOL_OPTION_MODE_EUROPEAN	Europäische Option: die Option kann nur am Fälligkeitsdatum ausgeübt werden
SYMBOL_OPTION_MODE_AMERICAN	Amerikanische Option: die Option kann an jedem Handelstag vor der Fälligkeit ausgeübt werden

Die Finanzinstrumente sind nach Wirtschaftssektoren kategorisiert. Ein Wirtschaftssektor ist ein Teil der wirtschaftlichen Tätigkeit, der spezifische Merkmale, wirtschaftliche Ziele, Funktionen und Verhaltensweisen aufweist, die es ermöglichen, diesen Sektor von anderen Teilen der Wirtschaft zu trennen. ENUM_SYMBOL_SECTOR listet die Wirtschaftssektoren auf, zu denen ein Handelsinstrument gehören kann.

ENUM_SYMBOL_SECTOR

ID	Beschreibung
SECTOR_UNDEFINED	Unbestimmt
SECTOR_BASIC_MATERIALS	Grundstoffe
SECTOR_COMMUNICATION_SERVICES	Kommunikationsdienste
SECTOR_CONSUMER_CYCLICAL	Konjunkturabhängige Konsumgüter
SECTOR_CONSUMER_DEFENSIVE	Defensive Konsumgüter
SECTOR_CURRENCY	Währungen
SECTOR_CURRENCY_CRYPTO	Kryptowährungen
SECTOR_ENERGY	Energie
SECTOR_FINANCIAL	Finanzsektor
SECTOR_HEALTHCARE	Gesundheitswesen
SECTOR_INDUSTRIALS	Industriell
SECTOR_REAL_ESTATE	Immobilien
SECTOR_TECHNOLOGY	Technologie
SECTOR_UTILITIES	Versorgungsunternehmen

© 2000-2025, MetaQuotes Ltd.

Jedes Finanzinstrument kann einer bestimmten Art von Industrie oder Wirtschaftszweigen zugeordnet werden. Eine Industrie ist ein Wirtschaftszweig, der eine eng verwandte Menge von Rohstoffen, Waren oder Dienstleistungen produziert. ENUM_SYMBOL_INDUSTRY listet Branchen auf, zu denen ein Handelsinstrument gehören kann.

ENUM_SYMBOL_INDUSTRY

ID	Beschreibung
INDUSTRY_UNDEFINED	Unbestimmt
Grundstoffe	
INDUSTRY_AGRICULTURAL_INPUTS	Landwirtschaftliche Betriebsmittel
INDUSTRY_ALUMINIUM	Aluminium
INDUSTRY_BUILDING_MATERIALS	Baustoffe
INDUSTRY_CHEMICALS	Chemikalien
INDUSTRY_COKING_COAL	Kokskohle
INDUSTRY_COPPER	Kupfer
INDUSTRY_GOLD	Gold
INDUSTRY_LUMBER_WOOD	Schnittholz und Holzproduktion
INDUSTRY_INDUSTRIAL_METALS	Andere Industriemetalle und Bergbau
INDUSTRY_PRECIOUS_METALS	Andere Edelmetalle und Bergbau
INDUSTRY_PAPER	Papier und Papierprodukte
INDUSTRY_SILVER	Silber
INDUSTRY_SPECIALTY_CHEMICALS	Spezialchemikalien
INDUSTRY_STEEL	Stahl
Kommunikationsdienste	
INDUSTRY_ADVERTISING	Werbeagenturen
INDUSTRY_BROADCASTING	Rundfunk
INDUSTRY_GAMING_MULTIMEDIA	Elektronische Spiele und Multimedia
INDUSTRY_ENTERTAINMENT	Unterhaltung
INDUSTRY_INTERNET_CONTENT	Internet-Inhalte und -Informationen
INDUSTRY_PUBLISHING	Verlagswesen
INDUSTRY_TELECOM	Telekommunikationsdienste
Konjunkturabhängige Konsumgüter	

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
INDUSTRY_APPAREL_MANUFACTURING	Bekleidungsherstellung
INDUSTRY_APPAREL_RETAIL	Bekleidungseinzelhandel
INDUSTRY_AUTO_MANUFACTURERS	Autohersteller
INDUSTRY_AUTO_PARTS	Autoteile
INDUSTRY_AUTO_DEALERSHIP	Auto- und Lkw-Händler
INDUSTRY_DEPARTMENT_STORES	Kaufhäuser
INDUSTRY_FOOTWEAR_ACCESSORIES	Schuhe und Zubehör
INDUSTRY_FURNISHINGS	Mobiliar, Einrichtungsgegenstände und Geräte
INDUSTRY_GAMBLING	Glücksspiel
INDUSTRY_HOME_IMPROV_RETAIL	Heimwerkereinzelhandel
INDUSTRY_INTERNET_RETAIL	Interneteinzelhandel
INDUSTRY_LEISURE	Freizeit
INDUSTRY_LODGING	Beherbergung
INDUSTRY_LUXURY_GOODS	Luxusgüter
INDUSTRY_PACKAGING_CONTAINERS	Verpackung und Behälter
INDUSTRY_PERSONAL_SERVICES	Personaldienstleistungen
INDUSTRY_RECREATIONAL_VEHICLES	Freizeit-Fahrzeuge
INDUSTRY_RESIDENT_CONSTRUCTION	Wohnbau
INDUSTRY_RESORTS_CASINOS	Ferienanlagen und Kasinos
INDUSTRY_RESTAURANTS	Restauration
INDUSTRY_SPECIALTY_RETAIL	Spezialisierter Einzelhandel
INDUSTRY_TEXTILE_MANUFACTURING	Textilherstellung
INDUSTRY_TRAVEL_SERVICES	Reisedienstleistungen
Defensive Konsumgüter	
INDUSTRY_BEVERAGES_BREWERS	Getränke - Brauereien
INDUSTRY_BEVERAGES_NON_ALCO	Getränke - alkoholfrei
INDUSTRY_BEVERAGES_WINERIES	Getränke - Weinkellereien und Destillerien
INDUSTRY_CONFECTIONERS	Konditoreien
INDUSTRY_DISCOUNT_STORES	Discountläden
INDUSTRY_EDUCATION_TRAINIG	Dienstleistungen im Bereich Bildung und Ausbildung

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
INDUSTRY_FARM_PRODUCTS	Landwirtschaftliche Produkte
INDUSTRY_FOOD_DISTRIBUTION	Lebensmittelvertrieb
INDUSTRY_GROCERY_STORES	Lebensmittelgeschäfte
INDUSTRY_HOUSEHOLD_PRODUCTS	Haushalt und persönliche Produkte
INDUSTRY_PACKAGED_FOODS	Verpackte Lebensmittel
INDUSTRY_TOBACCO	Tabak
Energie	
INDUSTRY_OIL_GAS_DRILLING	Öl- und Gasbohrung
INDUSTRY_OIL_GAS_EP	Gewinnung und Verarbeitung von Öl und Gas
INDUSTRY_OIL_GAS_EQUIPMENT	Ausrüstung und Dienstleistungen für Öl und Gas
INDUSTRY_OIL_GAS_INTEGRATED	Öl und Gas, integriert
INDUSTRY_OIL_GAS_MIDSTREAM	Öl- und Gas-Midstream
INDUSTRY_OIL_GAS_REFINING	Öl und Gas, Raffinierung und Marketing
INDUSTRY_THERMAL_COAL	Thermische Kohle
INDUSTRY_URANIUM	Uran
Finanzsektor	
INDUSTRY_EXCHANGE_TRADED_FUND	Exchange Traded Fund (ETF)
INDUSTRY_ASSETS_MANAGEMENT	Vermögensverwaltung
INDUSTRY_BANKS_DIVERSIFIED	Banken - Diversifiziert
INDUSTRY_BANKS_REGIONAL	Banken - Regional
INDUSTRY_CAPITAL_MARKETS	Capital markets
INDUSTRY_CLOSE_END_FUND_DEBT	Geschlossener Fonds - Verschuldung
INDUSTRY_CLOSE_END_FUND_EQUITY	Geschlossene Fonds - Aktien
INDUSTRY_CLOSE_END_FUND_FOREIGN	Geschlossener Fonds - Ausland
INDUSTRY_CREDIT_SERVICES	Kredit-Dienstleistungen
INDUSTRY_FINANCIAL_CONGLOMERATE	Finanzkonglomerate
INDUSTRY_FINANCIAL_DATA_EXCHANGE	Finanzdaten & Aktienbörse
INDUSTRY_INSURANCE_BROKERS	Versicherungsmakler
INDUSTRY_INSURANCE_DIVERSIFIED	Versicherung - Diversifiziert
INDUSTRY_INSURANCE_LIFE	Versicherung - Leben

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
INDUSTRY_INSURANCE_PROPERTY	Insurance - Property and casualty
INDUSTRY_INSURANCE_REINSURANCE	Versicherung - Rückversicherung
INDUSTRY_INSURANCE_SPECIALTY	Versicherung - Spezialgebiet
INDUSTRY_MORTGAGE_FINANCE	Hypotheken-Finanzierung
INDUSTRY_SHELL_COMPANIES	Mantelgesellschaften
Gesundheitswesen	
INDUSTRY_BIOTECHNOLOGY	Biotechnologie
INDUSTRY_DIAGNOSTICS_RESEARCH	Diagnostik und Forschung
INDUSTRY_DRUGS_MANUFACTURERS	Arzneimittelhersteller - Allgemein
INDUSTRY_DRUGS_MANUFACTURERS_SPEC	Arzneimittelhersteller - Spezialitäten und Generika
INDUSTRY_HEALTHCARE_PLANS	Krankenversicherungen
INDUSTRY_HEALTH_INFORMATION	Gesundheitsinformationsdienste
INDUSTRY_MEDICAL_FACILITIES	Einrichtungen der medizinischen Versorgung
INDUSTRY_MEDICAL_DEVICES	Medizinische Geräte
INDUSTRY_MEDICAL_DISTRIBUTION	Medizinischer Vertrieb
INDUSTRY_MEDICAL_INSTRUMENTS	Medizinische Instrumente und Zubehör
INDUSTRY_PHARM_RETAILERS	Pharmazeutische Einzelhändler
Industriell	
INDUSTRY_AEROSPACE_DEFENSE	Luft- und Raumfahrt und Verteidigung
INDUSTRY_AIRLINES	Fluggesellschaften
INDUSTRY_AIRPORTS_SERVICES	Flughäfen und Flugdienste
INDUSTRY_BUILDING_PRODUCTS	Bauprodukte und Ausrüstung
INDUSTRY_BUSINESS_EQUIPMENT	Geschäftsausstattung und -zubehör
INDUSTRY_CONGLOMERATES	Großkonzerne
INDUSTRY_CONSULTING_SERVICES	Beratungsdienste
INDUSTRY_ELECTRICAL_EQUIPMENT	Elektrische Ausrüstung & Teile
INDUSTRY_ENGINEERING_CONSTRUCTION	Technik und Baugewerbe
INDUSTRY_FARM_HEAVY_MACHINERY	Landwirtschaftliche Maschinen und schwere Baumaschinen
INDUSTRY_INDUSTRIAL_DISTRIBUTION	Industrievertrieb

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
INDUSTRY_INFRASTRUCTURE_OPERATIONS	Infrastruktur-Betrieb
INDUSTRY_FREIGHT_LOGISTICS	Integrierter Güterverkehr und Logistik
INDUSTRY_MARINE_SHIPPING	Seeschifffahrt
INDUSTRY_METAL_FABRICATION	Metall-Herstellung
INDUSTRY_POLLUTION_CONTROL	Kontrolle von Umweltverschmutzung und Behandlung
INDUSTRY_RAILROADS	Eisenbahnen
INDUSTRY_RENTAL_LEASING	Miet- und Leasing-Dienstleistungen
INDUSTRY_SECURITY_PROTECTION	Sicherheits- und Schutzdienste
INDUSTRY_SPEALITY_BUSINESS_SERVICES	Spezielle Dienstleistungen für Unternehmen
INDUSTRY_SPEALITY_MACHINERY	Spezialisierte Industriemaschinen
INDUSTRY_STUFFING_EMPLOYMENT	Arbeitsvermittlung
INDUSTRY_TOOLS_ACCESSORIES	Werkzeuge und Zubehör
INDUSTRY_TRUCKING	Frachtverkehr
INDUSTRY_WASTE_MANAGEMENT	Waste management
Immobilien	
INDUSTRY_REAL_ESTATE_DEVELOPMENT	Real estate - Development
INDUSTRY_REAL_ESTATE_DIVERSIFIED	Immobilien - Diversifiziert
INDUSTRY_REAL_ESTATE_SERVICES	Immobilien-Dienstleistungen
INDUSTRY_REIT_DIVERSIFIED	Immobilien-Investmentfonds - Diversifiziert
INDUSTRY_REIT_HEALTCARE	REIT - Medizinische Einrichtungen
INDUSTRY_REIT_HOTEL_MOTEL	REIT - Hotels und Motels
INDUSTRY_REIT_INDUSTRIAL	Immobilien-Investmentfonds - Industrie
INDUSTRY_REIT_MORTAGE	Immobilien-Investmentfonds - Hypothek
INDUSTRY_REIT_OFFICE	Immobilien-Investmentfonds - Büros
INDUSTRY_REIT_RESIDENTAL	Immobilien-Investmentfonds - Wohnraum
INDUSTRY_REIT_RETAIL	Immobilien-Investmentfonds - Einzelhandel
INDUSTRY_REIT_SPECIALITY	Immobilien-Investmentfonds - Besondere Räumlichkeiten
Technologie	
INDUSTRY_COMMUNICATION_EQUIPMENT	Kommunikationsausrüstung

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
INDUSTRY_COMPUTER_HARDWARE	Computer-Ausrüstung
INDUSTRY_CONSUMER_ELECTRONICS	Unterhaltungselektronik
INDUSTRY_ELECTRONIC_COMPONENTS	Elektronische Komponenten
INDUSTRY_ELECTRONIC_DISTRIBUTION	Elektronik- und Computervertrieb
INDUSTRY_IT_SERVICES	Informationstechnische Dienstleistungen
INDUSTRY_SCIENTIFIC_INSTRUMENTS	Wissenschaftliche und technologische Instrumente
INDUSTRY_SEMICONDUCTOR_EQUIPMENT	Halbleiter-Ausrüstung und -Materialien
INDUSTRY_SEMICONDUCTORS	Halbleiter
INDUSTRY_SOFTWARE_APPLICATION	Software - Anwendungen
INDUSTRY_SOFTWARE_INFRASTRUCTURE	Software - Infrastruktur
INDUSTRY_SOLAR	Solarenergie
Versorgungsunternehmen	
INDUSTRY_UTILITIES_DIVERSIFIED	Versorger - Diversifiziert
INDUSTRY_UTILITIES_POWERPRODUCERS	Versorger - Unabhängige Energieerzeuger
INDUSTRY_UTILITIES_RENEWABLE	Versorger - Erneuerbare Energie
INDUSTRY_UTILITIES_REGULATED_ELECTRIC	Versorger - Regulierte Elektrizitätsunternehmen
INDUSTRY_UTILITIES_REGULATED_GAS	Versorger - Regulierte Gasunternehmen
INDUSTRY_UTILITIES_REGULATED_WATER	Versorger - Regulierte Wasserversorger
INDUSTRY_UTILITIES_FIRST	Beginn der Aufzählung der Typen von Versorgungsdienstleistungen. Korrespondierend mit INDUSTRY_UTILITIES_DIVERSIFIED.
INDUSTRY_UTILITIES_LAST	Ende der Aufzählung der Typen von Versorgungsdienstleistungen. Korrespondierend mit INDUSTRY_UTILITIES_REGULATED_WATER.

© 2000-2025, MetaQuotes Ltd.

Information über das Konto

für den Erhalt von Informationen über das laufende Konto werden die Funktionen AccountInfoInteger(), AccountInfoDouble() und AccountInfoString() verwendet. Als Parameter werden diesen Funktionen die Werte aus den entsprechen Enumerationen ENUM_ACCOUNT_INFO übergeben.

für die Funktion AccountInfoInteger()

ENUM_ACCOUNT_INFO_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
ACCOUNT_LOGIN	Nummer des Kontos	long
ACCOUNT_TRADE_MODE	Handelsweise des Kontos	ENUM_ACCOUNT_TRADE_MODE
ACCOUNT_LEVERAGE	Höhe des Leverage	long
ACCOUNT_LIMIT_ORDERS	Maximale zulässige Anzahl der geltenden Wartеordern	int
ACCOUNT_MARGIN_SO_MODE	Modus des minimal zugänglichen Marge	ENUM_ACCOUNT_STOPOUT_MODE
ACCOUNT_TRADE_ALLOWED	Erlaubnis zu handeln für das laufende Konto	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
ACCOUNT_TRADE_EXPERT	Erlaubnis zu handeln für den Experten	bool
ACCOUNT_MARGIN_MODE	Berechnungsmodus der Marge	ENUM_ACCOUNT_MARGIN_MODE
ACCOUNT_CURRENCY_DIGITS	Die Anzahl der Nachkommastellen für Kontowährung, die für eine genaue Anzeige der Handelsergebnisse erforderlich sind	int
ACCOUNT_FIFO_CLOSE	Ein Kennzeichen, das anzeigt, dass Positionen nur mittels der FIFO-Regel	bool

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	geschlossen werden können. Wenn der Eigenschaftswert auf true gesetzt ist, dann werden alle Symbolpositionen in der gleichen Reihenfolge geschlossen, in der sie geöffnet werden, beginnend mit der ältesten. Im Falle eines Versuchs, Positionen in einer anderen Order	

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ der Eigenschaft
	zu schließen, erhält der Händler eine entsprechende Fehlermeldung. Für Konten, deren Positionen nicht im Hedging-Modus abgerechnet werden (ACCOUNT_MARGIN_MODE!=ACCOUNT_MARGIN_MODE_RETAIL_HEDGING), ist der Eigenschaftswert immer 'false'.	

für die Funktion AccountInfoDouble()

ENUM_ACCOUNT_INFO_DOUBLE

© 2000-2025, MetaQuotes Ltd.

Idenifikator	Beschreibung	Typ der Eigenschaft
ACCOUNT_BALANCE	Bilanz des Kontos in der Währung des Deposits	double
ACCOUNT_CREDIT	Grösse des gegebenen Kredits in der Währung des Deposits	double
ACCOUNT_PROFIT	Grösse des laufenden Gewinns auf dem Konto in der Währung des Deposits	double
ACCOUNT_EQUITY	Wer der Eigenmittel auf dem Konto in der Währung des Deposits	double
ACCOUNT_MARGIN	Wert des reservierten Pfandgeldes in der Währung des Deposits	double
ACCOUNT_MARGIN_FREE	Grösse der freien Mittel auf dem Konto in der	double

© 2000-2025, MetaQuotes Ltd.

Idenifikator	Beschreibung	Typ der Eigenschaft
	Währung des Deposits, die für die Öffnung der Position zugänglich sind	
ACCOUNT_MARGIN_LEVEL	Konto-Margenniveau in Prozenten	double
ACCOUNT_MARGIN_SO_CALL	Margin Call. Abhängig vom eingestellten ACCOUNT_MARGIN_SO_MODE wird in Prozenten oder in der Währung des Deposits ausgedrückt	double
ACCOUNT_MARGIN_SO_SO	Margin Stop Out. Abhängig vom eingestellten ACCOUNT_MARGIN_SO_MODE wird in Prozenten oder in der Währung des	double

© 2000-2025, MetaQuotes Ltd.

Idenifikator	Beschreibung	Typ der Eigenschaft
	Deposits ausgedrückt	
ACCOUNT_MARGIN_INITIAL	Initial Margin. Die reservierte Mittel auf dem Konto als die Garantie für alle ausstehenden Aufträge (Pending Ordern)	double
ACCOUNT_MARGIN_MAINTENANCE	Maintenance Margin. Die reservierte Mittel auf dem Konto als ein Sicherheitsmindestbetrag für alle offenen Positionen	double
ACCOUNT_ASSETS	Die Aktiva auf dem Konto	double
ACCOUNT_LIABILITIES	Aktuelle Verbindlichkeiten auf dem Konto	double
ACCOUNT_COMMISSION_BLOCKED	Die aktuelle blockierte Kommissi	double

© 2000-2025, MetaQuotes Ltd.

Idenifikator	Beschreibung	Typ der Eigenschaft
	on für das Konto	

für die Funktion AccountInfoString()

ENUM_ACCOUNT_INFO_STRING

Identifikator	Beschreibung	Typ der Eigenschaft
ACCOUNT_NAME	Name des Kunden	string
ACCOUNT_SERVER	Name des Handelsservers	string
ACCOUNT_CURRENCY	Depositenwaehrung	string
ACCOUNT_COMPANY	Name der Gesellschaft, die das Konto bedient	string

Es gibt verschiedene Typen von Konten, die auf dem Handelsserver geöffnet werden können. Um den Typ des Kontos, auf dem das MQL5-Programm lauft, zu finden, verwenden Sie ENUM_ACCOUNT_TRADE_MODE.

ENUM_ACCOUNT_TRADE_MODE

Identifikator	Beschreibung
ACCOUNT_TRADE_MODE_DEMO	Demokonto
ACCOUNT_TRADE_MODE_CONTEST	Wettbewerbskonto
ACCOUNT_TRADE_MODE_REAL	Realkonto

Wenn es nicht genügend Eigenkapital, um offene Positionen zu halten, gibt, geschieht eine Situation der erzwungenen Schließung Stop-Out. Die minimale Marge-Ebene, bei dem Stop-Out eintretet, kann als Prozentsatz oder in Form von Geld angegeben werden. Finden Sie heraus, welcher Modus für dieses Konto eingestellt ist, mit der Enumeration ENUM_ACCOUNT_STOPOUT_MODE.

ENUM_ACCOUNT_STOPOUT_MODE

Identifikator	Beschreibung
ACCOUNT_STOPOUT_MODE_PERCENT	Level wird in Prozenten vorgegeben
ACCOUNT_STOPOUT_MODE_MONEY	Level wird in Geld vorgegeben

© 2000-2025, MetaQuotes Ltd.

ENUM_ACCOUNT_MARGIN_MODE

Bezeichnung	Beschreibung
ACCOUNT_MARGIN_MODE_RETAIL_NETTING	Wird beim außerbörslichen Handel mit Netting Mode verwendet (nur eine Position pro Symbol). Die Marge wird je nach Symboltyp(SYMBOL_TRADE_CALC_MODE) berechnet.
ACCOUNT_MARGIN_MODE_EXCHANGE	Wird beim Handel an Devisenmärkten verwendet. Die Margin wird auf der Basis von Diskonten berechnet, die in den Einstellungen des Symbols angegeben sind. Diskonte werden vom Broker gesetzt, können aber nicht niedriger als die von der Börse gesetzten Werte sein.
ACCOUNT_MARGIN_MODE_RETAIL_HEDGING	Wird beim außerbörslichen Handel bei einer unabhängigen Verrechnung von Positionen verwendet (Hedge Modus, man kann mehrere Positionen pro Symbol haben). Die Marge wird je nach Symboltyp (SYMBOL_TRADE_CALC_MODE) und unter Berücksichtigung des Wertes der abgesicherten Marge (SYMBOL_MARGIN_HEDGED) berechnet.

Ein Beispiel für ein Skript, das kurze Information über das Konto anzeigt.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Firmenname
   string company=AccountInfoString(ACCOUNT_COMPANY);
//--- Kundenname
   string name=AccountInfoString(ACCOUNT_NAME);
//--- Kontonummer
   long login=AccountInfoInteger(ACCOUNT_LOGIN);
//--- Servername
   string server=AccountInfoString(ACCOUNT_SERVER);
//--- Kontowährung
   string currency=AccountInfoString(ACCOUNT_CURRENCY);
//--- Demo-, Konkurs- oder Realkonto
   ENUM_ACCOUNT_TRADE_MODE account_type=(ENUM_ACCOUNT_TRADE_MODE)AccountInfoInteger(AC
//--- Nun verwandeln wir den Enum-Wert in übersichtlicher Form
   string trade_mode;
   switch(account_type)
     {
      case  ACCOUNT_TRADE_MODE_DEMO:
```

© 2000-2025, MetaQuotes Ltd.

```
         trade_mode="Demo";
         break;
      case  ACCOUNT_TRADE_MODE_CONTEST:
         trade_mode="Konkurs";
         break;
      default:
         trade_mode="Real";
         break;
    ;}
//--- Stop Out wird in Prozent oder in Form von Geld angegeben
   ENUM_ACCOUNT_STOPOUT_MODE stop_out_mode=(ENUM_ACCOUNT_STOPOUT_MODE)AccountInfoInteg
//--- Erhalten die Werte der Ebenen, auf denen Margin Call und Stop-Out geschehen
   double margin_call=AccountInfoDouble(ACCOUNT_MARGIN_SO_CALL);
   double stop_out=AccountInfoDouble(ACCOUNT_MARGIN_SO_SO);
//--- Zeigen kurze Informationen über das Konto
   PrintFormat("Konto des Kunden '%s' #%d %s öffnet in '%s' am Server '%s'",
               name,login,trade_mode,company,server);
   PrintFormat("Kontowährung - %s, MarginCall und StopOut werden in %s" angegeben,
               currency,(stop_out_mode==ACCOUNT_STOPOUT_MODE_PERCENT)?"Prozent":" Gelc
   PrintFormat("MarginCall=%G, StopOut=%G",margin_call,stop_out);
 ;}
```

© 2000-2025, MetaQuotes Ltd.

Teststatistik

Nach dem Test sind verschiedene Parameter der Statistik der Trading-Ergebnisse berechnet. Sie können Werte der Parameter mit Hilfe der Funktion TesterStatistics() abrufen, mit der Angabe der Parameter-ID aus ENUM_STATISTICS.

Obwohl zwei Typen von Parametern (int und double) sind für die Berechnung der Statistik verwendet, gibt die Funktion alle Werte in der double-Form zurück. Alle statistischen Werte der double-Typs sind in der Einzahlung-Währung standardmäßig ausgedrückt, sofern nicht anders angegeben.

ENUM_STATISTICS

ID	Beschreibung des statistischen Parameters	Typ
STAT_INITIAL_DEPOSIT	Der Wert des Ersteinzahlung	double
STAT_WITHDRAWAL	Von einem Konto abgezogene Geld	double
STAT_PROFIT	Nettogewinn nach Test, Betrag von STAT_GROSS_PROFIT und STAT_GROSS_LOSS (STAT_GROSS_LOSS ist immer kleiner oder gleich Null)	double
STAT_GROSS_PROFIT	Total Gewinn, Betrag aller profitablen (positiven) Trades. Der Wert is größer oder gleich Null	double
STAT_GROSS_LOSS	Total Verlust, Betrag aller verlustbringenden (negativen) Trades. Der Wert is kleiner oder gleich Null	double
STAT_MAX_PROFITTRADE	Maximaler Gewinn - der höchste Wert unter allen profitablen Trades. Der Wert is größer oder gleich Null	double
STAT_MAX_LOSSTRADE	Maximaler Verlust - der kleinste Wert unter allen verlustbringenden	double

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung des statistischen Parameters	Typ
	Trades. Der Wert is kleiner oder gleich Null	
STAT_CONPROFITMAX	Maximaler Gewinn unter konsequent profitable Trades. Der Wert is größer oder gleich Null	double
STAT_CONPROFITMAX_TRADES	Anzahl der Trades, die STAT_CONPROFITMAX geformt haben (maximaler Gewinn unter konsequent profitable Trades)	int
STAT_MAX_CONWINS	Total Gewinn in der längsten Reihe von profitablen Trades	double
STAT_MAX_CONPROFIT_TRADES	Anzahl der Trades in der längste Reihe von profitablen Trades STAT_MAX_CONWINS	int
STAT_CONLOSSMAX	Maximaler Verlust unter konsequent profitable Trades. Der Wert is kleiner oder gleich Null	double
STAT_CONLOSSMAX_TRADES	Anzahl der Trades, die STAT_CONLOSSMAX geformt haben (maximaler Verlust unter konsequent verlustbringende Trades)	int
STAT_MAX_CONLOSSES	Total Verlust in der längsten Reihe von verlustbringenden Trades	double
STAT_MAX_CONLOSS_TRADES	Anzahl der Trades in der längste Reihe von verlustbringenden Trades	int

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung des statistischen Parameters	Typ
	STAT_MAX_CONLOSSES	
STAT_BALANCEMIN	Der minimale Wert der Bilanz	double
STAT_BALANCE_DD	Maximaler Drawdown der Bilanz in Geld. Im Prozeß der Trading kann ein Bilanz zahlreiche Drawdowns haben, hier nehmen wir den größte Wert	double
STAT_BALANCEDD_PERCENT	Drawdown der Bilanz als Prozentsatz, der zum Zeitpunkt von maximalem Bilanz-Drawdown in Geld aufgenommen wurde (STAT_BALANCE_DD)	double
STAT_BALANCE_DDREL_PERCENT	Maximaler Drawdown der Bilanz als Prozentsatz. Im Prozeß der Trading kann ein Bilanz zahlreiche Drawdowns haben, für jeden Drawdown wird sein Wert als Prozentsatz berechnet. Der größte Wert wird zurückgegeben	double
STAT_BALANCE_DD_RELATIVE	Drawdown der Bilanz in Geld, der zum Zeitpunkt von maximalem Bilanz-Drawdown in Prozent aufgenommen wurde (STAT_BALANCE_DDREL_PERCENT)	double
STAT_EQUITYMIN	Der minimale Wert des Eigenkapitals	double
STAT_EQUITY_DD	Maximaler Drawdown des Eigenkapitals in Geld. Im Prozeß der	double

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung des statistischen Parameters	Typ
	Trading kann ein Eigenkapital zahlreiche Drawdowns haben, hier nehmen wir den größte Wert	
STAT_EQUITYDD_PERCENT	Drawdown des Eigenkapitals in Prozent, der zum Zeitpunkt von maximalem Eigenkapitasdrawdown in Geld aufgenommen wurde (STAT_EQUITY_DD)	double
STAT_EQUITY_DDREL_PERCENT	Maximaler Drawdown des Eigenkapitals als Prozentsatz. Im Prozeß der Trading kann ein Eigenkapital zahlreiche Drawdowns haben, für jeden Drawdown wird sein Wert als Prozentsatz berechnet. Der größte Wert wird zurückgegeben	double
STAT_EQUITY_DD_RELATIVE	Drawdown des Eigenkapitals in Geld, der zum Zeitpunkt von maximalem Eigenkapitalsdrawdown in Prozent aufgenommen wurde (STAT_EQUITY_DDREL_PERCENT).	double
STAT_EXPECTED_PAYOFF	Erwartete Auszahlung	double
STAT_PROFIT_FACTOR	Profit-Faktor, gleich dem Verhältnis von STAT_GROSS_PROFIT /STAT_GROSS_LOSS. Wenn STAT_GROSS_LOSS=0, ist Profit-Faktor gleich DBL_MAX	double

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung des statistischen Parameters	Typ
STAT_RECOVERY_FACTOR	Recovery-Faktor, gleich dem Verhältnis von STAT_PROFIT/STAT_BALANCE_DD	double
STAT_SHARPE_RATIO	Sharpe Ratio	double
STAT_MIN_MARGINLEVEL	Der minimale Wert der Margeebene	double
STAT_CUSTOM_ONTESTER	Der Wert des berechneten Benutzer-Optimierungskriterium, der OnTester()-Funktion zurückgegeben wird	double
STAT_DEALS	Anzahl der Deals	int
STAT_TRADES	Anzahl der Trades	int
STAT_PROFIT_TRADES	Gewinn-Trades	int
STAT_LOSS_TRADES	Verlust-Trades	int
STAT_SHORT_TRADES	Kurze Trades	int
STAT_LONG_TRADES	Lange Trades	int
STAT_PROFIT_SHORTTRADES	Kurze Gewinn-Trades	int
STAT_PROFIT_LONGTRADES	Lange Gewinn-Trades	int
STAT_PROFITTRADES_AVGCON	Die durchschnittliche Länge einer Reihe von Gewinn-Trades	int
STAT_LOSSTRADES_AVGCON	Die durchschnittliche Länge einer Reihe von Verlust-Trades	int
STAT_COMPLEX_CRITERION	Komplexes Optimierungskriterium	

© 2000-2025, MetaQuotes Ltd.

Handelskonstanten

Verschidenartige Konstanten, die für Programmieren von Handelsstrategien verwendet werden, sind in folgende Gruppen aufgeteilt:

- Information über historische Daten des Instrumentes – Erhaltung der allgemeinenInformation über das finanzielle Instrument;
- Ordereigenschaften – Erhalten von Information über Handelsordern;
- Positionsegenschaften – Erhalten von Information über laufende Positionen;
- Dealeigenschaften – Erhalten von Information über abgeschlossene Deals;
- Typen der Handelsoperationen – Beschreibung der zugaenglichen Handelsoperationen;
- Typen der Handelstransaktionen - Beschreibung der möglichen Typen der Handelstransaktionen;
- Handelsordern in DOM - Trennung der Ordern nach der Richtung der angeforderten Handelsoperation.

© 2000-2025, MetaQuotes Ltd.

Information über historische Daten des Instrumentes

Beim Zugang zu Zeitreihen wird für Erhaltung der zusaetzlichen Information über das Instrument die Funktion SeriesInfoInteger() verwendet. Als Parameter dieser Funktion wird Identifikator der angeforderten Eigenschaft übertragen, der einen der Enumerationswerte ENUM_SERIES_INFO_INTEGER. annehmen kann

ENUM_SERIES_INFO_INTEGER

Identifikator	Beschreibung	Typ der Eigenschaft
SERIES_BARS_COUNT	Anzahl der Bars für Symbol-Periode zum jetzigen Zeitpunkt	long
SERIES_FIRSTDATE	Das erste Datum für Symbol-Periode zum jetzigen Zeitpunkt	datetime
SERIES_LASTBAR_DATE	Öffnungszeit der letzten Bar für Symbol-Periode	datetime
SERIES_SERVER_FIRSTDATE	Das erste Datum des Symbols auf dem Server unabhängig von der Periode	datetime
SERIES_TERMINAL_FIRSTDATE	Das erste Datum in der Geschichte des Symbols im Client-Terminal unabhängig von der Periode	datetime
SERIES_SYNCHRONIZED	Flagge der Synchronisierung von Symbol/Periode zum jetzigen Zeitpunkt	bool

© 2000-2025, MetaQuotes Ltd.

Ordereigenschaften

Befehlsanweisungen, Handelsoperationen durchzuführen, werden als Order formalisiert. Jede Order hat eine Menge der Eigenschaften für Lesen, die Information über die kann durch die Funktionen OrderGet...() und HistoryOrderGet...() erhalten werden.

für die Funktionen OrderGetInteger() und HistoryOrderGetInteger()

ENUM_ORDER_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ
ORDER_TICKET	Das Ticket der Order. Das ist eine einmalige Nummer, die jeder Order zugewiesen wird.	long
ORDER_TIME_SETUP	Set-up Zeit des Orders	datetime
ORDER_TYPE	Ordertyp	ENUM_ORDER_TYPE
ORDER_STATE	Orderstatus	ENUM_ORDER_STATE
ORDER_TIME_EXPIRATION	Ablauffrist einer Order	datetime
ORDER_TIME_DONE	Durchführungszeit oder Annullieren einer Order	datetime
ORDER_TIME_SETUP_MSC	Zeitpunkt der Erstellung von Order in Millisekunden seit 01.01.1970	long
ORDER_TIME_DONE_MSC	Zeit der Ausführung/Abschiebung des Orders in Millisekunden seit 01.01.1970	long
ORDER_TYPE_FILLING	Durchführungstyp nach dem Rest	ENUM_ORDER_TYPE_FILLING

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
ORDER_TYPE_TIME	Lebenszeit des Orders	ENUM_ORDER_TYPE_TIME
ORDER_MAGIC	Identifikator des Experten, der Order gestellt hat (bestimmt dafür, dass jeder Expert seine eigene unikale Nummer stellt)	long
ORDER_REASON	Grund oder Quelle der Platzierung einer Order	ENUM_ORDER_REASON
ORDER_POSITION_ID	Identifikator der Position, den Order nach seine Durchführung erhält. Jede durchgeführte Order erzeugt einen Deal, das eine neue Position eroeffnet oder eine schon existierende Position verändert. Durchgeführte Order erhält eben diesen Identifikator dieser Position.	long
ORDER_POSITION_BY_ID	Der Identifikator der Gegenposition für die Orders vom Typ	long

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
	ORDER_TYPE _CLOSE_BY.	

für die Funktionen OrderGetDouble() und HistoryOrderGetDouble()

ENUM_ORDER_PROPERTY_DOUBLE

Identifikator	Beschreibung	Typ
ORDER_VOLUME_INITIAL	Initialvolumen bei der Orderstellung	double
ORDER_VOLUME_CURRENT	eine nicht durchgeführte Order	double
ORDER_PRICE_OPEN	Preis angegeben in Order	double
ORDER_SL	Level Stop Loss	double
ORDER_TP	Level Take Profit	double
ORDER_PRICE_CURRENT	Laufender Preis des Ordersymbols	double
ORDER_PRICE_STOPLIMIT	Preis der Aufstellung der Limit Order bei Auslösung von StopLimit Order	double

für die Funktionen OrderGetString() und HistoryOrderGetString()

ENUM_ORDER_PROPERTY_STRING

Identifikator	Beschreibung	Typ
ORDER_SYMBOL	Symbol der Order	string
ORDER_COMMENT	Orderkommentar	string
ORDER_EXTERNAL_ID	ID der Order im	string

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
	Außenhandelssystem (an der Börse)	

Beim Senden einer Handelsanforderung durch die Funktion OrderSend() muss für einige Operationen Ordertyp angegeben werden. Ordertyp wird im Feld *type* der speziellen Struktur MqlTradeRequest angegeben, und kann verschiedene Werte aus der Enumeration ENUM_ORDER_TYPE annehmen.

ENUM_ORDER_TYPE

Identifikator	Beschreibung
ORDER_TYPE_BUY	Marktkauforder
ORDER_TYPE_SELL	Martverkauforder
ORDER_TYPE_BUY_LIMIT	Schwebende Order Buy Limit
ORDER_TYPE_SELL_LIMIT	Schwebende Order Sell Limit
ORDER_TYPE_BUY_STOP	Schwebende Order Buy Stop
ORDER_TYPE_SELL_STOP	Schwebende Order Sell Stop
ORDER_TYPE_BUY_STOP_LIMIT	Beim Erreichen des Orderpreises wird eine Warteorder Buy Limit zum Preis StopLimit gestellt
ORDER_TYPE_SELL_STOP_LIMIT	Beim Erreichen des Orderpreises wird eine Warteorder Sell Limit zum Preis StopLimit gestellt
ORDER_TYPE_CLOSE_BY	Order zum Schließen zur Gegenposition

Jede Order hat einen Status, der seinen Stand beschreibt. Für die Erhaltung der Information verwenden Sie die Funktion OrderGetInteger() oder HistoryOrderGetInteger() mit dem Modifikator ORDER_STATE. Zulaessige Werte werden in der Enumeration ENUM_ORDER_STATE aufbewahren.

ENUM_ORDER_STATE

Identifikator	Beschreibung
ORDER_STATE_STARTED	Order geprüft, aber vom Broker noch nicht angenommen
ORDER_STATE_PLACED	Order angenommen
ORDER_STATE_CANCELED	Order ist vom Kunden abgelehnt
ORDER_STATE_PARTIAL	Order ist teilweise durchgeführt
ORDER_STATE_FILLED	Order ist vollständig durchgeführt
ORDER_STATE_REJECTED	Order abgelehnt

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
ORDER_STATE_EXPIRED	Order ist nach Ablauffrist abgelehnt
ORDER_STATE_REQUEST_ADD	Order ist im Registrierungsstatus (Aussetzung in Handelssystem)
ORDER_STATE_REQUEST_MODIFY	Order ist im Änderungsstatus (Verändern von Parametern)
ORDER_STATE_REQUEST_CANCEL	Order ist im Löschungsstatus (Löschung aus dem Handelssystem)

Beim Senden einer Handelsanfrage für die Ausführung **zur aktuellen Zeit** (Time-In-Force) sollten der Preis und das gewünschte Kauf-/Verkaufsvolumen angegeben werden. Bedenken Sie aber, dass die Finanzmärkte keine Garantie dafür bieten, dass das gesamte angeforderte Volumen für ein bestimmtes Finanzinstrument zum gewünschten Preis verfügbar ist. Deshalb werden die Handelsoperationen in Echtzeit durch die Modi der Preis- und Volumenausführung geregelt. Die Modi, oder Ausführungsrichtlinien, definieren die Regeln für den Fall, dass der Preis sich geändert hat oder das angeforderte Volumen **momentan** nicht vollständig umgesetzt werden kann.

Der Preisausführungsmodus kann über die Symboleigenschaft SYMBOL_TRADE_EXEMODE erhalten werden, die die Kombination der Flags der Enumeration ENUM_SYMBOL_TRADE_EXECUTION enthält.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
Ausführungsmodus (Anfrage zur Ausführung)	Ausführung einer Markt-Order zu dem zuvor vom Broker erhaltenen Preis. Die Preise für eine bestimmte Markt-Order werden beim Broker angefo	SYMBOL_TRADE_EXECUTION_REQUEST

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	rdert, bevor die Order gesendet wird. Nach Erhalt der Preise kann die Auftragsausführung zu dem angegebenen Preis entweder bestätigt oder abgelehnt werden.	
Sofortige Ausführung (Sofortige Ausführung)	Sofortige Ausführung einer Markt-Order zu dem angegebenen Preis. Wenn eine Handelsanfrage zur Ausführung gesend	SYMBOL_TRADE_EXECUTION_INSTANT

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	et wird, fügt die Plattform automatisch die aktuellen Preise dem Auftrag hinzu. • Wenn der Broker den Preis akzeptiert, w	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	ird der Auftrag ausgeführt. • Wenn der Broker den Auftrag	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	tenPreisnichtakzeptiert, wird ein "Requote" gesendet	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	— der Broker gibt Preise zur Rückzudenen dieser Auftraga	

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	usgeführtwerdenkannn.	
Marktausführung (Marktausführung)	Der Broker trifft eine Entscheidung über den Ausführungspreis des Auftrags ohne zusätzliche Diskussion mit dem Händler. Das Absenden des Auftrags mit einem	SYMBOL_TRADE_EXECUTION_MARKET

© 2000-2025, MetaQuotes Ltd.

Ausführungsmodus	Beschreibung	Der Wert von ENUM_SYMBOL_TRADE_EXECUTION
	solchen Modus schließt die vorherige Zustimmung zu ihrer Ausführung zu diesem Preis ein.	
Börsliche Ausführung (Börsliche Ausführung)	Handelsgeschäfte werden zu den aktuellen Preisen des Marktes ausgeführt.	SYMBOL_TRADE_EXECUTION_EXCHANGE

Die Richtlinie der Volumenfüllung wird in der Auftragseigenschaft ORDER_TYPE_FILLING angegeben und darf nur die Werte aus der Enumeration ENUM_ORDER_TYPE_FILLING enthalten

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
Alles oder Nichts (Fill or Kill, FOK)	Ein Auftrag wird nur mit dem angegebenen Volumen ausgeführt.	ORDER_FILLING_FOK

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	Wenn die angeforderte Volumen eines Finanzinstruments derzeit nicht auf dem Markt verfügbar ist, wird der Auftrag nicht ausgeführt. Das gewünschte Volumen kann sich aus mehreren verfügbaren Angeboten zusammensetzen. Die Möglichkeit der Verwendung von	

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	FOK wird durch den Handelsserver festgelegt.	
Jetzt oder Nie (Immediate or Cancel, IOC)	Ein Händler akzeptiert, den Auftrag mit dem maximal am Markt verfügbaren Volumen innerhalb des in dem Auftrag angegebenen Volumens auszuführen. Wenn die Anfrage nicht vollständig umgesetzt werden kann, wird	ORDER_FILLING_IOC

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	ein Auftrag mit dem verfügbaren Volumen ausgeführt und das verbleibende Volumen wird annulliert. Die Möglichkeit der Verwendung von IOC-Orders wird auf dem Handelsserver festgelegt.	
Passiv (Buchen oder Stornieren, BoC)	Ein BoC-Auftrag setzt voraus, dass die Order nur in der Tiefe	ORDER_FILLING_BOC

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	des Marktes platziert werden kann und nicht sofort ausgeführt werden kann. Wenn der Auftrag bei der Platzierung sofort ausgeführt werden könnte, dann würde er storniert werden. Tatsächlich garantiert die BOC-Politik, dass der Preis des erteilten Auftrags	

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	schlechter sein wird als der aktuelle Markt. BoC-Aufträge werden zur Umsetzung des passiven Handels verwendet, damit der Auftrag nicht sofort bei der Platzierung ausgeführt und die aktuelle Liquidität nicht verändert wird. Es werden nur Limit- und	

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	Stop-Limit-Orders unterstützt (ORDER_TYPE_BUY_LIMIT, ORDER_TYPE_SELL_LIMIT, ORDER_TYPE_BUY_STOP_LIMIT, ORDER_TYPE_SELL_STOP_LIMIT).	
Rückkehren	Im Falle einer teilweisen Füllung wird ein Auftrag mit Restvolumen nicht storniert, sondern weiterbearbeitet. Diese Form	ORDER_FILLING_RETURN

© 2000-2025, MetaQuotes Ltd.

Füllrichtlinie	Beschreibung	Die Wert von ENUM_ORDER_TYPE_FILLING
	der Rückkehr nach einem gesendeten Auftrag sind im Modus der Marktausführung (Marktausführung - SYMBOL_TRADE_EXECUTION_MARKET) nicht zulässig.	

Beim Senden einer Handelsanfrage mit der Funktion OrderSend() kann im Feld *type_filling* die notwendige Volumenausführungspolitik eingestellt werden, und zwar in der speziellen Struktur MqlTradeRequest.Es stehen die Werte aus der Enumeration ENUM_ORDER_TYPE_FILLING zur Verfügung. Um den Eigenschaftswert in eines bestimmten aktiven/abgeschlossenen Auftrags zu erhalten, verwenden Sie die Funktion OrderGetInteger() oder HistoryOrderGetInteger() mit dem Modifikator ORDER_TYPE_FILLING.

Vor dem Senden eines Auftrags mit der aktuellen Ausführungszeit, für die korrekte Einstellung des Wertes von ORDER_TYPE_FILLING (Volumenausführungsart), können Sie die Funktion SymbolInfoInteger() mit jedem Finanzinstrument verwenden, um den Eigenschaftswert von SYMBOL_FILLING_MODE zu erhalten, der die Volumenausführungsarten, die für das Symbol erlaubt sind, als eine Kombination von Flags anzeigt. Der Fülltyp ORDER_FILLING_RETURN ist immer aktiviert, außer im Modus "Marktausführung" (SYMBOL_TRADE_EXECUTION_MARKET).

Die Verwendung der Fülltypen in Abhängigkeit vom Ausführungsmodus ist in der folgenden Tabelle dargestellt:

© 2000-2025, MetaQuotes Ltd.

Ausführungsart	Alles oder Nichts (FOK, ORDER_FILLING_FOK)	Jetzt oder Nie (IOC, IOC ORDER_FILLING_IOC)	Rückkehren (Return, ORDER_FILLING_RETURN)
Sofortige Ausführung (SYMBOL_TRADE_EXECUTION_INSTANT)	+ (unabhängig von der Einstellung eines Symbols)	+ (unabhängig von der Einstellung eines Symbols)	+ (immer)
Ausführungsanfrage SYMBOL_TRADE_EXECUTION_REQUEST	+ (unabhängig von der Einstellung eines Symbols)	+ (unabhängig von der Einstellung eines Symbols)	+ (immer)
Marktausführung SYMBOL_TRADE_EXECUTION_MARKET	+ (definiert den Symboleinstellungen)	+ (definiert den Symboleinstellungen)	- (deaktiviert, unabhängig von den Symboleinstellungen)
Börsliche Ausführung SYMBOL_TRADE_EXECUTION_EXCHANGE	+ (definiert den Symboleinstellungen)	+ (definiert den Symboleinstellungen)	+ (immer)

Bei Pending-Orders (schwebenden Aufträgen) sollte unabhängig von der Ausführungsart (SYMBOL_TRADE_EXEMODE) die Ausführungsart ORDER_FILLING_RETURN verwendet werden, da solche Aufträge zum Zeitpunkt des Absendens nicht zur Ausführung vorgesehen sind. Bei der Verwendung von Pending-Orders (schwebenden Aufträgen) akzeptiert ein Händler im Voraus, dass der Broker, wenn die Bedingungen für ein Geschäft mit diesem Auftrag erfüllt sind, die von der Börse unterstützte Ausführungsart verwendet.

Gültigkeitsfrist einer Order kann im Feld *type_time* der speziellen Struktur MqlTradeRequest angegeben werden beim Senden der Handelsanforderung durch die Funktion OrderSend(). Zulässig sind Werte aus der Enumeration ENUM_ORDER_TYPE_TIME. Für die Erhaltung des Wertes dieser Eigenschaft verwenden Sie die Funktion OrderGetInteger() oder HistoryOrderGetInteger() mit dem Modifikator ORDER_TYPE_TIME.

ENUM_ORDER_TYPE_TIME

Identifikator	Beschreibung
ORDER_TIME_GTC	Pending-Order bleibt gültig, bis sie manuell gelöscht wird (Good Till Cancel)
ORDER_TIME_DAY	Order wird nur innerhalb des Handelstages gültig
ORDER_TIME_SPECIFIED	Order wird bis zum Ablauffrist gültig
ORDER_TIME_SPECIFIED_DAY	Order wird bis 23:59:59 Uhr am angegebenen Datum gültig. Wird diese Zeit nicht auf dem Handelszeit fallen, wird der

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
	Ablauf in naher den Handelszeit auftreten.

Die Eigenschaft ORDER_REASON beinhaltet den Grund der Platzierung einer Order. Eine Order kann mithilfe eines MQL5-Programms, in einer mobilen Anwendung oder infolge eines StopOut Ereignisses usw. platziert werden. Die möglichen Werte von ORDER_REASON sind in der Aufzählung ENUM_ORDER_REASON beschrieben.

ENUM_ORDER_REASON

Identifier	Beschreibung
ORDER_REASON_CLIENT	Die Order wurde in einem Desktop-Terminal platziert
ORDER_REASON_MOBILE	Die Order wurde in einer mobilen Anwendung platziert
ORDER_REASON_WEB	Die Order wurde auf der Webplattform platziert
ORDER_REASON_EXPERT	Die Order wurde durch ein MQL5-Programm platziert - einen Expert Advisor oder Script
ORDER_REASON_SL	Die Order wurde infolge der Auslösung von Stop Loss platziert
ORDER_REASON_TP	Die Order wurde infolge der Auslösung von Take Profit platziert
ORDER_REASON_SO	Die Order wurde infolge des Ereginisses Stop Out platziert

© 2000-2025, MetaQuotes Ltd.

Eigenschaften der Positionen

Das Ergebnis der Handelsopeartionen führt zur Öffnung der Position, Veränderung ihres Volumens und/oder Richtung oder ihr Annulileren. Handelsoperationen werden aufgrund der Ordern durchgeführt, die durch die Funktion OrderSend() als Handelsanorderungen gesendet werden. Für jedes fianzielles Instrument (Symbol) ist nur eine offene Position moeglich. Die Position hat Set der Eigenschaften, dier für Lesen durch die Funktionen PositionGet...() zugaenglich sind.

für die Funktion PositionGetInteger()

ENUM_POSITION_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ
POSITION_TICKET	Das Ticket der Position. Das ist eine einmalige Nummer, die jeder neu eröffneten Position zugewiesen wird. In der Regel entspricht sie dem Ticket der Order, mit der die Position eröffnet wurde, außer den Fällen, wenn das Ticket infolge Operationen auf dem Server geändert wurde. Z.B. Anrechnung von Swaps durch eine erneute Eröffnung einer Position. Um die Order zu finden, mit der die Position eröffnet wurde, ist die Eigenschaft POSITION_IDENTIFIER zu verwenden. Der Wert von POSITION_TICKET entspricht	long

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
	MqlTradeRequest::position.	
POSITION_TIME	Eröffnungszeit der Position	datetime
POSITION_TIME_MSC	Eröffnungszeit der Position in Millisekunden seit 01.01.1970	long
POSITION_TIME_UPDATE	Positionsänderungszeit	datetime
POSITION_TIME_UPDATE_MSC	Positionsänderungszeit in Millisekunden seit 01.01.1970	long
POSITION_TYPE	Typ der Position	ENUM_POSITION_TYPE
POSITION_MAGIC	Magic number für die Position (sehen Sie ORDER_MAGIC)	long
POSITION_IDENTIFIER	Identifikator einer Position ist eine unikale Zahl, die jeder neu geoeffneten Position zugeordnet wird und innerhalb ihres ganzen Leben nicht Eine Umkehr der Positionsrichtung verändert den Identifikator der Position. Der Identifikator der Position wird in jeder Order (ORDER_POSITION_ID) und in jedem Trade (DEAL_POSITION_ID) angegeben, der diese eröffnet,	long

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
	modifiziert oder geschlossen hat. Verwenden Sie diese Eigenschaft für die Suche nach Ordern und Trades, die mit der Position verbunden sind. Bei der Umkehr einer Position im Netting-Modus (mithilfe eines einheitlichen in/out Trades) ändert sich POSITION_IDENTIFIER nicht. Dabei wird aber POSITION_TICKET durch das Ticket der Order ersetzt, die die Umkehr verursacht hat. Im Hedging-Modus ist die Umkehr einer Position nicht vorgehesen.	
POSITION_REASON	Grund der Eröffnung einer Position	ENUM_POSITION_REASON

für die Funktion PositionGetDouble()

ENUM_POSITION_PROPERTY_DOUBLE

Identifikator	Beschreibung	Typ
POSITION_VOLUME	Volumen der Position	double
POSITION_PRICE_OPEN	Positionspreis	double
POSITION_SL	Level Stop Loss für die offene Position	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
POSITION_TP	Level Take Profit für die offene Position	double
POSITION_PRICE_CURRENT	Laufender Preis des Symbols	double
POSITION_SWAP	Gesamtswap	double
POSITION_PROFIT	Laufender Gewinn	double

für die Funktion PositionGetString()

ENUM_POSITION_PROPERTY_STRING

Identifikator	Beschreibung	Typ
POSITION_SYMBOL	Symbol der Position	string
POSITION_COMMENT	Kommentar zur Position	string
POSITION_EXTERNAL_ID	Positionskennung eines anderen Handelsplatzes (einer anderen Börse)	string

Richtung der offenen Position (Kauf oder Verkauf) ist durch Wert von der Enumeration ENUM_POSITION_TYPE bestimmt. Für die Erhaltung des Typs der offenen Position muss die Funktion PositionGetInteger() mit dem Modifikator POSITION_TYPE verwendet werden.

ENUM_POSITION_TYPE

Identifikator	Beschreibung
POSITION_TYPE_BUY	Kauf
POSITION_TYPE_SELL	Verkauf

Die Eigenschaft POSITION_REASON beinhaltet den Grund der Eröffnung einer Position. Eine Position kann infolge der Ausführung einer Order eröffnet werden, die in einem Desktop-Terminal, einer mobilen Anwendung, durch einen Expert Advisor usw. platziert wurde. Die möglichen Werte von POSITION_REASON werden in der Aufzählung ENUM_POSITION_REASON beschrieben.

ENUM_POSITION_REASON

© 2000-2025, MetaQuotes Ltd.

Identifier	Beschreibung
POSITION_REASON_CLIENT	Die Position wurde infolge der Auslösung einer Order eröffnet, die in einem Desktop-Terminal platziert wurde
POSITION_REASON_MOBILE	Die Position wurde infolge der Auslösung einer Order eröffnet, die in einer mobilen Anwendung platziert wurde
POSITION_REASON_WEB	Die Position wurde infolge der Auslösung einer Order eröffnet, die auf der Webplattform platziert wurde
POSITION_REASON_EXPERT	Die Position wurde infolge der Auslösung einer Order eröffnet, die durch ein MQL5-Programm - einen Expert Advisor oder ein Script platziert wurde

© 2000-2025, MetaQuotes Ltd.

Eigenschaften der Deals

Geschäft ist die Spiegelung der Tatsache der Handelsoperation auf Grund der Order, die Handelsanforderung enthält. Jeder Deal wird durch Eigenschaften beschrieben, die ermöglichen, Information über sie zu bekommen. Für Lesen der Werte der Eigenschaften werden die Funktionen der Art HistoryDealGet...() verwendet, die die Werte aus den entsprechenden Enumerationen rückgeben.

für die Funktion HistoryDealGetInteger()

ENUM_DEAL_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ
DEAL_TICKET	Das Ticket des Trades. Das ist eine einmalige Nummer, die jedem Trade zugewiesen wird.	long
DEAL_ORDER	Order, auf deren Grund der Deal abgeschlossen wurde	long
DEAL_TIME	Zeit des Dealabschlusses	datetime
DEAL_TIME_MSC	Zeitpunkt der Transaktion in Millisekunden seit 01.01.1970	long
DEAL_TYPE	Typ des Deals	ENUM_DEAL_TYPE
DEAL_ENTRY	Dealsrichtung - Markteingang, Marktausgang oder Kehrwendung	ENUM_DEAL_ENTRY
DEAL_MAGIC	Magic number für Deal (sehen Sie ORDER_MAGIC)	long
DEAL_REASON	Grund oder Ursprung der Ausführung eines Abschlusses	ENUM_DEAL_REASON
DEAL_POSITION_ID	Indetifikator der Position, an deren Öffnung, Veränderung oder Schliessung sich der Deal teilnahm. Jede Position hat ihren unikalen Identifikator, der allen Deals zugeordnet wird, die im Instrument innerhalb des ganzen Lebens der Position abgeschlossen wurde.	long

für die Funktion HistoryDealGetDouble()

ENUM_DEAL_PROPERTY_DOUBLE

Identifikator	Beschreibung	Typ
DEAL_VOLUME	Dealvolumen	double
DEAL_PRICE	Dealpreis	double
DEAL_COMMISSION	Dealkommission	double

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
DEAL_SWAP	Gesamtswap beim Schliessen	double
DEAL_PROFIT	Finanzielles Ergebnis des Deals	double
DEAL_FEE	Gebühr für das Ausführen eines Deals, die unmittelbar nach der Ausführung erhoben wird	double
DEAL_SL	Stop-Loss • Eröffnungs- und Umkehr-Deals verwenden die Preise der Stop-Loss' von der ursprünglichen Order, auf deren Grundlage die Position eröffnet oder umgekehrt wurde. • Die Exit-Deals verwenden den Stop-Loss-Level einer Position zum Zeitpunkt der Positionsschließung	double
DEAL_TP	Take-Profit • Eröffnungs- und Umkehr-Deals verwenden die Preise der Take-Profits von der ursprünglichen Order, auf deren Grundlage die Position eröffnet oder umgekehrt wurde. • Die Exit-Deals verwenden den Wert des Take-Profit einer Position zum Zeitpunkt der Positionsschließung	double

für die Funktion HistoryDealGetString()

ENUM_DEAL_PROPERTY_STRING

Identifikator	Beschreibung	Typ
DEAL_SYMBOL	Dealssymbol	string
DEAL_COMMENT	Kommentar zum Deal	string
DEAL_EXTERNAL_ID	Identifikator des Deals im Außenhandelssystem (an der Börse)	string

© 2000-2025, MetaQuotes Ltd.

Jeder Deal wird durch Typ charakterisiert, zulaessige Werte werden in der Enumeration ENUM_DEAL_TYPE gegeben. Für Erhlatung der Information über den Typ des Deals verwenden Sie die Funktion HistoryDealGetInteger() mit dem Modifikator DEAL_TYPE.

ENUM_DEAL_TYPE

Identifikator	Beschreibung
DEAL_TYPE_BUY	Kauf
DEAL_TYPE_SELL	Verkauf
DEAL_TYPE_BALANCE	Saldo
DEAL_TYPE_CREDIT	Gutschrift
DEAL_TYPE_CHARGE	Lastschrift
DEAL_TYPE_CORRECTION	Korrektur
DEAL_TYPE_BONUS	Enumeration der Boni
DEAL_TYPE_COMMISSION	Zusätzliche Kommissionen
DEAL_TYPE_COMMISSION_DAILY	Kommission, die am Ende des Handelstages angerechnet wird
DEAL_TYPE_COMMISSION_MONTHLY	Kommission, die am Ende des Monats angerechnet wird
DEAL_TYPE_COMMISSION_AGENT_DAILY	Agentenkommission, die am Ende des Handelstages angerechnet wird
DEAL_TYPE_COMMISSION_AGENT_MONTHLY	Agentenkommission, die am Ende des Monats angerechnet wird
DEAL_TYPE_INTEREST	Anrechnungen von Zinsen auf freie Mittel
DEAL_TYPE_BUY_CANCELED	Abgebrochener Kaufdeal. Es kann eine Situation sein, wenn zuvor abgeschlossener Kaufdeal abgebrochen wird. In solchem Fall ändert sich der Typ des früher abgeschlossenen Deals (DEAL_TYPE_BUY) aufDEAL_TYPE_BUY_CANCELED, und seiner Gewinn/Verlust wird auf Null gesetzt. Zuvor erhaltener Gewinn/Verlust wird auf das Konto mittels gesonderten Bilanzoperation angerechnet/abgebucht
DEAL_TYPE_SELL_CANCELED	Abgebrochener Verkaufdeal. Es kann eine Situation sein, wenn zuvor abgeschlossener Verkaufdeal abgebrochen wird. In solchem Fall ändert sich der Typ des zuvor abgeschlossenen Deals (DEAL_TYPE_SELL) aufDEAL_TYPE_SELL_CANCELED, und seiner Gewinn/Verlust wird auf Null gesetzt. Zuvor erhaltener Gewinn/Verlust wird auf das Konto mittels gesonderten Bilanzoperation angerechnet/abgebucht
DEAL_DIVIDEND	Anrechnung der Dividenden
DEAL_DIVIDEND_FRANKED	Anrechnung von "franked dividends" (unterliegen nicht der Quellenbesteuerung)

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
DEAL_TAX	Anrechnung der Steuer

Deals unterscheiden sich nicht nur durch den Typ, der in der Enumeration ENUM_DEAL_TYPE vorgegeben wird , sondern auch durch die Methode der Positionsveränderung. Das kann einfach Öffnung der Position sein oder Akkumulation der früher geöffneten Position sein (Markteingang), Positionsschliessen durch den gegensätzlichen Deal mit dem entsprechenden Volumen (Marktausgang) oder Kehrwendung der Position falls entgegengerichteten Deal das Volumen der früher geöffneten Position überdeckt.

Alle diesen Situationen werden durch Werte aus der Enumeration ENUM_DEAL_ENTRY beschrieben. Für die Erhaltung dieser Information über den Deal verwenden Sie die Funktion HistoryDealGetInteger() mit dem Modifikator DEAL_ENTRY.

ENUM_DEAL_ENTRY

Identifikator	Beschreibung
DEAL_ENTRY_IN	Markteingang
DEAL_ENTRY_OUT	Marktausgang
DEAL_ENTRY_INOUT	Kehrwendung
DEAL_ENTRY_OUT_BY	Schließen zur Gegenposition

Die Eigenschaft DEAL_REASON beinhaltet den Grund der Ausführung eines Abschlusses. Ein Abschluss kann infolge der Auslösung einer in einer mobilen Anwendung oder in einem MQL5-Ptogramm platzierten Order, infolge des Ereignisses StopOut oder infolge der Anrechnung/Abbuchung der Variation Margin usw. ausgeführt werden. Die möglichen Werte von DEAL_REASON sind in der Aufzählung ENUM_DEAL_REASON beschrieben. Für non-trading Abschlüsse, die infolge Änderungen des Kontostandes, Kreditoperationen, Kommissionen usw. entstehen, wird als Grund DEAL_REASON_CLIENT angegeben.

ENUM_DEAL_REASON

Identifier	Beschreibung
DEAL_REASON_CLIENT	Der Abschluss wurde infolge der Auslösung einer Order ausgeführt, die in einem Desktop-Terminal platziert wurde
DEAL_REASON_MOBILE	Der Abschluss wurde infolge der Auslösung einer Order ausgeführt, die in einer mobilen Anwendung platziert wurde
DEAL_REASON_WEB	Der Abschluss wurde infolge der Auslösung einer Order ausgeführt, die auf der Webplattform platziert wurde
DEAL_REASON_EXPERT	Der Abschluss wurde infolge der Auslösung einer Order ausgeführt, die durch ein MQL5-Programm - einen Expert Advisor oder ein Script platziert wurde

© 2000-2025, MetaQuotes Ltd.

Identifier	Beschreibung
DEAL_REASON_SL	Der Abschluss wurde infolge der Auslösung von Stop Loss ausgeführt
DEAL_REASON_TP	Der Abschluss wurde infolge der Auslösung von Take Profit ausgeführt
DEAL_REASON_SO	Der Abschluss wurde infolge des Ereignisses Stop Out ausgeführt
DEAL_REASON_ROLLOVER	Der Abschluss wurde infolge der Verschiebung einer Position ausgeführt
DEAL_REASON_VMARGIN	Der Abschluss wurde infolge der Anrechnung/Abbuchung der Variation Margin ausgeführt
DEAL_REASON_SPLIT	Der Abschluss wurde infolge eines Splits (Preissenkung) eines Symbols, auf welchem im Moment des Splits eine offene Position vorhanden war
DEAL_REASON_CORPORATE_ACTION	Der Deal wurde als Folge einer Unternehmensmaßnahme durchgeführt: Zusammenlegung oder Umbenennung eines Wertpapiers, Übertragung eines Kunden auf ein anderes Konto usw.

© 2000-2025, MetaQuotes Ltd.

Typen der Handelsoperationen

Handel wird durch Senden der Befehlsanweisungen, Positionen zu öffnen und auch Befehlsanweisungen, wartende Order einzustellen, modifizieren und entfernen unter Verwendung der Funktion OrderSend(). Jede Handelsorder deutet auf den Typ der angeforderten Handelsoperation. Handelsoperationen werden in der Enumeration ENUM_TRADE_REQUEST_ACTIONS beschrieben.

ENUM_TRADE_REQUEST_ACTIONS

Identifikator	Beschreibung
TRADE_ACTION_DEAL	Handelsorder auf den sofortigen Dealabschluss mit den angegebenen Parametern stellen (Marktorder stellen)
TRADE_ACTION_PENDING	Handelsorder für Dealabschluss bei den angegebenen Bedingungen (wartende Order) stellen
TRADE_ACTION_SLTP	die Werte Stop Loss und Take Profit der offenen Position verändern
TRADE_ACTION_MODIFY	Parameter der früher eingestellten Order verändern
TRADE_ACTION_REMOVE	Wartende Handelsserver entfernen
TRADE_ACTION_CLOSE_BY	Close a position by an opposite one

Ein Beispiel der Transaktion TRADE_ACTION_DEAL für das Eröffnen einer Buy-Position:

```
#define EXPERT_MAGIC 123456   // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Eröffnen einer Buy-Position                                      |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
//--- Parameter der Anfrage
   request.action   =TRADE_ACTION_DEAL;                     // Typ der Transaktion
   request.symbol   =Symbol();                              // Symbol
   request.volume   =0.1;                                   // Volumen von 0.1 Lot
   request.type     =ORDER_TYPE_BUY;                        // Ordertyp
   request.price    =SymbolInfoDouble(Symbol(),SYMBOL_ASK); // Eröffnungspreis
   request.deviation=5;                                     // zulässige Abweichung vo
   request.magic    =EXPERT_MAGIC;                          // MagicNumber der Order
//--- Anfrage senden
   if(!OrderSend(request,result))
      PrintFormat("OrderSend error %d",GetLastError());     // wenn die Anfrage konnte
//--- Details zur Transaktion
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
  }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transaktion TRADE_ACTION_DEAL für das Eröffnen einer Sell-Position:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456   // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Eröffnen einer Sell-Position                                     |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
//--- Parameter der Anfrage
   request.action   =TRADE_ACTION_DEAL;                     // Typ der Transaktion
   request.symbol   =Symbol();                              // Symbol
   request.volume   =0.2;                                   // Volumen von 0.2 Lot
   request.type     =ORDER_TYPE_SELL;                       // Ordertyp
   request.price    =SymbolInfoDouble(Symbol(),SYMBOL_BID); // Eröffnungspreis
   request.deviation=5;                                     // zulässige Abweichung vo
   request.magic    =EXPERT_MAGIC;                          // MagicNumber der Order
//--- Anfrage senden
   if(!OrderSend(request,result))
      PrintFormat("OrderSend error %d",GetLastError());     // wenn die Anfrage konnte
//--- Details zur Transaktion
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
  }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transaktion TRADE_ACTION_DEAL für das Schließen von Positionen:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456   // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Schließen aller Position                                         |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren
   MqlTradeRequest request;
   MqlTradeResult  result;
   int total=PositionsTotal(); // Anzahl offener Positionen
//--- in allen offenen Positionen suchen
   for(int i=total-1; i>=0; i--)
     {
      //--- Parameter der Order
      ulong  position_ticket=PositionGetTicket(i);
      string position_symbol=PositionGetString(POSITION_SYMBOL);
      int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS);
      ulong  magic=PositionGetInteger(POSITION_MAGIC);
      double volume=PositionGetDouble(POSITION_VOLUME);
      ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
      //--- Details zur Position anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s [%I64d]",
                  position_ticket,
                  position_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  magic);
      //--- wenn MagicNumber übereinstimmt
      if(magic==EXPERT_MAGIC)
        {
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
         //--- Parameter der Transaktion setzen
         request.action   =TRADE_ACTION_DEAL;        // Typ der Transaktion
         request.position =position_ticket;          // das Ticket der Position
         request.symbol   =position_symbol;          // Symbol
         request.volume   =volume;                   // das Volumem der Position
         request.deviation=5;                        // zulässige Abweichung vom Preis
         request.magic    =EXPERT_MAGIC;             // MagicNumber der Position
         //--- den Preis und den Ordertyp je nach dem Positionstyp setzen
         if(type==POSITION_TYPE_BUY)
           {
            request.price=SymbolInfoDouble(position_symbol,SYMBOL_BID);
            request.type =ORDER_TYPE_SELL;
           }
         else
           {
            request.price=SymbolInfoDouble(position_symbol,SYMBOL_ASK);
            request.type =ORDER_TYPE_BUY;
           }
         //--- Informationen über das Schließen anzeigen
         PrintFormat("Close #%I64d %s %s",position_ticket,position_symbol,EnumToString
         //--- Senden der Anfrage
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kor
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
         //---
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transaktion TRADE_ACTION_PENDING für das Setzen einer Pending Order:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Ein Beispiel für das Platzieren von Pending Orders"
#property script_show_inputs
#define EXPERT_MAGIC 123456                                  // MagicNumber des Expert Advi
input ENUM_ORDER_TYPE orderType=ORDER_TYPE_BUY_LIMIT;   // Ordertyp
//+------------------------------------------------------------------+
//| Pending Orders setzen                                            |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
//--- Parameter für das Platzieren einer Pending Order
   request.action   =TRADE_ACTION_PENDING;                                  // Typ der Tran
   request.symbol   =Symbol();                                              // Symbol
   request.volume   =0.1;                                                   // das Volumen
   request.deviation=2;                                                     // zulässige Ab
   request.magic    =EXPERT_MAGIC;                                          // MagicNumber
   int offset = 50;                                                         // Abweichung v
   double price;                                                            // Preis, bei d
   double point=SymbolInfoDouble(_Symbol,SYMBOL_POINT);                     // Punktgröße
   int digits=SymbolInfoInteger(_Symbol,SYMBOL_DIGITS);                     // Stellen nach
   //--- den Typ der Transaktion überprüfen
   if(orderType==ORDER_TYPE_BUY_LIMIT)
     {
      request.type     =ORDER_TYPE_BUY_LIMIT;                              // Ordertyp
      price=SymbolInfoDouble(Symbol(),SYMBOL_ASK)-offset*point;            // Eröffnungsku
      request.price    =NormalizeDouble(price,digits);                     // normierter P
     }
   else if(orderType==ORDER_TYPE_SELL_LIMIT)
     {
      request.type     =ORDER_TYPE_SELL_LIMIT;                              // Ordertyp
      price=SymbolInfoDouble(Symbol(),SYMBOL_ASK)+offset*point;            // Eröffnungsk
      request.price    =NormalizeDouble(price,digits);                     // normierter
     }
   else if(orderType==ORDER_TYPE_BUY_STOP)
     {
      request.type =ORDER_TYPE_BUY_STOP;                                   // Ordertyp
      price        =SymbolInfoDouble(Symbol(),SYMBOL_ASK)+offset*point;    // Eröffnungsk
      request.price=NormalizeDouble(price,digits);                         // normierter
     }
   else if(orderType==ORDER_TYPE_SELL_STOP)
     {
      request.type     =ORDER_TYPE_SELL_STOP;                              // Ordertyp
      price=SymbolInfoDouble(Symbol(),SYMBOL_ASK)-offset*point;            // Eröffnungsp
      request.price    =NormalizeDouble(price,digits);                     // normierter
     }
   else Alert("Dieses Beispiel gilt nur für das Platzieren von Pending Orders");   //
//--- Senden einer Anfrage
   if(!OrderSend(request,result))
      PrintFormat("OrderSend error %d",GetLastError());                    // wenn die An
//--- Details zur Transaktion
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
  }
//+------------------------------------------------------------------+
```

Ein Beispiel für die Transaktion TRADE_ACTION_SLTP für die Änderung von Stop Loss und Take Profit bei einer offenen Position:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456  // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Stop Loss und Take Profit modifizieren                           |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren
   MqlTradeRequest request;
   MqlTradeResult  result;
   int total=PositionsTotal(); // Anzahl offener Positionen
//--- in allen offenen Positionen suchen
   for(int i=0; i<total; i++)
     {
      //--- Parameter der Order
      ulong  position_ticket=PositionGetTicket(i);// das Ticket der Position
      string position_symbol=PositionGetString(POSITION_SYMBOL); // Symbol
      int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS); // Stellen
      ulong  magic=PositionGetInteger(POSITION_MAGIC); // MagicNumber der Position
      double volume=PositionGetDouble(POSITION_VOLUME);    // Volumen der Position
      double sl=PositionGetDouble(POSITION_SL);  // Stop Loss der Position
      double tp=PositionGetDouble(POSITION_TP);  // Take Profit der Position
      ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
      //--- Details zur Position anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s  sl: %s  tp: %s  [%I64d]",
                  position_ticket,
                  position_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  DoubleToString(sl,digits),
                  DoubleToString(tp,digits),
                  magic);
      //--- wenn die MagicNumber übereinstimmt, sind Stop Loss und Take Profit nicht g
      if(magic==EXPERT_MAGIC && sl==0 && tp==0)
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- aktuelle Preis Levels berechnen
         double price=PositionGetDouble(POSITION_PRICE_OPEN);
         double bid=SymbolInfoDouble(position_symbol,SYMBOL_BID);
         double ask=SymbolInfoDouble(position_symbol,SYMBOL_ASK);
         int    stop_level=(int)SymbolInfoInteger(position_symbol,SYMBOL_TRADE_STOPS_L
         double price_level;
         //--- wenn der Level der minimalen Abweichung vom aktuellen Schlusskurs in Pu
         if(stop_level<=0)
            stop_level=150; // setzen wir eine Abweichung von 150 Punkten vom aktuelle
         else
            stop_level+=50; // sicherheitshalber nehmen wir (SYMBOL_TRADE_STOPS_LEVEL

         //--- Stop Loss und Take Profit berechnen und abrunden
         price_level=stop_level*SymbolInfoDouble(position_symbol,SYMBOL_POINT);
         if(type==POSITION_TYPE_BUY)
           {
            sl=NormalizeDouble(bid-price_level,digits);
            tp=NormalizeDouble(ask+price_level,digits);
           }
         else
           {
            sl=NormalizeDouble(ask+price_level,digits);
            tp=NormalizeDouble(bid-price_level,digits);
           }
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
         //--- Parameter der Transaktion setzen
         request.action  =TRADE_ACTION_SLTP; // Typ der Transaktion
         request.position=position_ticket;   // das Ticket der Position
         request.symbol=position_symbol;     // Symbol
         request.sl      =sl;                // Stop Loss der Position
         request.tp      =tp;                // Take Profit der Position
         request.magic=EXPERT_MAGIC;         // MagicNumber der Position
         //--- Informationen über die Modifizierung anzeigen
         PrintFormat("Modify #%I64d %s %s",position_ticket,position_symbol,EnumToStrin
         //--- Senden einer Anfrage
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kon
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
        }
     }
   }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transkation TRADE_ACTION_MODIFY für das Modifizieren von Preis-Levels von Pending Orders:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456  // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Modifizieren von Pending Orders                                  |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
   int total=OrdersTotal(); // Anzahl platzierter Pending Orders
//--- in allen gesetzten Pending Orders suchen
   for(int i=0; i<total; i++)
     {
      //--- Parameter der Order
      ulong  order_ticket=OrderGetTicket(i);                                  // das Ticket
      string order_symbol=Symbol();                                           // Symbol
      int    digits=(int)SymbolInfoInteger(order_symbol,SYMBOL_DIGITS);       // Stellen na
      ulong  magic=OrderGetInteger(ORDER_MAGIC);                              // MagicNumbe
      double volume=OrderGetDouble(ORDER_VOLUME_CURRENT);                     // das aktuel
      double sl=OrderGetDouble(ORDER_SL);                                     // der aktuel
      double tp=OrderGetDouble(ORDER_TP);                                     // der aktuel
      ENUM_ORDER_TYPE type=(ENUM_ORDER_TYPE)OrderGetInteger(ORDER_TYPE); // Ordertyp
      int offset = 50;                                                        // Abweichung
      double price;                                                           // Preis der
      double point=SymbolInfoDouble(order_symbol,SYMBOL_POINT);               // Punktgröße
      //--- Details zur Order anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s  sl: %s  tp: %s  [%I64d]",
                  order_ticket,
                  order_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  DoubleToString(sl,digits),
                  DoubleToString(tp,digits),
                  magic);
      //--- wenn die MagicNumber übereinstimmt, sind Stop Loss und Take Profit nicht s
      if(magic==EXPERT_MAGIC && sl==0 && tp==0)
        {
         request.action=TRADE_ACTION_MODIFY;                                  // Typ der Tran
         request.order = OrderGetTicket(i);                                   // das Ticket d
         request.symbol   =Symbol();                                          // Symbol
         request.deviation=5;                                                 // zulässige Ab
        //--- Preis-Level, Take Profit und Stop Loss der Order je nach Ordertyp setzen
         if(type==ORDER_TYPE_BUY_LIMIT)
           {
            price = SymbolInfoDouble(Symbol(),SYMBOL_ASK)-offset*point;
            request.tp = NormalizeDouble(price+offset*point,digits);
            request.sl = NormalizeDouble(price-offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                  // normier
           }
         else if(type==ORDER_TYPE_SELL_LIMIT)
           {
           price = SymbolInfoDouble(Symbol(),SYMBOL_BID)+offset*point;
            request.tp = NormalizeDouble(price-offset*point,digits);
            request.sl = NormalizeDouble(price+offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                   // normie
           }
         else if(type==ORDER_TYPE_BUY_STOP)
           {
           price = SymbolInfoDouble(Symbol(),SYMBOL_BID)+offset*point;
            request.tp = NormalizeDouble(price+offset*point,digits);
```

© 2000-2025, MetaQuotes Ltd.

```
            request.sl = NormalizeDouble(price-offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                  // normie
           }
         else if(type==ORDER_TYPE_SELL_STOP)
           {
           price = SymbolInfoDouble(Symbol(),SYMBOL_ASK)-offset*point;
            request.tp = NormalizeDouble(price-offset*point,digits);
            request.sl = NormalizeDouble(price+offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                  // normie
           }
         //--- Anfrage senden
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kor
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
        }
     }
  }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transaktion TRADE_ACTION_REMOVE für das Löschen von Pending Orders:

```
#define EXPERT_MAGIC 123456  // MagicNumber des Expert Advisors
//+------------------------------------------------------------------+
//| Pending Orders löschen                                           |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
   int total=OrdersTotal(); // Anzahl platzierter Pending Orders
//--- in allen platzierten Pending Orders suchen
   for(int i=total-1; i>=0; i--)
     {
      ulong  order_ticket=OrderGetTicket(i);                   // das Ticket der Orde
      ulong  magic=OrderGetInteger(ORDER_MAGIC);               // MagicNumber der Orde
      //--- wenn die MagicNumber übereinstimmt
      if(magic==EXPERT_MAGIC)
        {
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
         //--- Parameter der Transaktion setzen
         request.action=TRADE_ACTION_REMOVE;                   // Typ der Transaktion
         request.order = order_ticket;                         // das Ticket der Orde
         //--- Anfrage senden
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kor
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
        }
     }
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

Ein Beispiel der Transaktion TRADE_ACTION_CLOSE_BY für das Schließen zur Gegenposition:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456  // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Alle Positionen zu Gegenpositionen schließen                     |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren
   MqlTradeRequest request;
   MqlTradeResult  result;
   int total=PositionsTotal(); // Anzahl offener Positionen
//--- in allen offenen Positionen suchen
   for(int i=total-1; i>=0; i--)
     {
      //--- Parameter der Order
      ulong  position_ticket=PositionGetTicket(i);
      string position_symbol=PositionGetString(POSITION_SYMBOL);
      int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS);
      ulong  magic=PositionGetInteger(POSITION_MAGIC);
      double volume=PositionGetDouble(POSITION_VOLUME);
      double sl=PositionGetDouble(POSITION_SL);
      double tp=PositionGetDouble(POSITION_TP);
      ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
      //--- Details zur Position anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s  sl: %s  tp: %s  [%I64d]",
                  position_ticket,
                  position_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  DoubleToString(sl,digits),
                  DoubleToString(tp,digits),
                  magic);
      //--- wenn die MagicNumber übereinstimmt
      if(magic==EXPERT_MAGIC)
        {
         for(int j=0; j<i; j++)
           {
            string symbol=PositionGetSymbol(j); // das Symbol der Gegenposition
            //--- wenn das Symbol der Gegenposition mit dem Symbol der gesuchten übere
            if(symbol==position_symbol && PositionGetInteger(POSITION_MAGIC)==EXPERT_M
              {
               //--- Typ der Gegenposition setzen
               ENUM_POSITION_TYPE type_by=(ENUM_POSITION_TYPE)PositionGetInteger(POSIT
               //--- beenden, wenn die Typen der Positionen gleich sind
               if(type==type_by)
                  continue;
               //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
               ZeroMemory(request);
               ZeroMemory(result);
               //--- Parameter der Transaktion setzten
               request.action=TRADE_ACTION_CLOSE_BY;                    // Typ de
               request.position=position_ticket;                        // Ticket
               request.position_by=PositionGetInteger(POSITION_TICKET); // Ticket
               //request.symbol     =position_symbol;
               request.magic=EXPERT_MAGIC;                              // MagicN
               //--- Information über das Schließen zur Gegenposition anzeigen
               PrintFormat("Close #%I64d %s %s by #%I64d",position_ticket,position_sym
               //--- Anfrage senden
               if(!OrderSend(request,result))
                  PrintFormat("OrderSend error %d",GetLastError()); // wenn die Anfrag
```

© 2000-2025, MetaQuotes Ltd.

```
                //--- Details zur Transaktion
                PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result
               }
            }
         }
      }
   }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

Typen der Handelstransaktionen

Als Ergebnis der Durchführung von bestimmten Aktionen auf dem Handelskonto, ändert sich der Kontostand. Zu solchen Aktionen gehören:

- Senden der Handelsanfrage aus einer beliebigen MQL5-Anwendung in dem Client-Terminal mittels der Funktionen OrderSend und OrderSendAsync und ihre nachfolgende Ausführung;
- Senden der Handelsanfrage über die grafische Oberfläche des Terminals und ihre nachfolgende Ausführung;
- Auslösen von aufgeschobenen Ordern und Stop-Ordern auf dem Server;
- Ausführen von Operationen auf der Seite der Handel-Server.

Als Ergebnis dieser Aktionen werden die folgenden Handelstransaktionen durchgeführt:

- Die Verarbeitung der Handelsanfrage;
- Änderung der offenen Aufträge;
- Änderung der Ordergeschichte;
- Änderung der Dealgeschichte;
- Änderung der Position.

Zum Beispiel, beim Senden Marktkauforder, wird Marktkauforder behandelt und eine entsprechende Marktkauforder für das Konto erstellt. Ist die Order ausgeführt, wird sie aus der Auftragsliste entfernt und zur Ordergeschichte hinzugefügt. Dann wird der entsprechende Deal zur Geschichte hinzugefügt und eine neue Position wird erstellt. Alle diese Aktionen sind Handelstransaktionen.

Damit ein Programmierer die Aktionen verfolgen konnte, die bezüglich eines Handelskontos ausgeführt werden, ist die Funktion OnTradeTransaction vorgesehen. Mit Hilfe dieses Handlers kann man die auf das Konto angewendeten Handelstransaktionen im MQL5-Anwendung bekommen. Beschreibung der Handelstransaktion wird im ersten Parameter OnTradeTransaction mit Hilfe der Struktur MqlTradeTransaction übergeben.

Typ der Handelstransaktion wird im type Parameter der Struktur MqlTradeTransaction übergeben. Möglichen Typen der Handelstransaktionen werden von der folgenden Enumeration beschrieben:

ENUM_TRADE_TRANSACTION_TYPE

Identifikator	Beschreibung
TRADE_TRANSACTION_ORDER_ADD	Hinzufügen von einer neuen offenen Order.
TRADE_TRANSACTION_ORDER_UPDATE	Änderung der offenen Order. Zu solchen Änderungen gehören nicht nur die offensichtlichen Änderungen seitens des Client-Terminals oder Handel-Servers, sondern auch die Änderung ihres Zustandes bei der Ausstellung (zum Beispiel, Übergang vom Zustand ORDER_STATE_STARTED im ORDER_STATE_PLACED oder vom ORDER_STATE_PLACED im ORDER_STATE_PARTIAL usw.).
TRADE_TRANSACTION_ORDER_DELETE	Entfernung der Order aus der offenen Auftragsliste. Eine Order kann aus der offenen Auftragsliste als Ergebnis der Stellung der entsprechenden Anfrage

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
	oder als Ergebnis der Durchführung (Füllung) und Übertragung in die Geschichte entfernt werden.
TRADE_TRANSACTION_DEAL_ADD	Hinzufügen des Deals zur Geschichte. Es erfolgt als Ergebnis der Orderausfüllung oder Durchführung der Operationen mit einem Kontostand.
TRADE_TRANSACTION_DEAL_UPDATE	Änderung des Deals in der Geschichte. Es kann eine Situation sein, wenn zuvor ausgeführter Deal auf dem Server geändert wird. Zum Beispiel, ein Deal war im externen Handelssystem (die Börse) geändert, wohin er vom Broker übertragen wurde.
TRADE_TRANSACTION_DEAL_DELETE	Entfernung der Order aus der Geschichte. Es kann eine Situation sein, wenn zuvor ausgeführter Deal vom Server entfernt wird. Zum Beispiel, ein Deal war vom externen Handelssystem (die Börse) entfernt, wohin er vom Broker übertragen wurde.
TRADE_TRANSACTION_HISTORY_ADD	Hinzufügen der Order zur Geschichte als Ergebnis der Ausführung oder Stornierung.
TRADE_TRANSACTION_HISTORY_UPDATE	Änderung der Order, die sich in der Ordergeschichte befindet. Dieser Typ ist für die Erweiterung der Funktionalität auf der Seite des Handel-Servers vorgesehen.
TRADE_TRANSACTION_HISTORY_DELETE	Entfernung der Order aus der Ordergeschichte. Dieser Typ ist für die Erweiterung der Funktionalität auf der Seite des Handel-Servers vorgesehen.
TRADE_TRANSACTION_POSITION	Änderung der Position, die mit der Ausführung des Deals nicht verbunden ist. Dieser Typ der Transaktion zeigt, dass die Position auf der Seite des Handel-Servers geändert wurde. Volumen, Eröffnungspreis sowie Stop Loss und Take Profit Ebenen von Position können geändert werden. Informationen über Änderungen werden in der Struktur MqlTradeTransaction durch den Händler OnTradeTransaction übergeben. Änderung der Position (Hinzufügen, Änderung oder Liquidation) als Ergebnis des Abschlusses des Deals zieht kein Erscheinen der Transaktion TRADE_TRANSACTION_POSITION nach sich.
TRADE_TRANSACTION_REQUEST	Benachrichtigung, dass die Handelsanfrage vom Server bearbeitet ist und das Ergebnis seiner Bearbeitung empfangen ist. Für Transaktionen dieses Typs in der Struktur MqlTradeTransaction muss nur type Feld (Typ der Transaktion) analysiert werden. Für zusätzliche Informationen muss man die zweiten

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
	und dritten Parameter der Funktion OnTradeTransaction (request und result) analysieren.

Verschiedene Parameter werden je nach Typ der Handelstransaktion in der Struktur MqlTradeTransaction, die sie beschreibt, ausgefüllt. Die ausführliche Beschreibung der übertragenen Daten ist im Abschnitt "Struktur der Handelstransaktion" gegeben.

Sehen Sie auch

Struktur der Handelstransaktion, OnTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Handelsordern in DOM

für Boersensymbole ist das Fenster "Auftragswarteschlange", zugaenglich, wo laufende Kaufs- und Verkaufsordern zu sehen sind. Für jede Order ist die erwuenschte Richtung der Handelsoperation, das erforderliche Volumen und der angeforderte Preis zu sehen.

für Erhaltung der Information über laufendes DOM durch Mittel der Sprache MQL5 ist die Funktion MarketBookGet() bestimmt, die "screen shot" ins Feld der Strukturen MqlBookInfo stellt. Jedes Element dieses Feldes hat im Feld *type* die Information über die Richtung der Order – der Wert aus der Enumeration ENUM_BOOK_TYPE.

ENUM_BOOK_TYPE

Identifikator	Beschreibung
BOOK_TYPE_SELL	Verkaufsorder
BOOK_TYPE_BUY	Kaufsorder
BOOK_TYPE_SELL_MARKET	Verkaufsorder mit Marktpreis
BOOK_TYPE_BUY_MARKET	Kaufsorder mit Marktpreis

Sehen Sie auch

Strukturen und Klassen, DOM Struktur, Typen der Handelsoperationen, Marktinformation erhalten

© 2000-2025, MetaQuotes Ltd.

Eingenschaften von Signalen

Werte der Aufzählungen für das Arbeiten mit Handelssignalen und Einstellungen für Copy Trading.

Aufzählungen der Eigenschaften von Handelssignalen vom Typ double:

ENUM_SIGNAL_BASE_DOUBLE

Konstante	Beschreibung
SIGNAL_BASE_BALANCE	Kontostand
SIGNAL_BASE_EQUITY	Equity
SIGNAL_BASE_GAIN	Wachstum in Prozent
SIGNAL_BASE_MAX_DRAWDOWN	Maximaler Drawdown
SIGNAL_BASE_PRICE	Abo-Preis
SIGNAL_BASE_ROI	ROI (Return on Investment) des Signals in %

Aufzählungen der Eigenschaften von Handelssignalen vom Typ integer:

ENUM_SIGNAL_BASE_INTEGER

Konstante	Beschreibung
SIGNAL_BASE_DATE_PUBLISHED	Veröffentlichungsdatum des Signals (wann wurde es zum Abonnieren freigegeben)
SIGNAL_BASE_DATE_STARTED	Anfangsdatum der Beobachtung des Signals
SIGNAL_BASE_DATE_UPDATED	Datum der letzten Aktualisierung der Handelsstatistik des Signals
SIGNAL_BASE_ID	Signal-ID
SIGNAL_BASE_LEVERAGE	Hebel
SIGNAL_BASE_PIPS	Handelsergebnis in Pips
SIGNAL_BASE_RATING	Position im Ranking der Signale
SIGNAL_BASE_SUBSCRIBERS	Abonnentenzahl
SIGNAL_BASE_TRADES	Anzahl von Trades
SIGNAL_BASE_TRADE_MODE	Kontotyp (0-Realkonto, 1-Demokonto, 2-Wettbewerbskonto)

Aufzählung der Eigenschaften von Handelssignalen vom Typ string:

ENUM_SIGNAL_BASE_STRING

Konstante	Beschreibung
SIGNAL_BASE_AUTHOR_LOGIN	Benutzername des Signalgebers

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung
SIGNAL_BASE_BROKER	Name des Brokers (des Unternehmens)
SIGNAL_BASE_BROKER_SERVER	Server des Brokers
SIGNAL_BASE_NAME	Name des Signals
SIGNAL_BASE_CURRENCY	Währung des Signalkontos

Aufzählung der Eigenschaften von Einstellungen für das Kopieren von Handelssignalen vom Typ double:

ENUM_SIGNAL_INFO_DOUBLE

Konstante	Beschreibung
SIGNAL_INFO_EQUITY_LIMIT	Equity Limit
SIGNAL_INFO_SLIPPAGE	Slippage, mit welchem Marktorders bei der Synchronisierung von Positionen und beim Kopieren von Trades platziert werden
SIGNAL_INFO_VOLUME_PERCENT	Begrenzung von Equity für Signale, r/o

Aufzählungen der Eigenschaften von Einstellungen für das Kopieren von Handelssignalen vom Typ integer:

ENUM_SIGNAL_INFO_INTEGER

Konstante	Beschreibung
SIGNAL_INFO_CONFIRMATIONS_DISABLED	Flag der Erlaubnis der Synchronisierung ohne Anzeige des Bestätigungsdialogs
SIGNAL_INFO_COPY_SLTP	Flag des Kopierens von Stop Loss und Take Profit
SIGNAL_INFO_DEPOSIT_PERCENT	Begrenzung nach der Einlage (in %)
SIGNAL_INFO_ID	Signal-ID, r/o
SIGNAL_INFO_SUBSCRIPTION_ENABLED	Flag der Erlaubnis für das Kopieren von Trades nach dem Abonnement
SIGNAL_INFO_TERMS_AGREE	Flag der Akzeptierung der Nutzungsbedingungen des Signal-Services, r/o

Aufzählungen der Eigenschaften von Handelssignalen vom Typ string :

ENUM_SIGNAL_INFO_STRING

Konstante	Beschreibung
SIGNAL_INFO_NAME	Name des Signals, r/o

Siehe auch

Handelssignale

© 2000-2025, MetaQuotes Ltd.

Benannte Konstanten

Alle in der Sprache MQL5 verwendeten Konstanten können in folgende Gruppen aufgeteilt werden:

- Vorbestimmte Makrosubstitutionen - Werte werden in der Compilierungsetappe substituiert;
- Mathematische Konstanten - Werte einiger mathematischer Ausdrücke;
- Konstanten der numerischen Typen - Grenzbedingungen, die in einfache Typen gesteckt werden;
- Deinitialisierungsgruende - Beschreibung der Deinitialisierungsgruende;
- Pruefung des Objektanzeigers - Enumeration der Anzeigertypen, die durch die Funktion CheckPointer() rückgegeben werden ;
- Andere Konstanten - alles, was ausser anderer Konstantengruppen bleibt.

© 2000-2025, MetaQuotes Ltd.

Vorbestimmte Makrosubstitutionen

für die Erleichterung des ebugging und der Informationserhaltung über die Arbeit des mql5-Programms werden spezielle Konstanten-Makros eingeführt, deren Werte im Augenblick der Compilierung vorgegeben werden. Das einfachste Verfahren von Verwendung dieser Konstanten - Ausgabe der Werte mit der Funktion Print(), wie es im Beispiel gezeigt wird.

Konstante	Beschreibung
__CPU_ARCHITECTURE__	Name der Architektur (Befehlssatz), für den die EX5-Datei kompiliert wurde.
__DATE__	Datum der Kompilierung einer Date ohne die Uhrzeit (Stunden, Minuten und Sekunden sind 0)
__DATETIME__	Datum und Uhrzeit der Kompilierung einer Date
__LINE__	Nummer der Zeile im Ausgangskode, auf der sich der Makro befindet
__FILE__	Name der laufenden kompilierten Datei
__PATH__	Absoluter Pfad zu der laufenden kompilierten Datei
__FUNCTION__	Funktionsname, in deren Körper sich das Makro befindet
__FUNCSIG__	Signatur der Funktion, in deren Körper befindet sich das Makro. Protokollierung der vollständige Beschreibung der Funktionen kann bei der Identifizierung von überladenen Funktionen nützlich sein
__MQLBUILD__, __MQL5BUILD__	Buildnummer des Compilers
__COUNTER__	Der Compiler ersetzt für jede angetroffene Deklaration von __COUNTER__ den Zählerwert von 0 bis N-1, wobei N eine Anzahl von Verwendungen im Code ist. Der Befehl __COUNTER__ wird beim Neukompilieren des Quellcodes ohne Änderungen garantiert. Der Wert von __COUNTER__ wird auf folgende Weise berechnet: • der anfängliche Wert des Zählers ist 0, • nach jeder Verwendung des Zählers wird sein Wert um 1 erhöht, • zunächst expandiert der Compiler alle Makros und Templates lokal in Quellcode, • für jede spezielle Funktion eines Templates wird ein eigener Code erzeugt, • für jede spezielle Klasse/Struktur eines Templates wird ein separater Code erzeugt, • anschließend durchläuft der Compiler den erhaltenen Quellcode in dem definierten Auftrag und ersetzt jede erkannte Verwendung von __COUNTER__ durch den aktuellen Zählerwert.

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung
	Das Beispiel unten zeigt, wie der Compiler den Quellcode behandelt und alle Instanzen von __COUNTER__, auf die er trifft, durch sequentiell steigende Werte ersetzt.
__RANDOM__	Der Compiler fügt für jede Deklaration von __RANDOM__ einen zufälligen ulong-Wert ein.

Beispiel:

```
#property copyright "Copyright © 2009, MetaQuotes Software Corp."
#property link      "https://www.metaquotes.net"
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- Beispiel der Informationsausgabe bei der Ratgeberinitialisierung
   Print(" __FUNCTION__ = ",__FUNCTION__,"  __LINE__ = ",__LINE__);
//--- Einstellung des Intervals zwischen Ereignisse des Timers
   EventSetTimer(5);
//---
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Beispiel der Inforamtionsausgabe bei der Deinitialisierung des Ratgebers
   Print(" __FUNCTION__ = ",__FUNCTION__,"  __LINE__ = ",__LINE__);
//---
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {

//--- Ausgabe der Information beim Tickaufnahme
   Print(" __MQLBUILD__ = ",__MQLBUILD__,"  __FILE__ = ",__FILE__);
   Print(" __FUNCTION__ = ",__FUNCTION__,"  __LINE__ = ",__LINE__);
   test1(__FUNCTION__);
   test2();
//---
  }
//+------------------------------------------------------------------+
//| test1                                                            |
//+------------------------------------------------------------------+
void test1(string par)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Informationsausgabe innerhalb der Funktion
   Print(" __FUNCTION__ = ",__FUNCTION__," __LINE__ = ",__LINE__," par=",par);
 ;}
//+------------------------------------------------------------------+
//| test2                                                            |
//+------------------------------------------------------------------+
void test2()
  {
//--- Informationsausgabe innerhalb der Funktion
   Print(" __FUNCTION__ = ",__FUNCTION__," __LINE__ = ",__LINE__);
  }
//+------------------------------------------------------------------+
//| OnTimer event handler                                            |
//+------------------------------------------------------------------+
void OnTimer()
  {
//---
   Print(" __FUNCTION__ = ",__FUNCTION__," __LINE__ = ",__LINE__);
   test1(__FUNCTION__);
  }
```

Das Beispiel zur Verdeutlichung der Arbeit mit dem Makro __COUNTER__

```
//--- Erstellen eines Makro für eine schnelle Anzeige des Ausdrucks und seines Wertes
#define print(expr) Print(#expr,"=",expr)

//--- Definieren des nutzerdefinierten Makros MACRO_COUNTER mit Hilfe des vordefiniert
#define MACRO_COUNTER __COUNTER__

//--- den Eingabewerte der Variablen mit dem Makro __COUNTER__ setzen
input int InpVariable = __COUNTER__;

//--- Zuweisen des Wertes einer globalen Variablen mit dem Makro __COUNTER__ vor dem I
int ExtVariable = __COUNTER__;

//+------------------------------------------------------------------+
//| Die Funktion gibt den Wert von __COUNTER__ zurück                |
//+------------------------------------------------------------------+
int GlobalFunc(void)
  {
   return(__COUNTER__);
  }
//+------------------------------------------------------------------+
//| Das Funktionstemplate gibt den Wert von __COUNTER__ zurück       |
//+------------------------------------------------------------------+
template<typename T>
int GlobalTemplateFunc(void)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   return(__COUNTER__);
  }
//+------------------------------------------------------------------+
//| Die Struktur mit der Methode, die __COUNTER__ zurückgibt         |
//+------------------------------------------------------------------+
struct A
  {
   int               dummy;  // nicht verwendet

   int               Method(void)
     {
      return(__COUNTER__);
     }
  };
//+------------------------------------------------------------------+
//| Das Strukturtemplate mit der Methode, die __COUNTER__ zurückgibt |
//+------------------------------------------------------------------+
template<typename T>
struct B
  {
   int               dummy;  // nicht verwendet

   int               Method(void)
     {
      return(__COUNTER__);
     }
  };
//+------------------------------------------------------------------+
//| Die Struktur mit der Templatemethode, die __COUNTER__ zurückgibt |
//+------------------------------------------------------------------+
struct C
  {
   int               dummy;  // nicht verwendet

   template<typename T>
   int               Method(void)
     {
      return(__COUNTER__);
     }
  };
//+------------------------------------------------------------------+
//| die Funktion #2, die den Wert __COUNTER__ zurückgibt             |
//+------------------------------------------------------------------+
int GlobalFunc2(void)
  {
   return(__COUNTER__);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
// __COUNTER__ im Makro und den Variablen
   print(MACRO_COUNTER);
   print(InpVariable);
   print(ExtVariable);

//--- __COUNTER__ in den Funktionen
   print(GlobalFunc());
   print(GlobalFunc());                // der Wert hat sich nicht geändert
   print(GlobalTemplateFunc<int>());
   print(GlobalTemplateFunc<int>());   // der Wert hat sich nicht geändert
   print(GlobalTemplateFunc<double>());// der Wert hat sich nicht geändert
   print(GlobalFunc2());
   print(GlobalFunc2());               // der Wert hat sich nicht geändert

// __COUNTER__ in der Struktur
   A a1, a2;
   print(a1.Method());
   print(a2.Method());                 // der Wert hat sich nicht geändert

// __COUNTER__ im Strukturtemplate
   B<int> b1, b2;
   B<double> b3;
   print(b1.Method());
   print(b2.Method());                 // der Wert hat sich nicht geändert
   print(b3.Method());                 // der Wert hat sich geändert

// __COUNTER__ in der Struktur mit einer Templatefunktion
   C c1, c2;
   print(c1.Method<int>());
   print(c1.Method<double>());         // der Wert hat sich geändert
   print(c2.Method<int>());            // der gleiche Wert wie beim ersten Aufruf von

//--- Sehen wir uns noch einmal __COUNTER__ im Makro und die globale Variable an
   print(MACRO_COUNTER);  // der Wert hat sich geändert
   print(ExtGlobal2);
  }
//--- Wir setzen den Wert der globalen Variablen mit dem Makro __COUNTER__, nachdem wi
int ExtGlobal2 = __COUNTER__;
//+------------------------------------------------------------------+

/* Ergebnis
   __COUNTER__=3
   InpVariable=0
   ExtVariable=1
   GlobalFunc()=5
```

© 2000-2025, MetaQuotes Ltd.

```
   GlobalFunc()=5
   GlobalTemplateFunc<int>()=8
   GlobalTemplateFunc<int>()=8
   GlobalTemplateFunc<double>()=9
   GlobalFunc2()=7
   GlobalFunc2()=7
   a1.Method()=6
   a2.Method()=6
   b1.Method()=10
   b2.Method()=10
   b3.Method()=11
   c1.Method<int>()=12
   c1.Method<double>()=13
   c2.Method<int>()=12
   __COUNTER__=4
   ExtGlobal2=2

*/
```

© 2000-2025, MetaQuotes Ltd.

Mathematische Konstanten

für einige mathematische Funktionen werden spezielle Konstanten reserviert, die Werte enthalten. Diese Konstanten können in jeder Stelle des mql5-Programms statt der Berechnung ihrer Werte durch mathematische Funktionen.

Konstante	Beschreibung	Wert
M_E	e	2.71828182845904523536
M_LOG2E	log2(e)	1.44269504088896340736
M_LOG10E	log10(e)	0.43429448190325182765 1
M_LN2	ln(2)	0.69314718055994530941 7
M_LN10	ln(10)	2.30258509299404568402
M_PI	pi	3.14159265358979323846
M_PI_2	pi/2	1.57079632679489661923
M_PI_4	pi/4	0.78539816339744830961 6
M_1_PI	1/pi	0.31830988618379067153 8
M_2_PI	2/pi	0.63661977236758134307 6
M_2_SQRTPI	2/sqrt(pi)	1.12837916709551257390
M_SQRT2	sqrt(2)	1.41421356237309504880
M_SQRT1_2	1/sqrt(2)	0.70710678118654752440 1

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- geben wir Werte der Konstanten aus
   Print("M_E = ",DoubleToString(M_E,16));
   Print("M_LOG2E = ",DoubleToString(M_LOG2E,16));
   Print("M_LOG10E = ",DoubleToString(M_LOG10E,16));
   Print("M_LN2 = ",DoubleToString(M_LN2,16));
   Print("M_LN10 = ",DoubleToString(M_LN10,16));
   Print("M_PI = ",DoubleToString(M_PI,16));
   Print("M_PI_2 = ",DoubleToString(M_PI_2,16));
   Print("M_PI_4 = ",DoubleToString(M_PI_4,16));
   Print("M_1_PI = ",DoubleToString(M_1_PI,16));
   Print("M_2_PI = ",DoubleToString(M_2_PI,16));
   Print("M_2_SQRTPI = ",DoubleToString(M_2_SQRTPI,16));
   Print("M_SQRT2 = ",DoubleToString(M_SQRT2,16));
   Print("M_SQRT1_2 = ",DoubleToString(M_SQRT1_2,16));
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

Konstanten der numerischen Datentypen

Jeder einfache numerische Typ ist für gewisse Aufgaben bestimmt und ermöglicht Arbeit des mql5-Programms beim richtigen Verwenden zu optimieren. Für bessere Kodelesbarkeit und richtige Bearbeitung der Berechnungsergebnisse sind Konstanten eingeführt, die Informationen über Einschraenkungen zu bekommen ermöglichen, durch die einer oder anderer Typ einfacher Daten bechraenkt werden.

Konstante	Beschreibung	Wert
CHAR_MIN	Minimaler Wert, der durch den Typ char dargestellt werden kann	-128
CHAR_MAX	Maximaler Wert, der durch den Typ char dargestell werden kann	127
UCHAR_MAX	Maximaler Wert, der durch den Typ uchar dargestell werden kann	255
SHORT_MIN	Minimaler Wert, der durch den Typ short dargestellt werden kann	-32768
SHORT_MAX	Maximaler Wert, der durch den Typ short dargestellt werden kann	32767
USHORT_MAX	Maximaler Wert, der durch den Typ ushort dargestellt werden kann	65535
INT_MIN	Minimaler Wert, der durch den Typ int dargestellt werden kann int	-2147483648
INT_MAX	Maximaler Wert, der durch den Typ int dargestellt werden kann	2147483647
UINT_MAX	Maximaler Wert, der durch den Typ uint dargestellt werden kann	4294967295
LONG_MIN	Minimaler Wert, der durch den Typ long dargestellt werden kann	-9223372036854775808
LONG_MAX	Maximaler Wert, der durch den Typ long dargestellt werden kann	9223372036854775807
ULONG_MAX	Maximaler Wert, der durch den Typ ulong dargestellt werden kann	18446744073709551615
DBL_MIN	Minimaler positiver Wert, der durch den Typ double dargestellt werden kann	2.2250738585072014e-308
DBL_MAX	Maximaler Wert, der durch den Typ double dargestellt werden kann	1.7976931348623158e+308
DBL_EPSILON	Der minimale Wert, für den Die Bedingung 1.0+DBL_EPSILON != 1.0 erfuellt wird	2.2204460492503131e-016

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Wert
DBL_DIG	Anzahl der bedeutenden Dezimalzeichen	15
DBL_MANT_DIG	Anzahl der Bits in mantissa	53
DBL_MAX_10_EXP	Maximaler dezimaler Wert der Exponentenstufung	308
DBL_MAX_EXP	Maximaler binaerer Wert der Exponentenstufung	1024
DBL_MIN_10_EXP	Minimaler dezimaler Wert der Exponenetenstufung	(-307)
DBL_MIN_EXP	Minimaler binaerer Wert der Exponentenstufung	(-1021)
FLT_MIN	Minimaler positiver Wert, der durch den Typ float repraesentiert werden kann	1.175494351e-38
FLT_MAX	Maximaler Wert, der durch den Typ float repraesentiert werden kann	3.402823466e+38
FLT_EPSILON	Der minimale Wert, für den die Bedingung 1.0+FLT_EPSILON != 1.0 erfuellt wird	1.192092896e-07
FLT_DIG	Anzahl der bedeutenden Dezimalzeichen	6
FLT_MANT_DIG	Anzahl der Bits in mantissa	24
FLT_MAX_10_EXP	Maximaler dezimaler Wert der Exponentenstufung	38
FLT_MAX_EXP	Maximaler binaerer Wert der Exponentenstufung	128
FLT_MIN_10_EXP	Minimaler dezimaler Wert der Exponentenstufung	-37
FLT_MIN_EXP	Minimaler binaerer Wert der Exponentenstufung	(-125)

Beispiel:

```
void OnStart()
  {
//--- geben wir Werte der Konstanten aus
   printf("CHAR_MIN = %d",CHAR_MIN);
   printf("CHAR_MAX = %d",CHAR_MAX);
   printf("UCHAR_MAX = %d",UCHAR_MAX);
   printf("SHORT_MIN = %d",SHORT_MIN);
   printf("SHORT_MAX = %d",SHORT_MAX);
```

© 2000-2025, MetaQuotes Ltd.

```
   printf("USHORT_MAX = %d",USHORT_MAX);
   printf("INT_MIN = %d",INT_MIN);
   printf("INT_MAX = %d",INT_MAX);
   printf("UINT_MAX = %u",UINT_MAX);
   printf("LONG_MIN = %I64d",LONG_MIN);
   printf("LONG_MAX = %I64d",LONG_MAX);
   printf("ULONG_MAX = %I64u",ULONG_MAX);
   printf("EMPTY_VALUE = %.16e",EMPTY_VALUE);
   printf("DBL_MIN = %.16e",DBL_MIN);
   printf("DBL_MAX = %.16e",DBL_MAX);
   printf("DBL_EPSILON = %.16e",DBL_EPSILON);
   printf("DBL_DIG = %d",DBL_DIG);
   printf("DBL_MANT_DIG = %d",DBL_MANT_DIG);
   printf("DBL_MAX_10_EXP =  %d",DBL_MAX_10_EXP);
   printf("DBL_MAX_EXP = %d",DBL_MAX_EXP);
   printf("DBL_MIN_10_EXP = %d",DBL_MIN_10_EXP);
   printf("DBL_MIN_EXP = %d",DBL_MIN_EXP);
   printf("FLT_MIN = %.8e",FLT_MIN);
   printf("FLT_MAX = %.8e",FLT_MAX);
   printf("FLT_EPSILON = %.8e",FLT_EPSILON);
/*
   CHAR_MIN = -128
   CHAR_MAX = 127
   UCHAR_MAX = 255
   SHORT_MIN = -32768
   SHORT_MAX = 32767
   USHORT_MAX = 65535
   INT_MIN = -2147483648
   INT_MAX = 2147483647
   UINT_MAX = 4294967295
   LONG_MIN = -9223372036854775808
   LONG_MAX = 9223372036854775807
   ULONG_MAX = 18446744073709551615
   EMPTY_VALUE = 1.7976931348623157e+308
   DBL_MIN = 2.2250738585072014e-308
   DBL_MAX = 1.7976931348623157e+308
   DBL_EPSILON = 2.2204460492503131e-16
   DBL_DIG = 15
   DBL_MANT_DIG = 53
   DBL_MAX_10_EXP =  308
   DBL_MAX_EXP = 1024
   DBL_MIN_10_EXP = -307
   DBL_MIN_EXP = -1021
   FLT_MIN = 1.17549435e-38
   FLT_MAX = 3.40282347e+38
   FLT_EPSILON = 1.19209290e-07
*/
  }
```

© 2000-2025, MetaQuotes Ltd.

Deinitialisierungsgruende

Koden der Deinitialisierungsgruende von Expert, die durch die Funktion UninitializeReason() zurückgegeben werden. können jede der folgenden Werte haben:

Konstante	Wert	Beschreibung
REASON_PROGRAM	0	Expert beendet seine Operation durch Aufruf der Funktion ExpertRemove()
REASON_REMOVE	1	Das Programm ist entfernt vom Chart
REASON_RECOMPILE	2	Das Programm ist umcompiliert
REASON_CHARTCHANGE	3	Symbol oder Chartperiode wurde verändert
REASON_CHARTCLOSE	4	Das Chart wurde abgeschlossen
REASON_PARAMETERS	5	Eingabeparameter wurden vom Benutzer verändert
REASON_ACCOUNT	6	Ein anderes Konto wurde aktiviert oder das Konto wurde mit dem Handelsserver neu verbunden, nachdem die Kontoeinstellungen geändert worden
REASON_TEMPLATE	7	andere Chartschablone wurde angewendet
REASON_INITFAILED	8	Der Wert bedeutet, dass Bearbeiter OnInit() Nichtnullwert zurückgegeben hat
REASON_CLOSE	9	Das Terminal wurde geschlossen

Kode des Deinitialisierungsgrundes wird auch als Parameter der vorbestimmten Funktion übertragen OnDeinit(const int reason).

Beispiel:

```
//+------------------------------------------------------------------+
//| get text description                                             |
//+------------------------------------------------------------------+
string getUninitReasonText(int reasonCode)
  {
   string text="";
//---
   switch(reasonCode)
     {
      case REASON_ACCOUNT:
         text="Account was changed";break;
      case REASON_CHARTCHANGE:
         text="Symbol or timeframe was changed";break;
      case REASON_CHARTCLOSE:
         text="Chart was closed";break;
      case REASON_PARAMETERS:
         text="Input-parameter was changed";break;
      case REASON_RECOMPILE:
```

© 2000-2025, MetaQuotes Ltd.

```
         text="Program "+__FILE__+" was recompiled";break;
      case REASON_REMOVE:
         text="Program "+__FILE__+" was removed from chart";break;
      case REASON_TEMPLATE:
         text="New template was applied to chart";break;
      default:text="Another reason";
     }
//---
   return text;
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Zweiter Weg , Kode des Deinitialisierungsgrundes zu bekommen
   Print(__FUNCTION__,"_Kode des Deinitialisierungsgrundes = ",reason);
//--- Zweiter Weg, Kode des Deinitialisierungsgrundes zu bekommen
   Print(__FUNCTION__,"_UninitReason = ",getUninitReasonText(_UninitReason));
  }
```

© 2000-2025, MetaQuotes Ltd.

Pruefung des Objektanzeigers

für die Pruefung des Typs von Objektanzeiger ist die Funktion CheckPointer() vorausbestimmt, die den Wert aus der Enumeration ENUM_POINTER_TYPE rückgibt. Wenn ein unkorrekter Anzeiger verwendet wird, wird die Programmausführung sofort unterbrochon.

Objekte, erzeugt von der Anweisung new, sind Objekte des Typs POINTER_DYNAMIC. Ausschliesslich für solche Anzeiger kann und muss man delete() Anweisung verwenden.

Alle anderen Anzeiger gehören zum Typ POINTER_AUTOMATIC, was bedeutet, dass dieses Objekt automatisch vom Medium des mql5_programms erzeugt wurde. Solche Objekte werden nach Verwendung auch automatisch entfernt.

ENUM_POINTER_TYPE

Konstante	Beschreibung
POINTER_INVALID	Unkorrekter Anzeiger
POINTER_DYNAMIC	Anzeiger des Objektes, erzeugt von der Anweisung new()
POINTER_AUTOMATIC	Anzeiger jedes Objektes, erzeugt automatisch (ohne new())

Sehen Sie auch

Ausführungsfehler, Anweisung der Objektentfernung delete, CheckPointer()

© 2000-2025, MetaQuotes Ltd.

Andere Konstanten

Konstante CLR_NONE wird verwendet, um Farbefehlen anzudeuten, d.h. graphisches Objekt oder graphische Serie des Indikatoren wird nicht dargestellt. Diese Konstante gehört zur Liste der Benennungen von Web-Farben nicht, kann aber überall verwendet werden, wo Farbenargumente angefordert werden.

Konstante INVALID_HANDLE kann bei der Pruefung von Dateihandles (s. die Funktionen FileOpen() und FileFindFirst()) verwendet werden.

Konstante	Beschreibung	Wert
CHARTS_MAX	Maximal mögliche Zahl der gleichzeitig offenen Charts im Terminal	100
clrNONE	Abwesenheit der Farbe	-1
EMPTY_VALUE	Leerwert im Indikatorpuffer	DBL_MAX
INVALID_HANDLE	unkorrektes handle	-1
IS_DEBUG_MODE	Flagge der Arbeit des mq5-Programm in Debugging-Mode	true in Debugging-Mode, anderenfalls false
IS_PROFILE_MODE	Flagge der Arbeit des mq5-Programms im Modus von Code Profiling	im Profiling-Modus ist nicht gleich null, sonst 0
NULL	Null des jeden Typs	0
WHOLE_ARRAY	Bedeutet die Anzahl von Elementen, die bis zum Feldende bleiben, bis das ganze Feld verarbeitet wird	-1
WRONG_VALUE	Konstante kann implizit reduziert werden zum Typ jeder Enumeration.	-1

Konstante EMPTY_VALUE entspricht gewoehnlich den Werten des Indikators, die auf dem Chart nicht gezeigt werden. ZB. Für den eingebauten Indikator Standard Deviation mit der Periode wird die Linie für die ersten 19 Bars in der Geschichte nicht ausgegeben. Wenn Sie handle dieses Indikators durch die Funktion iStdDev() erzeugen und die Indikatorwerte für diese Bars in Feld durch CopyBuffer() kopieren, werden diese Werte EMPTY_VALUE gleich sein.

Man kann selbsstaendig seinen eigenen Leerwert des Indikators im Benutzerindikator spezifizieren, wobei der Indikator auf dem Chart nicht gezeigt zu werden braucht. Verwenden Sie dafür die Funktion PlotIndexSetDouble() mit dem Modifikator PLOT_EMPTY_VALUE.

Konstante NULL kann einer Variable jedes einfachen Typs oder Anzeiger des Objektes oder der Struktur oder der Klasse zugeordnet werden. Die Zuordnung von NULL einer Zeilenvariable bedeutet die vollstaendige Deinitialisierung dieser Variable.

© 2000-2025, MetaQuotes Ltd.

Konstante WRONG_VALUE ist für die Faellen bestimmt, wenn es erforderlich ist, den Wert der Enumeration rückzugeben und das muss ein unkorrekter Wert sein muss. ZB man muss angeben, dass dieser Wert ein Wert dieser Enumeration ist. Als Beispiel kann man eine gewisse Funktion CheckLineStyle() nehmen, die Linienstil für Objekt rückgibt, das mit dem Namen angegeben wird. Wenn bei der Stilanforderung durch die Funktion ObjectGetInteger() das Ergebnis true sein wird, kehrt ein Wert der Enumeration ENUM_LINE_STYLE zurück, anderenfalls kehrt WRONG_VALUE zurück.

```
void OnStart()
  {
   if(CheckLineStyle("MyChartObject")==WRONG_VALUE)
      printf("Error line style getting.");
  }
//+------------------------------------------------------------------+
//| Kehrt Stil der Linie für Objekt, das mit dem Namen angegeben wird |
//+------------------------------------------------------------------+
ENUM_LINE_STYLE CheckLineStyle(string name)
  {
   long style;
//---
   if(ObjectGetInteger(0,name,OBJPROP_STYLE,0,style))
      return((ENUM_LINE_STYLE)style);
   else
      return(WRONG_VALUE);
  }
```

Konstante WHOLE_ARRAY ist für die Funktionen bestimmt, die die Andeutung der Anzahl der Elementen in verarbeiteten Felder fordern:

- ArrayCopy();
- ArrayMinimum();
- ArrayMaximum();
- FileReadArray();
- FileWriteArray().

Wenn man spezifizieren muss, dass alle Feldwerte von der angegebenen Position bis zum Ende verarbeitet werden muessen, reicht es, den Wert WHOLE_ARRAY. anzugeben.

Konstante IS_PROFILE_MODE erlaubt die Arbeit des Programms für die korrekte Datenerfassung im Profiling-Modus zu ändern. Profiling erlaubt die Ausführungszeit von einzelnen Fragmente des Programms (in der Regel eine Funktion) zu messen, und auch die Anzahl solcher Aufrufe zu berechnen. Um die korrekte Information über die Ausführungszeit im Profiling-Modus zu erhalten, kann man die Aufrufe der Funktion Sleep() wie folgt deaktivieren:

```
//--- Sleep kann das Ergebnis von Profiling
if(!IS_PROFILE_MODE) Sleep(100) stark beeinflussen(verzerren); // verbieten wir den Au
```

Wert der Konstante IS_PROFILE_MODE wird vom Compiler bei der Compilierung angegeben, und in gewöhnlichen Modus gleich Null gestellt. Beim Programmstart im Profiling-Modu wird eine spezielle Compilierung durchgeführt, und in diesem Fall statt IS_PROFILE_MODE wird einen von null verschiedenen Wert ersetzt.

© 2000-2025, MetaQuotes Ltd.

Das Verwenden der Konstante IS_DEBUG_MODE ist für die leichte Veranderung der Arbeit mql5-Programms in Debuggin-Mode nuetzlich. ZB. in Debugging-Mode kann es erforderlich sein, zusaettzliche Debugging Information in Log des Terminals auszugeben oder zusaetzliche graphische Objekte auf dem Chart zu erzeugen.

Das unten angeführte Beispiel erzeugt das Objekt Label und gibt seine Beschreibung und Farbe abhängig vom Mode des Scriptablaufs. Für Ablauf des Scripts in Debugging-Mode von MetaEditor, drücken Sie die Taste F5. Wenn Sie wollen, Script vom Browserwindow im Terminal ablaufen lassen, wird Farbe und Objekttext von Label anders sein.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Check_DEBUG_MODE.mq5 |
//|                        Copyright © 2009, MetaQuotes Software Corp. |
//|                                             https://www.metaquotes.net |
//+------------------------------------------------------------------+
#property copyright "Copyright © 2009, MetaQuotes Software Corp."
#property link      "https://www.metaquotes.net"
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   string label_name="invisible_label";
   if(ObjectFind(0,label_name)<0)
     {
      Print("Object ",label_name," not found. Error code = ",GetLastError());
      //--- erzeugen wir das Objekt Label
      ObjectCreate(0,label_name,OBJ_LABEL,0,0,0);
      //--- stellen wir die Koordinate X ein
      ObjectSetInteger(0,label_name,OBJPROP_XDISTANCE,200);
      //--- stellen wir die Koordinate Y ein
      ObjectSetInteger(0,label_name,OBJPROP_YDISTANCE,300);
      ResetLastError();
      if(IS_DEBUG_MODE) // Debugging-Mode
        {
         //--- Nachricht über Mode des Scriptablaufs zeigen
         ObjectSetString(0,label_name,OBJPROP_TEXT,"DEBUG MODE");
         //--- geben wir rote Farbe des Textes vor
         if(!ObjectSetInteger(0,label_name,OBJPROP_COLOR,clrRed))
            Print("Nicht in der Lage, Farbe einzustellen. Fehler ",GetLastError());
        }
      else               // Arbeitmode
         ObjectSetString(0,label_name,OBJPROP_TEXT,"RELEASE MODE");
         //--- stellen wir unsichtbare Textfarbe vor
         if(!ObjectSetInteger(0,label_name,OBJPROP_COLOR,CLR_NONE))
            Print("Misskungen, Farbe einzustellen. Fehler ",GetLastError());
        }
      ChartRedraw();
      DebugBreak();     // hier wird Unterbrechung erfolgen, wenn wir in Debugging-Mode
     }
  }
```

Methoden der Datenverschlüsselung

© 2000-2025, MetaQuotes Ltd.

Für die Angabe der Methode der Datenumwandlung (Verschlüsselung und Berechnung von Hash-Werten) wird in den CryptEncode() und CryptDecode() Funktionen die ENUM_CRYPT_METHOD Aufzählung verwendet.

ENUM_CRYPT_METHOD

Konstante	Beschreibung
CRYPT_BASE64	BASE64 Verschlüsselung (Umkodierung)
CRYPT_AES128	AES Verschlüsselung mit einem 128-Bit-Schlüssel (16 Byte)
CRYPT_AES256	AES Verschlüsselung mit einem 256-Bit-Schlüssel (32 Byte)
CRYPT_DES	DES Verschlüsselung mit einem 56-Bit-Schlüssel (7 Byte)
CRYPT_HASH_SHA1	Berechnung HASH SHA1
CRYPT_HASH_SHA256	Berechnung HASH SHA256
CRYPT_HASH_MD5	Berechnung HASH MD5
CRYPT_ARCH_ZIP	ZIP-Archivierung

Sehen Sie auch

DebugBreak, Information über das ausgeführte MQL5-Programm, CryptEncode(), CryptDecode()

© 2000-2025, MetaQuotes Ltd.

Datenstrukturen

In MQL5 gibt es 12 vordefinierte Strukturen, die für Speicherung und Übertragung der Sonderinformation bestimmt sind:

- MqlDateTime ist bestimmt für die Darstellung des Datums und der Zeit;
- MqlParam ermöglicht die Eingabeparameter beim Erzeugen eines Handles für einen Indikator durch die Funktion IndicatorCreate() zu übertragen;
- MqlRates bietet Information über historische Daten, sie enthalten Preis, Volumen und Erweiterung;
- MqlBookInfo für den Empfang der Information von DOM (Fenster mit den Kursen) ;
- MqlTradeRequest wird verwendet für das Erzeugen von Handelsanforderung bei der Durchführung von Handelsoperationen;
- MqlTradeResult empfängt die Antwort des Handelsservers auf die Handelsanforderung, die durch die Funktion OrderSend() gesendet wurden;
- MqlTradeTransaction enthält die Beschreibung einer Handelstransaktion;
- MqlTick wird verwendet, um schnell die wichtigsten Informationen über die laufenden Preise zu bekommen.
- Die Strukturen des Wirtschaftskalenders werden verwendet, um Daten über die an die MetaTrader 5-Plattform gesendeten Ereignisse des Wirtschaftskalenders in Echtzeit zu erhalten. Die Funktion des Wirtschaftskalenderskönnen die makroökonomischer Parameter unmittelbar nach der Veröffentlichung neuer Berichte analysieren, da relevante Werte ohne Verzögerung direkt von der Quelle übertragen werden.

© 2000-2025, MetaQuotes Ltd.

MqlDateTime

In der Datumstruktur gibt es acht Felder des Typs int.

```
struct MqlDateTime
  {
   int year;            // Jahr
   int mon;             // Monat
   int day;             // Tag
   int hour;            // Stunde
   int min;             // Minuten
   int sec;             // Sekunden
   int day_of_week;     // Wochentag (0-Sonntag, 1-Montag)
   int day_of_year;     // Ordnungszahl des Jahres (1. Jannuar - ist Tag Null im Jahr)
  };
```

Hinweis

Ordnungszahl des Jahres day_of_year im Schaltjahr, angefangen mit Maerz, wird sich von der Ordnungszahl des entspechenden Tages im Nichtschaltjahr unterscheiden.

Beispiel:

```
void OnStart()
  {
//---
   datetime date1=D'2008.03.01';
   datetime date2=D'2009.03.01';

   MqlDateTime str1,str2;
   TimeToStruct(date1,str1);
   TimeToStruct(date2,str2);
   printf("%02d.%02d.%4d, day of year = %d",str1.day,str1.mon,
          str1.year,str1.day_of_year);
   printf("%02d.%02d.%4d, day of year = %d",str2.day,str2.mon,
          str2.year,str2.day_of_year);
  }
/*  Ergebnis
   01.03.2008, day of year = 60
   01.03.2009, day of year = 59
*/
```

Sehen Sie auch

TimeToStruct, Strukturen und Klassen

© 2000-2025, MetaQuotes Ltd.

Struktur der Eingabeparameter des Indikators (MqlParam)

Struktur MqlParam ist speziell für übertragung der Eingabeparameter ausgearbeitet bei der Erzeugung von handle des technischen Indikators durch die Funktion IndicatorCreate().

```
struct MqlParam
  {
   ENUM_DATATYPE     type;                    // Typ des Eingabeparmeters, Wert der Enumeration ENUM
   long              integer_value;           // Feld für Speicherung des ganzzahligen Wertes
   double            double_value;            // Feld für Speicherung des Wertes double oder float
   string            string_value;            // Feld für Speicherun des Wertes des Zeilentyps
  };
```

Alle Eingabeparameter des Indikators werden als Feld des Typs MqlParam übertragen, Feld *type* jedes Elementes dieses Feldes deutet auf den Datentyp, die von diesem Element übertragen werden. Die Werte der Indikatorparameter muessen vorlaeufug in entsprechende Felder jedes Elementes stellen in *integer_value*, in *double_value* oder in *string_value*) abhängig davon welcher Wert der Enumeration ENUM_DATATYPE im Feld *type* gibt.

Wenn der Wert IndicatorCreate() als Indikatortyp vom dritten Parameter durch die Funktion IND_CUSTOM übertragen wird , muss das erste Element des Feldes der Eingabeparameter Feld *type* mit dem Wert TYPE_STRING aus der Enumeration ENUM_DATATYPE haben und Feld *string_value* muss den Namen des Benutzerindikators enthalten.

© 2000-2025, MetaQuotes Ltd.

MqlRates

Die Struktur bewahrt die Information über die Preise, Volumen und Spread auf.

```
struct MqlRates
  {
   datetime time;         // Anfangszeit der Periode
   double   open;         // Eroeffnungspreis
   double   high;         // der hoechste Preis der Periode
   double   low;          // der niedrigste Preis der Periode
   double   close;        // Schlusspreis
   long     tick_volume;  // Tickvolumen
   int      spread;       // Spread
   long     real_volume;  // Lagervolumen
  };
```

Beispiel:

```
void OnStart()
  {
   MqlRates rates[];
   int copied=CopyRates(NULL,0,0,100,rates);
   if(copied<=0)
      Print("Fehler des Kopieren von Preisdaten ",GetLastError());
   else Print("Kopiert ",ArraySize(rates)," Bars");
  }
```

Sehen Sie auch

CopyRates, Zugang zu Timeserien

© 2000-2025, MetaQuotes Ltd.

MqlBookInfo

In der Struktur gibt es Information über DOM.

```
struct MqlBookInfo
  {
   ENUM_BOOK_TYPE    type;          // Auftragstyp der Enumeration ENUM_BOOK_TYPE
   double            price;         // Preis
   long              volume;        // Volumen
   double            volume_real;   // Volumen mit größerer Genauigkeit
  };
```

Hinweis

MqlBookInfo ist eine vordefinierte Struktur, so braucht sie keine Deklarierung und Beschreibung. Um die Struktur zu benutzen, nur deklarieren Sie eine Variable dieses Typs.

DOM ist nur für finanzielle Symbole zugaenglich.

Beispiel:

```
   MqlBookInfo priceArray[];
   bool getBook=MarketBookGet(NULL,priceArray);
   if(getBook)
     {
      int size=ArraySize(priceArray);
      Print("MarketBookInfo für ",Symbol());
     }
   else
     {
      Print("Misslungen, DOm des Symbols zu erhlaten ",Symbol());
     }
```

Sehen Sie auch

MarketBookAdd, MarketBookRelease, MarketBookGet, Handelsordern in DOM, Datentypen

© 2000-2025, MetaQuotes Ltd.

Struktur der Handelsanforderung (MqlTradeRequest)

Interaktion von Client-Terminal und Handelsserverfür Durchführung für Operation der Orderstellung wird mittels Handelanforderungen durchgeführt. Anforderung ist durch die spezielle vorbestimmte Struktur MqlTradeRequest dargestellt, die alle erforderlichen Felder für Abschluss der Handelsorder enthält. Ergebnis der Verarbeitung von Anforderung ist durch die Struktur MqlTradeResult repraesentiert.

```
struct MqlTradeRequest
  {
   ENUM_TRADE_REQUEST_ACTIONS    action;           // Operationstyp
   ulong                         magic;            // Expert Advisor ID (Identifikato
   ulong                         order;            // Orderticket
   string                        symbol;           // Name des Handelssymbols
   double                        volume;           // angefordertes Dealvolumen in Lo
   double                        price;            // Preis
   double                        stoplimit;        // Orderlevel StopLimit
   double                        sl;               // Oderlevel Stop Loss
   double                        tp;               // Orderlevel  Take Profit
   ulong                         deviation;        // maximal moegliche Abweichnung v
   ENUM_ORDER_TYPE               type;             // Ordertyp
   ENUM_ORDER_TYPE_FILLING       type_filling;     // Durchführungstyp der Order
   ENUM_ORDER_TYPE_TIME          type_time;        // Typ der Ablauffrist der Order
   datetime                      expiration;       // Ablauffrist der Pending-Order (
   string                        comment;          // Oprderkommentar
   ulong                         position;         // Position ticket
   ulong                         position_by;      // The ticket of an opposite posit
  };
```

Beschreibung der Felder

Feld	Beschreibung
action	Typ der Handelsoperation. Wert kann einer der Enumerationswerte ENUM_TRADE_REQUEST_ACTIONS sein
magic	Identifikator des Experten. Ermoeglicht, analytische Verabeitung der Handelsordern zu gestalten. Jeder Experte kann seinen unikalen Identifikator beim Senden der Handelanforderung stellen
order	Ticket der Order. Für Modifikation der Warteordern erforderlich.
symbol	Symbolname der Order. Nicht erforderlich bei Modifikation der Ordern und Schliessen der Positionen.
volume	Angefordertes Dealvolumen in Lots. Realwert des Volumens wird vom Durchführungstyp der Order abhängen
price	Preis, bei dem Order ausgeführt werden muss, für Marktordern mit dem Durchführungstyp "Market Execution" (SYMBOL_TRADE_EXECUTION_MARKET), die den Typ TRADE_ACTION_DEAL haben, ist es nicht erforderlich, Preis anzugeben

© 2000-2025, MetaQuotes Ltd.

Feld	Beschreibung
stoplimit	Preis, bei dem Warteorder Limit gestellt wird, wenn der Preis den Wert price reicht(diese Bedingung ist obligatorisch). Bis dahin wird Warteorder ins Handelssystem nicht ausgegeben
sl	Stop Loss des Preises bei unguenstiger Bewegung des Preises
tp	Take Profit des Preises bei guenstiger Bewegung des Preises
deviation	Maximal moefliche Abweichung vom angeforderten Preis vorgegeben in Punkten
type	Ordertyp. Wert kann einer der Enumerationswerte ENUM_ORDER_TYPE sein
type_filling	Durchführungstyp der Order. Kann einer der Enumerationswerte ENUM_ORDER_TYPE_FILLING sein
type_time	Ablauftyp der Order. Kann einer der Enumerationswerte ENUM_ORDER_TYPE_TIME sein
expiration	Ablauffrist der Warteordern (für Ordern des Typs ORDER_TIME_SPECIFIED)
comment	Kommentar zur Order

Beim Modifizieren oder Schließen einer Position im Hedging-Modus müssen Sie ihr Ticket (MqlTradeRequest::position) angeben. Im Netting-Modus können Sie das Ticket auch angeben, aber die Identifikation erfolgt anhand des Symbolnamens.

für Senden der Order Handelsoperationen durchzuführen, muss die Funktion OrderSend() verwendet werden. Für jede Handelsoperation muessen obligatorische Felde angegeben werden und können optionale Felder ausgefuellt werden. Insgesamt werden 7 Varianten des Sendens der Handelsanforderung vorausgesehen:

Request Execution

Handelsorder, Positionen im Request Execution Mode zu oeffnen (Handelsmode bei der Anforderung der laufenden Preise). Man muss 9 Felder spezifizieren:

- action
- symbol
- volume
- price
- sl
- tp
- deviation
- type
- type_filling

Man kann auch Feldwerte magic und comment vorgeben.

Instant Execution

Handelsorder, Positionen im Instant Execution Mode zu oeffnen (Handel mit Flowpreisen). Man muss 9 Felder spezifizieren:

© 2000-2025, MetaQuotes Ltd.

- action
- symbol
- volume
- price
- sl
- tp
- deviation
- type
- type_filling

Man kann auch Feldwerte magic und comment vorgeben.

Market Execution

Handelsorder, Positionen im Market Execution Mode zu oeffnen (Durchführung der Handelsorder im Mark). Man muss 5 Felder spezifizieren:

- action
- symbol
- volume
- type
- type_filling

Man kann auch Feldwerte magic und comment vorgeben.

Exchange Execution

Handelsorder, Positionen im Exchange Execution Modus zu öffnen (Börse-Modus der Ausführung von Handelsordern). Man muss 5 Felder angeben:

- action
- symbol
- volume
- type
- type_filling

Man kann auch Feldwerte magic und comment vorgeben.

Ein Beispiel der Transaktion TRADE_ACTION_DEAL für das Eröffnen einer Buy-Position:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456   // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Eröffnen einer Buy-Position                                      |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
//--- Parameter der Anfrage
   request.action   =TRADE_ACTION_DEAL;                    // Typ der Transaktion
   request.symbol   =Symbol();                             // Symbol
   request.volume   =0.1;                                  // Volumen von 0.1 Lot
   request.type     =ORDER_TYPE_BUY;                       // Ordertyp
   request.price    =SymbolInfoDouble(Symbol(),SYMBOL_ASK); // Eröffnungspreis
   request.deviation=5;                                    // zulässige Abweichung vo
   request.magic    =EXPERT_MAGIC;                         // MagicNumber der Order
//--- Anfrage senden
   if(!OrderSend(request,result))
      PrintFormat("OrderSend error %d",GetLastError());     // wenn die Anfrage konnte
//--- Details zur Transaktion
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
  }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transaktion TRADE_ACTION_DEAL für das Eröffnen einer Sell-Position:

```
#define EXPERT_MAGIC 123456   // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Eröffnen einer Sell-Position                                     |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
//--- Parameter der Anfrage
   request.action   =TRADE_ACTION_DEAL;                    // Typ der Transaktion
   request.symbol   =Symbol();                             // Symbol
   request.volume   =0.2;                                  // Volumen von 0.2 Lot
   request.type     =ORDER_TYPE_SELL;                      // Ordertyp
   request.price    =SymbolInfoDouble(Symbol(),SYMBOL_BID); // Eröffnungspreis
   request.deviation=5;                                    // zulässige Abweichung vo
   request.magic    =EXPERT_MAGIC;                         // MagicNumber der Order
//--- Anfrage senden
   if(!OrderSend(request,result))
      PrintFormat("OrderSend error %d",GetLastError());     // wenn die Anfrage konnte
//--- Details zur Transaktion
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
  }
//+------------------------------------------------------------------+
```

Ein Beispiel der Transaktion TRADE_ACTION_DEAL für das Schließen von Positionen:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456   // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Schließen aller Position                                         |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren
   MqlTradeRequest request;
   MqlTradeResult  result;
   int total=PositionsTotal(); // Anzahl offener Positionen
//--- in allen offenen Positionen suchen
   for(int i=total-1; i>=0; i--)
     {
      //--- Parameter der Order
      ulong  position_ticket=PositionGetTicket(i);
      string position_symbol=PositionGetString(POSITION_SYMBOL);
      int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS);
      ulong  magic=PositionGetInteger(POSITION_MAGIC);
      double volume=PositionGetDouble(POSITION_VOLUME);
      ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
      //--- Details zur Position anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s [%I64d]",
                  position_ticket,
                  position_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  magic);
      //--- wenn MagicNumber übereinstimmt
      if(magic==EXPERT_MAGIC)
        {
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
         //--- Parameter der Transaktion setzen
         request.action   =TRADE_ACTION_DEAL;        // Typ der Transaktion
         request.position =position_ticket;          // das Ticket der Position
         request.symbol   =position_symbol;          // Symbol
         request.volume   =volume;                   // das Volumem der Position
         request.deviation=5;                        // zulässige Abweichung vom Preis
         request.magic    =EXPERT_MAGIC;             // MagicNumber der Position
         //--- den Preis und den Ordertyp je nach dem Positionstyp setzen
         if(type==POSITION_TYPE_BUY)
           {
            request.price=SymbolInfoDouble(position_symbol,SYMBOL_BID);
            request.type =ORDER_TYPE_SELL;
           }
         else
           {
            request.price=SymbolInfoDouble(position_symbol,SYMBOL_ASK);
            request.type =ORDER_TYPE_BUY;
           }
         //--- Informationen über das Schließen anzeigen
         PrintFormat("Close #%I64d %s %s",position_ticket,position_symbol,EnumToStrin
         //--- Senden der Anfrage
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kor
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
         //---
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
//+------------------------------------------------------------------+
```

SL & TP Modification

Handelsorder, Levels StopLoss und/oder TakeProfit zu modifizieren. Man muss 4 Felder angeben:

- action
- symbol
- sl
- tp

Ein Beispiel für die Transaktion TRADE_ACTION_SLTP für die Änderung von Stop Loss und Take Profit bei einer offenen Position:

```
#define EXPERT_MAGIC 123456  // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Stop Loss und Take Profit modifizieren                           |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren
   MqlTradeRequest request;
   MqlTradeResult  result;
   int total=PositionsTotal(); // Anzahl offener Positionen
//--- in allen offenen Positionen suchen
   for(int i=0; i<total; i++)
     {
      //--- Parameter der Order
      ulong  position_ticket=PositionGetTicket(i);// das Ticket der Position
      string position_symbol=PositionGetString(POSITION_SYMBOL); // Symbol
      int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS); // Stellen
      ulong  magic=PositionGetInteger(POSITION_MAGIC); // MagicNumber der Position
      double volume=PositionGetDouble(POSITION_VOLUME);    // Volumen der Position
      double sl=PositionGetDouble(POSITION_SL);  // Stop Loss der Position
      double tp=PositionGetDouble(POSITION_TP);  // Take Profit der Position
      ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
      //--- Details zur Position anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s  sl: %s  tp: %s  [%I64d]",
                  position_ticket,
                  position_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  DoubleToString(sl,digits),
                  DoubleToString(tp,digits),
                  magic);
      //--- wenn die MagicNumber übereinstimmt, sind Stop Loss und Take Profit nicht g
      if(magic==EXPERT_MAGIC && sl==0 && tp==0)
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- aktuelle Preis Levels berechnen
         double price=PositionGetDouble(POSITION_PRICE_OPEN);
         double bid=SymbolInfoDouble(position_symbol,SYMBOL_BID);
         double ask=SymbolInfoDouble(position_symbol,SYMBOL_ASK);
         int    stop_level=(int)SymbolInfoInteger(position_symbol,SYMBOL_TRADE_STOPS_L
         double price_level;
         //--- wenn der Level der minimalen Abweichung vom aktuellen Schlusskurs in Pu
         if(stop_level<=0)
            stop_level=150; // setzen wir eine Abweichung von 150 Punkten vom aktuelle
         else
            stop_level+=50; // sicherheitshalber nehmen wir (SYMBOL_TRADE_STOPS_LEVEL

         //--- Stop Loss und Take Profit berechnen und abrunden
         price_level=stop_level*SymbolInfoDouble(position_symbol,SYMBOL_POINT);
         if(type==POSITION_TYPE_BUY)
           {
            sl=NormalizeDouble(bid-price_level,digits);
            tp=NormalizeDouble(bid+price_level,digits);
           }
         else
           {
            sl=NormalizeDouble(ask+price_level,digits);
            tp=NormalizeDouble(ask-price_level,digits);
           }
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
         //--- Parameter der Transaktion setzen
         request.action  =TRADE_ACTION_SLTP; // Typ der Transaktion
         request.position=position_ticket;   // das Ticket der Position
         request.symbol=position_symbol;     // Symbol
         request.sl      =sl;                // Stop Loss der Position
         request.tp      =tp;                // Take Profit der Position
         request.magic=EXPERT_MAGIC;         // MagicNumber der Position
         //--- Informationen über die Modifizierung anzeigen
         PrintFormat("Modify #%I64d %s %s",position_ticket,position_symbol,EnumToStrin
         //--- Senden einer Anfrage
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kon
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
        }
     }
  }
//+------------------------------------------------------------------+
```

Pending Order

Handelsorder, eine Handelsorder einzustellen. Man muss 11 Felder angeben:

- action
- symbol
- volume
- price
- stoplimit
- sl
- tp
- type

© 2000-2025, MetaQuotes Ltd.

- type_filling
- type_time
- expiration

Man kann auch Feldwerte magic und comment vorgeben.

Ein Beispiel der Transaktion TRADE_ACTION_PENDING für das Setzen einer Pending Order:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Ein Beispiel für das Platzieren von Pending Orders"
#property script_show_inputs
#define EXPERT_MAGIC 123456                                  // MagicNumber des Expert Advi
input ENUM_ORDER_TYPE orderType=ORDER_TYPE_BUY_LIMIT;    // Ordertyp
//+------------------------------------------------------------------+
//| Pending Orders setzen                                            |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
//--- Parameter für das Platzieren einer Pending Order
   request.action   =TRADE_ACTION_PENDING;                                  // Typ der Tran
   request.symbol   =Symbol();                                              // Symbol
   request.volume   =0.1;                                                   // das Volumen
   request.deviation=2;                                                     // zulässige Ab
   request.magic    =EXPERT_MAGIC;                                          // MagicNumber
   int offset = 50;                                                         // Abweichung v
   double price;                                                            // Preis, bei d
   double point=SymbolInfoDouble(_Symbol,SYMBOL_POINT);                     // Punktgröße
   int digits=SymbolInfoInteger(_Symbol,SYMBOL_DIGITS);                     // Stellen nach
   //--- den Typ der Transaktion überprüfen
   if(orderType==ORDER_TYPE_BUY_LIMIT)
     {
      request.type     =ORDER_TYPE_BUY_LIMIT;                               // Ordertyp
      price=SymbolInfoDouble(Symbol(),SYMBOL_ASK)-offset*point;             // Eröffnungsku
      request.price    =NormalizeDouble(price,digits);                      // normierter P
     }
   else if(orderType==ORDER_TYPE_SELL_LIMIT)
     {
      request.type     =ORDER_TYPE_SELL_LIMIT;                               // Ordertyp
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID)+offset*point;              // Eröffnungsk
      request.price    =NormalizeDouble(price,digits);                       // normierter
     }
   else if(orderType==ORDER_TYPE_BUY_STOP)
     {
      request.type =ORDER_TYPE_BUY_STOP;                                    // Ordertyp
      price        =SymbolInfoDouble(Symbol(),SYMBOL_ASK)+offset*point;     // Eröffnungsk
      request.price=NormalizeDouble(price,digits);                          // normierter
     }
   else if(orderType==ORDER_TYPE_SELL_STOP)
     {
      request.type     =ORDER_TYPE_SELL_STOP;                               // Ordertyp
      price=SymbolInfoDouble(Symbol(),SYMBOL_BID)-offset*point;             // Eröffnungsp
      request.price    =NormalizeDouble(price,digits);                      // normierter
     }
   else Alert("Dieses Beispiel gilt nur für das Platzieren von Pending Orders");   //
//--- Senden einer Anfrage
   if(!OrderSend(request,result))
      PrintFormat("OrderSend error %d",GetLastError());                     // wenn die An
//--- Details zur Transaktion
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
  }
//+------------------------------------------------------------------+
```

Modify Pending Order

Handelsorder, Preislevels der Warteorder zu modifizieren. Man muss 7 Felder spezifizieren:

- action

© 2000-2025, MetaQuotes Ltd.

- order
- price
- sl
- tp
- type_time
- expiration

Ein Beispiel der Transaktion TRADE_ACTION_MODIFY für das Modifizieren von Preis-Levels von Pending Orders:

© 2000-2025, MetaQuotes Ltd.

```
#define EXPERT_MAGIC 123456  // MagicNumber des EAs
//+------------------------------------------------------------------+
//| Modifizieren von Pending Orders                                  |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
   int total=OrdersTotal(); // Anzahl platzierter Pending Orders
//--- in allen gesetzten Pending Orders suchen
   for(int i=0; i<total; i++)
     {
      //--- Parameter der Order
      ulong  order_ticket=OrderGetTicket(i);                                  // das Ticket
      string order_symbol=Symbol();                                           // Symbol
      int    digits=(int)SymbolInfoInteger(order_symbol,SYMBOL_DIGITS);       // Stellen na
      ulong  magic=OrderGetInteger(ORDER_MAGIC);                              // MagicNumbe
      double volume=OrderGetDouble(ORDER_VOLUME_CURRENT);                     // das aktuel
      double sl=OrderGetDouble(ORDER_SL);                                     // der aktuel
      double tp=OrderGetDouble(ORDER_TP);                                     // der aktuel
      ENUM_ORDER_TYPE type=(ENUM_ORDER_TYPE)OrderGetInteger(ORDER_TYPE); // Ordertyp
      int offset = 50;                                                        // Abweichung
      double price;                                                           // Preis der
      double point=SymbolInfoDouble(order_symbol,SYMBOL_POINT);               // Punktgröße
      //--- Details zur Order anzeigen
      PrintFormat("#%I64u %s  %s  %.2f  %s  sl: %s  tp: %s  [%I64d]",
                  order_ticket,
                  order_symbol,
                  EnumToString(type),
                  volume,
                  DoubleToString(PositionGetDouble(POSITION_PRICE_OPEN),digits),
                  DoubleToString(sl,digits),
                  DoubleToString(tp,digits),
                  magic);
      //--- wenn die MagicNumber übereinstimmt, sind Stop Loss und Take Profit nicht s
      if(magic==EXPERT_MAGIC && sl==0 && tp==0)
        {
         request.action=TRADE_ACTION_MODIFY;                                  // Typ der Tran
         request.order = OrderGetTicket(i);                                   // das Ticket d
         request.symbol   =Symbol();                                          // Symbol
         request.deviation=5;                                                 // zulässige Ab
        //--- Preis-Level, Take Profit und Stop Loss der Order je nach Ordertyp setzen
         if(type==ORDER_TYPE_BUY_LIMIT)
           {
            price = SymbolInfoDouble(Symbol(),SYMBOL_ASK)-offset*point;
            request.tp = NormalizeDouble(price+offset*point,digits);
            request.sl = NormalizeDouble(price-offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                  // normier
           }
         else if(type==ORDER_TYPE_SELL_LIMIT)
           {
           price = SymbolInfoDouble(Symbol(),SYMBOL_BID)+offset*point;
            request.tp = NormalizeDouble(price-offset*point,digits);
            request.sl = NormalizeDouble(price+offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                   // normie
           }
         else if(type==ORDER_TYPE_BUY_STOP)
           {
           price = SymbolInfoDouble(Symbol(),SYMBOL_ASK)+offset*point;
            request.tp = NormalizeDouble(price+offset*point,digits);
```

© 2000-2025, MetaQuotes Ltd.

```
            request.sl = NormalizeDouble(price-offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                  // normie
           }
         else if(type==ORDER_TYPE_SELL_STOP)
           {
           price = SymbolInfoDouble(Symbol(),SYMBOL_BID)-offset*point;
            request.tp = NormalizeDouble(price-offset*point,digits);
            request.sl = NormalizeDouble(price+offset*point,digits);
            request.price    =NormalizeDouble(price,digits);                  // normie
           }
         //--- Anfrage senden
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kor
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
        }
     }
  }
//+------------------------------------------------------------------+
```

Delete Pending Order

Handelsorder, eine Warteorder zu entfernen. Man muss 2 Felder spezifizieren:

- action
- order

Ein Beispiel der Transaktion TRADE_ACTION_REMOVE für das Löschen von Pending Orders:

```
#define EXPERT_MAGIC 123456  // MagicNumber des Expert Advisors
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Pending Orders löschen                                           |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Anfrage und das Ergebnis deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result={};
   int total=OrdersTotal(); // Anzahl platzierter Pending Orders
//--- in allen platzierten Pending Orders suchen
   for(int i=total-1; i>=0; i--)
     {
      ulong  order_ticket=OrderGetTicket(i);                   // das Ticket der Order
      ulong  magic=OrderGetInteger(ORDER_MAGIC);               // MagicNumber der Orde
      //--- wenn die MagicNumber übereinstimmt
      if(magic==EXPERT_MAGIC)
        {
         //--- die Werte der Anfrage und des Ergebnisses auf Null setzen
         ZeroMemory(request);
         ZeroMemory(result);
         //--- Parameter der Transaktion setzen
         request.action=TRADE_ACTION_REMOVE;                   // Typ der Transaktion
         request.order = order_ticket;                         // das Ticket der Order
         //--- Anfrage senden
         if(!OrderSend(request,result))
            PrintFormat("OrderSend error %d",GetLastError());  // wenn die Anfrage kor
         //--- Details zur Transaktion
         PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,
        }
     }
  }
//+------------------------------------------------------------------+
```

Sehen Sie auch

Strukturen und Klassen, Handelsfunktionen, Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

Struktur der Ergebnisse der Prüfung der Handelsanforderung (MqlTradeCheckResult)

Bevor Sie dem Handelsserver Anforderung auf eine Handelsoperation senden, muss man es prüfen. Die Prüfung wird mittels der Funktion OrderCheck() durchgeführt, der die geprüfte Anforderung und die Variable des Typs Strukturen MqlTradeCheckResult ubertragen werden. In diese Variable wird das Ergebnis der Prüfung geschrieben.

```
struct MqlTradeCheckResult
  {
   uint          retcode;             // Antwortkode
   double        balance;             // Bilanz nach Dealabschluss
   double        equity;              // Equity nach Dealabschluss
   double        margin;              // Margenanforderungen
   double        margin_free;         // Freie Marge
   double        margin_level;        // Level der Marge
   string        comment;             // Kommentar zum Antwortkode (Beschreibung des Fehlers)
  };
```

Beschreibung der Felder

Feld	Beschreibung
retcode	Rückgabekode
balance	Wert der Bilanz nach Ausführung der Handelsoperation
equity	Wert der Eigenmittel nach Ausführung der Handelsoperation
profit	Wert des fließenden Gewinns nach Ausführung der Handelsoperation
margin	Grösse der Marge, die für die erforderliche Handelsoperation notwendig ist
margin_free	Grösse der freien Eigenmittel, die nach Ausführung der erforderlichen Handelsoperation bleiben
margin_level	Levekl der Marge nach Ausführung der erforderlichen Handelsoperation
comment	Kommentar zum Antwortkode, Beschreibung des Fehlers

Sehen Sie auch

Struktur der Handelsanforderung, Struktur für Erhaltung der laufenden Preise, OrderSend, OrderCheck

© 2000-2025, MetaQuotes Ltd.

Struktur des Ergebnisses der Handelsanforderung (MqlTradeResult)

Als Antwort auf Handelsanforderung, gibt das Handelsserver Daten über Bearbeitungsergebnis der Handelsanforderung als eine vorbestimmte Sonderstruktur MqlTradeResult zurück.

```
struct MqlTradeResult
  {
   uint     retcode;          // Kode des Operationsergebnisses
   ulong    deal;             // Ticket des Deals, wenn es abgeschlossen ist
   ulong    order;            // Ticket der Order, wenn sie gestellt ist
   double   volume;           // Dealvolumen, bestaetigt vom Broker
   double   price;            // Dealpreis, bestaetigt vom Broker
   double   bid;              // Laufender Bidpreis (Requote Preis)
   double   ask;              // Laufender Askpreis (Requote Preis)
   string   comment;          // Kommentar des Brokers zur Opearion (als Default durch Beschreibung
   uint     request_id;       // Identifikator der Anforderung, der vom Terminal beim Senden eingese
   int      retcode_external; // Return Code eines externen Handelssystems
  };
```

Beschreibung der Felder

Feld	Beschreibung
retcode	Rückkehrkode des Handelsservers
deal	Ticket von Deal, wenn es abgeschlossen ist. Zugänglich bei der Handelsoperation TRADE_ACTION_DEAL
order	Ticket der Order, wenn sie gestellt ist. Gemeldet bei der Handelsoperation TRADE_ACTION_PENDING
volume	Dealvolumen, bestätigt vom Broker. Hängt vom Typ der Orderausfuellung
price	Preis im Deal, bestätigt vom Broker. Hängt vom Feld *deviation* in der Handelsanforderung und/oder vom Typ der Handelsoperation ab
bid	Laufender Bidpreis (Requote Preis)
ask	Laufender Askpreis (Requote Preis)
comment	Kommentar des Brokers zur Operation (als Default durch Beschreibung ausgefüllt Rückkehrcode des Handels Servers)
request_id	Identifikator der Anforderung, der vom Terminal beim Senden eingesetzt wird
retcode_external	Der Code des Fehlers, den ein externes Handelssystem zurückgegeben hat. Das Hinzufügen und die Typen dieser Fehler hängen mit dem Broker und dem externen Handelssystem zusammen, an das Transaktionen gesendet werden

Ergebnis der Handelsoperation kehrt zur Variable des Typs MqlTradeResult zurück, die vom zweiten Parameter in die Funktion OrderSend() für Durchführung der Handelsoperationen übertragen wird .

© 2000-2025, MetaQuotes Ltd.

Das Terminal schreibt den Identifikator der Anforderung in das Feld request_id, wenn er an den Handel-Server durch Funktionen OrdersSend() und OrderSendAsync() gesendet wird. Vom Handel-Server erhält das Terminal Nachrichten über die ausgeführten Handelstransaktionen und übergibt sie für Bearbeitung in die OnTradeTransaction () Funktion, die als die Parameter enthält:

- Beschreibung der Handelstransaktion in der Struktur MqlTradeTransaction;
- Beschreibung der Handelsanforderung, der aus der Funktion OrderSend() oder OrdersSendAsync() gesendet ist. Identifikator der Anforderung wird vom Terminal an den Handle-Server gesendet, und die Anforderung und seinen request_id werden im Terminalspeicher aufbewahrt;
- das Ergebnis der Ausführung der Handelsanforderung als die Struktur MqlTradeResult, in dem das Feld request_id den Identifikator dieser Anforderung enthält.

Funktion OnTradeTransaction() bekommt drei Eingabeparameter, aber die beiden letzten Parameter sollten nur für die Handelstransaktionen mit dem Typ TRADE_TRANSACTION_REQUEST analysiert werden. In allen anderen Fällen werden die Daten über die Handelsanforderung und das Ergebnis seiner Ausführung nicht ausgefüllt. Das Beispiel der Analyse der Parameter ist im Abschnitt Struktur der Handelstransaktion gegeben.

Die Einstellung des Identifikators request_id vom Terminal für die Handelsanforderung dient zur Arbeit mit der asynchronen Funktion OrderSendAsync() beim Senden an den Server. Dieser Identifikator erlaubt die ausgeführte Aktion (Aufruf der Funktion OrderSend oder OrderSendAsync) mit dem in die OnTradeTransaction() übergebenen Ergebnis dieser Aktion zu verbinden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Senden der Handelsanforderung mit Ergebnisverarbeitung           |
//+------------------------------------------------------------------+
bool MyOrderSend(MqlTradeRequest request,MqlTradeResult result)
  {
//--- stellen wir Kode des letzten Fehlers auf Null
   ResetLastError();
//--- senden wir Anforderung
   bool success=OrderSend(request,result);
//--- wenn das Ergebnis erfolglos ist - versuchen wir zu erfahren warum
   if(!success)
     {
      int answer=result.retcode;
      Print("TradeLog:Trade request failed. Error = ",GetLastError());
      switch(answer)
        {
         //--- Requote
         case 10004:
           {
            Print("TRADE_RETCODE_REQUOTE");
            Print("request.price = ",request.price,"   result.ask = ",
                  result.ask," result.bid =",result.bid);
            break;
           }
         //--- Order ist vom Server nicht akzeptiert
         case 10006:
           {
            Print("TRADE_RETCODE_REJECT");
            Print("request.price = ",request.price,"   result.ask = ",
                  result.ask," result.bid = ",result.bid);
            break;
           }
         //--- Falscher Preis
         case 10015:
           {
            Print("TRADE_RETCODE_INVALID_PRICE");
            Print("request.price = ",request.price,"   result.ask = ",
                  result.ask," result.bid = ",result.bid);
            break;
           }
         //--- falscher SL und/oder TP
         case 10016:
           {
            Print("TRADE_RETCODE_INVALID_STOPS");
            Print("request.sl = ",request.sl," request.tp = ",request.tp);
            Print("result.ask = ",result.ask," result.bid = ",result.bid);
            break;
           }
         //--- unkorrektes Volumen
         case 10014:
           {
            Print("TRADE_RETCODE_INVALID_VOLUME");
            Print("request.volume = ",request.volume,"   result.volume = ",
                  result.volume);
            break;
           }
         //--- nicht genug Geld für Handelsoperation
         case 10019:
           {
            Print("TRADE_RETCODE_NO_MONEY");
            Print("request.volume = ",request.volume,"   result.volume = ",
```

© 2000-2025, MetaQuotes Ltd.

```
                  result.volume,"   result.comment = ",result.comment);
            break;
           }
         //--- irgendwelcher anderer Grund, Kode der Serverantwort ist zu melden
         default:
           {
            Print("Other answer = ",answer);
           }
        }
      //--- erfolgloses Ergebnis der Handelsanfoderung durch Rückkehr false wird gemel
      return(false);
     }
//--- OrderSend() gibt true zurück - wiederholen wir die Antwort
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

Struktur der Handelstransaktion (MqlTradeTransaction)

Als Ergebnis der Durchführung von bestimmten Aktionen auf dem Handelskonto, ändert sich der Kontostand. Zu solchen Aktionen gehören:

- Senden der Handelsanfrage aus einer beliebigen MQL5-Anwendung in dem Client-Terminal mittels der Funktionen OrderSend und OrderSendAsync und ihre nachfolgende Ausführung;
- Senden der Handelsanfrage über die grafische Oberfläche des Terminals und ihre nachfolgende Ausführung;
- Auslösen von aufgeschobenen Ordern und Stop-Ordern auf dem Server;
- Ausführen von Operationen auf der Seite der Handel-Server.

Als Ergebnis dieser Aktionen werden die folgenden Handelstransaktionen durchgeführt:

- Die Verarbeitung der Handelsanfrage;
- Änderung der offenen Aufträge;
- Änderung der Ordergeschichte;
- Änderung der Dealgeschichte;
- Änderung der Position.

Zum Beispiel, beim Senden Marktkauforder, wird Marktkauforder behandelt und eine entsprechende Marktkauforder für das Konto erstellt. Ist die Order ausgeführt, wird sie aus der Auftragsliste entfernt und zur Ordergeschichte hinzugefügt. Dann wird der entsprechende Deal zur Geschichte hinzugefügt und eine neue Position wird erstellt. Alle diese Aktionen sind Handelstransaktionen.

Um die auf das Konto angewendeten Handelstransaktionen zu erhalten, ist der spezielle Handler OnTradeTransaction im MQL5 vorgesehen. In den ersten Parameter wird die Struktur MqlTradeTransaction übergeben, die Handelstransaktionen beschreibt.

```
struct MqlTradeTransaction
  {
   ulong                         deal;             // Ticket des Deals
   ulong                         order;            // Ticket der Order
   string                        symbol;           // Name des Handelsinstrumentes
   ENUM_TRADE_TRANSACTION_TYPE   type;             // Typ der Handelstransaktion
   ENUM_ORDER_TYPE               order_type;       // Typ der Order
   ENUM_ORDER_STATE              order_state;      // Zustand der Order
   ENUM_DEAL_TYPE                deal_type;        // Typ des Deals
   ENUM_ORDER_TYPE_TIME          time_type;        // Typ der Order nach der Zeit der Aktion
   datetime                      time_expiration;  // Ablaufzeit der Order
   double                        price;            // Preis
   double                        price_trigger;    // Aktivierungspreis von Stop-Limit-Order
   double                        price_sl;         // Stop Loss Ebene
   double                        price_tp;         // Take Profit Ebene
   double                        volume;           // Volumen in Lots
   ulong                         position;         // Position ticket
   ulong                         position_by;      // Ticket of an opposite position
  };
```

Felder Beschreibung

Feld	Beschreibung
deal	Ticket des Deals

© 2000-2025, MetaQuotes Ltd.

Feld	Beschreibung
order	Ticket der Order
symbol	Name des Handelsinstrumente, nach dem die Transaktion durchgeführt wurde.
type	Typ der Handelstransaktion Der Wert kann einer der Enumerationswerte ENUM_TRADE_TRANSACTION_TYPE sein.
order_type	Typ der Handelsorder. Der Wert kann einer der Enumerationswerte ENUM_ORDER_TYPE sein.
order_state	Zustand der Handelsorder. Der Wert kann einer der Enumerationswerte ENUM_ORDER_STATE sein.
deal_type	Typ des Deals. Der Wert kann einer der Enumerationswerte ENUM_DEAL_TYPE sein.
time_type	Typ der Order nach Ablauf. Der Wert kann einer der Enumerationswerte ENUM_ORDER_TYPE_TIME sein.
time_expiration	Ablaufzeit der aufgeschobenen Order (für die Order von Typ ORDER_TIME_SPECIFIED und ORDER_TIME_SPECIFIED_DAY).
price	Preis. Je nach Typ der Handelstransaktion kann es ein Preis von einer Order, eines Deals oder einer Position sein.
price_trigger	Stop-Kurs (Aktivierungspreis) von Stop-Limit-Order (ORDER_TYPE_BUY_STOP_LIMIT und ORDER_TYPE_SELL_STOP_LIMIT).
price_sl	Stop Loss Preis. Je nach Typ der Handelstransaktion kann es sich auf eine Order, einen Deal oder eine Position beziehen.
price_tp	Take Profit Preis. Je nach Typ der Handelstransaktion kann es sich auf eine Order, einen Deal oder eine Position beziehen.
volume	Volumen in Lots. Je nach Typ der Handelstransaktion kann es das aktuelle Volumen von einer Order, einem Deal oder einer Position zeigen.
position	Das Ticket der Position, die die Transaktion beeinflußt hat.
position_by	Das Ticket der Gegenposition. Wird beim Schließen einer Position zur Gegenposition verwendet, die auf demselben Symbol, aber in einer entgegengesetzten Richtung eröffnet wurde.

Der bestimmende Parameter für die Analyse der empfangenen Transaktion ist sein Typ, der ins Feld **type** übergeben wird. Zum Beispiel, wenn eine Transaktion der Typ TRADE_TRANSACTION_REQUEST (das Ergebnis der Bearbeitung der Handelsanforderung vom Server ist erhalten) ist, hat die Struktur nur ein ausgefülltes Feld **type**. Die anderen Felder müssen nicht analysiert werden. In diesem Fall kann man zwei zusätzliche Parameter **request** und **result** analysieren, die an den Handler OnTradeTransaction() übergeben werden, wie im Beispiel unten gezeigt.

Wenn Sie den Typ der Handelsoperation wissen, können Sie über eine Analyse des aktuellen Zustandes von Ordern, Positionen und Deals auf dem Handelskonto entscheiden. Es ist zu beachten, dass eine Handelsanforderung, die aus dem Terminal an den Server gesendet wird, mehrere

© 2000-2025, MetaQuotes Ltd.

Handelstransaktionen auslösen kann, deren Reihenfolge des Eingangs ins Terminal nicht gewährleistet wird.

Die Struktur MqlTradeTransaction wird auf unterschiedlicher Weise je nach Typ der Handelstransaktion (ENUM_TRADE_TRANSACTION_TYPE) ausgefüllt:

TRADE_TRANSACTION_ORDER_* und TRADE_TRANSACTION_HISTORY_*

Für die Handelstransaktionen betreffend die Bearbeitung der offenen Ordern (TRADE_TRANSACTION_ORDER_ADD, TRADE_TRANSACTION_ORDER_UPDATE und TRADE_TRANSACTION_ORDER_DELETE) und der Ordergeschichte (TRADE_TRANSACTION_HISTORY_ADD, TRADE_TRANSACTION_HISTORY_UPDATE, TRADE_TRANSACTION_HISTORY_DELETE), werden in der Struktur MqlTradeTransaction die folgenden Felder ausgefüllt:

- order - Ticket der Order;
- symbol - Name eines Finanzinstrumentes in der Order;
- type - Typ der Handelstransaktion;
- order_type - Typ der Order;
- orders_state - aktueller Zustand der Order;
- time_type - Typ des Ablaufs der Order;
- time_expiration - Ablaufzeit der Order (für die Ordern mit dem Typ des Ablaufs von ORDER_TIME_SPECIFIED und ORDER_TIME_SPECIFIED_DAY);
- price - Preis in der Order, der von einem Kunden angegeben ist;
- price_trigger - Stop-Kurs des Auslösens von Stop-Limit-Order (nur für ORDER_TYPE_BUY_STOP_LIMIT und ORDER_TYPE_SELL_STOP_LIMIT);
- price_sl - Stop Loss der Order (wird ausgefüllt, wenn in der Order angegeben ist);
- price_tp - Take Profit Preis der Order (wird ausgefüllt, wenn in der Order angegeben ist);
- volume - aktuelles Volumen der Order (nicht ausgeführt). Anfangsvolumen der Order kann man in der Ordergeschichte mit Hilfe der Funktion HistoryOrders* finden.
- position - das Ticket der Position, die durch die Ausführung einer Order eröffnet, modifiziert oder geschlossen wurde. Das Feld ist nur für Marktorders auszufüllen. Für TRADE_TRANSACTION_ORDER_ADD wird das Feld nicht ausgefüllt.
- position_by - das Ticket der Gegenposition. Ist nur für die Orders auszufüllen, die eine Position zur Gegenposition schließen (close by).

TRADE_TRANSACTION_DEAL_*

Für die Handelstransaktionen betreffend die Bearbeitung des Deals (TRADE_TRANSACTION_DEAL_ADD, TRADE_TRANSACTION_DEAL_UPDATE und TRADE_TRANSACTION_DEAL_DELETE), werden in der Struktur MqlTradeTransaction die folgenden Felder ausgefüllt:

- deal - Ticket des Deals;
- order - Ticket der Order, auf dessen Grundlage der Deal abgeschlossen wurde;
- symbol - Name eines Finanzinstrumentes im Deal;
- type - Typ der Handelstransaktion;
- deal_type - Typ des Deals;
- price - Preis, zu dem einen Deal abgeschlossen wurde;
- price_sl - Stop Loss Preis (wird ausgefüllt, wenn in der Order angegeben ist, auf dessen Grundlage der Deal abgeschlossen wurde);

© 2000-2025, MetaQuotes Ltd.

- price_tp - Take Profit Preis (wird ausgefüllt, wenn in der Order angegeben ist, auf dessen Grundlage der Deal abgeschlossen wurde);
- volume - Volumen des Deals in Lots.
- position - das Ticket der Position, die durch die Ausführung einer Order eröffnet, modifiziert oder geschlossen wurde.
- position_by - das Ticket der Gegenposition. Nur für die Trades auszufüllen, die eine Position zur Gegenposition schließen (out by).

TRADE_TRANSACTION_POSITION

Für die Handelstransaktionen betreffend die Änderungen von Positionen, die mit der Ausführung des Deals nicht verbunden sind (TRADE_TRANSACTION_POSITION), werden in der Struktur MqlTradeTransaction die folgenden Felder ausgefüllt:

- symbol - Name eines Finanzinstrumentes der Position;
- type - Typ der Handelstransaktion;
- deal_type - Typ der Position (DEAL_TYPE_BUY oder DEAL_TYPE_SELL);
- price - gewogener Durchschnittskurs der Eröffnung von Position;
- price_sl - Stop Loss Preis;
- price_tp - Take Profit Preis;
- volume - Volumen der Position in Lots, wenn es geändert wurde.
- position - das Ticket der Position.

> Änderung der Position (Hinzufügen, Änderung oder Liquidation) als Ergebnis des Abschlusses des Deals zieht kein Erscheinen der Transaktion TRADE_TRANSACTION_POSITION nach sich.

TRADE_TRANSACTION_REQUEST

Nur ein Feld wird in der Struktur MqlTradeTransaction für die Handelstransaktionen ausgefüllt, die die Tatsache beschreiben, dass die Handelsanfrage vom Server bearbeitet ist und das Ergebnis seiner Bearbeitung empfangen ist (TRADE_TRANSACTION_REQUEST):

- type - Typ der Handelstransaktion;

> Für Transaktionen dieses Typs muss nur type Feld (Typ der Handelstransaktion) analysiert werden. Für zusätzliche Informationen muss man die zweiten und dritten Parameter der Funktion OnTradeTransaction (request und result) analysieren.

Beispiel:

```
input int MagicNumber=1234567;

//--- die Klasse CTrade einfügen und die Variable deklarieren
#include <Trade\Trade.mqh>
CTrade trade;
//--- Flags für das Platzieren und Löschen einer Pending Order
bool pending_done=false;
bool pending_deleted=false;
//--- das Ticket der Pending Order hier speichern
ulong order_ticket;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
int OnInit()
  {
//--- setzen wir eine MagicNumber, mit deren alle unsere Orders markiert werden
   trade.SetExpertMagicNumber(MagicNumber);
//--- Trade Requests werden im asynchronen Modus mithilfe der Funktion OrderSendAsync
   trade.SetAsyncMode(true);
//--- die Variable mit Null initialisieren
   order_ticket=0;
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- Pending Order platzieren
   if(!pending_done)
     {
      double ask=SymbolInfoDouble(_Symbol,SYMBOL_ASK);
      double buy_stop_price=NormalizeDouble(ask+1000*_Point,(int)SymbolInfoInteger(_Sy
      bool res=trade.BuyStop(0.1,buy_stop_price,_Symbol);
      //--- wenn die Funktion BuyStop() erfolgreich war
      if(res)
        {
         pending_done=true;
         //--- das Ergebnis der Anfrage aus ctrade erhalten
         MqlTradeResult trade_result;
         trade.Result(trade_result);
         //--- request_id für die gesendete Anfrage erhalten
         uint request_id=trade_result.request_id;
         Print("Anfrage zum Platzieren einer Pending Order gesendet. Identifikator der
         //--- das Ticket der Order speichern (im asynchronen Modus in CTrade gleich N
         order_ticket=trade_result.order;
         //--- alles erledigt, OnTick() vorzeitig verlassen
         return;
        }
     }
//--- Pending Order löschen
   if(!pending_deleted)
      //--- zusätzliche Überprüfung
      if(pending_done && (order_ticket!=0))
        {
         //--- versuchen, eine Pending Order zu löschen
         bool res=trade.OrderDelete(order_ticket);
         Print("OrderDelete=",res);
         //--- beim erfolgreichen Senden der Löschanfrage
         if(res)
```

© 2000-2025, MetaQuotes Ltd.

```
            {
             pending_deleted=true;
             //--- das Ergebnis der Ausführung der Anfrage erhalten
             MqlTradeResult trade_result;
             trade.Result(trade_result);
             //--- den Identifikator dem dem Ergebnis der Anfrage entnehmen
             uint request_id=trade_result.request_id;
             //--- im Journal anzeigen
             Print("Löschanfrage für die Order #",order_ticket, gesendet
                   ". Der Identifikator der Anfrage Request_ID=",request_id,
                   "\r\n");
             //--- das Ticket der Order aus dem Ergebnis der Anfrage schreiben
             order_ticket=trade_result.order;
            }
         }
//---
  }
//+------------------------------------------------------------------+
//| TradeTransaction function                                        |
//+------------------------------------------------------------------+
void OnTradeTransaction(const MqlTradeTransaction &trans,
                        const MqlTradeRequest &request,
                        const MqlTradeResult &result)
  {
//--- den Typ der Transaktion als einen Wert der Aufzählung erhalten
   ENUM_TRADE_TRANSACTION_TYPE type=(ENUM_TRADE_TRANSACTION_TYPE)trans.type;
//--- wenn die Transaktion das Ergebnis der Anfrage darstellt, nur ihren Namen anzeige
   if(type==TRADE_TRANSACTION_REQUEST)
     {
      Print(EnumToString(type));
      //--- die Beschreibung der bearbeiteten Anfrage anzeigen
      Print("------------RequestDescription\r\n",RequestDescription(request));
      //--- die Beschreibung des Ergebnisses der Anfrage anzeigen
      Print("------------ResultDescription\r\n",TradeResultDescription(result));
      //--- das Ticket der Order speichern, um diese bei der nächsten Bearbeitung in C
      if(result.order!=0)
        {
         //--- die Order beim nächsten Aufruf von OnTick() nach dem Ticket löschen
         order_ticket=result.order;
         Print(" Ticket der Pending Order ",order_ticket,"\r\n");
        }
     }
   else // für Transaktionen von einem anderen Typ eine vollständige Beschreibung anze
//--- die Beschreibung der Transaktion im Journal anzeigen
      Print("------------TransactionDescription\r\n",TransactionDescription(trans));

//---
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Gibt eine textuelle Beschreibung der Transaktion zurück                  |
//+------------------------------------------------------------------+
string TransactionDescription(const MqlTradeTransaction &trans)
  {
//---
   string desc=EnumToString(trans.type)+"\r\n";
   desc+="Symbol: "+trans.symbol+"\r\n";
   desc+="Deal ticket: "+(string)trans.deal+"\r\n";
   desc+="Deal type: "+EnumToString(trans.deal_type)+"\r\n";
   desc+="Order ticket: "+(string)trans.order+"\r\n";
   desc+="Order type: "+EnumToString(trans.order_type)+"\r\n";
   desc+="Order state: "+EnumToString(trans.order_state)+"\r\n";
   desc+="Order time type: "+EnumToString(trans.time_type)+"\r\n";
   desc+="Order expiration: "+TimeToString(trans.time_expiration)+"\r\n";
   desc+="Price: "+StringFormat("%G",trans.price)+"\r\n";
   desc+="Price trigger: "+StringFormat("%G",trans.price_trigger)+"\r\n";
   desc+="Stop Loss: "+StringFormat("%G",trans.price_sl)+"\r\n";
   desc+="Take Profit: "+StringFormat("%G",trans.price_tp)+"\r\n";
   desc+="Volume: "+StringFormat("%G",trans.volume)+"\r\n";
   desc+="Position: "+(string)trans.position+"\r\n";
   desc+="Position by: "+(string)trans.position_by+"\r\n";
//--- den erhaltenen String zurückgeben
   return desc;
  }
//+------------------------------------------------------------------+
//| Gibt die textuelle Beschreibung der Handelsanfrage zurück                |
//+------------------------------------------------------------------+
string RequestDescription(const MqlTradeRequest &request)
  {
//---
   string desc=EnumToString(request.action)+"\r\n";
   desc+="Symbol: "+request.symbol+"\r\n";
   desc+="Magic Number: "+StringFormat("%d",request.magic)+"\r\n";
   desc+="Order ticket: "+(string)request.order+"\r\n";
   desc+="Order type: "+EnumToString(request.type)+"\r\n";
   desc+="Order filling: "+EnumToString(request.type_filling)+"\r\n";
   desc+="Order time type: "+EnumToString(request.type_time)+"\r\n";
   desc+="Order expiration: "+TimeToString(request.expiration)+"\r\n";
   desc+="Price: "+StringFormat("%G",request.price)+"\r\n";
   desc+="Deviation points: "+StringFormat("%G",request.deviation)+"\r\n";
   desc+="Stop Loss: "+StringFormat("%G",request.sl)+"\r\n";
   desc+="Take Profit: "+StringFormat("%G",request.tp)+"\r\n";
   desc+="Stop Limit: "+StringFormat("%G",request.stoplimit)+"\r\n";
   desc+="Volume: "+StringFormat("%G",request.volume)+"\r\n";
   desc+="Comment: "+request.comment+"\r\n";
//--- den erhaltenen String zurückgeben
   return desc;
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Gibt die textuelle Beschreibung der bearbeiteten Anfrage zurück     |
//+------------------------------------------------------------------+
string TradeResultDescription(const MqlTradeResult &result)
  {
//---
   string desc="Retcode "+(string)result.retcode+"\r\n";
   desc+="Request ID: "+StringFormat("%d",result.request_id)+"\r\n";
   desc+="Order ticket: "+(string)result.order+"\r\n";
   desc+="Deal ticket: "+(string)result.deal+"\r\n";
   desc+="Volume: "+StringFormat("%G",result.volume)+"\r\n";
   desc+="Price: "+StringFormat("%G",result.price)+"\r\n";
   desc+="Ask: "+StringFormat("%G",result.ask)+"\r\n";
   desc+="Bid: "+StringFormat("%G",result.bid)+"\r\n";
   desc+="Comment: "+result.comment+"\r\n";
//--- den erhaltenen String zurückgeben
   return desc;
  }
```

Sehen Sie auch

Typen der Handelstransaktionen, OnTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Struktur für Erfassung der laufenden Preise (MqlTick)

Das ist die Struktur für Speicherung der letzten Preise des Symbols. Bestimmt für schnelles Erhalten der gefragten Information über laufende Preise.

```
struct MqlTick
  {
   datetime     time;          // Zeit des letzten Preisaktualisierung
   double       bid;           // aktueller Bid-Preis (Geldkurs)
   double       ask;           // aktueller Ask-Preis (Briefkurs)
   double       last;          // Preis des letzten Geschäfts (Last)
   ulong        volume;        // Volumen des aktuellen Last-Preises
   long         time_msc;      // Zeitpunkt des letzten Last-Preises in Millisekunden
   uint         flags;         // Tick-Flags
   double       volume_real;   // Volumen des aktuellen Last-Preises mit größerer Gena
  };
```

Variable des Typs MqlTick ermöglicht mit einem Anruf der Funktion SymbolInfoTick() Werte von Ask, Bid, Last und Volumen zu bekommen.

Alle Parameter sind immer für jeden Tick gefüllt, unabhängig davon, ob die Daten im Vergleich zum vorherigen Tick verändert haben. So können Sie immer einen aktuellen Preisstand haben, ohne auf die vorherigen Werte in der Tick-Historie zu suchen. Beispielsweise kann nur der Bid-Preis mit dem Tick ändern, aber die andere Parameter werden auch in der Preisstruktur angegeben: der vorherige Bid-Preis, Volumen usw.

Um herauszufinden, welche Daten im aktuellen Tick geändert sind, analysieren Sie seine Flags:

- TICK_FLAG_BID - der Tick hat den Bid-Preis geändert
- TICK_FLAG_ASK - der Tick hat den Ask-Preis geändert
- TICK_FLAG_LAST - der Tick hat den Last-Preis geändert
- TICK_FLAG_VOLUME - der Tick hat das Volumen geändert
- TICK_FLAG_BUY - der Tick hat als Ergebnis eines Kauf-Deals erschienen
- TICK_FLAG_SELL - der Tick hat als Ergebnis eines Verkauf-Deals erschienen

Beispiel:

```
void OnTick()
  {
   MqlTick last_tick;
//---
   if(SymbolInfoTick(Symbol(),last_tick))
     {
      Print(last_tick.time,": Bid = ",last_tick.bid,
            " Ask = ",last_tick.ask,"  Volume = ",last_tick.volume);
     }
   else Print("SymbolInfoTick() failed, error = ",GetLastError());
//---
  }
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Strukturen und Klassen, CopyTicks(), SymbolInfoTick()

© 2000-2025, MetaQuotes Ltd.

Struktur des Wirtschaftskalenders

Dieser Abschnitt beschreibt die Funktionen für das Arbeiten mit dem Wirtschaftskalender, die direkt auf der MetaTrader-Plattform verfügbar sind. Der Wirtschaftskalender ist eine vorgefertigte Enzyklopädie mit Beschreibungen makroökonomischer Indikatoren, deren Veröffentlichungsterminen und Wichtigkeitsmerkmalen. Relevante Werte makroökonomischer Indikatoren werden direkt zum Zeitpunkt der Veröffentlichung an die MetaTrader-Plattform gesendet und in einem Chart als Tags angezeigt, mit denen Sie die erforderlichen Indikatoren nach Ländern, Währungen und Bedeutung visuell verfolgen können.

Die Funktionen des Wirtschaftskalenders ermöglichen die automatische Analyse eingehender Ereignisse nach benutzerdefinierten Wichtigkeitskriterien aus der Perspektive der benötigten Länder/Währungspaare.

Länderbeschreibungen werden von der Struktur MqlCalendarCountry gesetzt. Sie wird in den Funktionen CalendarCountryById() und CalendarCountries() verwendet.

```
struct MqlCalendarCountry
  {
   ulong                             id;                // Länder-ID (ISO 3166-1
   string                            name;              // Ländername in Textfor
   string                            code;              // Länder-Code (ISO 3166
   string                            currency;          // Währungs-Code des Lan
   string                            currency_symbol;   // Symbol der Landeswähr
   string                            url_name;          // URL-Landesname, wie e
  };
```

Ereignisbeschreibungen werden mit der Struktur MqlCalendarEvent dargestellt. Sie wird in den Funktionen CalendarEventById(), CalendarEventByCountry() und CalendarEventByCurrency() verwendet.

```
struct MqlCalendarEvent
  {
   ulong                             id;                // Ereignis-ID
   ENUM_CALENDAR_EVENT_TYPE          type;              // Ereignis-Typ entsprec
   ENUM_CALENDAR_EVENT_SECTOR        sector;            // der Sektor des Ereign
   ENUM_CALENDAR_EVENT_FREQUENCY     frequency;         // Häufigkeit des Ereign
   ENUM_CALENDAR_EVENT_TIMEMODE      time_mode;         // Zeitmodus eines Ereig
   ulong                             country_id;        // Länder-ID
   ENUM_CALENDAR_EVENT_UNIT          unit;              // Maßeinheit des Wertes
   ENUM_CALENDAR_EVENT_IMPORTANCE    importance;        // Wichtigkeit des Ereig
   ENUM_CALENDAR_EVENT_MULTIPLIER    multiplier;        // Multiplikator des Wer
   uint                              digits;            // Anzahl der Dezimalste
   string                            source_url;        // URL der Quelle, wo da
   string                            event_code;        // Ereignis-Code
   string                            name;              // Textname des Ereignis
  };
```

© 2000-2025, MetaQuotes Ltd.

Ereigniswerte werden mit der Struktur MqlCalendarValue dargestellt. Sie wird in den Funktionen CalendarValueById(), CalendarValueHistoryByEvent(), CalendarValueHistory(), CalendarValueLastByEvent() und CalendarValueLast() verwendet

```
struct MqlCalendarValue
  {
   ulong                             id;                // Werte-ID
   ulong                             event_id;          // Ereignis-ID
   datetime                          time;              // Datum und Zeit des Er
   datetime                          period;            // Ereignisberichtszeitr
   int                               revision;          // Revision des veröffen
   long                              actual_value;      // aktueller Wert in ppr
   long                              prev_value;        // vorheriger Wert in pp
   long                              revised_prev_value;// revidierter vorherige
   long                              forecast_value;    // prognostizierter Wert
   ENUM_CALENDAR_EVENT_IMPACT        impact_type;       // mögliche Auswirkung a
  //--- Funktionen zur Prüfung der Werte
   bool                         HasActualValue(void) const;   // Rückgabe von true, we
   bool                         HasPreviousValue(void) const; // Rückgabe von true, we
   bool                         HasRevisedValue(void) const;  // Rückgabe von true, we
   bool                         HasForecastValue(void) const; // Rückgabe von true, we
  //--- Funktionen zur Rückgabe der Werte
   double                       GetActualValue(void) const;   // Rückgabe des aktuelle
   double                       GetPreviousValue(void) const; // Rückgabe des vorherig
   double                       GetRevisedValue(void) const;  // Rückgabe des revidier
   double                       GetForecastValue(void) const; // Rückgabe des prognost
  };
```

Die Struktur von MqlCalendarValue bietet Methoden zum Prüfen und Setzen von Werten der Felder *actual_value*, *forecast_value*, *prev_value* und *revised_prev_value*. Wenn kein Wert angegeben wird, speichert das Feld LONG_MIN (-9223372036854775808).

Bitte beachten Sie, dass die in diesem Feld gespeicherten Werte mit einer Million multipliziert werden. Dies bedeutet, dass, wenn Sie Werte von MqlCalendarValue mithilfe der Funktionen CalendarValueById, CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent und CalendarValueLast abrufen, sollten Sie überprüfen, ob die Feldwerte gleich LONG_MIN sind; wenn in einem Feld ein Wert angegeben ist, sollten Sie den Wert durch 1.000.000 dividieren, um den Wert zu erhalten. Eine andere Methode zum Abrufen der Werte besteht darin, die Werte mit den Funktionen der Struktur MqlCalendarValue zu überprüfen und abzurufen.

Ein Beispiel für den Umgang mit Kalenderereignissen:

```
//--- Erstellen einer Struktur zum Speichern von Kalenderereignissen mit reellen Werte
struct AdjustedCalendarValue
  {
   ulong                             id;                // Werte-ID
   ulong                             event_id;          // Ereignis-ID
   datetime                          time;              // Datum und Zeit des Er
   datetime                          period;            // Ereignisberichtszeitr
   int                               revision;          // Revision des veröffen
```

© 2000-2025, MetaQuotes Ltd.

```
   double                              actual_value;          // aktueller Wert
   double                              prev_value;            // vorheriger Wert
   double                              revised_prev_value;    // revidierter, vorherig
   double                              forecast_value;        // prognostizierter Wert
   ENUM_CALENDAR_EVENT_IMPACT          impact_type;           // mögliche Auswirkung a
  };
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
//--- Länder-Code der EU (ISO 3166-1 Alpha-2)
   string EU_code="EU";
//--- Abrufen aller Ereigniswerte der EU
   MqlCalendarValue values[];
//--- Festlegen der Zeitgrenzen des Intervalls für die Ereignisse
   datetime date_from=D'01.01.2021';  // Abfrage aller Ereignisse von 2021
   datetime date_to=0;                // 0 bedeutet alle bekannten Ereignisse, einschl
//--- Abfrage der Ereignishistorie der EU seit 2021
   if(!CalendarValueHistory(values, date_from, date_to, EU_code))
     {
      PrintFormat("Fehler! Es konnten keine Ereignisse für country_code=%s, EU_code ab
      PrintFormat("Fehlernummer: %d", GetLastError());
      return;
     }
   else
      PrintFormat("Erhaltene Ereigniswerte für country_code=%s: %d",
                  EU_code, ArraySize(values));
//--- die Größe des Arrays für die Ausgabe an das Journal zu reduzieren
   if(ArraySize(values)>5)
      ArrayResize(values, 5);
//--- Ausgabe von Ereigniswerten in das Journal, wie sie sind, ohne Prüfung oder Umwan
   Print("Ausgabe der Kalenderwerte wie sie sind");
   ArrayPrint(values);

//--- Überprüfung der Feldwerte und Konvertierung in tatsächliche Werte
//--- Option 1 zum Prüfen und Abrufen der Werte
   AdjustedCalendarValue values_adjusted_1[];
   int total=ArraySize(values);
   ArrayResize(values_adjusted_1, total);
//--- Kopieren der Werte mit Kontrollen und Anpassungen
   for(int i=0; i<total; i++)
     {
      values_adjusted_1[i].id=values[i].id;
      values_adjusted_1[i].event_id=values[i].event_id;
      values_adjusted_1[i].time=values[i].time;
      values_adjusted_1[i].period=values[i].period;
      values_adjusted_1[i].revision=values[i].revision;
```

© 2000-2025, MetaQuotes Ltd.

```
      values_adjusted_1[i].impact_type=values[i].impact_type;
      //--- Werte prüfen und durch 1.000.000 dividieren
      if(values[i].actual_value==LONG_MIN)
         values_adjusted_1[i].actual_value=double("nan");
      else
         values_adjusted_1[i].actual_value=values[i].actual_value/1000000.;

      if(values[i].prev_value==LONG_MIN)
         values_adjusted_1[i].prev_value=double("nan");
      else
         values_adjusted_1[i].prev_value=values[i].prev_value/1000000.;

      if(values[i].revised_prev_value==LONG_MIN)
         values_adjusted_1[i].revised_prev_value=double("nan");
      else
         values_adjusted_1[i].revised_prev_value=values[i].revised_prev_value/1000000.

      if(values[i].forecast_value==LONG_MIN)
         values_adjusted_1[i].forecast_value=double("nan");
      else
         values_adjusted_1[i].forecast_value=values[i].forecast_value/1000000.;
     }
   Print("Die erste Methode zum Prüfen und Abrufen der Kalenderwerte");
   ArrayPrint(values_adjusted_1);

//--- Option 2 zum Prüfen und Abrufen der Werte
   AdjustedCalendarValue values_adjusted_2[];
   ArrayResize(values_adjusted_2, total);
//--- Kopieren der Werte mit Kontrollen und Anpassungen
   for(int i=0; i<total; i++)
     {
      values_adjusted_2[i].id=values[i].id;
      values_adjusted_2[i].event_id=values[i].event_id;
      values_adjusted_2[i].time=values[i].time;
      values_adjusted_2[i].period=values[i].period;
      values_adjusted_2[i].revision=values[i].revision;
      values_adjusted_2[i].impact_type=values[i].impact_type;
      //--- Prüfen und Abrufen der Werte
      if(values[i].HasActualValue())
         values_adjusted_2[i].actual_value=values[i].GetActualValue();
      else
         values_adjusted_2[i].actual_value=double("nan");

      if(values[i].HasPreviousValue())
         values_adjusted_2[i].prev_value=values[i].GetPreviousValue();
      else
         values_adjusted_2[i].prev_value=double("nan");

      if(values[i].HasRevisedValue())
```

© 2000-2025, MetaQuotes Ltd.

```
         values_adjusted_2[i].revised_prev_value=values[i].GetRevisedValue();
      else
         values_adjusted_2[i].revised_prev_value=double("nan");

      if(values[i].HasForecastValue())
         values_adjusted_2[i].forecast_value=values[i].GetForecastValue();
      else
         values_adjusted_2[i].forecast_value=double("nan");
     }
   Print("Die zweite Methode zum Prüfen und Abrufen der Kalenderwerte");
   ArrayPrint(values_adjusted_2);

//--- Option 3 zur Ermittlung der Werte - ohne Prüfungen
   AdjustedCalendarValue values_adjusted_3[];
   ArrayResize(values_adjusted_3, total);
//--- Kopieren der Werte mit Kontrollen und Anpassungen
   for(int i=0; i<total; i++)
     {
      values_adjusted_3[i].id=values[i].id;
      values_adjusted_3[i].event_id=values[i].event_id;
      values_adjusted_3[i].time=values[i].time;
      values_adjusted_3[i].period=values[i].period;
      values_adjusted_3[i].revision=values[i].revision;
      values_adjusted_3[i].impact_type=values[i].impact_type;
      //--- Abrufen der Werte ohne Prüfung
      values_adjusted_3[i].actual_value=values[i].GetActualValue();
      values_adjusted_3[i].prev_value=values[i].GetPreviousValue();
      values_adjusted_3[i].revised_prev_value=values[i].GetRevisedValue();
      values_adjusted_3[i].forecast_value=values[i].GetForecastValue();
     }
   Print("Die dritte Methode zum Abrufen der Kalenderwerte - ohne Prüfung");
   ArrayPrint(values_adjusted_3);
  }
/*
   Wir haben die Ereigniswerte für country_code=EU erhalten:1051
  Ausgabe der Kalenderwerte wie sie sind
         [id] [event_id]              [time]            [period] [revision]        [act
   [0] 144520  999500001 2021.01.04 12:00:00 2020.12.01 00:00:00          3
   [1] 144338  999520001 2021.01.04 23:30:00 2020.12.29 00:00:00          0
   [2] 147462  999010020 2021.01.04 23:45:00 1970.01.01 00:00:00          0 -922337203
   [3] 111618  999010018 2021.01.05 12:00:00 2020.11.01 00:00:00          0
   [4] 111619  999010019 2021.01.05 12:00:00 2020.11.01 00:00:00          0
  Die erste Methode zum Prüfen und Abrufen von Kalenderwerten
         [id] [event_id]              [time]            [period] [revision] [actual_va
   [0] 144520  999500001 2021.01.04 12:00:00 2020.12.01 00:00:00          3       55.2
   [1] 144338  999520001 2021.01.04 23:30:00 2020.12.29 00:00:00          0      143.1
   [2] 147462  999010020 2021.01.04 23:45:00 1970.01.01 00:00:00          0
   [3] 111618  999010018 2021.01.05 12:00:00 2020.11.01 00:00:00          0       11.0
   [4] 111619  999010019 2021.01.05 12:00:00 2020.11.01 00:00:00          0        3.1
```

© 2000-2025, MetaQuotes Ltd.

```
  Die zweite Methode zum Prüfen und Abrufen von Kalenderwerten
         [id] [event_id]              [time]            [period] [revision] [actual_va
   [0] 144520  999500001 2021.01.04 12:00:00 2020.12.01 00:00:00          3       55.2
   [1] 144338  999520001 2021.01.04 23:30:00 2020.12.29 00:00:00          0      143.1
   [2] 147462  999010020 2021.01.04 23:45:00 1970.01.01 00:00:00          0
   [3] 111618  999010018 2021.01.05 12:00:00 2020.11.01 00:00:00          0       11.0
   [4] 111619  999010019 2021.01.05 12:00:00 2020.11.01 00:00:00          0        3.1
  Die dritte Methode zum Abrufen von Kalenderwerten - ohne Prüfung
         [id] [event_id]              [time]            [period] [revision] [actual_va
   [0] 144520  999500001 2021.01.04 12:00:00 2020.12.01 00:00:00          3       55.2
   [1] 144338  999520001 2021.01.04 23:30:00 2020.12.29 00:00:00          0      143.1
   [2] 147462  999010020 2021.01.04 23:45:00 1970.01.01 00:00:00          0
   [3] 111618  999010018 2021.01.05 12:00:00 2020.11.01 00:00:00          0       11.0
   [4] 111619  999010019 2021.01.05 12:00:00 2020.11.01 00:00:00          0        3.1
*/
```

Die Häufigkeit eines Ereignisses wird in der Struktur MqlCalendarEvent dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_FREQUENCY**

ID	Beschreibung
CALENDAR_FREQUENCY_NONE	Veröffentlichungshäufigkeit ist nicht angegeben
CALENDAR_FREQUENCY_WEEK	Veröffentlichung einmal im Monat
CALENDAR_FREQUENCY_MONTH	Veröffentlichung einmal im Monat
CALENDAR_FREQUENCY_QUARTER	Veröffentlichung einmal im Quartal
CALENDAR_FREQUENCY_YEAR	Veröffentlichung einmal im Jahr
CALENDAR_FREQUENCY_DAY	Veröffentlichung einmal am Tag

Ereignis-Typ wird mit der Struktur MqlCalendarEvent dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_TYPE**

ID	Beschreibung
CALENDAR_TYPE_EVENT	Ereignis (Sitzung, Rede etc.)
CALENDAR_TYPE_INDICATOR	Indikator
CALENDAR_TYPE_HOLIDAY	Feiertag

Der wirtschaftliche Sektor eines Ereignisses wird in der Struktur MqlCalendarEvent dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_SECTOR**

ID	Beschreibung
CALENDAR_SECTOR_NONE	Ein Sektor wurde nicht angegeben

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
CALENDAR_SECTOR_MARKET	Markt, Börse
CALENDAR_SECTOR_GDP	Bruttoinlandsprodukt (BIP)
CALENDAR_SECTOR_JOBS	Arbeitsmarkt
CALENDAR_SECTOR_PRICES	Preise
CALENDAR_SECTOR_MONEY	Geld
CALENDAR_SECTOR_TRADE	Handel
CALENDAR_SECTOR_GOVERNMENT	Regierung
CALENDAR_SECTOR_BUSINESS	Geschäft
CALENDAR_SECTOR_CONSUMER	Konsum
CALENDAR_SECTOR_HOUSING	Bau
CALENDAR_SECTOR_TAXES	Steuern
CALENDAR_SECTOR_HOLIDAYS	Feiertage

Die Wichtigkeit des Ereignisses wird in der Struktur MqlCalendarEvent dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_IMPORTANCE**

ID	Beschreibung
CALENDAR_IMPORTANCE_NONE	Wichtigkeit wurde nicht bestimmt
CALENDAR_IMPORTANCE_LOW	Geringe Wichtigkeit
CALENDAR_IMPORTANCE_MODERATE	Mittlere Wichtigkeit
CALENDAR_IMPORTANCE_HIGH	Hohe Wichtigkeit

Maßeinheit für die Anzeige der Ereigniswerte wird in der Struktur MqlCalendarEvent dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_UNIT**

ID	Beschreibung
CALENDAR_UNIT_NONE	Maßeinheit wurde nicht angegeben
CALENDAR_UNIT_PERCENT	Prozentsatz
CALENDAR_UNIT_CURRENCY	Nationale Währung
CALENDAR_UNIT_HOUR	Stunden
CALENDAR_UNIT_JOB	Jobs
CALENDAR_UNIT_RIG	Förderanlagen

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
CALENDAR_UNIT_USD	USD
CALENDAR_UNIT_PEOPLE	Menschen
CALENDAR_UNIT_MORTGAGE	Hypothekarkredite
CALENDAR_UNIT_VOTE	Stimmen
CALENDAR_UNIT_BARREL	Barrels
CALENDAR_UNIT_CUBICFEET	Kubikfuß
CALENDAR_UNIT_POSITION	Nichtkommerzielle Nettoposition
CALENDAR_UNIT_BUILDING	Immobilen

In einigen Fällen benötigen wirtschaftliche Werte einen Multiplikator, die mit der Struktur MqlCalendarEvent dargestellt werden. Im Folgenden aufgelistete Multiplikatoren sind möglich **ENUM_CALENDAR_EVENT_MULTIPLIER**

ID	Beschreibung
CALENDAR_MULTIPLIER_NONE	Ein Multiplikator wurde nicht bestimmt
CALENDAR_MULTIPLIER_THOUSANDS	Tausend
CALENDAR_MULTIPLIER_MILLIONS	Million
CALENDAR_MULTIPLIER_BILLIONS	Milliarde
CALENDAR_MULTIPLIER_TRILLIONS	Billion

Die mögliche Auswirkung auf die Währungskurse wird in der Struktur MqlCalendarValue dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_IMPACT**

ID	Beschreibung
CALENDAR_IMPACT_NA	Wichtigkeit ist nicht bestimmt
CALENDAR_IMPACT_POSITIVE	Positive Wirkung
CALENDAR_IMPACT_NEGATIVE	Negative Wirkung

Der Zeitpunkt eines Ereignisses wird in der Struktur MqlCalendarEvent dargestellt. Im Folgenden aufgelistete Werte sind möglich**ENUM_CALENDAR_EVENT_TIMEMODE**

ID	Beschreibung
CALENDAR_TIMEMODE_DATETIME	Die Quelle veröffentlicht eine genaue Uhrzeit eines Ereignisses

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
CALENDAR_TIMEMODE_DATE	Das Ereignis dauert den ganzen Tag
CALENDAR_TIMEMODE_NOTIME	Die Quelle veröffentlicht keine Zeit für ein Ereignis
CALENDAR_TIMEMODE_TENTATIVE	Die Quelle veröffentlicht einen Tag statt eines genauen Zeitpunktes für ein Ereignis Die Zeit wird beim Eintreten des Ereignisses angegeben.

Siehe auch

Wirtschaftskalender

© 2000-2025, MetaQuotes Ltd.

Kodes der Fehler und Warnungen

Im Abschnitt gibt es folgende Beschreibungen:

- Rückkehrkodes des Handelsservers - Analyse der Ergebnisse von Handelsserver, gesendet durch die Funktion OrderSend();
- Compiler Warnungen - Kodes der Warnmedungen, die bei der Compilierung erscheinen (sind keine Fehler);
- Compilierungsfehler - Kodes der Fehlernachrichten beim unerfolgreichen Versuch der Compilierung;
- Laufzeitfehler - Fehlerkodes bei der Durchführung des mql5-Programms, die durch die Funktion GetLastError() erhalten werden können.

© 2000-2025, MetaQuotes Ltd.

Rückgabecodes des Handelsservers

Alle Befehlsanweisungen, Handelsoperationen durchzuführen, werden als Struktur der Handelsanforderung MqlTradeRequest mittels der Funktion OrderSend() gesendet. Das Ergebnis der Erfüllung dieser Funktion wird in die Struktur MqlTradeResult eingetragen, deren Feld *retcode* Rückkehrkode des Handelsservers enthält.

Kode	Identifikator	Beschreibung
10004	TRADE_RETCODE_REQUOTE	Requote
10006	TRADE_RETCODE_REJECT	Anforderung ist abgelehnt
10007	TRADE_RETCODE_CANCEL	Anforderung ist vom Trader aufgehoben
10008	TRADE_RETCODE_PLACED	Order ist platziert
10009	TRADE_RETCODE_DONE	Anforderung ist erfüllt
10010	TRADE_RETCODE_DONE_PARTIAL	Anforderung ist partiell erfüllt
10011	TRADE_RETCODE_ERROR	Fehler der Anforderung Verarbeitung
10012	TRADE_RETCODE_TIMEOUT	Anforderung abgelehnt nach Zeitablaufs
10013	TRADE_RETCODE_INVALID	Ungültige Anforderung
10014	TRADE_RETCODE_INVALID_VOLUME	Ungültiges Volumen in der Anforderung
10015	TRADE_RETCODE_INVALID_PRICE	Ungültiger Preis in der Anforderung
10016	TRADE_RETCODE_INVALID_STOPS	Ungültige Stops in der Anforderung
10017	TRADE_RETCODE_TRADE_DISABLED	Handel ist verboten
10018	TRADE_RETCODE_MARKET_CLOSED	Markt ist geschlossen
10019	TRADE_RETCODE_NO_MONEY	Nicht genug Geld, um Anforderung zu erfüllen
10020	TRADE_RETCODE_PRICE_CHANGED	Preise haben sich verändert
10021	TRADE_RETCODE_PRICE_OFF	Es gibt keine Preise für die Verarbeitung der Anforderung
10022	TRADE_RETCODE_INVALID_EXPIRATION	Ungültiges Datum des Orderablaufs in der Anforderung
10023	TRADE_RETCODE_ORDER_CHANGED	Order Status hat sich verändert
10024	TRADE_RETCODE_TOO_MANY_REQUESTS	Zuviele offene Anfragen
10025	TRADE_RETCODE_NO_CHANGES	Keine Veränderungen in der Anforderung
10026	TRADE_RETCODE_SERVER_DISABLES_AT	Autotrading ist vom Server untersagt
10027	TRADE_RETCODE_CLIENT_DISABLES_AT	Autotrading ist vom Client-Terminal untersagt
10028	TRADE_RETCODE_LOCKED	Anforderung ist für Verarbeitung blockiert

© 2000-2025, MetaQuotes Ltd.

Kode	Identifikator	Beschreibung
10029	TRADE_RETCODE_FROZEN	Order oder Position sind eingefroren
10030	TRADE_RETCODE_INVALID_FILL	Nicht unterstützter Durchführungstyp der Order nach dem Rest
10031	TRADE_RETCODE_CONNECTION	keine Verbindung mit dem Handelsserver
10032	TRADE_RETCODE_ONLY_REAL	Operation ist nur für reelle Kontos erlaubt
10033	TRADE_RETCODE_LIMIT_ORDERS	Limit für Anzahl der Warteordern ist erreicht
10034	TRADE_RETCODE_LIMIT_VOLUME	Limit für Volumen der Ordern und Positionen für dieses Symbol ist erreicht
10035	TRADE_RETCODE_INVALID_ORDER	Ungültiger oder verbotener Ordertyp
10036	TRADE_RETCODE_POSITION_CLOSED	Die Position mit dem angegebenen POSITION_IDENTIFIER wurde bereits geschlossen
10038	TRADE_RETCODE_INVALID_CLOSE_VOLUME	Das zu schließende Volumen übersteigt das aktuelle Volumen der Position
10039	TRADE_RETCODE_CLOSE_ORDER_EXIST	Für die angegebene Position ist bereits eine Verkaufsorder vorhanden. Kann im Hedging Modus auftreten: • beim Versuch, eine Position zur Gegenposition zu schließen, wenn es bereits Orders zum Schließen dieser Position gibt • beim Versuch, eine Position ganz oder teilweise zu schließen, wenn das Gesamtvolumen der vorhandenen Verkaufsorders und der zu platzierenden Order das aktuelle Volumen der Position übersteigt
10040	TRADE_RETCODE_LIMIT_POSITIONS	Die Anzahl offener Positionen, die man zur gleichen Zeit auf einem Konto haben kann, kann durch die Einstellungen des Servers beschränkt werden. Wenn ein Limit erreicht ist, gibt der Server beim Versuch, eine Order zu platzieren, den Fehler TRADE_RETCODE_LIMIT_POSITIONS zurück. Die Beschränkung funktioniert je nach dem Modus der Verrechnung von Positionen auf dem Konto: • Netting-Modus — die Anzahl offener Positionen wird berücksichtigt. Wenn ein Limit erreicht ist, lässt die Plattform keine weiteren Orders

© 2000-2025, MetaQuotes Ltd.

Kode	Identifikator	Beschreibung
		platzieren, deren Ausführung die Erhöhung der Anzahl offener Positionen verursachen kann. In Wirklichkeit lässt die Plattform Orders nur für die Symbole platzieren, für welche es offene Positionen bereits gibt. Im Netting-Modus werden aktuelle Pending Orders nicht berücksichtigt, denn die Ausführung einer Pending Order kann zur Änderung aktueller Positionen und nicht zur Erhöhung ihrer Anzahl führen. • Hedging-Modus — Pending Orders werden zusammen mit offenen Positionen berücksichtigt, denn die Auslösung einer Pending Order führt immer zur Eröffnung einer neuen Position. Wenn ein Limit erreicht ist, lässt die Plattform keine Marktorders für Eröffnung von Positionen sowie keine Pending Orders platzieren.
10041	TRADE_RETCODE_REJECT_CANCEL	Die Anfrage auf die Aktivierung der Pending Order wurde abgelehnt, die Order wurde abgebrochen
10042	TRADE_RETCODE_LONG_ONLY	Die Anfrage wurde abgelehnt, weil für das Symbol die Regel "**Nur Long-Positionen erlaubt**" (POSITION_TYPE_BUY) festgelegt wurde
10043	TRADE_RETCODE_SHORT_ONLY	Die Anfrage wurde abgelehnt, weil für das Symbol die Regel "**Nur Short-Positionen erlaubt**" (POSITION_TYPE_SELL) festgelegt wurde
10044	TRADE_RETCODE_CLOSE_ONLY	Die Anfrage wurde abgelehnt, weil für das Symbol die Regel "**Nur das Schließen vorhandener Postionen erlaubt**" festgelegt wurde
10045	TRADE_RETCODE_FIFO_CLOSE	Der Auftrag wurde zurückgewiesen, weil das Flag "**Positionsschließung nur nach FIFO-Regel erlaubt**" für das Handelskonto gesetzt ist (ACCOUNT_FIFO_CLOSE=true)
10046	TRADE_RETCODE_HEDGE_PROHIBITED	Die Anfrage wurde abgelehnt, weil für das Handelskonto die Regel festgelegt wurde "**Gegensätzliche Positionen ein und desselben Symbols ist deaktiviert**". Wenn das Konto z.B. eine Kaufposition hat, kann ein Nutzer keine Verkaufsposition eröffnen oder einen

© 2000-2025, MetaQuotes Ltd.

Kode	Identifikator	Beschreibung
		schwebenden Verkaufsauftrag platzieren. Die Regel wird nur auf Konten mit Hedging-Buchhaltungssystem angewendet (ACCOUNT_MARGIN_MODE=ACCOUNT_MARGIN_MODE_RETAIL_HEDGING).

© 2000-2025, MetaQuotes Ltd.

Compiler Warnungen

Compiler Warnungen haben Informationszwecke und sind keine Fehlernachrichten.

Nummer	Beschreibung
21	Unvollständige Datumaufzeichnung in Zeile datetime
22	Fehlerzahlen in der Zeile datetime für Datum, Anforderungen: Jahr 1970<=X<=3000 Monat 0<X<=12 Tag 0<X<= 31/30/28(29)....
23	Fehlerzahlen in der Zeile datetime für Zeit, Anforderungen: Stunde 0<=X<24 Minute 0<=X<60
24	Ungültige Farbe in RGB Format : eine der Komponenten RGB ist weniger als 0 oder mehr als 255
25	Unbekanntes Symbol der escape Folge. Bekannte: \n \r \t \\ \" \' \X \x
26	Volumen der lokalen Variablen der Funktion ist zu gross (>512Kb), reduzieren Sie die Zahl
29	Enumeration ist schon definiert (Duplizieren) - Glieder werden zur ersten Definition hinzugefügt
30	Neudefinieren des Makros
31	Variable ist erklärt, aber wird nicht verwendet
32	Konstruktor muss den Typ void haben
33	Destruktor muss den Typ void haben
34	Konstante passt nicht in Bereich von Ganzzahlen (X>_UI64_MAX || X<_I64_MIN) und wird in Typ double umgewandelt
35	Zu langes HEX - mehr als 16 Symbole (ältere Nibbles werden weggeschnitten)
36	Es gibt kein einziges Nibble in der HEX Zeile "0x"
37	Es gibt keine einzige Funktion - nichts zu erfüllen
38	verwendet wird eine nicht initialisierte Variable
41	Funktion hat keinen Körper und wird nicht aufgerufen
43	Mögliche Datenverluste bei Typenreduzierung. Beispiel: int x=(double)z;
44	Verlust der Genauigkeit (der Daten) bei der Umwandlung von Konstante. Konstante: int x=M_PI
45	Unterschiede der Zeichen von Operanden in Vergleichsoperationen. Beispiel: (char) c1>(uchar)c2

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
46	Probleme mit Funktionsimportieren - Erklärung von #import ist angefordert oder Import der Funktionen ist schon geschlossen
47	Beschreibung ist zu gross - Zusatzsymbole werden nicht in Ausführungsdatei einbeschlossen werden
48	Anzahl der erklärten Indikatorpuffer ist mehr als angefordert
49	Keine Farbe für Darstellung der graphischen Serie im Indikator
50	Es gibt keine einzige graphische Serie für Indikatordarstellung
51	Funktion-Bearbeiter 'OnStart" ist in Script nicht gefunden
52	Funktion-Bearbeiter 'OnStart" ist mit falschen Parametern definiert
53	Funktion 'OnStart' kann nur in Script definiert werden
54	Funktion 'OnInit' wird mit falschen Parametern definiert
55	Funktion 'OnInit' wird nicht in Scripts verwendet
56	Funktion 'OnDeinit' ist mit falschen Parametern definiert
57	Funktion 'OnDeinit' wird nicht in Scripts verwendet
58	Zwei Funktionen 'OnCalculate' sind definiert. OnCalculate() wird auf einem Preisfeld verwendet.
59	Überlauf bei Berechnung der komplizierten ganzzahligen Konstante ist festgestellt
60	Variable kann nicht initialisiert sein.
61	Diese Erklärung macht Aufruf der lokalen Variable , die in der angegebenen Zeile erklärt wird, nicht zugänglich
62	Diese Erklärung mach Aufruf der globalen Variable, die in der angegebenen Zeile erklärt wird, nicht zugänglich
63	Kann nicht für statische Felder verwendet werden
64	Diese Erklärung macht Aufruf der vordefinierte Variable nicht zugänglich
65	Der Wert des Ausdrucks ist immer true/false
66	Verwendung einer Variable oder Ausdruck des Typs bool in mathematischen Operationen ist unsicher
67	Das Ergebnis der Anwendung des unären Minus-Operators zum unsigned Typ ulong ist unbestimmt
68	Die Version in der Eigenschaft #property version kann nicht in Sektion Markt verwendet werden, das richtige Format ist #property version "XXX.YYY"
69	Leere kontrollierte Anweisung aufgetreten
70	Ungültig Rückgabetyp der Funktion oder falsche Parameter bei der Erklärung einer Event-Handler-Funktion

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
71	Eine explizite Umwandlung von Strukturen zu einem Typ ist erforderlich
72	Diese Erklärung macht einen direkten Zugang zum Mitglied einer in der angegebenen Zeichenfolge erklärten Klasse unmöglich. Der Zugang wird nur möglich sein, durch Operation mit Kontexterlaubnis ::
73	Die Konstante in dem binären Schreibung ist zu groß, die höherwertigen Bits werden verworfen werden
74	Parameter in der Methode der geerbten Klasse verschiedet sich in der Modifier const, hat die Kindfunktion die Basisfunktion überladen
75	Negativer oder zu großer für Offsetwert in Bit-Operation der Verschiebung, ist das Ergebnis undefiniert
76	Die Funktion muss einen Wert zurückgeben
77	Eine Funktion vom Typ void muss keinen Wert zurückgeben
78	Nicht alle Ausführungsvarianten geben den Rückgabewert zurück
79	Ausdrücke auf der globalen Ebene sind nicht erlaubt
80	Mögliche Fehler in der Folge von Operationen, verwenden Sie Klammern um die Reihenfolge explizit anzugeben
81	Zwei Aufrufe der Funktion OnCalCulate() sind gefunden. Die Variante mit Zeitrahmen OHLC wird verwendet
82	Die Struktur enthält keine Mitglieder, wird die Größe auf 1 Byte gesetzt
83	Keine Bearbeitung des Ergebnisses der Funktion
84	Der Ressource-Indikator ist im Debug-Modus kompiliert. Dies verringert die Leistung. Um die Geschwindigkeit zu erhöhen kompilieren Sie den Indikator neu
85	Eine zu große Zeichencode in der Zeile, muss im Bereich von 0 bis 65535 sein
86	Nicht erkannte Sonderzeichen in der Zeichenfolge
87	Keine Fenster-Eigenschaft für den Indikator (die die Anzeige im Hauptfenster oder im Unterfenster festsetzt). Eigenschaft #property indicator_chart_window wird anwendet

© 2000-2025, MetaQuotes Ltd.

Kompilierungsfehler

MetaEdtior 5, Editor der mql5-Programme, zeigt Fehlermeldungen des Programms, die vom eingebauten Compiler während der Compilierung detektiert werden. Die Liste der Fehler ist in der Tabelle unten angegeben. Für Kompilierung des Ausgangskodes in den ablaufenden Kode drücken Sie **F7**. Programme mit Fehlern können nicht kompiliert werden, bis die vom Compiler angegebene Fehler eliminiert werden.

Nummer	Beschreibung
100	Dateilesen Fehler
101	Fehler der *.EX5 Dateieröffnung, um sie für Speicherung zu schreiben
103	Freier Speicher nicht genug, um Kompilierung zu beenden
104	Leere syntaktische Einheit, die vom Compiler nicht erkannt wird
105	Unkorrekter Dateiname in #include
106	Dateizugang in #include (Datei kann schon nicht existieren)
108	Ungeeigneter Name für #define
109	Unbekannter Befehl des Preprozessors (gültig #include,#define,#property,#import)
110	Unbekanntes für Compiler Symbol
111	Funktion nicht realisiert (Beschreibung ist vorhanden, Körper gibt es nicht)
112	Doppeltes Anführungszeichen ist ausgelassen (")
113	Öffnende Eckklammer (<)oder doppeltes Anführungszeichen (") ist ausgelassen
114	Einzelnes Anführungszeichen ist ausgelassen (')
115	Schließende Winkelklammer ">" ist ausgelassen
116	In der Erklärung ist der Typ nicht angegeben
117	Keine Rückgabeanweisung return oder nicht in allen Branchen der Erfüllung
118	Öffnende Klammer der Aufrufparameter wurde erwartet
119	Schreibfehler EX5
120	Unkorrekter Zugang zum Feldelement
121	Funktion ist nicht des Typs void und Anweisung return muss den Wert zurückgeben
122	Unkorrekte Erklärung des Destruktors
123	Doppelpunkt fehlt ":"
124	Variable ist schon erklärt
125	Variable mit diesem Identifikator ist schon erklärt
126	Name der Variable ist zu lang (>250 Symbole)

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
127	Struktur mit demselben Identifikator ist schon definiert
128	Struktur ist nicht definiert
129	Strukturglied mit demselben Namen ist schon definiert
130	Kein solches Strukturglied
131	Paarigkeit der eckigen Klammern ist verletzt
132	Öffnende runde Klammer "(" ist zu erwarten
133	Nicht balancierte geschweifte Klammern (es fehlt "}")
134	Kompliziert für Kompilierung (zu viele Branchen, interner Levelsteck ist übervoll)
135	Fehler der Dateieröffnung für Lesen
136	Nicht genug Speicherplatzes , um Ausgangsdatei in den Speicher zu laden
137	Variable ist zu erwarten
138	Referenz kann nicht initialisiert werden
140	Zuordnung war zu erwarten (entsteht bei der Erklärung)
141	Öffnende geschweifte Klammer "{" ist zu erwarten
142	Parameter kann nur ein dynamisches Feld sein
143	Verwenden des Typs "void" ist ungültig
144	Kein Paar für ")" oder "]", d.h. es fehlt "(" oder "["
145	Kein Paar für "(" oder "[",d.h. es fehlt ")" oder "]"
146	Unkorrekte Feldgröße
147	Zu viele Parameter (>64)
149	Dieses token ist nicht hier zu erwarten
150	Ungültiges Verwenden der Operation (falsche Operanden)
151	Ausdruck des Typs void ist unzulässig
152	Anweisung ist zu erwarten
153	Falsches Verwenden von break
154	Semikolon ist zu erwarten ";"
155	Komma ist zu erwarten ","
156	Typ muss als eine Klasse, nicht als eine Struktur bestimmt werden
157	Ausdruck ist zu erwarten
158	In HEX tritt "nicht HEX Symbol" auf oder zu lange Zahl (Zifferanzahl > 511)

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
159	Zeile-Konstante hat mehr als 65534 Symbole
160	Funktionsdefinieren ist hier unzugänglich
161	Unerwartetes Programmende
162	Vorrangige Erklärung für Strukturen ist verboten
163	Funktion mit demselben Namen ist schon definiert und hat einen anderen Typ des rückgegebenen Wertes
164	Funktion mit demselben Namen ist schon definiert und hat einen anderen Parametervorrat
165	Funktion mit demselben Namen ist schon definiert und implementiert
166	Überladen der Funktion für diesen Anruf ist nicht gefunden
167	Funktion mit dem Rückgabewert des Typs void kann den Wert nicht rückgeben
168	Funktion ist nicht definiert
170	Wert ist zu erwarten
171	Im Ausdruck case sind nur ganzzahlige Konstanten gültig
172	Wert für case in diesem switch wurde schon verwendet
173	Ganzzahliger Wert ist zu erwarten
174	Im Ausdruck #import ist Dateiname zu erwarten
175	Ausdrücke sind nicht auf dem globalen Niveau zugelassen
176	Runde Klammer ")" vor ";" ist ausgelassen
177	Links von Gleichheitszeichen ist die Variable erwartet
178	Ausdrucksergebnis wird nicht verwendet
179	Variablenerklärung in case ist nicht zugelassen
180	Implizite Umwandlung der Zeile in die Zahl
181	Implizite Umwandlung der Zahl in die Zeile
182	Nichteindeutiger Aufruf der überladenen Funktion (mehrere Überladungen passen)
183	Ungültiges else ohne entsprechendes if
184	Ungültiges case oder default ohne entsprechendes switch
185	Ungültiges Verwenden der Ellipse
186	Initialisirungsfolge hat mehr Elementen als Initialisierungsvariable
187	Konstante für case ist zu erwarten
188	Konstantausdruck ist angefordert

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
189	Konstantvariable kann nicht verändert werden
190	Rechte Klammer oder Komma ist zu erwarten (Erklärung des Feldgliedes)
191	Enum ID wird bereits verwendet
192	Enumeration kann keine Zugangsmodifikatoren haben (const, extern, static)
193	Enumerationsglied ist schon mit dem anderen Wert erklärt
194	Es gibt eine Variable, die mit demselben Namen definiert ist
195	Es gibt eine Struktur, die mit demselben Namen definiert wird
196	Name des Enumerationsgliedes ist zu erwarten
197	Ganzzahliger Ausdruck ist zu erwarten
198	Dividieren durch Null im Konstantausdruck
199	Falsche Parameterzahl in der Funktion
200	Parameter per Referenz muss Variable sein
201	Variable desselben Typs für Übertragung durch Referenz wird erwartet
202	Konstantvariable kann nicht durch nicht Konstantreferenz übertragen werden
203	Ganzzahlige positive Konstante ist angefordert
204	Fehler des Zugangs zum geschützten Glied der Klasse
205	Import ist schon anders definiert
208	Durchführungsdatei nicht erzeugt
209	für Indikator ist kein Eingangspunkt gefunden 'OnCalculate'
210	Anweisung continue kann nur innerhalb des Zyklus verwendet werden
211	Fehler des Zuganges zum private(geschlossenen) Glied der Klasse
213	Strukturmethode oder Klassenmethode ist nicht erklärt
214	Fehler des Zugangs zur private(geschlossenen) Methode der Klasse
216	Kopieren der Strukturen mit Objekten ist nicht zulässig
218	Index außerhalb der Feldgrenzen
219	Felderinitialisierung in Erklärung der Struktur oder der Klasse ist ungültig
220	Konstruktor der Klasse kann keine Parameter haben
221	Destruktor der Klasse kann keine Parameter haben
222	Methode der Klasse oder der Struktur mit denselben Parametern ist schon erklärt
223	Operand ist zu erwarten

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
224	Es gibt schon die Methode mit diesem Namen, aber mit anderen Parametern (Erklärung!=Implementierung)
225	Zu importierende Funktion ist nicht beschrieben
226	Die Funktion ZeroMemory() kann nicht für Klassen mit geschützten Members oder Vererbung
227	Unklarer Aufruf der überladenen Funktion (exaktes Zusammenfallen der Parameter für mehrere Überladungen)
228	Variablenname ist zu erwarten
229	Referenz kann nicht an dieser Stelle erklärt werden
230	Schon verwendet als Enumerationsname
232	Klasse oder Struktur ist zu erwarten
235	delete kann nicht für Feldentfernung aufgerufen werden
236	Anweisung ' while ' ist zu erwarten
237	In delete muss den Anzeiger geben
238	Es gibt schon default für dieses switch
239	Syntaktischer Fehler
240	Escape-Folge kann nur in Zeilen auftreten (fängt mit '\' an)
241	Feld angefordert - Eckige Klammer '[' gehört zum Feld nicht oder oder keine Felder als Feldparameters
242	Kann nicht durch intialisierende Folge initialisiert werden
243	Import ist nicht definiert
244	Fehler des Optimierers auf dem syntaktischen Baum
245	Zu viele Strukturen sind erklärt (versuchen Sie das Programm zu vereinfachen)
246	Parameterumwandlung ist ungültig
247	Unkorrektes Verwenden der Anweisung delete
248	Anzeiger für Referenz kann nicht erklärt werden
249	Es ist nicht zugelassen Referenz für Referenz zu erklären
250	Es ist nicht zugelassen, Anzeiger für Anzeiger zu erklären
251	Strukturerklärung in der Parameterliste ist ungültig
252	Ungültige Operation der Typenreduzierung
253	Anzeiger kann nur für Klasse oder Struktur erklärt werden
256	Nicht erklärter Identifikator

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
257	Fehler des Optimierers des Durchführungskodes
258	Fehler der Generierung des Durchführungskodes
260	Ungültiger Ausdruck für Anweisung switch
261	Pool der Zeilenkonstanten ist überfüllt, vereinfachen Sie das Programm
262	Unmöglich zur Enumeration zu reduzieren
263	*virtual* kann nicht für Daten verwendet werden (Glieder der Klasse oder der Struktur)
264	Geschützte Methode der Klasse kann nicht aufgerufen werden
265	Umdefinierte virtuelle Funktion gibt anderen Typ zurück
266	Klasse kann nicht von der Struktur geerbt werden
267	Struktur kann nicht von der Klasse geerbt werden
268	Konstruktor kann nicht virtuell sein (Spezifikator virtual ist ungültig)
269	Struktur kann keine virtuelle Methoden haben
270	Funktion muss den Körper haben
271	Überladen der Systemfunktionen (Funktionen des Terminals) ist verboten
272	Spezifikator const ist für die Funktionen, die keine Klassen- oder Strukturglieder sind, ungültig
274	Klassenglieder können nicht in der Konstantmethode geändert werden
276	Unpassende Initialisierungsfolge
277	Defaultwert für Parameter ist ausgelassen (Spezifik der Erklärung von Defaultparametern)
278	Neubestimmen des Default-Parameters (in der Erklärung und Realisierung sind die Werte verschieden)
279	Nicht-Konstantmethode für Konstantobjekt kann nicht aufgerufen werden
280	Für Zugang zu Gliedern wird das Objekt gefordert (Punkt für eine Nichtklasse/Struktur gestellt)
281	Name der schon erklärten Struktur kann nicht bei der Erklärung verwendet werden
284	Nicht autorisierte Umwandlung (bei der geschlossenen Vererbung)
285	Strukturen und Felder können nicht als input-Variablen verwendet werden
286	Spezifikator const ist für Konstruktor/Destruktor ungültig
287	Falsche Zeilenausdruck für Typ datetime
288	unbekannte Eigenschaft (#property)

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
289	unkorrekter Wert für Eigenschaft
290	unkorrekter Index für Eigenschaft in #property
291	Anrufparameter ist ausgelassen - < func(x,) >
293	Objekt muss per Referenz übertragen werden
294	Feld muss per Referenz übertragen werden
295	Funktion wurde als zu exportierende Funktion erklärt
296	Funktion wurde nicht als zu exportierende Funktion erklärt
297	Zu importierende Funktion kann nicht exportiert werden
298	Zu importierende Funktion kann nicht diesen Parameter haben (Anzeiger, Klasse oder Struktur, die dynamisches Feld, Anzeiger, Llasse usw. enthalten, können nicht übertragen werden)
299	Muss eine Klassen sein
300	#import ist nicht abgeschlossen
302	Nicht-Kompatibilität der Typen
303	extern-Variable ist schon initialisiert
304	Keine einzige exportierte Funktion oder Standardeingangspunkt ist gefunden
305	Expliziter Aufruf des Konstruktors ist verboten
306	Methode wurde als konstant erklärt
307	Methode wurde nicht als konstant erklärt
308	Unkorrekte Größe der Ressource-Datei
309	Unkorrekter Ressource-Name
310	Fehler beim Öffnen der Ressource-Datei
311	Fehler beim Lesen der Ressource-Datei
312	Unbekannter Typ der Ressource
313	Unkorrekte Pfad zur Ressource-Datei
314	Der angegebene Ressource-Name wird bereits verwendet
315	Erwartete Parameter des Makros
316	Nach dem Makronamen muss ein Leerzeichen stehen
317	Fehler in der formalen Parameter des Makros
318	Falsche Anzahl von Parametern bei der Verwendung eines Makros
319	Zu viele Parameter für ein Makro

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
320	Zu komplex, das Makro soll vereinfacht werden
321	Parameter für EnumToString() kann nur Enumeration sein
322	Ressourcename ist zu lang
323	Bildformate ist nicht unterstützt (nur BMP mit 24 oder 32 Bit Farbtiefe ist erlaubt)
324	Ein Array kann nicht im Operator erklärt werden
325	Die Funktion kann nur auf globalen Bereich erklärt werden
326	Die Erklärung ist nicht für den aktuellen Sichtbereich erlaubt
327	Initialisierung der statischen Variablen mit Werte der lokalen Variablen ist ungültig
328	Unerlaubte Deklaration der Array von Objekten, die keinen Standardkonstruktor haben
329	Initialisierungsliste ist nur für Konstruktoren erlaubt
330	Es gibt keine Definition der Function nach der Initialisierungsliste
331	Die Initialisierungsliste ist leer
332	Array-Initialisierung im Konstruktor ist verboten
333	Initialisierung der Mitglieder der Elternklasse in der Initialisierungsliste ist verboten
334	Ein Ausdruck des Typs Integer wurde erwartet
335	Erforderlicher Speicher für das Array übersteigt den maximalen Wert
336	Erforderlicher Speicher für die Struktur übersteigt den maximalen Wert
337	Erforderlicher Speicher für die auf globaler Ebene deklarierte Variablen übersteigt den maximalen Wert
338	Erforderlicher Speicher für die lokale Variablen übersteigt den maximalen Wert
339	Der Konstruktor ist nicht definiert
340	Ungültige Name der Symboldatei
341	Konnte nicht die Symboldatei unter dem angegebenen Pfad öffnen
342	Ungültig Symboldateiformat, es muss ICO sein
343	Neuinitialisierung eines Mitglieds in einer Klasse/Struktur Konstruktor mit der Initialisierungsliste
344	Initialisierung der statischen Mitglieder in der Initialisierungsliste des Konstruktors ist nicht erlaubt
345	Initialisierung der nichtstatischen Mitglied einer Klasse/Struktur auf globaler Ebene ist nicht erlaubt
346	Der Name der Methode der Klasse/Struktur stimmt mit der zuvor deklarierten Namen eines Mitglieds

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
347	Der Name des Mitglieds der Klasse/Struktur stimmt mit der zuvor deklarierten Namen einer Methode
348	Virtuelle Funktion kann nicht als static deklariert werden
349	Modifikator const ist für eine statische Funktion nicht erlaubt
350	Konstruktor oder Destruktor können nicht statisch sein
351	Nicht-statischer Mitglied/Methode einer Klasse oder einer Struktur kann nicht von einer statischen Funktion zugegriffen werden
352	Nach dem Schlüsselwort operator wurde eine überladene Operation (+,-,[],++,-- usw.) erwartet
353	Nicht alle Operationen können in MQL5 überladen werden
354	Definition entspricht nicht der Deklaration
355	Eine ungültige Anzahl der Parameter ist für den Operator angegeben
356	Event-Handling-Funktion nicht gefunden
357	Methoden können nicht exportiert sein
358	Ein Zeiger auf das konstante Objekt kann nicht in einen Zeiger auf ein nicht-konstantes Objekt umwandelt werden
359	Klasse-Schablonen werden noch nicht unterstützt
360	Überladen von Funktionsschablonen wird noch nicht unterstützt
361	Sie können nicht eine Funktionsschablone benutzen
362	Unklare Parameter in einer Funktionsschablone (verschiedene Typen von Parametern passen)
363	Es ist unmöglich zu bestimmen, in welchen Parametertyp den Argument von Funktionsschablone umzuwandeln
364	Falsche Anzahl von Parametern in einer Funktionsschablone
365	Funktionsschablone kann nicht virtuell sein
366	Funktionsschablonen können nicht exportiert werden
367	Funktionsschablonen können nicht importiert werden
368	Strukturen mit Objekten nicht akzeptabel sind
369	String Arrays und Strukturen, die Objekte enthalten, sind ungültig
370	Ein Statisches Mitglied der Klasse/Struktur soll explizit initialisiert werden
371	Die Beschränkung der Compiler: eine Ziele kann nicht mehr als 65.535 Zeichen enthalten
372	Inkonsistent #ifdef/#endif

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
373	Funktionsergebnis kann kein Objekt der Klasse sein, weil es keine Kopierkonstruktor gibt
374	Verwenden Sie keine nicht-statische Glieder und/oder Methoden für Initialisierung statischer Variablen
375	OnTesterInit() kann nicht ohne Deklaration von OnTesterDeinit() verwendet werden
376	Name der Lokalvariable ist mit dem Namen eines der Funktionsparameter gleich
377	Makros __FUNCSIG__ und __FUNCTION__ können nicht außerhalb des Funktionskörpers verwendet werden
378	Ungültiger Rückgabetyp. Dies ist ein Fehler für aus DLL importierten Funktionen, die eine Struktur oder einen Zeiger zurückgeben
379	Fehler bei der Anwendung der Vorlage
380	Nicht verwendet
381	Ungültige Syntax bei der Deklaration einer rein virtuellen Funktion, nur "=NULL" oder "=0" erlaubt
382	Nur virtuelle Funktionen können mit einem reinen Spezifizierer ("=NULL" oder "=0") deklariert werden
383	Instanz von abstrakter Klasse kann nicht erstellt werden
384	Für die dynamische Umwandlung mithilfe von dynamic_cast muss der Pointer auf benutzerdefinierten Typ als Zieltyp dienen
385	Typ "Pointer auf Funktion" wird erwartet
386	Pointer auf Methoden sind nicht verfügbar
387	Fehler - der Typ des Pointers auf Funktion kann nicht definiert werden
388	Typumwandlung wegen der privaten Vererbung nicht möglich
389	Die Variable mit dem Modifikator const muss beim Deklarieren initialisiert werden
393	Nur Methoden mit dem öffentlichen Zugriff können in einer Schnittstelle deklariert werden
394	Ungültige Verschachtelung einer Schnittstelle innerhalb einer anderen Schnittstelle
395	Eine Schnittstelle kann nur von einer anderen Schnittstelle abgeleitet werden
396	Eine Schnittstele wird erwartet
397	Die Schnittstellen unterstützen nur öffentliche Vererbung
398	Die Schnittstelle kann keine Mitglieder enthalten
399	Objekte der Schnittstelle können nicht direkt erstellt werden, verwenden Sie nur Vererbung
400	Ein Spezifikator kann nicht in einer Vorwärtsdeklaration verwendet werden.

© 2000-2025, MetaQuotes Ltd.

Nummer	Beschreibung
401	Ein Ableitung aus der Klasse ist nicht möglich, da sie mit dem Spezifikator final deklariert wurde.
402	Eine Methode, die mit dem Spezifikator final deklariert wurde, kann nicht redefiniert werden.
403	Der Spezifikator final kann nur auf virtuelle Funktionen angewendet werden.
404	Die Methode, die durch den Spezifikator override gekennzeichnet ist, überschreibt tatsächlich keine Basisklassenfunktion.
405	Ein Spezifikator darf nicht bei der Definition einer Funktion, sondern nur bei einer Deklaration verwendet werden.
406	Der Typ nicht auf den angegebenen umgewandelt werden
407	Der Typ kann nicht für eine Ressource-Variable verwendet werden.
408	Fehler in der Projektdatei
409	Kann nicht als Mitglied einer union verwendet werden.
410	Mehrdeutige Wahl des Namens, der Verwendungskontext sollte explizit definiert werden.
411	Die Struktur kann nicht aus der DLL verwendet werden.
412	Kann keine Funktion aufrufen, die durch den Spezifikator delete deklariert wurde.
413	MQL4 wird nicht unterstützt. Um das Programm zu kompilieren, müssen Sie den MetaEditor aus dem Installationsverzeichnis des MetaTrader 4 verwenden.

© 2000-2025, MetaQuotes Ltd.

Ausführungsfehler

GetLastError() - Funktion, die die letzte Fehlerzahl zurückgibt, die in der vorbestimmten Variable _LastError gespeichert wird. Der Wert dieser Variable muss durch die Funktion ResetLastError() auf Null gestellt werden.

Konstante	Wert	Beschreibung
ERR_SUCCESS	0	Die Operation wurde erfolgreich ausgeführt
ERR_INTERNAL_ERROR	4001	Unerwarteter interner Fehler
ERR_WRONG_INTERNAL_PARAMETER	4002	Fehlerwert beim internen Aufruf des Client-Terminals
ERR_INVALID_PARAMETER	4003	Fehlerwert beim Aufruf der Systemfunktion
ERR_NOT_ENOUGH_MEMORY	4004	Nicht genug Speicherplatz für Ausführung der Systemfunktion
ERR_STRUCT_WITHOBJECTS_ORCLASS	4005	Struktur hat Zeilenobjekte und/oder Objekte der dynamischen Felder und/oder Strukturen mit solchen Objekten und/oder Klassen
ERR_INVALID_ARRAY	4006	Array mit unpassendem Typ, falscher Größe oder beschädigtes Objekt des dynamischen Arrays
ERR_ARRAY_RESIZE_ERROR	4007	Zu kleine Arraygröße für die geforderte Größe oder der Versuch der Größenänderung eines statischen Arrays
ERR_STRING_RESIZE_ERROR	4008	Zeichenkette ist zu groß
ERR_NOTINITIALIZED_STRING	4009	nicht initialisierte Zeichenkette
ERR_INVALID_DATETIME	4010	Ungültiger Wert des Datums und/oder der Zeit
ERR_ARRAY_BAD_SIZE	4011	Die Gesamtzahl der Elemente im Array darf nicht größer als 2147483647 sein
ERR_INVALID_POINTER	4012	Ungültiger Anzeiger
ERR_INVALID_POINTER_TYPE	4013	Falscher Typ des Pointers
ERR_FUNCTION_NOT_ALLOWED	4014	Systemfunktion ist für Anruf nicht erlaubt

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_RESOURCE_NAME_DUPLICATED	4015	Die Namen der dynamischen und statischen Ressource sind gleich
ERR_RESOURCE_NOT_FOUND	4016	Ressource mit diesem Namen ist nicht in EX5 gefunden
ERR_RESOURCE_UNSUPPORTED_TYPE	4017	Nicht unterstützte Ressource-Typ oder Größe von mehr als 16 MB
ERR_RESOURCE_NAME_IS_TOO_LONG	4018	Der Name der Ressource ist mehr als 63 Zeichen
ERR_MATH_OVERFLOW	4019	Es ist ein Speicherüberlauf bei der Berechnung der mathematischen Funktion aufgetreten
ERR_SLEEP_ERROR	4020	Das aufgerufene Sleep() überschreitet den Zeitpunkt des Testendes.
ERR_PROGRAM_STOPPED	4022	Der Test wurde zwangsweise von außen beendet. Beispielsweise wurde die Optimierung unterbrochen, das visuelle Testfenster geschlossen oder der Test-Agent angehalten.
ERR_INVALID_TYPE	4023	Ungültiger Typ
ERR_INVALID_HANDLE	4024	Ungültiges Handle
ERR_TOO_MANY_OBJECTS	4025	Obergrenze der Objektanzahl erreicht.
Charts		
ERR_CHART_WRONG_ID	4101	Falscher Identifikator des Charts
ERR_CHART_NO_REPLY	4102	Chart antwortet nicht
ERR_CHART_NOT_FOUND	4103	Chart ist nicht gefunden
ERR_CHART_NO_EXPERT	4104	Im Chart gibt es keinen Expert Advisor, der das Ereignis verarbeiten kann
ERR_CHART_CANNOT_OPEN	4105	Fehler bei Charteröffnung
ERR_CHART_CANNOT_CHANGE	4106	Fehler bei Veränderung des Symbols und der Periode des Charts
ERR_CHART_WRONG_PARAMETER	4107	Ein fehlerhafter Parameterwert für Funktionen für die Arbeit mit dem Chart

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_CHART_CANNOT_CREATE_TIMER	4108	Fehler bei Timererzeugung
ERR_CHART_WRONG_PROPERTY	4109	Falsche Charteigenschaft ID
ERR_CHART_SCREENSHOT_FAILED	4110	Fehler bei Screenshot Erzeugung
ERR_CHART_NAVIGATE_FAILED	4111	Fehler der Navigierung durch Chart
ERR_CHART_TEMPLATE_FAILED	4112	Fehler der Schablonenverwendung
ERR_CHART_WINDOW_NOT_FOUND	4113	Subfenster mit angegebenen Indikator nicht gefunden
ERR_CHART_INDICATOR_CANNOT_ADD	4114	Fehler beim Hinzufügen eines Indikators zu einem Chart
ERR_CHART_INDICATOR_CANNOT_DEL	4115	Fehler beim Entfernen des Indikators aus dem Chart
ERR_CHART_INDICATOR_NOT_FOUND	4116	Der Indikator kann nicht auf dem angegebenen Chart gefunden werden
Graphische Objekte		
ERR_OBJECT_ERROR	4201	Fehler bei Arbeit mit dem graphischen Objekt
ERR_OBJECT_NOT_FOUND	4202	Graphisches Objekt ist nicht gefunden
ERR_OBJECT_WRONG_PROPERTY	4203	Fehleridentifikator der Eigenschaft des graphischen Objektes
ERR_OBJECT_GETDATE_FAILED	4204	Datum entsprechend dem Wert kann nicht erhalten werden
ERR_OBJECT_GETVALUE_FAILED	4205	Wert entsprechend dem Datum kann nicht erhalten werden
MarketInfo		
ERR_MARKET_UNKNOWN_SYMBOL	4301	Unbekanntes Symbol
ERR_MARKET_NOT_SELECTED	4302	Symbol nicht gewählt in MarketWatch
ERR_MARKET_WRONG_PROPERTY	4303	Fehleridentifikator der Symboleigenschaft
ERR_MARKET_LASTTIME_UNKNOWN	4304	Zeit des Letzten Ticks ist unbekannt (es gab keine Ticks)

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_MARKET_SELECT_ERROR	4305	Fehler beim Hinzufügung oder Löschung eines Symbols in MarketWatch
ERR_MARKET_SELECT_LIMIT	4306	Das Limit der ausgewählten Symbole in MarketWatch wurde überschritten.
ERR_MARKET_SESSION_INDEX	4307	Falsche Session-ID beim Aufruf der Funktionen SymbolInfoSessionQuote/SymbolInfoSessionTrade.
Zugang zur Geschichte		
ERR_HISTORY_NOT_FOUND	4401	Angeforderte Historie ist nicht gefunden
ERR_HISTORY_WRONG_PROPERTY	4402	Falscher Identifikator der Eigenschaften der Historie
ERR_HISTORY_TIMEOUT	4403	Zeitrüberschreitung beim Abruf der Historie
ERR_HISTORY_BARS_LIMIT	4404	Die Anzahl der abgerufenen Balken ist in den Einstellungen des Terminals begrenzt
ERR_HISTORY_LOAD_ERRORS	4405	Zahlreiche Fehler beim Laden der Historie
ERR_HISTORY_SMALL_BUFFER	4407	Das aufnehmende Array ist zu klein, um alle abgerufenen Daten zu speichern
Global_Variables		
ERR_GLOBALVARIABLE_NOT_FOUND	4501	Globale Variable des Client-Terminals ist nicht gefunden
ERR_GLOBALVARIABLE_EXISTS	4502	Globale Variable des Client-Terminals mit demselben Namen existiert schon
ERR_GLOBALVARIABLE_NOT_MODIFIED	4503	Globale Variablen wurden nicht modifiziert
ERR_GLOBALVARIABLE_CANNOTREAD	4504	Die Datei mit den Werten der globalen Variablen konnte nicht geöffnet und gelesen werden
ERR_GLOBALVARIABLE_CANNOTWRITE	4505	Die Datei mit den Werten der globalen Variablen konnte nicht geschrieben werden

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_MAIL_SEND_FAILED	4510	Fehler beim Sender der E-Mail
ERR_PLAY_SOUND_FAILED	4511	Fehler beim Abspielen eines Tons
ERR_MQL5_WRONG_PROPERTY	4512	Falscher Identifikator der Programmeigenschaft
ERR_TERMINAL_WRONG_PROPERTY	4513	Falscher Identifikator der Terminalseigenschaft
ERR_FTP_SEND_FAILED	4514	Fehler beim Dateisenden via ftp
ERR_NOTIFICATION_SEND_FAILED	4515	Benachrichtigung konnte nicht gesendet werden
ERR_NOTIFICATION_WRONG_PARAMETER	4516	Ungültige Parameter, um Benachrichtigung zu senden - ein leerer String oder NULL war in die Funktion SendNotification() übergeben
ERR_NOTIFICATION_WRONG_SETTINGS	4517	Falsche Einstellung von Benachrichtigungen im Terminal (ID ist nicht eingegeben oder keine Berechtigung ist aufgesetzt)
ERR_NOTIFICATION_TOO_FREQUENT	4518	Zu häufige Zusendung von Benachrichtigungen
ERR_FTP_NOSERVER	4519	Ftp-Server wurde nicht in den Einstellungen definiert
ERR_FTP_NOLOGIN	4520	Ftp-Benutzername wurde nicht in den Einstellungen definiert
ERR_FTP_FILE_ERROR	4521	Datei existiert nicht
ERR_FTP_CONNECT_FAILED	4522	Verbindung mit dem Ftp-Server fehlgeschlagen
ERR_FTP_CHANGEDIR	4523	Upload-Verzeichnis auf dem Ftp-Server nicht gefunden
Puffer der Benutzerindikatoren		
ERR_BUFFERS_NO_MEMORY	4601	Nicht genug Speicherplatz für Neuzuordnung der Indikatorpuffer
ERR_BUFFERS_WRONG_INDEX	4602	Falscher Index des Indikatorpuffers
Eigenschaften der Benutzerindikatoren		
ERR_CUSTOM_WRONG_PROPERTY	4603	Falscher Identifikator der Eigenschaft von Benutzerindikator

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
Account		
ERR_ACCOUNT_WRONG_PROPERTY	4701	Falscher Identifikator der Kontoeigenschaft
ERR_TRADE_WRONG_PROPERTY	4751	Falscher Identifikator der Handelseigenschaft
ERR_TRADE_DISABLED	4752	Handel für Expert ist verboten
ERR_TRADE_POSITION_NOT_FOUND	4753	Position ist nicht gefunden
ERR_TRADE_ORDER_NOT_FOUND	4754	Order ist nicht gefunden
ERR_TRADE_DEAL_NOT_FOUND	4755	Deal ist nicht gefunden
ERR_TRADE_SEND_FAILED	4756	Fehler beim Senden der Handelsanforderung
ERR_TRADE_CALC_FAILED	4758	Berechnung des Profits oder der Margin fehlgeschlagen
Indikator		
ERR_INDICATOR_UNKNOWN_SYMBOL	4801	Unbekanntes Symbol
ERR_INDICATOR_CANNOT_CREATE	4802	Indikator kann nicht erzeugt werden
ERR_INDICATOR_NO_MEMORY	4803	Nicht genug Speicherplatz, um den Indikator zu starten
ERR_INDICATOR_CANNOT_APPLY	4804	Indikator kann nicht auf den anderen Indikator angewendet werden
ERR_INDICATOR_CANNOT_ADD	4805	Fehler beim Hinzufügen des Indikators
ERR_INDICATOR_DATA_NOT_FOUND	4806	Angeforderte Daten nicht gefunden
ERR_INDICATOR_WRONG_HANDLE	4807	Handle des Indikators ist ungültig
ERR_INDICATOR_WRONG_PARAMETERS	4808	Falsche Parameterzahl beim Indikator
ERR_INDICATOR_PARAMETERS_MISSING	4809	Fehlende Parameter beim Indikator
ERR_INDICATOR_CUSTOM_NAME	4810	Der erste Parameter im Feld muss der Name des nutzerdefinierten Indikators sein
ERR_INDICATOR_PARAMETER_TYPE	4811	Falscher Parametertyp für den Indikator

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_INDICATOR_WRONG_INDEX	4812	Falscher Index des angeforderten Indikatorpuffers
Markttiefe		
ERR_BOOKS_CANNOT_ADD	4901	DOM kann nicht zugefügt werden
ERR_BOOKS_CANNOT_DELETE	4902	DOM kann nicht entfernt werden
ERR_BOOKS_CANNOT_GET	4903	DOM Dasten können nicht erhalten werden
ERR_BOOKS_CANNOT_SUBSCRIBE	4904	Fehler in Subskription neue DOM Daten zu erhalten
Dateioperationen		
ERR_TOO_MANY_FILES	5001	Mehr als 64 Dateien können nicht gleichzeitig eröffnet werden
ERR_WRONG_FILENAME	5002	Ungültiger Dateiname
ERR_TOO_LONG_FILENAME	5003	Zu langer Dateiname
ERR_CANNOT_OPEN_FILE	5004	Fehler der Dateieröffnung
ERR_FILE_CACHEBUFFER_ERROR	5005	Nicht genug Speicherplatz, um Cache zu lesen
ERR_CANNOT_DELETE_FILE	5006	Fehler beim Löschen der Datei
ERR_INVALID_FILEHANDLE	5007	Datei mit diesem Handle ist schon geschlossen oder konnte nicht erstellt werden
ERR_WRONG_FILEHANDLE	5008	Falsches Handle der Datei
ERR_FILE_NOTTOWRITE	5009	Datei muss für Schreiben eröffnet werden
ERR_FILE_NOTTOREAD	5010	Datei muss für Lesen eröffnet werden
ERR_FILE_NOTBIN	5011	Datei muss als binäre Datei eröffnet werden
ERR_FILE_NOTTXT	5012	Datei muss als Textdatei eröffnet werden
ERR_FILE_NOTTXTORCSV	5013	Datei muss als Textdatei oder CSV eröffnet werden
ERR_FILE_NOTCSV	5014	Datei muss als CSV eröffnet werden
ERR_FILE_READERROR	5015	Fehler des Datei Lesen

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_FILE_BINSTRINGSIZE	5016	Es muss Zeilengröße angegeben werden, denn die Datei ist als binäre Datei eröffnet
ERR_INCOMPATIBLE_FILE	5017	für Zeilenfelder muss Textdatei sein, für andere - binäre Datei
ERR_FILE_IS_DIRECTORY	5018	Das ist keine Datei, sondern ein Verzeichnis
ERR_FILE_NOT_EXIST	5019	Datei existiert nicht
ERR_FILE_CANNOT_REWRITE	5020	Datei kann nicht neugeschrieben werden
ERR_WRONG_DIRECTORYNAME	5021	Falscher Verzeichnisname
ERR_DIRECTORY_NOT_EXIST	5022	Verzeichnis existiert nicht
ERR_FILE_ISNOT_DIRECTORY	5023	Das ist eine Datei, nicht Verzeichnis
ERR_CANNOT_DELETE_DIRECTORY	5024	Verzeichnis kann nicht entfernt werden
ERR_CANNOT_CLEAN_DIRECTORY	5025	Misslungen, das Verzeichnis zu löschen (eine oder mehrere Dateien können blockiert werden und Operation der Entfernung ist misslungen)
ERR_FILE_WRITEERROR	5026	Ressource konnte nicht in die Datei geschrieben werden
ERR_FILE_ENDOFFILE	5027	Es ist unmöglich, das nächste Datenfragment aus der CSV-Datei (FileReadString, FileReadNumber, FileReadDatetime, FileReadBool) zu lesen, Dateiende erreicht
Zeilenreduzierung		
ERR_NO_STRING_DATE	5030	In der Zeile gibt es kein Datum
ERR_WRONG_STRING_DATE	5031	Falsches Datum in der Zeile
ERR_WRONG_STRING_TIME	5032	Falsche Zeit in der Zeile
ERR_STRING_TIME_ERROR	5033	Fehler der Umwandlung der Zeile in Datum
ERR_STRING_OUT_OF_MEMORY	5034	Nicht genug Speicherplatz für Zeile
ERR_STRING_SMALL_LEN	5035	Zeilenlänge ist kleiner als erwartet

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_STRING_TOO_BIGNUMBER	5036	Zu lange Zahl, länger als ULONG_MAX
ERR_WRONG_FORMATSTRING	5037	Falsche Formatzeile
ERR_TOO_MANY_FORMATTERS	5038	Zahl der Formatspezifikationen ist größer, als die Parameterzahl
ERR_TOO_MANY_PARAMETERS	5039	Parameterzahl ist mehr als Zahl der Formatspezifikatoren
ERR_WRONG_STRING_PARAMETER	5040	Beschädigter Parameter des Typs
ERR_STRINGPOS_OUTOFRANGE	5041	Position außerhalb der Zeile
ERR_STRING_ZEROADDED	5042	Zum Zeilenende wird 0 hinzugefügt, eine sinnlose Operation
ERR_STRING_UNKNOWNTYPE	5043	Unbekannter Datentyp bei Umwandlung in die Zeile
ERR_WRONG_STRING_OBJECT	5044	Beschädigtes Zeichenkettenobjekt
Operationen mit Feldern		
ERR_INCOMPATIBLE_ARRAYS	5050	Kopieren von inkompatiblen Arrays. Ein Zeichenarray kann nur in ein Zeichenarray kopiert werden, numerisches Array in ein numerisches.
ERR_SMALL_ASSERIES_ARRAY	5051	Zielarray ist als AS_SERIES erklärt, und es ist von der unzureichender Größe
ERR_SMALL_ARRAY	5052	Zu kleines Feld, Startposition außerhalb des Feldes
ERR_ZEROSIZE_ARRAY	5053	Array mit der Größe Null
ERR_NUMBER_ARRAYS_ONLY	5054	Muss ein numerisches Array sein
ERR_ONEDIM_ARRAYS_ONLY	5055	Muss eindimensionales Array sein
ERR_SERIES_ARRAY	5056	Zeitreihe kann nicht verwendet werden
ERR_DOUBLE_ARRAY_ONLY	5057	Muss ein Array des Typs double sein
ERR_FLOAT_ARRAY_ONLY	5058	Muss ein Array des Typs float sein
ERR_LONG_ARRAY_ONLY	5059	Muss ein Array des Typs long sein
ERR_INT_ARRAY_ONLY	5060	Muss ein Array des Typs int sein

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_SHORT_ARRAY_ONLY	5061	Muss ein Array des Typs short sein
ERR_CHAR_ARRAY_ONLY	5062	Muss ein Array des Typs char sein
ERR_STRING_ARRAY_ONLY	5063	Nur Array vom Typ string
Arbeit mit OpenCL		
ERR_OPENCL_NOT_SUPPORTED	5100	OpenCL-Funktionen werden auf diesem Computer nicht unterstützt
ERR_OPENCL_INTERNAL	5101	Interner Fehler bei der Ausführung von OpenCL
ERR_OPENCL_INVALID_HANDLE	5102	Ungültiges OpenCL-Handle
ERR_OPENCL_CONTEXT_CREATE	5103	Fehler beim Erstellen von OpenCL-Kontext
ERR_OPENCL_QUEUE_CREATE	5104	Es konnte kein Lauf-Warteschlange in OpenCL erstellen
ERR_OPENCL_PROGRAM_CREATE	5105	Fehler beim Kompilieren eines OpenCL-Programms
ERR_OPENCL_TOO_LONG_KERNEL_NAME	5106	Zu lange Kernel-Name(OpenCL-Kernel)
ERR_OPENCL_KERNEL_CREATE	5107	Fehler beim Erstellen vom OpenCL-Kernel
ERR_OPENCL_SET_KERNEL_PARAMETER	5108	Fehler bei Eingabe von Parameters für den OpenCL-Kernel
ERR_OPENCL_EXECUTE	5109	Laufzeitfehler vom OpenCL-Programm
ERR_OPENCL_WRONG_BUFFER_SIZE	5110	Ungültige Größe des OpenCL-Puffers
ERR_OPENCL_WRONG_BUFFER_OFFSET	5111	Ungültiges Offset im OpenCL-Puffer
ERR_OPENCL_BUFFER_CREATE	5112	Fehler beim Erstellen des OpenCL-Puffers
ERR_OPENCL_TOO_MANY_OBJECTS	5113	Maximale Anzahl der OpenCL-Objekte überschritten
ERR_OPENCL_SELECTDEVICE	5114	Fehler bei der Auswahl des OpenCl-Device

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
Arbeiten mit der Datenbank		
ERR_DATABASE_INTERNAL	5120	Interner Datenbankfehler
ERR_DATABASE_INVALID_HANDLE	5121	Ungültiges Datenbankhandle
ERR_DATABASE_TOO_MANY_OBJECTS	5122	Übersteigt die maximal erlaubte Anzahl von Datenbankobjekten
ERR_DATABASE_CONNECT	5123	Verbindungsfehler zur Datenbank
ERR_DATABASE_EXECUTE	5124	Fehler bei der Anfrageausführung
ERR_DATABASE_PREPARE	5125	Fehler bei der Anfrageerstellung
ERR_DATABASE_NO_MORE_DATA	5126	Keine zu lesenden Daten mehr
ERR_DATABASE_STEP	5127	Der Wechsel zur nächsten Anfrage ist fehlgeschlagen
ERR_DATABASE_NOT_READY	5128	Die Daten zum Lesen der Anfrageergebnisse sind noch nicht bereit
ERR_DATABASE_BIND_PARAMETERS	5129	Das automatische Ersetzen der Parameter für eine SQL-Anfrage ist fehlgeschlagen
Operationen mit WebRequet		
ERR_WEBREQUEST_INVALID_ADDRESS	5200	Ungültige URL
ERR_WEBREQUEST_CONNECT_FAILED	5201	Verbindung mit der angegebenen URL fehlgeschlagen
ERR_WEBREQUEST_TIMEOUT	5202	Zeitliches Limit für das Erhalten von Daten überschritten
ERR_WEBREQUEST_REQUEST_FAILED	5203	Bei der Ausführung der HTTP-Anfrage ist ein Fehler aufgetreten
Operationen mit einem Netzwerk (Sockets)		
ERR_NETSOCKET_INVALIDHANDLE	5270	Ungültiges Handle eines Socket wurde der Funktion übergeben
ERR_NETSOCKET_TOO_MANY_OPENED	5271	Zu viele offene Sockets (max 128)
ERR_NETSOCKET_CANNOT_CONNECT	5272	Verbindung zum entfernten Host ist fehlgeschlagen
ERR_NETSOCKET_IO_ERROR	5273	Senden/Empfangen der Daten vom Socket ist fehlgeschlagen
ERR_NETSOCKET_HANDSHAKE_FAILED	5274	Das Herstellen einer sicheren Verbindung (TLS Handshake) ist fehlgeschlagen

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_NETSOCKET_NO_CERTIFICATE	5275	Das Zertifikat zum Schutz der Verbindung enthält keine Daten
Benutzerdefinierte Symbole		
ERR_NOT_CUSTOM_SYMBOL	5300	Ein benutzerdefinierter Symbol muss angegeben werden
ERR_CUSTOM_SYMBOL_WRONG_NAME	5301	Ungültiger Name des benutzerdefinierten Symbols. Der Name des Symbols darf nur lateinische Buchstaben ohne Satzzeichen, Leerzeichen und Sonderzeichen (".", "_", "&" und "#") enthalten. Es ist nicht empfehlenswert, Symbole <, >, :, ", /,\, |, ?, * zu verwenden.
ERR_CUSTOM_SYMBOL_NAME_LONG	5302	Der Name des benutzerdefinierten Symbols ist zu lang. Der Name des Symbols darf nicht länger als 32 Zeichen inklusive der abschließenden 0 sein
ERR_CUSTOM_SYMBOL_PATH_LONG	5303	Der Pfad zum benutzerdefinierten Symbol ist zu lang. Der Pfad darf nicht länger als 128 Zeichen einschließlich "Custom\\", Symbolnamens, Trennzeichens von Gruppen und der abschließenden 0 sein
ERR_CUSTOM_SYMBOL_EXIST	5304	Ein benutzerdefinierter Symbol mit diesem Namen existiert bereits
ERR_CUSTOM_SYMBOL_ERROR	5305	Fehler beim Erstellen, Löschen oder Ändern eines benutzerdefinierten Symbols
ERR_CUSTOM_SYMBOL_SELECTED	5306	Versuch, das in der Marktübersicht (Market Watch) ausgewählte benutzerdefinierte Symbol zu löschen
ERR_CUSTOM_SYMBOL_PROPERTY_WRONG	5307	Falsche Eigenschaft des benutzerdefinierten Symbols
ERR_CUSTOM_SYMBOL_PARAMETER_ERROR	5308	Falscher Parameter beim Setzen der Eigenschaft des benutzerdefinierten Symbols

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_CUSTOM_SYMBOL_PARAMETER_LONG	5309	Der String-Parameter beim Setzen der Eigenschaft des benutzerdefinierten Symbols ist zu lang
ERR_CUSTOM_TICKS_WRONG_ORDER	5310	Die Ticks im Array sind nicht in zeitlicher Reihenfolge geordnet
Wirtschaftskalender		
ERR_CALENDAR_MORE_DATA	5400	Arraygröße ist zu klein für die Übernahme aller Werte
ERR_CALENDAR_TIMEOUT	5401	Zeitlimit der Anforderung ist überschritten
ERR_CALENDAR_NO_DATA	5402	Land wurde nicht gefunden
Arbeiten mit der Datenbank		
ERR_DATABASE_ERROR	5601	Allgemeiner Fehler
ERR_DATABASE_LOGIC	5602	SQLite interner Logikfehler
ERR_DATABASE_PERM	5603	Zugriff verweigert
ERR_DATABASE_ABORT	5604	Durch die Rückrufroutine bedingter Abbruch
ERR_DATABASE_BUSY	5605	Die Datei der Datenbank ist blockiert
ERR_DATABASE_LOCKED	5606	Die Tabelle der Datenbank ist blockiert
ERR_DATABASE_NOMEM	5607	Nicht genügend Speicherplatz, die Operation zu beenden
ERR_DATABASE_READONLY	5608	Versuch, in eine schreibgeschützte Datenbank zu schreiben
ERR_DATABASE_INTERRUPT	5609	Operation wurde durch sqlite3_interrupt() beendet
ERR_DATABASE_IOERR	5610	E/A-Fehler der Festplatten
ERR_DATABASE_CORRUPT	5611	Das Festplattenabbild der Datenbank ist beschädigt.
ERR_DATABASE_NOTFOUND	5612	Unbekannter Operationscode in sqlite3_file_control()
ERR_DATABASE_FULL	5613	Das Einfügen ist fehlgeschlagen, weil die Datenbank voll ist

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_DATABASE_CANTOPEN	5614	Die Datei der Datenbank kann nicht geöffnet werden
ERR_DATABASE_PROTOCOL	5615	Fehler im Sperrprotokoll der Datenbank
ERR_DATABASE_EMPTY	5616	Ausschließlich für den internen Gebrauch
ERR_DATABASE_SCHEMA	5617	Verändertes Schema der Datenbank
ERR_DATABASE_TOOBIG	5618	Zeichenkette oder BLOB überschreitet die Größenbeschränkung
ERR_DATABASE_CONSTRAINT	5619	Abbruch wegen Verletzung der Einschränkungen
ERR_DATABASE_MISMATCH	5620	Nicht übereinstimmende Datentypen
ERR_DATABASE_MISUSE	5621	Fehlerhafte Verwendung der Bibliothek
ERR_DATABASE_NOLFS	5622	Verwendete Betriebssystemfunktionen werden vom Host nicht unterstützt
ERR_DATABASE_AUTH	5623	Authentifizierung ist fehlgeschlagen
ERR_DATABASE_FORMAT	5624	nicht verwendet
ERR_DATABASE_RANGE	5625	Parameterfehler bei der Einbindung, Indexfehler
ERR_DATABASE_NOTADB	5626	Geöffnete Datei ist keine Datenbankdatei
Matrizen und Vektoren		
ERR_MATRIX_INTERNAL	5700	Interner Fehler des Matrix/Vektor-Ausführungssubsystems
ERR_MATRIX_NOT_INITIALIZED	5701	Matrix/Vektor wurde nicht initialisiert
ERR_MATRIX_INCONSISTENT	5702	Inkonsistente Größe von Matrizen/Vektoren im lfd. Betrieb
ERR_MATRIX_INVALID_SIZE	5703	Ungültige Matrix-/Vektorgröße
ERR_MATRIX_INVALID_TYPE	5704	Ungültiger Matrix-/Vektor-Typ

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
ERR_MATRIX_FUNC_NOT_ALLOWED	5705	Funktion nicht verfügbar für diese Matrix/Vektor
ERR_MATRIX_CONTAINS_NAN	5706	Matrix/Vektor enthält Nicht-Zahlen (Nan/Inf)
ONNX Modelle		
ERR_ONNX_INTERNAL	5800	ONNX interner Fehler
ERR_ONNX_NOT_INITIALIZED	5801	ONNX Initialisierungsfehler zur Laufzeit des APIs
ERR_ONNX_NOT_SUPPORTED	5802	Von MQL5 nicht unterstützte Eigenschaft oder Wert
ERR_ONNX_RUN_FAILED	5803	ONNX Fehler zur Laufzeit des APIs
ERR_ONNX_INVALID_PARAMETERS_COUNT	5804	Ungültige Anzahl von Parametern, die an OnnxRun übergeben wurden
ERR_ONNX_INVALID_PARAMETER	5805	Ungültiger Parameterwert
ERR_ONNX_INVALID_PARAMETER_TYPE	5806	Ungültiger Parametertyp
ERR_ONNX_INVALID_PARAMETER_SIZE	5807	Ungültige Parametergröße
ERR_ONNX_WRONG_DIMENSION	5808	Tensor-Dimension nicht gesetzt oder ungültig
Benutzerdefinierte Fehler		
ERR_USER_ERROR_FIRST	65536	Mit diesem Kode fangen benutzerdefinierte Fehler an

Siehe auch

Rückgabecodes des Handelsservers

© 2000-2025, MetaQuotes Ltd.

Eingabe/Ausgabe Konstanten

Konstanten:

- Flaggen der Dateieroeffnung
- Dateieigenschaften
- Positionieren innerhalb der Datei
- Kodeseite Verwenden
- MessageBox

© 2000-2025, MetaQuotes Ltd.

Flaggen der Dateieroeffnung

Werte von Flaggen, die das Regime von Arbeit mit Dateien bestimmen. Flaggen werden folgenderweise definiert:

Identifikator	Wert	Beschreibung
FILE_READ	1	Datei wird für Lesen geoeffnet. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet. Bei der Öffnung der Datei muss unbedingt die Flagge FILE_WRITE und/oder die Flagge FILE_READ angegeben wird.
FILE_WRITE	2	Dateinwird für Schreiben verwendet. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet. Bei der Öffnung der Datei muss unbedingt die Flagge FILE_WRITE und/oder die Flagge FILE_READ angegeben wird.
FILE_BIN	4	Binares Regime von Lesen-Shreiben (ohne Umwandlung von Zeile in die Zeile). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet
FILE_CSV	8	Datei des Typs csv (alle geschriebenen Elemente werden in die Zeilen des entsprechenden Typs verwendet, unicode oder ansi, und durch Begrenzungszeichen getrennt). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet
FILE_TXT	16	Einfache Textdatei (ebensolcher wie csv, aber der Begrenzungszeichen wird nicht beachtet). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet
FILE_ANSI	32	Zeilen des Typs ANSI (1-Byte Symbole). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet
FILE_UNICODE	64	Zeilen des Typs UNICODE (2-Byte-Symbole). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet
FILE_SHARE_READ	128	Gemeinsamer Lesezugang von mehreren Progarmmen. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet, aber ersetzt es nicht die Notwendigkeit, beim Dateieröffneng FILE_WRITE und/oder FILE_READ anzugeben
FILE_SHARE_WRITE	256	Gemeinsamer Schreibezugang von mehreren Programmen. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet, aber ersetzt es nicht die Notwendigkeit, beim Dateieröffneng FILE_WRITE und/oder FILE_READ anzugeben
FILE_REWRITE	512	Moeglichket der Dateiumspeicherung durch die Funktionen FileCopy() undnden sein oder für FileMove(). Datei muss vorhanden sein oder für

© 2000-2025, MetaQuotes Ltd.

Identifikator	Wert	Beschreibung
		Schreiben geoeffnet werden. Anders wird die Datei nicht geoeffnet werden
FILE_COMMON	4096	Datei /// im Verzeichnis aller Client-Terminals \Terminal\Common\Files. Flagge wird beim Oeffnen von Dateien (FileOpen()), Dateikopieren (FileCopy(), FileMove()) und Pruefung von Dateianesenheit (FileIsExist()) verwendet

Bei der Öffnung der Datei kann eine oder mehrere Flaggen angegeben werden, diese Verbindung heisst Kombination der Flaggen. Kombination der Flaggen wird mittels des Zeichens der Operation des logischen ODER (|) geschrieben, das zwischen den aufgezählten Flaggen gestellt wird. ZB um eine Datei in CSV Format für Lesen und Schreiben zu öffnen, kann die Kombination FILE_READ|FILE_WRITE|FILE_CSV angegeben werden.

Beispiel:

```
int filehandle=FileOpen(filename,FILE_READ|FILE_WRITE|FILE_CSV);
```

Es gibt einige Besonderheiten der Arbeit bei Andeutung der Flaggen des Lesens und der Schreibung:

- Wenn FILE_READ angegeben wird, versucht man, die schon existierende Datei zu öffnen. Wenn die Datei existiert, gelingt es nicht die Datei zu öffnen, eine neue Datei wird nicht erzeugt.
- Wenn FILE_READ|FILE_WRITE - eine neue Date wird erzeugt, wenn die Datei mit diesem Namen fehlt.
- Wenn FILE_WRITE - Datei wird mit der Null-Grösse erzeugt.

Bei der Öffnung der Datei muss unbedingt die Flagge FILE_WRITE und/oder die Flagge FILE_READ angegeben wird.

Flaggen, die den Typ der offenen Datei bestimmen, haben die Priorität. Die älteste Flagge ist FILE_CSV, dann folgt die zweitälteste Flagge FILE_BIN, und die kleinste Priorität hat die Flagge FILE_TXT. So wenn mehrere Dateien gleichzeitig (FILE_TXT|FILE_CSV oder FILE_TXT|FILE_BIN oder FILE_BIN|FILE_CSV) angegeben werden, wird die älteste Flagge verwendet werden.

Flaggen, die den Typ der Kodierung bestimmen, haben auch eine Priorität. Flagge FILE_UNICODE hat die ältere Priorität als die Flagge FILE_ANSI. Darum wird bei der Andeutung der Kombination FILE_UNICODE|FILE_ANSI die Flagge FILE_UNICODE verwendet.

Wenn FILE_UNICODE, und FILE_ANSI angegeben wird, wird FILE_UNICODE gemeint. Wenn FILE_CSV, und FILE_BIN nicht angegeben wird, und FILE_TXT, wird FILE_CSV gemeint

Wenn die Datei als Textdatei für Lesen geöffnet wird (FILE_TXT oder FILE_CSV), und dabei wird am Anfang der Datei ein spezielles Zwei-Byte Zeichen 0xff,0xfe gefunden, wird Flagge der Kodierung FILE_UNICODE sein, auch wenn die Flagge FILE_ANSI angegeben wurde.

Sehen Sie auch

Dateioperationen

© 2000-2025, MetaQuotes Ltd.

Dateieigenschaften

Die Funktion FileGetInteger(), ist für den Erhalt von Dateieigenschaften verwendet. Der Identifier der gewünschten Eigenschaft ist dieser Funktion aus ENUM_FILE_PROPERTY_INTEGER beim Aufruf übergegeben.

ENUM_FILE_PROPERTY_INTEGER

ID	ID Beschreibung
FILE_EXISTS	Prüfung der Existenz
FILE_CREATE_DATE	Erstellungsdatum
FILE_MODIFY_DATE	Datum der letzten Überarbeitung
FILE_ACCESS_DATE	Datum der letzten Zugriff zur Datei
FILE_SIZE	Dateigröße in Byte
FILE_POSITION	Position eines Zeigers in der Datei
FILE_END	Zeichen der Dateiende erhalten
FILE_LINE_END	Zeichen der Zeilenende erhalten
FILE_IS_COMMON	Die Datei wird in einem freigegebenen Ordner für allen Terminals geöffnet (Sehen Sie FILE_COMMON)
FILE_IS_TEXT	Die Datei ist als eine Textdatei geöffnet (Sehen Sie FILE_TXT)
FILE_IS_BINARY	Die Datei ist als eine binäre Datei geöffnet (Sehen Sie FILE_BIN)
FILE_IS_CSV	Die Datei ist als CSV geöffnet (Sehen Sie FILE_CSV)
FILE_IS_ANSI	Die Datei ist als ANSI geöffnet (Sehen Sie FILE_ANSI)
FILE_IS_READABLE	Die geöffnete Datei ist lesbar (Sehen Sie FILE_READ)
FILE_IS_WRITABLE	Die geöffnete Datei ist beschreibbar (Sehen Sie FILE_WRITE)

Die Funktion FileGetInteger() hat zwei Varianten der Abruf In der ersten Version, um die Eigenschaften der Datei zu erhalten, eingeben Sie ihr Handle, der beim Öffnen Der Datei mit Funktion FileOpen() erhalten war. Diese Version ermöglicht es Ihnen, alle Dateieigenschaften zu bekommen.

Die zweite Version der Funktion FileGetInteger() gibt Werte der Dateieigenschaften durch ihren Namen zurück. In dieser Version können Sie nur die folgende allgemeine Eigenschaften erhalten:

- FILE_EXISTS - ob die Datei mit dem eingegebenen Namen existiert;
- FILE_CREATE_DATE - Erstellungsdatum der Datei mit dem eingegebenen Namen;
- FILE_MODIFY_DATE - Datum der letzten Überarbeitung der Datei mit dem eingegebenen Namen;
- FILE_ACCESS_DATE - Datum der letzten Zugriff zur Datei mit dem eingegebenen Namen;
- FILE_SIZE - Größe der Datei mit dem eingegebenen Namen.

Wenn Sie andere Eigenschaften als die oben genannten zu erhalten versuchen, gibt die zweite Version der FileGetInteger() einen Fehler zurück.

© 2000-2025, MetaQuotes Ltd.

Positionieren innerhalb der Dateien

Die meisten Dateifunktionen sind mit der Funktionen von Lesen/Aufzeichnung der Information verbunden. Dabei kann durch die Funktion FileSeek() der Dateianzeiger in der Position innerhalb der Datei angegeben werden, von der die naechste Lese- oder Schreibeoperation durchgeführt wird. Enumeration ENUM_FILE_POSITION enthält zulaessige Anzeigerpositionen, in Bezug auf die Verschiebungen in Bytes für naechste Operation spezifiziert werden können.

ENUM_FILE_POSITION

Identifikator	Beschreibung
SEEK_SET	Dateianfang
SEEK_CUR	Laufende Position des Dateianzeigers
SEEK_END	Dateiende

Sehen Sie auch

FileIsEnding, FileIsLineEnding

© 2000-2025, MetaQuotes Ltd.

Kodeseite Verwenden in Zeilenumwandlung

Beim Konvertieren der Zeilenvariablen in Felder des Typs char und umgekehrt wird in der Sprache MQL5 Kodieren verwendet, das dem laufenden ANSI Kodieren des Operationssystems Windows (CP_ACP) entspricht. Wenn man braucht, einen anderen Kodierungstyp anzugeben, kann man es durch Zusatzparameter für Funktionen CharArrayToString(), StringToCharArray() und FileOpen() einstellen.

In der Tabelle werden eingebaute Konstanten für einige am meisten angeforderte Kodeseiten angegeben. Nicht angegebene Kodeseiten können mit Hilfe des Kodes angegeben werden, der dieser Seite entspricht.

Eingebaute Konstanten der Kodeseiten

Konstante	Wert	Beschreibung
CP_ACP	0	Laufende Kodeseite ANSIim Operationssystem Windows
CP_OEMCP	1	Laufende Kodeseite OEM.
CP_MACCP	2	Laufende Kodeseite Macintosh. **Bemerkung:** Dieser Wert wird vor allem in früher erzeugten Programmkodes verwendet und ist nicht notwendig, denn moderne Computers Macintosh verwenden Unicode Kodieren.
CP_THREAD_ACP	3	Kodieren Windows ANSI für laufenden Thread.
CP_SYMBOL	42	Kodeseite Symbol
CP_UTF7	65000	Kodeseite UTF-7.
CP_UTF8	65001	Kodeseite UTF-8.

Sehen Sie auch

Zustand von Client Terminal

© 2000-2025, MetaQuotes Ltd.

Konstanten des Dialogfensters MessageBox

Rückgabekodes der Funktion MessageBox(). Wenn Message window Schaltfläche Cancel hat, gibt die Funktion den Wert IDCANCEL bei der gedrückten Taste ESC oder Schaltfläche Cancel zurück. Wenn message window keine Schaltfläche Cancel hat, wird das Drücken ESC keinen Effekt haben.

Konstante	Wert	Beschreibung
IDOK	1	Schaltfläche OK wird ausgewählt
IDCANCEL	2	Schaltfläche Cancel wird ausgewählt
IDABORT	3	Schaltfläche Abort wird ausgewählt
IDRETRY	4	Schaltfläche Retry wird ausgewählt
IDIGNORE	5	Schaltfläche Ignore wird ausgewählt
IDYES	6	Schaltfläche Yes wird ausgewählt
IDNO	7	Schaltfläche No wird ausgewählt
IDTRYAGAIN	10	Schaltfläche Try Again wird ausgewählt
IDCONTINUE	11	Schaltfläche Continue wird ausgewählt

Die Hauptflaggen der Funktion MessageBox() bestimmen den Inhalt und das Verhalten des Dialogfensters. Dieser Wert kann Flaggenkombination aus folgennden Flaggengruppen sein:

Konstante	Wert	Beschreibung
MB_OK	0x00000000	Message Window hat eine Schaltfläche: OK. Default
MB_OKCANCEL	0x00000001	Message Window hat zwei Schaltflächen: OK und Cancel
MB_ABORTRETRYIGNORE	0x00000002	Message Window hat drei Schaltflächen: Abort, Retry und Ignore
MB_YESNOCANCEL	0x00000003	Message Window hat drei Schaltflächen: Yes, No und Cancel
MB_YESNO	0x00000004	Message Window hat zwei Schaltflächen: Yes und No
MB_RETRYCANCEL	0x00000005	Message Window hat zwei Schaltflächen: Retry und Cancel
MB_CANCELTRYCONTINUE	0x00000006	Message Window hat drei Schaltflächen: Cancel, Try Again, Continue

für Darstellung der Ikone muss man in Message Window zusätzliche Flaggen spezifiziert werden:

© 2000-2025, MetaQuotes Ltd.

Konstante	Wert	Beschreibung
MB_ICONSTOP, MB_ICONERROR, MB_ICONHAND	0x00000010	Darstellung von STOP Zeichen
MB_ICONQUESTION	0x00000020	Darstellung von Fragezeichen
MB_ICONEXCLAMATION, MB_ICONWARNING	0x00000030	Darstellung von Aufrufzeichen
MB_ICONINFORMATION, MB_ICONASTERISK	0x00000040	Darstellung aus dem eingekreisten Zeichen i

Default -0 Schaltfläche werden durch folgende Flaggen vorgegeben:

Konstante	Wert	Beschreibung
MB_DEFBUTTON1	0x00000000	Die erste Schaltfläche MB_DEFBUTTON1 - ist Default, wenn MB_DEFBUTTON2, MB_DEFBUTTON3, oder MB_DEFBUTTON4 nicht spezifiziert werden
MB_DEFBUTTON2	0x00000100	zweite Schaltfläche - Default-Wert
MB_DEFBUTTON3	0x00000200	dritte Schaltfläche - Default-Wert
MB_DEFBUTTON4	0x00000300	Vierte Schaltfläche - Default-Wert

© 2000-2025, MetaQuotes Ltd.

Programme MQL5

fürdie Arbeit des mql5-Progarmms muss es kompiliert werden (Druckknopf "Kompilieren" oder Taste F7). Compilierung muss ohne Fehler vergehen (zugelassen sind Warnungen, die analysiert werden muessen). Dabei muss im entsprechenden Verzeichnis, *terminal_dir\MQL5\Experts*, *terminal_dir\MQL5\indicators* oder *terminal_dir\MQL5\scripts*, Durchführungsdatei mit demselben Namen und Erweiterung EX5 erzeugt werden. Gerade diese Datei muss abgelaufen lassen.

Besonderheiten der mql5-Programme werden in folgenden Abschnitten beschrieben:

- Programmausführung – Verfahren des Aufrufs der vorbestimmten Funktionen-Bearbeiter der Ereignisse;
- Testen von Handelsstrategien – Besonderheiten von Programmen im Strategie-Tester;
- Ereignisse des Client-Terminals – Beschreibung der Ereignisse, die in Programmen verarbeitet werden können;
- Aufruf der importierten Funktionen – Verfahren der Beschreibung, zulaessige Parameter, Suchedaten und Abkommen der importierten Funktionen;
- Durchführungsfehler – Erhalten der Information über Durchführungsfehler und kritische Fehler.

Experten, Benutzeranzeiger und Scripts werden zum einen der offenen Charts durch Drag'n'Drop aus dem Fenster "Navigator" des Client-Terminals auf entsprechenden Chart angehängt. Mql5-Programme können nur beim angeschaltenen Client-Terminals funktionieren.

Damit Expert zu arbeiten stoppt, muss er vom Chart durch die Wahl "Ratgeber - Entfernen" vom Kontextmenue des Charts entfernt werden. Arbeit der Ratgeber werden auch vom Verhalten des Knopfes "Ratgeber Erlauben/verbieten" beinflusst.

Damit Benutzeranzeiger zu arbeiten stoppt, muss man ihn vom Chart entfernen.

Benutzeranzeiger und Ratgeber arbeiten bis sie explizit vom Chart entfernt werden; Information über angehängte Ratgeber und Benutzeranzeiger wird zwischen Starts des Client-Terminals gespeichert.

Scripts werden einmal ausgeführt und nach Arbeitabschluss oder bei Veränderung des Verhaltens des laufenden Charts oder nach Abschluss des Client-Terminals automatisch entfernt. Beim neuen Start des Client-Terminals werden Scripts nicht ablaufen lassen, denn die Information über sie wird nicht gespeichert.

Auf einem Chart können ein Expert, ein Script und unbegrenzte Zahl der Anzeiger funktionieren.

Dienste erfordern keine Bindung an ein Chart, um zu funktionieren, und sind für die Ausführung von Hilfsfunktionen konzipiert. Mit einem Dienst können Sie beispielsweise ein nutzerspezifisches Symbol erstellen, dessen Chart öffnen, Daten dafür in einer Endlosschleife mit den Netzwerkfunktionen empfangen und ständig aktualisieren.

© 2000-2025, MetaQuotes Ltd.

Durchführung der Programme

Jedes Skript, jeder Dienst und jeder Expert Advisor läuft in einem separaten Thread. Alle Indikatoren, die auf einem Symbol berechnet werden, auch wenn sie verschiedenen Charts zugeordnet sind, arbeiten im gleichen Thread. Somit teilen sich alle Indikatoren auf einem Symbol die Ressourcen eines Threads.

Alle anderen Aktionen, die mit einem Symbol verbunden sind, wie die Verarbeitung von Ticks und die Synchronisation der Geschichte werden auch konsequent in dem selben Thread mit Indikatoren durchgeführt. Dies bedeutet, dass, wenn der Indikator hält unendliche Aktion, alle anderen Ereignisse in diesem Symbol nie durchgeführt werden.

Beim Starten eines Expert Advisors, stellen Sie sicher dass er aktuelle Handelsumwelt hat und Geschichte des Symbols für die Periode, und synchronisieren Sie Daten zwischen dem Terminal und dem Server. Für alle diese Verfahren stellt das Terminal eine Startverzögerung von nicht mehr als 5 Sekunden, wonach der Expert Advisor mit verfügbaren Daten gestartet wird. Daher, wenn es kein Verbindung zum Server gibt, kann dies zu einer Verzögerung des Starts des Expert Advisors führen.

Eine kurze Zusammenfassung des MQL5-Programmen ist in der Tabelle dargestellt:

Programm	Durchführung	Bemerkung
Service (Dienst)	Ein separater Thread, die Anzahl der Threads für Dienste ist gleich der Anzahl der Dienste.	Ein Dienst mit einer Schleife kann den Betrieb anderer Programme nicht unterbrechen.
Skript	In einem eigenen Thread, wie viele Skripte, so viele Ausführungsthreads für sie	Ein endloses Skript kann nicht Ausführung andere Programme brechen
Expert Advisor	In einem eigenen Thread, wie viele Expert Advisors, so viele Ausführungsthreads für sie	Ein endloser Expert Advisor nicht Ausführung andere Programme brechen
Indikator	Ein Thread für alle Indikatoren auf ein Symbol. Wie viele Symbole mit Indikatoren, so viele Ausführungsthreads für sie	Eine Endlosschleife in einem des Indikators wird Ausführung aller anderen Indikatoren auf diesem Symbol stoppen

Sofort nach Anhängen des Programms zum Chart Erfolgt sein Einspeichern in Client-Terminal und in Speicher des Client-Terminals und Initialisierung der globalen Variablen. Wenn eine der globalen Variablen des Typen Klasse Konstrukteur hat, wird dieser Konstrukteur während der Initialisierung der globalen Variablen aufgerufen werden.

Danach wartet das Programm auf das Ereignis vom Client-Terminal. Jedes mql5-Programm muss wenigstens eine Funktion-Bearbeiter des Ereignisses haben, sonst wird das Programm nicht durchgeführt werden. Funktionen-Bearbeiter haben vorbestimmte Namen, vorbestimmte Parametervorraete und vorbestimmte Rückkehrtypen.

© 2000-2025, MetaQuotes Ltd.

Typ	Funktionsname	Parameter	Anwendung	Bemerkung
int	OnInit	keine	Experten und Anzeiger	Bearbeiter des Ereignisses Init. Zugelassen Typ des Rückgabewertes void.
void	OnDeinit	const int reason	Experten und Anzeiger	Bearbeiter des Ereignisses Deinit.
void	OnStart	keine	Skripte und Dienste	Start Ereignisbehandlung.
int	OnCalculate	const int rates_total, const int prev_calculated, const datetime &Time[], const double &Open[], const double &High[], const double &Low[], const double &Close[], const long &TickVolume[], const long &Volume[], const int &Spread[]	Anzeiger	Bearbeiter des Ereignisses Calculate für alle Preisdaten.
int	OnCalculate	const int rates_total, const int prev_calculated, const int begin, const double &price[]	Anzeiger	Ereignisbearbeiter Calculate in einem Datenfeld. In Indikator können nicht 2 Bearbeiter Calculate gleichzeitig sein. In diesem Fall wird nur ein Ereignisbearbeiter Calculate in einem Datenfeld funktionieren.
void	OnTick	keine	Experten	Ereignisbearbeiter NewTick. Wenn das Ereignis des neuen Ticks verarbeitet wird, werden andere Ereignisse dieses Typs nicht aufgenommen.
void	OnTimer	keine	Experten und Anzeiger	Ereignisbearbeiter Timer.

© 2000-2025, MetaQuotes Ltd.

Typ	Funktionsname	Parameter	Anwendung	Bemerkung
void	OnTrade	keine	Experten	Ereignisbearbeiter Trade.
double	OnTester	keine	Experten	Ereignisbearbeiter Tester.
void	OnChartEvent	const int id, const long &lparam, const double &dparam, const string &sparam	Experten und Indikatoren	Ereignisbearbeiter ChartEvent.
void	OnBookEvent	const string &symbol_name	Experten und Indikatoren	Ereignisbearbeiter BookEvent.

Der Client-Terminal sendet die neuen Ereignisse in die entsprechende offene Charts. Darüber hinaus können die Ereignisse durch Charts (Chartsereignisse) oder MQL5-Programme (benutzerdefinierte Ereignisse) generiert werden. Generierung von Ereignisse der Erstellung oder Löschen von graphischen Objekten auf einen Chart kann durch CHART_EVENT_OBJECT_CREATE und CHART_EVENT_OBJECT_DELETE Chartseigenschaften aktiviert oder deaktiviert werden. Jedes MQL5-Programm und jeder Chart hat eine eigene Warteschlange von Ereignissen, in die alle neu eingehenden Ereignisse hinzugefügt werden.

Ein Programm erhält Ereignisse nur aus dem Chart, auf dem es läuft. Alle Ereignisse werden nacheinander in der Reihenfolge des Eingangs bearbeitet. Wenn eine Warteschlange bereits eine NewTick Ereignis hat, oder dieses Ereignis ist gerade in Bearbeitung, dann wird nicht die neue NewTick Ereignis in der Warteschlange des MQL5 Programms platziert. Auch wenn ChartEvent bereits in der Warteschlange ist, oder dieses Ereignis in Bearbeitung ist, wird kein neues Ereignis dieser Art eingerichtet werden. Die Timer-Ereignisse werden auf die gleiche Weise behandelt – wenn ein Timer-Ereignis in der Warteschlange ist oder behandelt wird, die neue Timer-Ereignis wird nicht in der Warteschlange platziert.

Ereignis-Warteschlangen haben eine begrenzte, aber ausreichende Größe, so dass die Warteschlange-Überlauf für gut geschriebene Programme unwahrscheinlich ist. Im Falle einer Warteschlange-Überlauf, werden neue Ereignisse ohne Wartezeiten verworfen.

Es wird dringend empfohlen, keine Endlosschleifen zur Behandlung von Ereignissen zu verwenden. Mögliche Ausnahmen sind Skripte und Dienste, die ein einzelnes Start-Ereignis behandeln.

Bibliotheken verarbeiten keine Ereignisse.

Das Verbot der Verwendung von Funktionen in Indikatoren und Experten

© 2000-2025, MetaQuotes Ltd.

Indikatoren, Skripte und Expert Advisors sind in MQL5 geschriebene ausführbare Programme. Sie sind für verschiedene Arten von Aufgaben ausgelegt. Deshalb gibt es einige Beschränkungen für die Verwendung bestimmter Funktionen, abhängig von der Art des Programms. Die folgenden Funktionen werden in Indikatoren verboten:

- OrderCalcMargin();
- OrderCalcProfit();
- OrderCheck();
- OrderSend();
- SendFTP();
- Sleep();
- ExpertRemove();
- MessageBox().

Alle Funktionen für Indikatoren sind in Expert Advisors und Skripte verboten:

- SetIndexBuffer();
- IndicatorSetDouble();
- IndicatorSetInteger();
- IndicatorSetString();
- PlotIndexSetDouble();
- PlotIndexSetInteger();
- PlotIndexSetString();
- PlotIndexGetInteger.

Die Bibliothek ist kein eigenständiges Programm und wird im Rahmen des Programms MQL5, die sie aufgerufen hat, ausgeführt: Skript, Indikator oder Expert Advisor. Dementsprechend werden die oben genannten Beschränkungen für die genannte Bibliothek angewendet.

Die in den Diensten verbotenen Funktionen

Die Dienste akzeptieren keine Ereignisse, da sie nicht an ein Chart gebunden sind. Die folgenden Funktionen sind bei Diensten verboten:

ExpertRemove();

EventSetMillisecondTimer();

EventSetTimer();

EventKillTimer();

SetIndexBuffer();

IndicatorSetDouble();

IndicatorSetInteger();

IndicatorSetString();

PlotIndexSetDouble();

© 2000-2025, MetaQuotes Ltd.

PlotIndexSetInteger();

PlotIndexSetString();

PlotIndexGetInteger();

Laden und Ausladen der Indikatoren

Indikatoren werden in folgenden Fällen ausgeladen:

- Anhängen des Indikators zum Chart;
- Start des Terminals (wenn Indikator vor dem früheren Terminalschliessen zum Chart angehängt wurde);
- Laden der Schablone (wenn in der Schablone Indikator angegeben wird, angehängt zum Chart);
- Profilwechsel (wenn Indikator zum einen der Profilcharts angehängt wird);
- Wechsel des Symbols und/oder Chartperiode, zu dem der Indikator angehängt wird;
- Änderung des Kontos, zu dem das Terminal verbunden ist;
- nach erfolgreicher Neucompilierung des Indikators, wenn dieser Anzeiger zum Chart angehängt wurde.
- Wechsel der Eingabeparameter des Indikators.

Indikatoren werden in folgenden Fällen ausgeladen:

- bei Abtrennung des Indikators vom Chart;
- Terminalschliessen (wenn Indikator zum Chart angehängt wurde);
- Laden der Schablone, Indikator zum Chart angehängt wird;
- Schliessen des Charts, zu dem Indikator angehängt wurde;
- Profilwechsel, wenn Indikator zu einem der Charts des Wechselprofils angehängt wird;
- Wechsel des Symbols und/oder Chartperiode, zu dem Indikator angehängt wird;
- Änderung des Kontos, zu dem das Terminal verbunden ist;
- Wechsel der Eingabeparameter des Indikators.

Laden und Ausladen der Experte

Laden des Experten erfolgt in folgenden Fällen:

- Anhängen von Experten zum Chart;
- Start des Terminals (wenn Expert vor dem früheren Terminalabschluss zum Chart angehängt wurde);
- Laden der Schablone (wenn in der Schablone Expert angegeben wird, angehängt zum Chart);
- nach erfolgreicher Neucompilierung des Experten, wenn dieser Expert zum Chart angehängt wurde.
- Profilwechsel (Wenn Expert zum einen der Profilcharts angehängt wird);
- Verbindung zu einem Konto, auch wenn die Kontonummer nicht geändert wurde (wenn der Expert Advisor wurde um den Chart, bevor die Zulassung des Terminals auf dem Server, verbunden).

© 2000-2025, MetaQuotes Ltd.

Ausladen des Experten, angehängt zum Chart erfolgt in folgenden Fällen:

- bei Abtrennung des Experten vom Chart;
- beim Anhängen des Experten zum Chart - wenn auf dem Chart einen anderen Experten gab, wird dieser Expert ausgeladen;
- Terminalschliessen, wenn der Expert zum Chart angehängt wurde;
- Laden der Schablone, wenn der Expert zum Chart angehängt wird;
- Schliessen des Charts, zu dem Expert angehängt wurde;
- Profilwechsel, wenn der Expert zum einen der Charts der Wechselprofile angehängt wird;
- Änderung des Kontos, zu dem das Terminal verbunden ist (wenn der Expert Advisor wurde um den Chart, bevor die Zulassung des Terminals auf dem Server, verbunden);
- Aufruf der Funktion ExpertRemove().

Beim Wechsel des Symbols oder Timeframe des Charts, zu dem Expert angehängt wird, erfolgt Laden und Ausladen des Experten nicht. Dabei werden Bearbeiter OnDeinit() auf altem Symbol/Timeframe und OnInit() auf neuem Symbol/Timeframe (wenn es die solchen geben) serienweise aufgerufen, Werte der globalen Variablen und statischer Variablen werden nicht auf Null gesetzt. Alle Ereignisse, die vor dem Initialisierungsende (Funktion OnInit()) erscheinen, werden weggelassen.

Laden und Ausladen von Scripts

Scripts werden sofort nach Anhängen zum Chart geladen und sofort nach Operationsende ausgeladen. Dabei werden die Funktionen OnInit() und OnDeinit() für Scripts nicht aufgerufen.

Beim Programmausladen (Programmenfernen vom Chart) erfolgt Deinitialisierung der globalen Variablen und Entfernung der Queue von Ereignissen. In diesem Fall bedeutet Deinitialisierung Freigabe der Variablen des Typs string, Freigabe der Objekte der dynamischen Felder und Aufruf der Destruktors, wenn sie es geben.

Laden und Löschen von Diensten

Dienste werden unmittelbar nach dem Start des Terminals geladen, wenn sie zum Zeitpunkt der Terminalabschaltung gestartet wurden. Das Löschen der Dienste erfolgt unmittelbar nach Beendigung ihrer Arbeit.

Dienste haben nur eine einzige Funktion, OnStart(), in der Sie eine Endlosschleife für Empfang und Behandeln der Daten implementieren können, z.B. um nutzerdefinierte Symbole mit Hilfe der Netzwerkfunktionen zu erstellen und zu aktualisieren.

Im Gegensatz zu Expert Advisors, Indikatoren und Skripten sind Dienste nicht an ein bestimmtes Chart gebunden, daher wird ein separater Mechanismus bereitgestellt, um sie zu starten. Im Navigator wird mit dem Befehl "Dienst starten" eine neue Instanz des Dienstes angelegt. Eine Dienst-Instanz kann über das entsprechende Menü der Instanz gestartet, gestoppt und entfernt werden. Um alle Instanzen zu verwalten, verwenden Sie das Menü der Dienste.

© 2000-2025, MetaQuotes Ltd.

Für bessere Verständigung der Expertenarbeit ist es empfehlenswert, Kode des im Beispiel angeführten Experten zu kompilieren und Operationen von Laden/Ausladen der Experten, Schablonen-, Symbol-, Timeframewechsel usw. durchführen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   TestExpert.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"

class CTestClass
  {
public:
   CTestClass() { Print("CTestClass constructor"); }
   ~CTestClass() { Print("CTestClass destructor"); }
  };
CTestClass global;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   Print("Initialisation");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Print("Deinitialisation with reason",reason);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
```

Scripts werden sofort nach Anhängen zum Chart geladen und sofort nach Operationsende ausgeladen.

Sehen Sie auch

Ereignisse des Client-Terminals, Ereignisbearbeiter

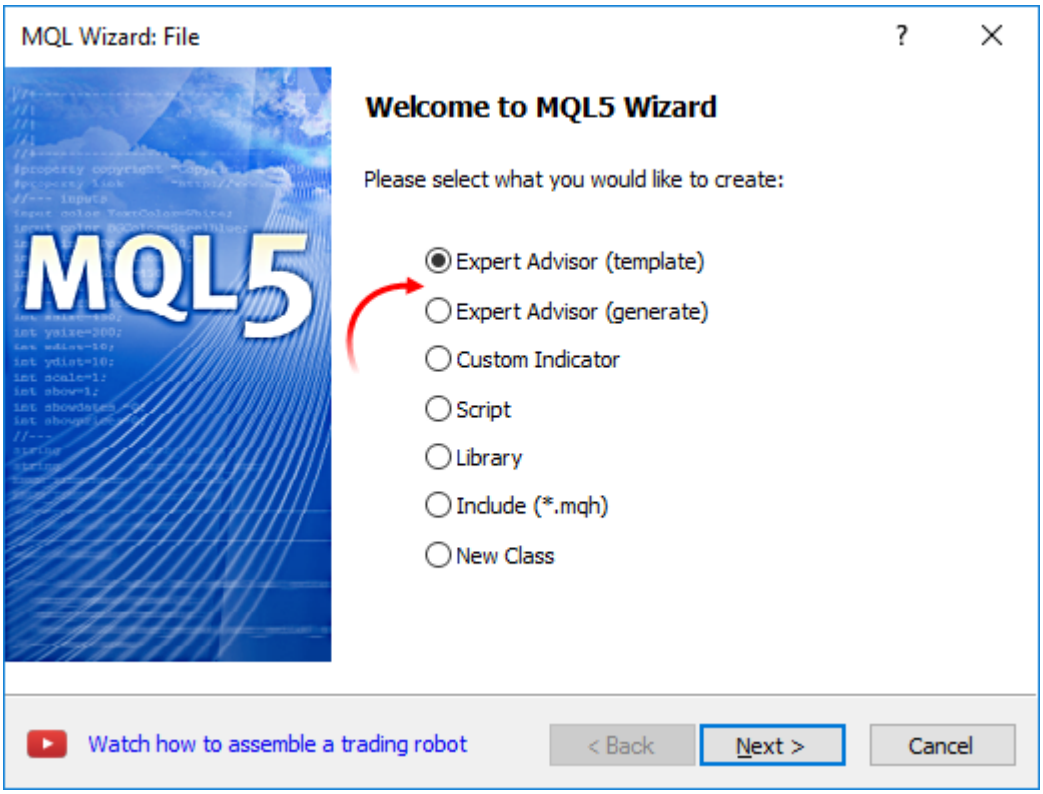
© 2000-2025, MetaQuotes Ltd.

Handelserlaubnis

Handelsautomatisierung

Die MQL5 Sprache hat eine spezielle Gruppe von Handelsfunktionen, mit denen man automatisierte Handelssysteme erstellen kann. Programme für den automatisierten Handel ohne menschliche Eingriffe sind Experten (Expert Advisor) oder Trading Roboter genannt. Um einen Experte in MetaEditor zu erstellen, starten sie den Assistent zur Erstellung von Experten MQL5 Wizard, und wählen Sie eine der Optionen:

- Expert Advisor (template) - ermöglicht eine Vorlage mit bereite Event-Handling-Funktionen erstellen, die soll mit allen notwendigen Funktionen durch Selbstprogrammierung ergänzt werden.
- Expert Advisor (generate) - ermöglicht einen fertigen Trading Roboter zu erstellendurch die Auswahl der Module, die Sie benötigen: Modul der Handelssignale, Geld-Management-Modul und Trailing-Stop-Modul.



Handelsfunktionen können nur in der Experten und Skripte arbeiten, in Indikatoren ist Handel verboten.

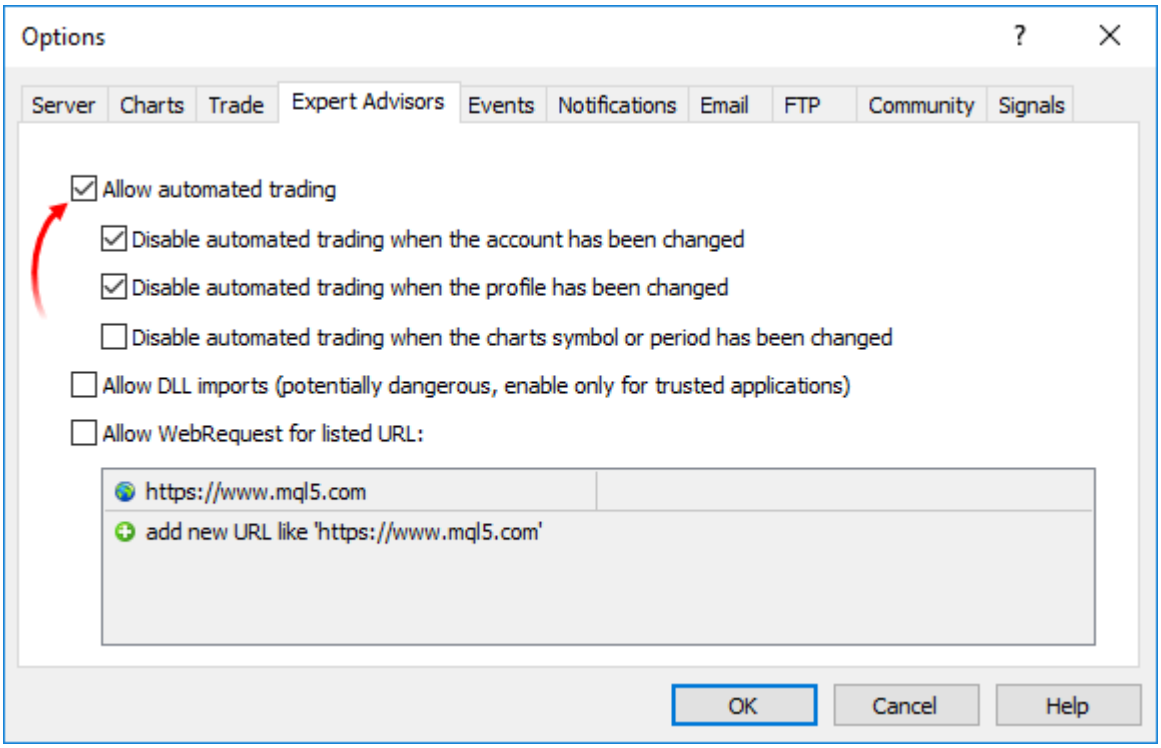
Überprüfung der Erlaubnis zum automatischen Handel

Um einen zuverlässigen Experten, der ohne menschliche Kontrolle automatisch funktionieren könnte, zu erstellen, ist es notwendig, eine Reihe von erforderlichen Kontrollen zu organisieren. Zuerst sollten wir programmatisch überprüfen, ob Handel überhaupt erlaubt ist. Dies ist eine Grundprüfung, die bei der Entwicklung eines automatisierten Systems unerlässlich ist.

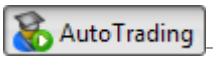
Überprüfung der Erlaubnis zum automatischen Handel im Terminal

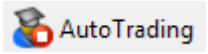
© 2000-2025, MetaQuotes Ltd.

Automatischer Handel für alle Programme kann im Terminal-Einstellungen verhindert oder erlaubt werden.



Sie können die Möglichkeit des automatisierten Handels direkt auf Standard-Panel des Terminals wechseln:

- 

- automatischer Handel ist erlaubt, Verwendung von Handelsfunktionen in laufenden Anwendungen ist erlaubt.
- 

- automatischer Handel ist verboten, die laufende Anwendungen funktionieren, aber die Handelsfunktionen können nicht ausgeführt werden.

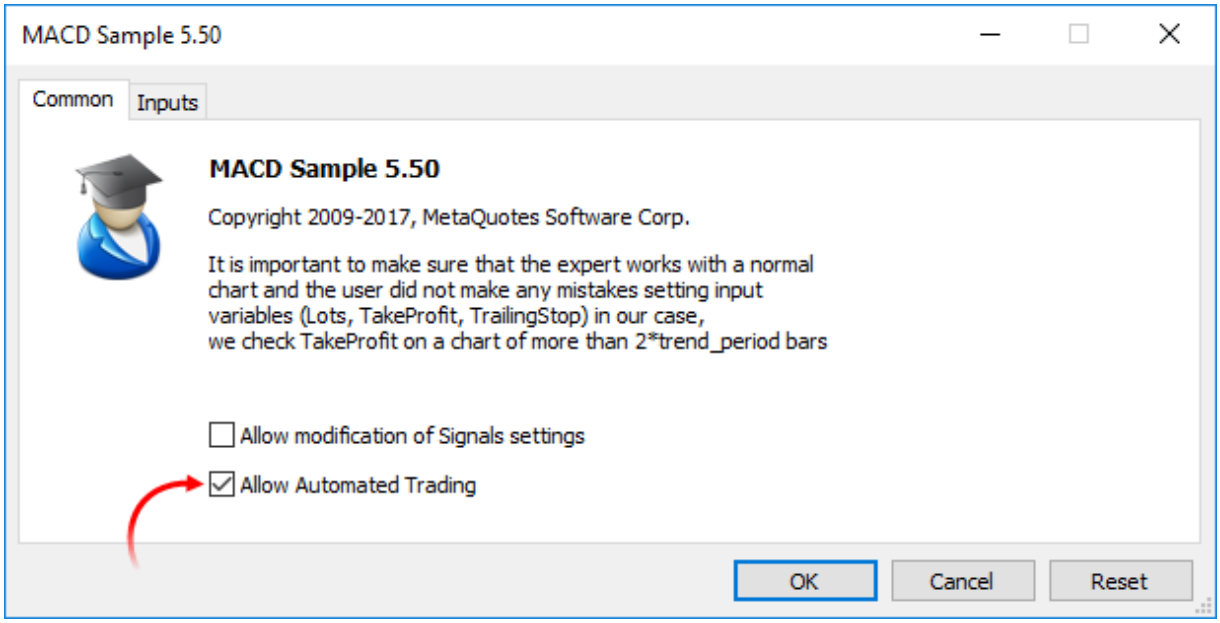
Beispiel der Überprüfung:

```
if (!TerminalInfoInteger(TERMINAL_TRADE_ALLOWED))
   Alert("Überprüfen Sie Erlaubnis zum automatischen Handel in Terminal-Einstellungen!
```

Überprüfung der Erlaubnis zum Handel für den laufenden Experte/Skript

Wenn Sie eine Anwendung starten, können Sie automatischen Handel speziell für sie erlauben oder verhindern. Um dies zu tun, verwenden Sie die spezielle Kontrollkästchen in den Programmeigenschaften.

© 2000-2025, MetaQuotes Ltd.



Beispiel der Überprüfung:

```
   if(!TerminalInfoInteger(TERMINAL_TRADE_ALLOWED))
      Alert("Überprüfen Sie Erlaubnis zum automatischen Handel in Terminal-Einstellung
   else
     {
      if(!MQLInfoInteger(MQL_TRADE_ALLOWED))
         Alert("Automatischer Handel ist verboten in Programmeigenschaften für ",__FI
     }
```

Überprüfung der Erlaubnis zum Handel für Experten/Skripte für dieses Konto

Automatisierter Handel kann auf der Sote des Handelsservers verboten sein. Beispiel der Überprüfung:

```
   if(!AccountInfoInteger(ACCOUNT_TRADE_EXPERT))
      Alert("Automatischer Handel ist für das Konto ",AccountInfoInteger(ACCOUNT_LOGIN
      " auf der Seite des Handelsservers verboten");
```

Wenn automatisierter Handel ist für ein Handelskonto deaktiviert, werden Handelsoperationen von Experten/Skripte nicht ausgeführt.

Überprüfung der Erlaubnis zum Handel für dieses Konto

In einigen Fällen sind alle Handelsoperationen für ein bestimmtes Handelskonto verboten - man kann nicht entweder manuell oder mit der Hilfe von Experten handeln. Beispiel der Überprüfung, wenn man zum Konto mit dem Investor-Passwort verbinden versuchte:

```
   if(!AccountInfoInteger(ACCOUNT_TRADE_ALLOWED))
      Comment("Handel ist verboten für das Konto ",AccountInfoInteger(ACCOUNT_LOGIN),
            ".\n Vielleicht ist Investor-Passwort für die Verbindung mit dem Handelsko
            "\n Überprüfen Sie das Terminal-Protokoll, ob es einen solches Rekord gibt
            "\n\'",AccountInfoInteger(ACCOUNT_LOGIN),"\': trading has been disabled -
```

© 2000-2025, MetaQuotes Ltd.

AccountInfoInteger(ACCOUNT_TRADE_ALLOWED) kann in folgenden Fälle false zurück geben:

- keine Verbindung mit dem Handelsserver. Kann durch TerminalInfoInteger(TERMINAL_CONNECTED) überprüft werden;
- Handelskonto ist auf den Modus read-only)in das Archiv gesendet) übertragen;
- Handel auf das Konto ist auf der Seite des Handelsservers verboten;
- Verbindung zum Handelskonto ist im Investor-Modus.

Sehen Sie auch

Zustand des Terminals, Information über das Konto, Information über das ausgeführte MQL5-Programm

© 2000-2025, MetaQuotes Ltd.

Ereignisse des Client-Terminals

Init

Sofort nach Laden des Programms (des Experten oder des Benutzeranzeigers) und Start der Initialisierung der globalen Variablen wird das Ereignis Init gesendet, das durch die Funktion vom Client-Terminal OnInit() verarbeietet wird, wenn sie vorhanden ist. Dieses Ereignis wird auch nach Wechsel des Finanzinstrumentes und/oder der Chartperiode, nach Umcompilierung des Programms im Editor MetaEditor, nach Wechsel der Eingabeparameter vom Fenster der Einstellung von Expert oder Benutzerindikator. Ratgeber wird nach Kontowechsel initialisiert. Für Scripts wird das Ereignis Init nicht generiert.

Deinit

Vor Deinitialisierung der globalen Variablen und Ausladen des Programms (des Experten oder des Benutzeranzeigers) sendet das Client-Terminal dem Programm das Ereignis Deinit. Das Ereignis Deinit wird auch nach Operationsende des Client-Terminals, beim Chartschliessen, unmittelbar vor dem Wechsel des Finanzinstrumentes und/oder der Chartperiode, bei erfolgreicher Umcompilierung des Programms, beim Wechsel der Eingabeparameter und beim Kontowechsel generiert.

Grund der Deinitialisierung kann man vom Parameter erhalten, der in die Funktion OnDeinit() übertragen wird. Durchführen der Funktion OnDeinit() wird von 2.5 Sekunden begrenzt. Wenn die Funktion für diese Zeit ihre Arbeit nicht schliesst, wird ihre Ausführung zwangslaeufig beendet. Für Scripts wird das Ereignis Deinit nicht generiert.

Start

Start ist ein besonderes Ereignis, um ein Skript oder einen Dienst nach dem Laden zu starten. Es wird von der Funktion OnStart behandelt. Das Startereignis wird nicht an EAs und benutzerdefinierte Indikatoren übergeben.

NewTick

Ereignis NewTick wird beim Erscheinen der neuen Quotationen generiert und durch die Funktion OnTick() der angehängten Ratgeber. Wenn beim Erscheinen der neuen Quotation die Funktion OnTick ausgeführt wurde, die in der früheren Quotation gestartet wurde, wird die errscheinende Quotation vom Ratgeberignoriert, denn der entsprechende Wert wird nicht in die Queue der Expertereignisse eingereiht.

Alle während der Programmausführung erscheinenden Quotationen werden ignoriert bis Ausführung der Funktion OnTick() beendet. Dann wird die Funktion nach Erscheinen der neuen Quotation initialisiert. Das Ereignis NewTick wird, wenn Verwenden der Ratgeber (Schaltfläche "Ratgeber erlauben/verbieten") verboten ist.

Das Ereignis NewTick wird unabhängig davon generiert, ob der automatische Handel erlaubt oder verboten ist (Schaltfläche "Autohandel erlauben/verboten"). Verbot des automatischen Handels bedeutet nur Verbot, Handelsanforderungen aus dem Expert zu senden, die Arbeit des Experten wird nicht abgebrochen.

Verbot des automatischen Handels beim Drücken auf die angegebene Schaltfläche bricht nicht die laufende Ausführung der Funktion OnTick().

© 2000-2025, MetaQuotes Ltd.

Calculate

Ereignis Calculate wird nur für Anzeiger sofort nach Senden vom Ereignis Init und bei jedem Wechsel der Preisdaten. Wird durch die Funktion OnCalculate verarbeitet.

Timer

Ereignis Timer wird in regelmässige vom Client-Terminal für Experten generiert, der durch die Funktion generiert EventSetTimer Timer aktiviert hat. Gewöhnlich wird diese Funktion in der Funktion OnInit aufgerufen. Das Ereignis Timer wird durch Funktion OnTimer verarbeitet. Nach dem Operationsende des Experten muss man den erzeugten Timer durch die Funktion EventKillTimer entfernen, die gewöhnlich in der Funktion OnDeinit auf gerufen wird.

Trade

Ereignis Trade wird beim Schliessen der Handelsoperation auf dem Handelsserver generiert. Verarbeitung des Ereignisses Trade wird durch die Funktio OnTrade() für folgende Handelsoperationen durchgeführt:

- Einstellung, Modifikation oder Entfernung der wartenden Order;
- Abbruch der wartenden Order beim Geldmangel oder Gueltigkeitsablauf;
- Aktivierung der wartenden Order ;
- Öffnen, Zusetzen oder Schliessen der Position (oder eines Teiles der Position);
- Modifikation der offenen Position (Veränderung von Stops).

TradeTransaction

Als Ergebnis der Durchführung von bestimmten Aktionen auf dem Handelskonto, ändert sich der Kontostand. Zu solchen Aktionen gehören:

- Senden der Handelsanforderung aus einer beliebigen MQL5-Anwendung in dem Client-Terminal mittels der Funktionen OrderSend und OrderSendAsync und ihr nachfolgendes Ausführen;
- Senden der Handelsanforderung über die grafische Oberfläche des Terminals und ihr nachfolgendes Ausführen;
- Auslösen von aufgeschobenen Ordern und Stop-Ordern auf dem Server;
- Ausführen von Operationen auf der Seite der Handel-Server.

Als Ergebnis dieser Aktionen werden die folgenden Handelstransaktionen durchgeführt:

- Die Verarbeitung der Handelanforderung;
- Änderung der offenen Aufträge;
- Änderung der Ordergeschichte;
- Änderung der Dealgeschichte;
- Änderung der Position.

Zum Beispiel, beim Senden Marktkauforder, wird Marktkauforder behandelt und eine entsprechende Marktkauforder für das Konto erstellt. Ist die Order durchgeführt, wird sie aus der Auftragsliste entfernt und zur Ordergeschichte hinzugefügt. Dann wird der entsprechende Deal zur Geschichte hinzugefügt und eine neue Position wird erstellt. Alle diese Aktionen sind Handelstransaktionen.

© 2000-2025, MetaQuotes Ltd.

Ankunft jeder solchen Transaktion im Terminal ist ein Ereignis TradeTransaction. Dieses Ereignis wird durch die Funktion OnTradeTransaction verarbeitet.

Tester

Ereignis Tester wird nach Testen des Experten in historischen Daten generiert. Bearbeitung des Ereignisses Tester wird durch die Funktion OnTester() durchgeführt.

TesterInit

Ereignis TesterInit wird beim Start der Optimierung im Strategie-Tester vor dem ersten Durchlauf generiert. Bearbeitung des Ereignisses TesterInit wird durch die Funktion OnTesterInit() durchgeführt.

TesterPass

Ereignis TesterPass wird beim Empfang eines neuen Datenframes generiert. Bearbeitung des Ereignisses TesterPass wird durch die Funktion OnTesterPass() durchgeführt.

TesterDeinit

Ereignis TesterDeinit wird nach der Optimierung des Expert Advisors im Strategie-Tester generiert. Bearbeitung des Ereignisses TesterDeinit wird durch die Funktion OnTesterDeinit() durchgeführt.

ChartEvent

Ereignis ChartEvent wird vom Client-Terminal generiert bei der Arbeit des Benutzers mit Chart:

- Keyboard Drücken, wenn Chartfenster fokussiert ist; ;
- Erzeugung des graphischen Objekts;
- Entfernung des graphischen Objekts;
- Mausklicken auf dem graphischen Objekt, das zum Chart gehört;
- Bewegung des graphischen Objekts mit der Maus;
- Ende des Texteditierens im Eingabefeld des graphischen Objektes LabelEdit.

Es gibt auch ein Benutzerereignis ChartEvent, das jedes mql5-Programm durch die Funktion EventChartCustom senden kann. Ereignis wird durch die Funktion OnChartEvent verarbeitet.

BookEvent

Das Ereignis BookEvent wird vom Client-Terminal bei DOM Veränderung generiert und durch die Funktion OnBookEvent verarbeitet. Damit das Client-Terminal Ereignisse BookEvent für konkretes Symbol zu generieren anfängt, reicht es, auf Erhalten dieser Ereignisse durch die Funktion MarketBookAdd zu subskribieren.

Um die Subskription auf Erhalten des Ereignisses für konkretes Symbol zu annullieren, muss die Funktion BookEvent MarketBookRelease aufgerufen werden. Das Ereignis BookEvent ist broadcast – das bedeutet, wenn ein Expert auf Erhalten des Ereignisses BookEvent durch die Funktion MarketBookAdd subskribiert, werden alle anderen Experten mit dem Handler OnBookEvent dieses Ereignisses bekommen. Darum muss der Symbolname, der in Bearbeiter als Parameter übertragen wird, analysieren.

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

Funktionen der Ereignisverabeitung, Progarammausführung

© 2000-2025, MetaQuotes Ltd.

Ressourcen

Benutzung der Grafik und Sound in MQL5 Programme

Die MQL5 Programme ermöglichen es Ihnen, mit Sound und Grafik-Dateien arbeiten:

- PlaySound() spielt eine Sounddatei;
- ObjectCreate() ermöglicht die Erstellung von Benutzeroberflächen mit Hilfe grafischer Objekte OBJ_BITMAP und OBJ_BITMAP_LABEL .

PlaySound()

Beispiel für Aufruf der Funktion PlaySound():

```
//+------------------------------------------------------------------+
//| Funktion ruft Standard-Funktion OrderSend() und spielt Sound     |
//+------------------------------------------------------------------+
void OrderSendWithAudio(MqlTradeRequest  &request, MqlTradeResult &result)
  {
  //--- Eine Anfrage an den Server senden
   OrderSend(request,result);
   //--- Wenn die Anfrage angenommen wird, Ok.wav spielen
   if(result.retcode==TRADE_RETCODE_PLACED) PlaySound("Ok.wav");
   //--- Wenn nicht, spielen Alarm aus Datei timeout.wav
   else PlaySound("timeout.wav");
  }
```

Das Beispiel zeigt, wie Sounds aus Dateien Ok.wav und timeout.wav, die in das Standard-Terminal-Paket enthalten sind, spielen. Diese Dateien liegen im Ordner **Terminal_Verzeichnis\Sounds**. Hier **Terminal_Verzeichnis** bedeutet den Verzeichnis aus dem MetaTrader 5 Client Terminal gestartet wird. Der Standort des Terminals Verzeichnis kann direkt von einem MQL5 Programm in folgender Weise gefunden werden:

```
//--- Verzeichnis in dem Terminaldaten liegen
   string terminal_path=TerminalInfoString(TERMINAL_PATH);
```

Sie können die Audio-Dateien nicht nur aus dem Ordner **Terminal_Verzeichnis\Sounds** verwenden, sondern auch von jedem Unterordner im Ordner **Terminaldaten_Verzeichnis\MQL5**. Lage des Verzeichnisses den Terminaldaten auf einem Computer können Sie durch das Terminal-Menü "File - Open Directory-Daten", oder programmatisch herausfinden:

```
//--- Verzeichnis in dem Terminaldaten liegen
   string terminal_data_path=TerminalInfoString(TERMINAL_DATA_PATH);
```

Zum Beispiel, wenn Demo.wav liegt in Terminaldaten_Verzeichnis\MQL5\Files, dann Aufruf von PlaySound() soll auf folgende Weise geschrieben werden:

```
//--- Demo.wav aus Terminaldaten_Verzeichnis\MQL5\Files\Demo.wav spielen
   PlaySound("\\Files\\Demo.wav");
```

© 2000-2025, MetaQuotes Ltd.

Bitte beachten Sie, dass im Kommentar der Pfad zur Datei mit Backslash "\" geschrieben wird, und in der Funktion "\\" verwendet wird.

Bei der Angabe des Pfades, immer nur den doppelten Backslash als Trennzeichen benutzen, da ein einzelner Backslash ist ein Kontrolsymbol des Compilers beim Umgang mit konstanten Zeichenfolgen und Symbolkonstanten im Programm-Quellcode.

Um Wiedergabe der Datei zu stoppen, rufen Sie die Funktion PlaySound() mit Parameter NULL auf:

```
//--- Aufruf von Playsound() mit einem Parameter NULL stoppt Audiowiedergabe
   PlaySound(NULL);
```

ObjectCreate()

Ein Beispiel eines Expert Advisors, der mit Hilfe ObjectCreate() erzeugt ein Objekt "Grafischer Zeichen" (OBJ_BITMAP_LABEL).

```
string label_name="currency_label";         // Objektname OBJ_BITMAP_LABEL
string euro      ="\\Images\\euro.bmp";     // Pfad zu Terminaldaten_Verzeichnis\MQL5\I
string dollar    ="\\Images\\dollar.bmp";   // Pfad zu Terminaldaten_Verzeichnis\MQL5\I
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Schaltfläche OBJ_BITMAP_LABEL erstellen, wenn es noch nicht erstellt worden
   if(ObjectFind(0,label_name)<0)
     {
      //--- Versuchen wir OBJ_BITMAP_LABEL zu erstellen
      bool created=ObjectCreate(0,label_name,OBJ_BITMAP_LABEL,0,0,0);
      if(created)
        {
         //--- Binden wir die Schaltfläche zur linken oberen Ecke des Charts
         ObjectSetInteger(0,label_name,OBJPROP_CORNER,CORNER_RIGHT_UPPER);
         //--- Konfigurieren wir nun die Eigenschaften des Objekts
         ObjectSetInteger(0,label_name,OBJPROP_XDISTANCE,100);
         ObjectSetInteger(0,label_name,OBJPROP_YDISTANCE,50);
         //--- Wird der letzte Fehlercode auf 0 fallen
         ResetLastError();
         //--- Laden wir das Bild für den Zustand der Schaltflächen "gedrückt"
         bool set=ObjectSetString(0,label_name,OBJPROP_BMPFILE,0,euro);
         //--- Ergebnis prüfen
         if(!set)
           {
            PrintFormat("Bild konnte nicht aus der Datei %s geladen werden. Fehlercode
           }
         ResetLastError();
         //--- Laden wir das Bild für den Zustand der Schaltflächen "nicht gedrückt"
```

© 2000-2025, MetaQuotes Ltd.

```
         set=ObjectSetString(0,label_name,OBJPROP_BMPFILE,1,dollar);

         if(!set)
           {
            PrintFormat("Bild konnte nicht aus der Datei %s geladen werden. Fehlercode
           }
         //---Geben wir dem Chart einen Befehl zu aktualisieren, so dass die Schaltflä
         ChartRedraw(0);
       ;}
      else
        {
         //--- Objekt konnte nicht erstellt werden, melden
         PrintFormat("Objekt OBJ_BITMAP_LABEL konnte nicht erstellt werden. Fehlercode
        }
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Objekt aus dem Chart löschen
   ObjectDelete(0,label_name);
  }
```

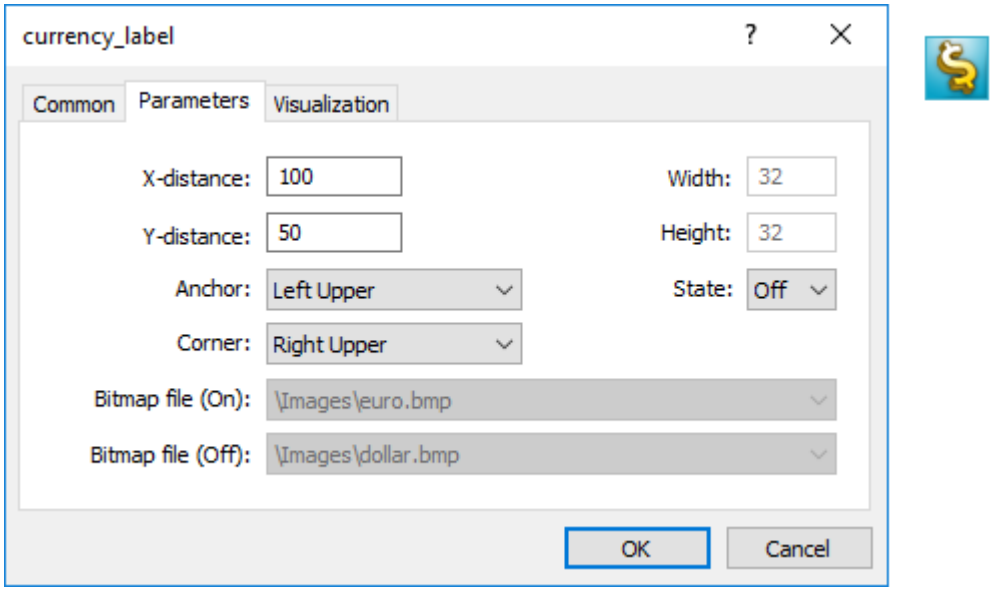
Erstellen und Konfigurieren eines grafischen Objekts namens currency_label sind in der Funktion OnInit() durchgeführt. Pfade zum grafischen Dateien werden in der globalen Variablen **euro** und **dollar** gegeben, doppelter Backslash für ein Separator verwendet ist:

```
string euro      ="\\Images\\euro.bmp";    // Pfad zu Terminaldaten_Verzeichnis\MQL5\I
string dollar    ="\\Images\\dollar.bmp";  // Pfad zu Terminaldaten_Verzeichnis\MQL5\I
```

Die Dateien legen im Ordner **Terminaldaten_Verzeichnis**\MQL5\Images.

Objekt OBJ_BITMAP_LABEL ist eigentlich eine Schaltfläche, die eine der beiden Bilder zeigt, je nach Status der Schaltfläche (gedrückt oder nicht gedrückt): euro.bmp oder dollar.bmp.

© 2000-2025, MetaQuotes Ltd.



Die Größe der Schaltfläche mit einer grafischen Oberfläche wird automatisch an die Größe des Bildes angepasst. Das Bild wird durch einen linken Mausklick auf das Objekt OBJ_BITMAP_LABEL geändert (**im Eigenschaften soll "Disable selection" gesetzt werden**). Objekt OBJ_BITMAP wird die gleiche Weise erstellt - es ist für die Erstellung der Hintergrund mit einem notwendigen Bild verwendet.

Der Wert der OBJPROP_BMPFILE Eigenschaft, die verantwortlich für das Erscheinungsbild der Objekte OBJ_BITMAP und OBJ_BITMAP_LABEL ist, kann dynamisch geändert werden. Dies ermöglicht das Erstellen verschiedener interaktiver Benutzeroberflächen für MQL5 Programme.

Die Einschließung von Ressourcen in ausführbaren Dateien beim Kompilation der MQL5-Programme

Ein MQL5 Programm kann viele verschiedene herunterladbare Ressourcen in Form von Bild-und Audio-Dateien brauchen. Um die Notwendigkeit alle diese Dateien ins anderen MQL5-Programm zu übertragen zu beseitigen, verwenden Sie die Compiler-Direktive #resource:

```
#resource Pfad_zu_Ressource-Datei
```

Befehl #resource sagt dem Compiler, dass die Ressource an dem angegebenen Pfad **Pfad_zu_Ressourcedatei** in die ausführbare Datei EX5 eingeschlossen werden sollten. Damit sind alle notwendigen Bilder und Sounds direkt in einer Datei EX5 gelegt, so dass es keine Notwendigkeit gibt, die Dateine zu übertragen, wenn Sie das Programm auf einem anderen Terminal ausführen möchten. Jede EX5 Datei kann Ressource haben, und jedes EX5 Programm kann Ressourcen aus einer anderen EX5 Programm verwenden.

BMP- und WAV-Dateien werden automatisch komprimiert, bevor sie in eine ausführbare Datei EX5 eingeschlossen sind. Dies bedeutet, dass die Verwendung der Ressourcen nicht nur erlaubt Ihnen MQL5 Programmen zu schaffen, sondern reduziert auch die Gesamtgröße der Dateien, die ein Chart für Grafiken und Sounds bracuht.

Die Dateigröße der Ressource konnte nicht mehr als 16 MB sein.

© 2000-2025, MetaQuotes Ltd.

Suche nach bestimmten Ressourcen von einem Compiler

Eine Ressource ist mit dem Befehl #resource "<Pfad zu Ressource Datei>" eingefügt.

```
#resource "<Pfad_zu_Ressourcedatei>"
```

Die Länge des konstanten String <Pfad_zu_Ressourcedatei> sollte nicht mehr als 63 Zeichen.

Der Compiler sucht nach einer Ressource auf dem angegebenen Pfad in folgender Reihenfolge:

- wenn der Backslash "\" Separator (geschrieben "\\") wird am Anfang des Pfads platziert, sucht es für die Ressource relativ zum Verzeichnis **Terminaldaten_Verzeichnis**\MQL5,
- wenn es kein Backslash gibt, wird die Ressource auf den Speicherort der Quelldatei, in der diese Ressource ist vorgeschrieben, gesucht.

Die Ressource-Pfad kann man nicht die Teilstrings "..\\" und "\\" verwenden.

Beispiele:

```
//--- Korrekte Angabe von Ressourcen
#resource "\\Images\\euro.bmp" // euro.bmp befindet sich in Terminaldaten_Verzeichnis\
#resource "picture.bmp"        // picture.bmp befindet sich in dasselbe Verzeichnis, w
#resource "Resource\\map.bmp"  // Ressource liegt in Quelldatei_Verzeichnis\Resource\r

//--- Falsche Angabe von Ressourcen
#resource ":picture_2.bmp"     // ":" kann nicht verwendet werden
#resource "..\\picture_3.bmp"  // ".." kann nicht verwendet werden
#resource "\\Files\\Images\\Folder_First\\My_panel\\Labels\\too_long_path.bmp" //mehr
```

Die Verwendung der Ressourcen

Ressource-Name

Nachdem die Ressource mit der Direktive #resource deklariert ist, kann es in jedem Teil des Programms verwendet werden. Ressource-Name wird seinen Pfad ohne einen Backslash am Anfang des Strings, die den Pfad zu der Ressource definiert. Um eine eigene Ressource zu verwenden, spezielles Attribut "::" soll vor dem Ressource-Namen hinzufügt werden.

Beispiele:

```
//--- Beispiele der Ressourcen und ihre Namen in den Kommentaren
#resource "\\Images\\euro.bmp"          // Ressource-Name - Images\euro.bmp
#resource "picture.bmp"                 // Ressource-Name - picture.bmp
#resource "Resource\\map.bmp"           // Ressource-Name - Resource\map.bmp
#resource "\\Files\\Pictures\\good.bmp" // Ressource-Name - Files\Pictures\good.bmp
#resource "\\Files\\Demo.wav";          // Ressource-Name - Files\Demo.wav"
#resource "\\Sounds\\thrill.wav";       // Ressource-Name - Sounds\thrill.wav"
...

//--- Verwendung der Ressourcen
ObjectSetString(0,bitmap_name,OBJPROP_BMPFILE,0,"::Images\\euro.bmp");
```

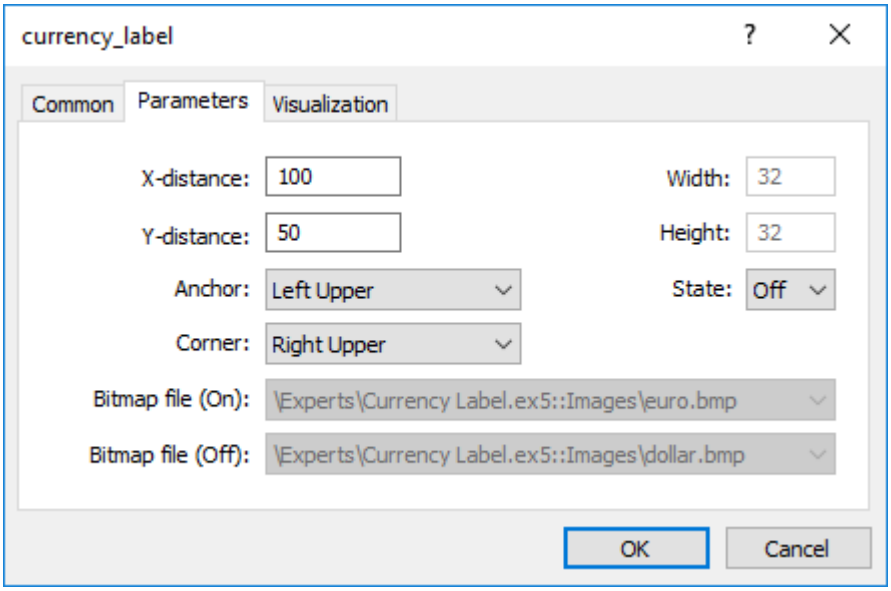
© 2000-2025, MetaQuotes Ltd.

```
...
ObjectSetString(0,my_bitmap,OBJPROP_BMPFILE,0,"::picture.bmp");
...
set=ObjectSetString(0,bitmap_label,OBJPROP_BMPFILE,1,"::Files\\Pictures\\good.bmp");
...
PlaySound("::Files\\Demo.wav");
...
PlaySound("::Sounds\\thrill.wav");
```

Anzumerken ist, dass bei der Einstellung von Bildern aus einer Ressource an den OBJ_BITMAP und OBJ_BITMAP_LABEL Objekten, der Wert der Eigenschaft OBJPROP_BMPFILE kann nicht manuell geändert werden. Zum Beispiel für das Erstellen OBJ_BITMAP_LABEL, verwenden wir unserer Ressourcen euro.bmp und dollar.bmp.

```
#resource "\\Images\\euro.bmp";    // euro.bmp liegt in Terminaldaten_Verzeichnis\MQL5
#resource "\\Images\\dollar.bmp";  // dollar.bmp liegt in Terminaldaten_Verzeichnis\MQ
```

Dann, in der Eigenschaften dieses Objekts sehen wir, dass die Eigenschaften BitMap File (On) und BitMap File (Off) sind grau und können nicht manuell geändert werden:



Verwendung der Ressourcen anderer mql5-Programmen

Die Verwendung der Mittel hat einen weiteren Vorteil - in jedem MQL5-Programm kann Ressource aus einer beliebigen Datei EX5 verwendet werden. So können Ressourcen aus einer Datei EX5 in vielen anderen MQL5-Programmen verwendet werden.

Um eine Ressource-Name aus einer anderen Datei zu verwenden, soll sie als <Pfad_EX5_Dateiname> angegeben werden. Zum Beispiel, in Skript Draw_Triangles_Script.mq5 wird eine Ressource zum Bild in der Datei triangle.bmp angegeben:

```
#resource "\\Files\\triangle.bmp"
```

© 2000-2025, MetaQuotes Ltd.

Dann ist ihr Name "Files\triangle.bmp" für das Skript. Um es zu verwenden, soll Attribut "::" zum Ressource-Namen hinzufügt werden.

```
//--- Verwendung der Ressource in dem Skript
ObjectSetString(0,my_bitmap_name,OBJPROP_BMPFILE,0,"::Files\\triangle.bmp");
```

Um die gleiche Ressource aus einem anderen Programm zu verwenden, z. B. aus einem Expert Advisor, müssen wir auf die Ressource Name fügen den Pfad zu der EX5-Datei relativ zu **Terminaldaten_Verzeichnis\MQL5** und den Namen der EX5-Datei des Skripts - **Draw_Triangles_Script.ex5** hinzufügen. Nehmen wir an, dass das Skript liegt in der Standard-Ordner **Terminaldaten_Verzeichnis\MQL5**\Scripts\, dann soll der Anruf wie folgt geschrieben werden:

```
//--- Verwendung der Ressource aus dem Skript im Expert Advisor
ObjectSetString(0,my_bitmap_name,OBJPROP_BMPFILE,0,"\\Scripts\\Draw_Triangles_Script.e
```

Wenn beim Zugriff auf Ressourcen in einer anderen EX5-Datei gibt es kein Pfad zur ausführbaren Datei, dann ist die ausführbare Datei im selben Ordner, wo das Program das ruft die Ressource liegt, gesucht. Das heißt, wenn der Expert Advisor fordert eine Ressource aus Datei Draw_Triangles_Script.ex5 ohne Angabe des Pfades, etwa so:

```
//--- Aufruf der Ressource des Skripts im Expert Advisor ohne Pfad
ObjectSetString(0,my_bitmap_name,OBJPROP_BMPFILE,0,"Draw_Triangles_Script.ex5::Files\\
```

dann wird die Datei in **Terminaldaten_Verzeichnis\MQL5**\Experts\ durchgesucht, wenn der Expert Advisor sich in **Terminaldaten_Verzeichnis\MQL5**\Experts\ befindet.

Arbeiten mit als Ressourcen verbundenen benutzerdefinierten Indikatoren

MQL5-Programme können eine oder mehrere benutzerdefinierte Indikatoren brauchen, können sie alle in den Code der ausführbaren MQL5-Programm aufgenommen werden. Aufnahme von Indikatoren als Ressourcen vereinfacht die Verteilung von Programmen.

Beispiel für den Anschluss und Verwendung eines benutzerdefinierten Indikators SampleIndicator.ex5 im Verzeichnis: **terminal_data_folder**\MQL5\Indicators\:

```
//+------------------------------------------------------------------+
//|                                                     SampleEA.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#resource "\\Indicators\\SampleIndicator.ex5"
int handle_ind;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   handle_ind=iCustom(_Symbol,_Period,"::Indicators\\SampleIndicator.ex5");
   if(handle_ind==INVALID_HANDLE)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("Expert: iCustom call: Error code=",GetLastError());
       return(INIT_FAILED);
      }
//--- ...
   return(INIT_SUCCEEDED);
  }
```

Der Fall, wo der benutzerdefinierte Indikator in der Funktion OnInit() erzeugt eine oder mehrere Kopien von sich selbst, erfordert gesonderte Betrachtung. Für die Nutzung der Ressource aus einer MQL5-Programm ist es notwendig, die Ressource in der folgenden Form anzugeben: **<Pfad_Dateiname_EX5>::<Ressourcenname>**.

Zum Beispiel, wenn der Indikator SampleIndicator.ex5 ist in Expert Advisor SampleEA.ex5 als Ressource aufgenommen ist, wird der Pfad zu sich selbst, der bei der Aufruf von iCustom() in der Funktion der Initialisierung der benutzerdefinierten Indikator angegeben war, wird wie folgt sein: "\\Experts\\SampleEA.ex5::Indikators\\SampleIndicator.ex5". Wenn Sie diesen Pfad explizit angeben, ist der benutzerdefinierte Indikator SampleIndicator.ex5 fest mit dem Expert Advisor SampleEA.ex5 gebunden und verliert die Fähigkeit selbst zu arbeiten.

Ein Pfad zu sich selbst kann mit der Funktion GetRelativeProgramPath() erhalten werden. Hier ist ein Beispiel davon:

```
//+------------------------------------------------------------------+
//|                                                SampleIndicator.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property indicator_separate_window
#property indicator_plots 0
int handle;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Der falsche Pfad der Angabe einer Referenz auf sich selbst
//--- string path="\\Experts\\SampleEA.ex5::Indicators\\SampleIndicator.ex5";
//--- Der richtige Pfad vom Erhalten einer Referenz auf sich selbst
  string path=GetRelativeProgramPath();
//--- indicator buffers mapping
   handle=iCustom(_Symbol,_Period,path,0,0);
   if(handle==INVALID_HANDLE)
     {
      Print("Indicator: iCustom call: Error code=",GetLastError());
      return(INIT_FAILED);
    ;}
   else Print("Indicator handle=",handle);
//---
   return(INIT_SUCCEEDED);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
///....
//+------------------------------------------------------------------+
//| GetRelativeProgramPath                                           |
//+------------------------------------------------------------------+
string GetRelativeProgramPath()
  {
   int pos2;
//--- erhalten wir den absoluten Pfad zum Programm
   string path=MQLInfoString(MQL_PROGRAM_PATH);
//--- Finden Sie die Position eines Teilstrings "\MQL5\"
   int    pos =StringFind(path,"\\MQL5\\");
//--- Teilstring ist nicht gefunden - Fehler
   if(pos<0)
      return(NULL);
//--- Verzeichnis "\MQL5" auslassen
   pos+=5;
//--- extra '\' auslassen
   while(StringGetCharacter(path,pos+1)=='\\')
      pos++;
//--- Wenn es eine Ressource ist, geben wir den Pfad relativ zu MQL5-Verzeichnis zurüc
   if(StringFind(path,"::",pos)>=0)
      return(StringSubstr(path,pos));
//--- Finden wir ein Trennzeichen für die erste Unterverzeichnis in MQL5 (zB MQL5\Indi
//--- Wenn es gibt es nicht, geben wir den Pfad relativ zu MQL5-Verzeichnis zurück
   if((pos2=StringFind(path,"\\",pos+1))<0)
      return(StringSubstr(path,pos));
//--- geben wir den Pfad relativ zu Verzeichnis zurück (zB MQL5\Indicators)
   return(StringSubstr(path,pos2+1));
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const int begin,
                const double& price[])
  {
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Ressourcenvariablen

Man kann Ressourcen mithilfe von Ressourcenvariablen deklarieren und mit ihnen so umgehen, als ob sie eine Variable des entsprechenden Typs wären. Format der Deklaration:

© 2000-2025, MetaQuotes Ltd.

```
#resource Pfad_zur_Ressourcendatei as Typ_der_Ressourcenvariablen Name_der_Ressourcenv
```

Deklarationsbeispiel:

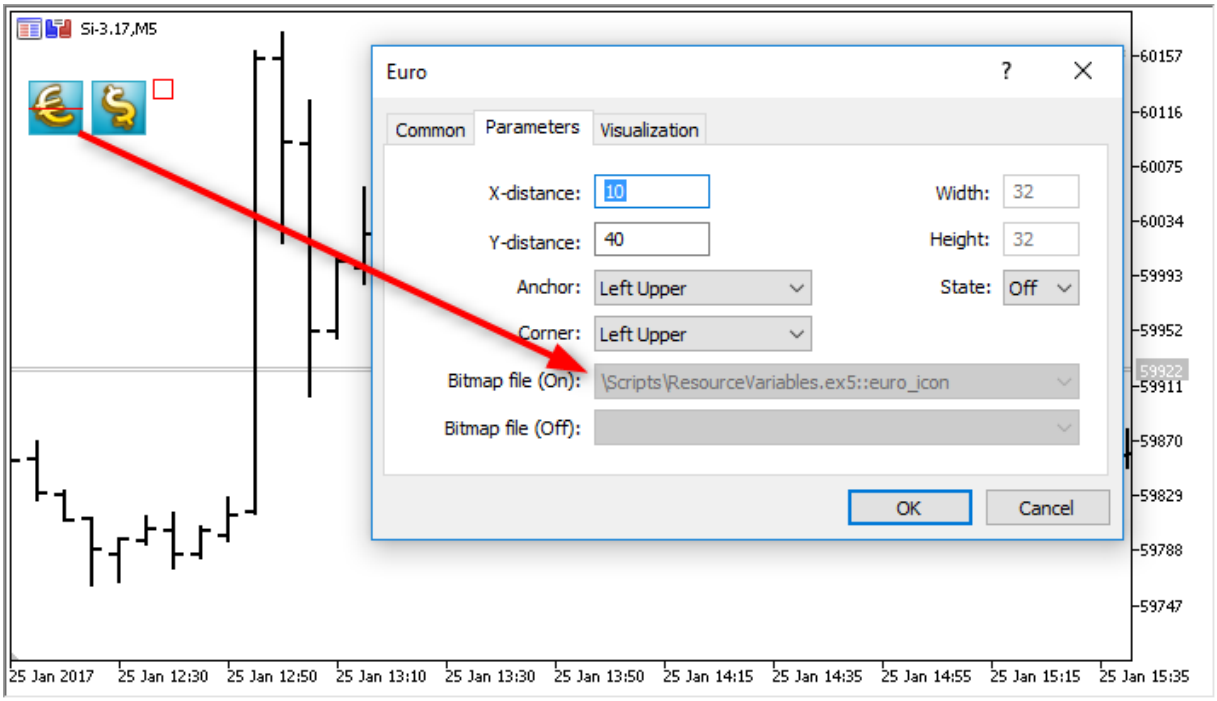
```
#resource "data.bin" as int ExtData[]              // Deklaration des Arrays vom nummer
#resource "data.bin" as MqlRates ExtData[]         // Deklaration des Arrays einfacher
//--- Strings
#resource "data.txt" as string ExtCode             // Deklaration des Strings, der die
//--- grafische Ressourcen
#resource "image.bmp" as bitmap ExtBitmap[]        // Deklaration des eindimensionalen
#resource "image.bmp" as bitmap ExtBitmap2[][]     // Deklaration des zweidimensionaler
```

Bei solcher Deklaration können die Daten der Ressourcen nur über eine Variable adressiert werden, die automatische Adressierung über "::<rsource name>" funktioniert nicht.

```
#resource "\\Images\\euro.bmp" as bitmap euro[][]
#resource "\\Images\\dollar.bmp"
//+------------------------------------------------------------------+
//|  Funktion für die Erstellung des Objekts OBJ_BITMAP_LABEL mithilfe der Ressource
//+------------------------------------------------------------------+
void Image(string name,string rc,int x,int y)
  {
   ObjectCreate(0,name,OBJ_BITMAP_LABEL,0,0,0);
   ObjectSetInteger(0,name,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(0,name,OBJPROP_YDISTANCE,y);
   ObjectSetString(0,name,OBJPROP_BMPFILE,rc);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- geben wir die Bildgröße [width, height] aus, das in der Ressourcenvariablen euro
   Print(ArrayRange(euro,1),", ",ArrayRange(euro,0));
//--- ändern wir das Bild in euro - zeichnen wir eine rote waagerechte Streife in der
   for(int x=0;x<ArrayRange(euro,1);x++)
      euro[ArrayRange(euro,1)/2][x]=0xFFFF0000;
//--- erstellen wir eine grafische Ressource mithilfe der Ressourcenvariablen
   ResourceCreate("euro_icon",euro,ArrayRange(euro,1),ArrayRange(euro,0),0,0,ArrayRang
//--- erstellen wir das Objekt grafische Beschriftung Euro und setzen wir ihm das Bild
   Image("Euro","::euro_icon",10,40);
//--- eine andere Methode der Verwendung der Ressource, wir können in ihr nicht zeichn
   Image("USD","::Images\\dollar.bmp",15+ArrayRange(euro,1),40);
//--- die direkte Methode der Adressierung von euro.bmp ist nicht zugänglich, denn es
   Image("E2","::Images\\euro.bmp",20+ArrayRange(euro,1)*2,40); // Fehler der Ausführu
  }
```

Das Ergebnis der Ausführung des Skripts: es sind nur zwei Objekte OBJ_BITMAP_LABEL aus drei erstellt. Dabei sehen wir auf der Abbildung des ersten Objekt eine rote Streife in der Mitte.

© 2000-2025, MetaQuotes Ltd.



Ein wichtiger Vorteil der Verwendung von Ressourcen besteht darin, dass Ressourcendateien, bevor sie in eine ausführbare EX5-Datei vor der Kompilation hinzugefügt werden, automatisch komprimiert werden. Die Verwendung von Ressourcenvariablen erlaubt notwendige Daten der ausführbaren EX5-Datei direkt hinzuzufügen und verringert die Anzahl sowie die Gesamtgröße der Dateien im Vergleich zur konventionellen Methode des Schreibens eines mql5-Programms.

Die Verwendung von Ressourcenvariablen ist besonders hilfreich bei der Veröffentlichung von Produkten im Market.

Eigenschaften

- Der spezielle Typ der Ressourcenvariablen *bitmap* zeigt dem Compiler, dass die Ressource eine grafische Abbildung darstellt. Solche Ressourcenvariablen bekommen den Typ uint.
- Die Array-Ressourcenvariable vom Typ *bitmap* kann zwei Dimensionen haben, in diesem Fall wird die Arraygröße als [Bildhöhe][Bildbreite] definiert. Wenn das Arrays eindimensional ist, wird die Anzahl der Elemente als Bildhöhe*Bildbreite definiert.
- Beim Herunterladen einer 24-Bit-Abbildung wird die Komponente des Alpha-Kanals für alle seine Pixel auf 255 gesetzt.
- Beim Herunterladen einer 32-Bit-Abbildung ohne Alpha-Kanal wird die Komponente des Alpha-Kanals für alle seine Pixel auch auf 255 gesetzt.
- Beim Herunterladen einer 32-Bit-Abbildung ohne Alpha-Kanal mit dem Alpha-Kanal werden die Pixel nicht verarbeitet.
- Die Datei der Ressource darf nicht größer als 128 Mb sein.
- Für String-Dateien wird die Kodierung automatisch nach BOM (Überschrift) bestimmt. Wenn BOM fehlt, wird die Kodierung nach dem Inhalt ermittelt. Es werden die Kodierungen ANSI, UTF-8 und UTF-16 unterstützt. Beim Lesen von Daten aus den Dateien werden alle Strings in Unicode umgewandelt.

OpenCL Programme

© 2000-2025, MetaQuotes Ltd.

Die Verwendung von Ressourcenvariablen kann die Entwicklung einiger Programme wesentlich vereinfachen. Sie können, zum Beispiel, den Code eines OpenCL-Programms in einer separaten CL-Datei schreiben und dann diese Datei als einen String den Ressourcen ihres MQL5-Programms hinzufügen.

```
#resource "seascape.cl" as string cl_program
...
int context;
if((cl_program=CLProgramCreate(context,cl_program)!=INVALID_HANDLE)
  {
   //--- weitere Operationen mit dem OpenCL Programm
  }
```

Ohne Ressourcenvariablen *cl_program* hätten Sie den ganzen Code als eine große String-Variablen beschreiben müssen.

Sehen Sie auch

ResourceCreate(), ResourceSave(), PlaySound(), ObjectSetInteger(), ChartApplyTemplate(), Dateioperationen

© 2000-2025, MetaQuotes Ltd.

Aufruf der Importfunktionen

für Import der Funktionen während der Ausführung des mql5-Programmes wird frühe Bindung verwendet. Das bedeutet wenn es im Programm Aufruf der Importfunktion gibt, wird der entsprechende Modul (ex5 oder dll) beim Programmladen geladen. Bibliotheken MQL5 und DLL Werden im Thread des Aufrufmoduls durchgeführt.

Es ist nicht empfehlenswert, den vollstaendig bestimmten Namen des geladenen Moduls der Art *Drive:\Directory\FileName.Ext verwenden*. Bibliotheken MQL5 werden vom Ordner *terminal_dir\MQL5\Libraries ausgeladen*. Wenn Bibliothek nicht gefunden wurde, versucht man, Bibliothek vom Ordner *terminal_dir\experts* herunterzuladen.

Systembibliotheken werden nach Regeln (DLL) des Operationssystems geladen. Wenn die Bibliothek schon geladen wird (zB. von anderen Experten oder aus anderem Client-Terminal, das parallel startet) wird die schon geladene Bibliothek aufgerufen.Anderenfalls erfolgt die Suche in der Folge:

1. Verzeichnis, von dem das dll importierende Modul abgelaufen wurde. Unter Modul wird Expert, Script, Indikator oder Bibliothek EX5 verstanden;
2. Verzeichnis Katalog_Data_Terminals\MQL5\Libraries (TERMINAL_DATA_PATH\MQL5\Libraries);
3. Verzeichnis, von dem das Client-Terminal MetaTrader 5 gestartet wurde;
4. Systemverzeichnis;
5. Verzeichnis Windows;
6. Laufendes Verzeichnis;
7. Verzeichnisse, die in Systemvariable PATH aufgelistet sind.

Wenn die Bibliothek DLL eine andere DLL in ihrer Arbeit verwendet, dann kann die erste DLL beim Fehlen der zweiten DLL nicht geladen werden.

Vor dem Laden des Experten (Script, Indikator) wird die gesamte Liste aller Bibliothekmoduln EX5 gebildet, die sowohl von dem geladenen Experten(Script, Indikator), als auch von der Bibliothek dieser Liste zu verwenden gedacht. So erfolgt das einmalige Laden der mehrmalig verwendeten Bibliothekmoduln EX5. Bibliotheken verwenden vorbestimmte Variablen des Experten (Script, Indikator), der sie aufgerufen hat.

Suche der importierten Bibliothek wird in der Folge durchgeführt:

1. Verzeichnis, Pfad zu dem in Bezug auf Verzeichnis des importierenden EX5 Experten (Script, Indikator) vorgegeben wird;
2. Verzeichnis Katalog_Terminals\MQL5\Libraries;
3. Verzeichnis MQL5\Libraries im Gesamtverzeichnis aller Client-Terminals MetaTrader 5(Common\MQL5\Libraries).

Funktionen, die aus DLL in mql5-Programm , importiert werden, müssen Windows API Abkommen garantieren. Für Garantieren dieses Abkommens wird im Ausgangstext der Programme, die mit den Sprachen C oder C++ geschrieben werden, das Schluesselwort __stdcall verwendet, das für Compiler der Gesellschaft Microsoft(r) spezifisch ist. Das Abkommen ist dadurch charakterisiert:

- ausrufende Funktion (hier mq5-Programm) muss "sehen" Prototyp der aufgerufenen Funktion (importierten aus DLL), um Parameter zum Stack richtig zusammenzustellen;

© 2000-2025, MetaQuotes Ltd.

- ausrufende Funktion (hier mql5-Programm) stellt Parameter auf Stack in umgekehrter Reihenfolge, von rechts nach links - gerade in dieser Reihenfolge liest die importierte Funktion die ihr übertragenen Parameter ein;
- Parameter werden nach Wert übertragen ausser deneb, die explizit durch Referenz übertragen werden (in diesem Fall sind es Zeilen)
- Importfunktion, klaert Stack selbst ab beim Einlesen der ihr übertragenen Parameter.

Beim Beschreiben des Prototyps der Importfunktion können Parameter mit Defaultwerten verwendet werden.

Falls die entsprechende Bibliothek nicht in der Lage war, geladen zu werden oder es ist untersagt, DLL zu verwenden oder die importierte Funktion nicht gefunden war - stoppt Expert seine Arbeit mit der entsprechenden Nachricht "expert stopped" im Journal. Dabei wird Expert nicht ablaufen lassen, bevor er neuinitialisiert wird. Expert kann neuinitialisiet werden im Ergebnis der Umcompilierung oder der Öffnung der Eigenschaften der Tabelle und Drücken der Schaltfläche OK.

Parameterübertragung

Alle Parameter der einfachen Typen werden nach Wert übertagen, wenn nicht explizit angegeben wird, dass sie durch Refernz übertragen werden. Bei der Übertragung einer Zeile wird Pufferadresse der kopierten Zeile übertragen; wenn eine Zeile durch Refenz übertragen wird, wird in die aus DLL importierte Funktion Pufferadresse gerader dieser Zeile Zeile ohne Kopieren übertragen.

Strukturen, die dynamische Felder, Zeilen, Klassen, andere komplizierte Strukturen, aber auch statische oder dynamische Felder der aufgelisteten Objekte enthalten, können nicht als Parameter in eine importierte Funktion übertragen werden.

Bei der Übertragung des Feldes in DLL wird immer (unabhängig von der Flagge AS_SERIES) die Adresse des Pufferanfangs übertragen. Funktion innerhalb DLL weiss nichts von der Flagge AS_SERIES, das übertragene Feld ist ein statisches Feld der unbekannten Laenge, für Spezifizieren der Feldgroesse verwenden Sie einen zusätzlichen Parameter.

© 2000-2025, MetaQuotes Ltd.

Ausführungsfehler

Im ausführenden Subsystem des Client-Terminals kann man den Fehlerkode aufbewahren, wenn er bei der Ausführung des mql5-Programms entsteht. für jedes ausführbare mql5-Programm gibt es die vorbestimmte Variable _LastError.

Vor dem Start der Funktion OnInit wird die Variable _LastError ausgenullt. Bei der fehlerhaften Situation während der Berechnungen oder beim Aufruf der eingebauten Funktion akzeptiert die Variable _LastError Kode des entsprechenden Fehlers. D n Wert, der in dieser Variable aufbewahren wird, kann man durch die Funktion GetLastError() erhalten.

Es gibt eine Reihe kritischer Fehler, deren Entstehung das Programm sofort brechen:

- Dividieren durch Null;
- Feldbereichsüberschreitung;
- Verwendung des unkorrekten Objektanzeigers.

© 2000-2025, MetaQuotes Ltd.

Testen von Handelsstrategien

Die Idee des automatisierten Handels ist attraktiv, weil ein Handelsroboter unermüdlich 24/7 arbeiten kann. Der Roboter wird nicht müde, zweifelhaft und hat keine Angst, er ist völlig frei von irgendwelchen psychischen Problemen. Sie müssen nur klar die Handelsregeln formalisieren und sie in den Algorithmen implementieren, und der Roboter ist fertig zu arbeiten. Aber zuerst müssen Sie sicherstellen, dass die folgenden zwei wichtige Bedingungen erfüllt sind:

- Der Expert Advisor führt Handelsoperationen in Übereinstimmung mit den Regeln des Handelssystems;
- Die Handelsstrategie, die im EA implementiert ist, zeigt einen Gewinn aus der Geschichte.

Um Antworten auf diese Fragen zu bekommen, verwenden wir den Strategie-Tester, der ein Teil von MetaTrader 5 Client Terminal ist.

In diesem Abschnitt werden die Merkmale von Testen und Optimierung der Programme in der Strategie-Tester beschrieben:

- Beschränkungen für die Funktionen im Strategie-Tester
- Modi Von Tick-Generierung
- Simulation von Spread
- Verwendung realer Ticks beim Testen
- Globale Variablen des Client-Terminals
- Berechnung von Indikatoren beim Testen
- Herunterladen von Geschichte beim Testen
- Multi-Currency-Testen
- Simulation der Zeit im Tester
- Grafische Objekte beim Testen
- OnTimer()-Funktion im Tester
- Sleep()-Funktion im Tester
- Print()-Funktion im Tester
- Verwenden des Testers für Optimierungsprobleme in mathematischen Berechnungen
- Synchronization von Balken beim Testen in "Open Prices Only"-Modus
- IndicatorRelease()-Funktion im Tester
- Behandlung von Ereignissen im Tester
- Testagenten
- Der Datenaustausch zwischen dem Terminal und dem Agenten
- Verwendung von einen freigegebenen Ordner von allen Client-Terminals
- Verwendung von DLL

Speicher- und Festplattenplatzbeschränkungen im MQL5 Cloud Network

© 2000-2025, MetaQuotes Ltd.

Für Optimierungen, die im MQL5 Cloud Network ausgeführt werden, gilt folgende Einschränkung: Der Expert Advisor darf nicht mehr als 4 GB an Informationen auf die Festplatte schreiben oder mehr als 4 GB an RAM verwenden. Wird diese Grenze überschritten, kann der Netzwerk-Agent die Berechnung nicht korrekt abschließen, und Sie erhalten kein Ergebnis. Allerdings wird Ihnen die gesamte für die Berechnungen aufgewendete Zeit in Rechnung gestellt.

Wenn Sie Informationen aus jedem Optimierungsdurchlauf benötigen, senden Sie Frames ohne auf die Festplatte zu schreiben. Um die Verwendung von Dateioperationen in Expert Advisors während der Berechnungen im MQL5 Cloud Network zu vermeiden, können Sie die folgende Prüfung verwenden:

```
int handle=INVALID_HANDLE;
bool file_operations_allowed=true;
if(MQLInfoInteger(MQL_OPTIMIZATION) || MQLInfoInteger(MQL_FORWARD))
   file_operations_allowed=false;

if(file_operations_allowed)
  {
   ...
   handle=FileOpen(...);
   ...
  }
```

Beschränkungen für die Funktionen im Strategie-Tester

Es gibt Beschränkungen für einige Funktionen im Strategie-Tester von Client Terminal.

Funktionen Comment(), Print() und PrintFormat()

Um die Leistung während Optimierung der Parameter von Experten zu erhöhen, werden die Funktionen Comment(), Print() und PrintFormat() nicht ausgeführt. Die Ausnahme ist die Verwendung dieser Funktionen innerhalb des OnInit()-Handlers. Dies erleichtert es, die Ursachen von Fehlern zu suchen, wenn sie auftreten.

Funktionen Alert(), MessageBox(), PlaySound(), SendFTP, SendMail(), SendNotification(), WebRequest()

Die Funktionen der Interaktion mit der "Außenwelt" Alert(), MessageBox(), PlaySound(), SendFTP(), SendMail(), SendNotification() und WebRequest() werden im Strategie-Tester nicht ausgeführt.

Modi Von Tick-Generierung

Ein Expert Advisor auf MQL5 ist ein Programm, das jedes Mal als Reaktion auf eine äußere Einwirkung - Ereignis - läuft. Für jedes vordefinierte Ereignis hat der EA die Funktion, die diesem Ereignis entspricht - Event-Handler.

© 2000-2025, MetaQuotes Ltd.

Das wichtigste Ereignis für den Expert Advisor ist eine Preisänderung - NewTick, und deshalb, um Expert Advisors zu testen, müssen wir Tick-Sequenz generieren. Es gibt drei Modi von Tick-Generation im Strategie-Tester von MetaTrader 5 Client-Terminal:

- Jeder Tick
- 1 Minute OHLC (OHLC Preise mit 1 Minute-Balken)
- Nur Eröffnungspreise

Die Grund- und detaillierteste ist der "Jeder Tick"-Modus. Sind die zwei anderen Modi die Vereinfachungen des basis Modus "Jeder Tick". Betrachten wir alle drei Modi, um die Unterschiede zwischen ihnen zu verstehen.

Jeder Tick

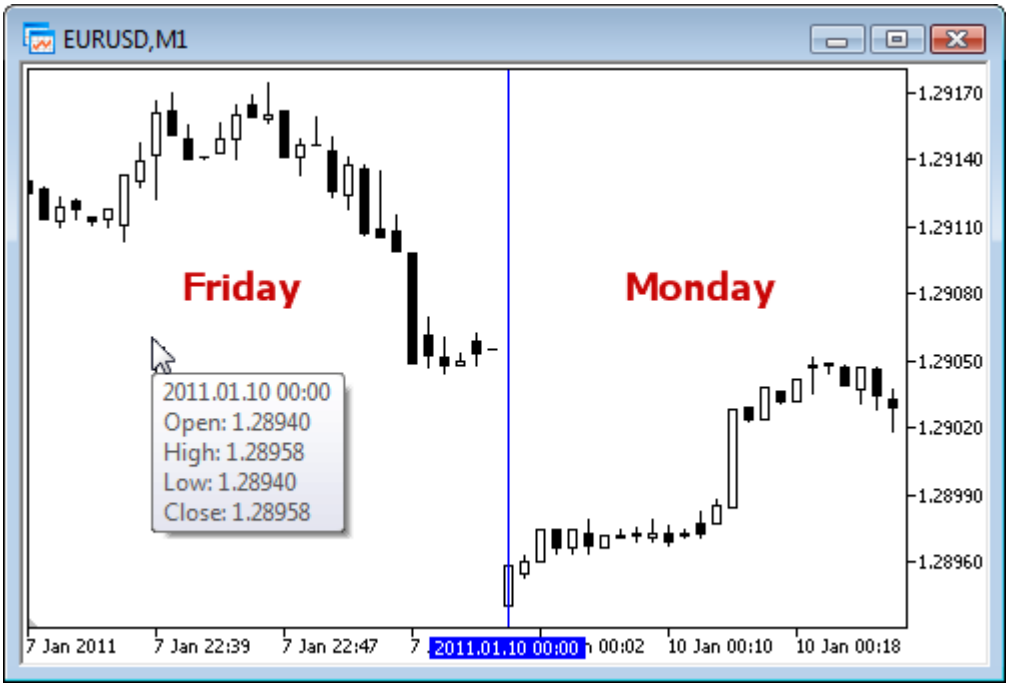
Die historische Preisdaten von Finanzinstrumenten werden aus dem Handelsserver an den MetaTrader 5 Client-Terminal in Form von sparsam verpackten Blöcke von 1-Minuten Balken übertragen. Für weitere Informationen über die Abfrage und Bauen der gewünschten Zeitrahmen lesen Sie bitte die Hilfethema Datenzugriff organisieren.

Das minimale Element der Preisgeschichte ist 1-Minute Balken, von dem Sie Informationen über die vier Werte der Preise bekommen können:

- Open - Preis, zu dem der 1-Minute Balken eröffnet wurde;
- High - das Maximum, das während dieses 1-Minute Balkens erreicht wurde;
- Low - das Minimum, das während dieses 1-Minute Balkens erreicht wurde;
- Close - Schlusspreis des Balkens.

Ein neue1 1-Minute-Balken öffnet nicht beim Start einer neuen Minute (Anzahl der Sekunden ist gleich 0), aber wenn ein Tick kommt - eine Preisänderung mindestens auf einen Punkt. Die Abbildung zeigt den ersten 1-Minute-Balken des neuen Handelswoche, der die Öffnungszeit des 2011.01.10 00:00 Uhr hat. Der Preisunterschied zwischen Freitag und Montag, den wir auf dem Chart sehen, ist weit verbreitet, da Wechselkurse schwankt auch am Wochenende als Reaktion auf eingehende Nachrichten.

© 2000-2025, MetaQuotes Ltd.

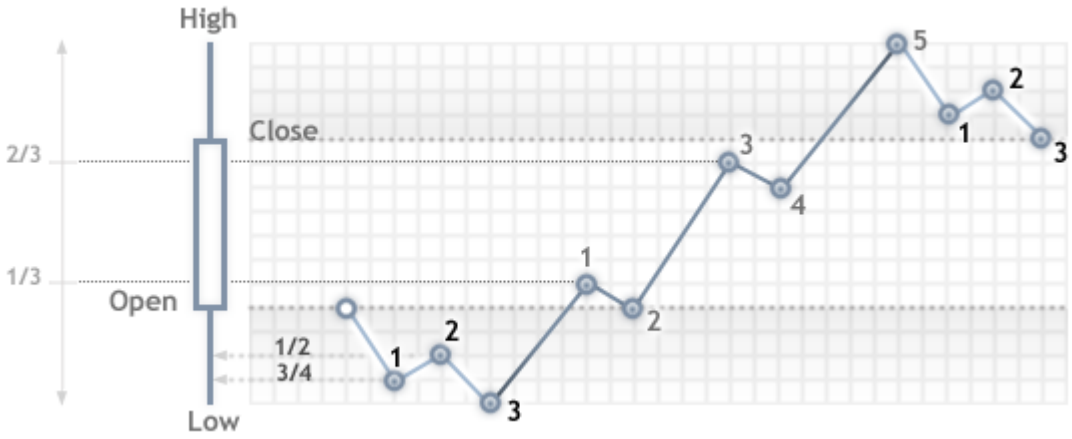


Für diesen Balken wir wissen nur, dass der 1-Minute-Balken auf 10 Januar 2011 in 00 Stunden 00 Minuten geöffnet wurde, aber wir wissen nichts über die Sekunden. Er könnte am 00.00.12 oder 00.00.36 (12 oder 36 Sekunden nach dem Start eines neuen Tages) oder jedem anderen Zeitpunkt innerhalb dieser Minute geöffnet worden sein. Aber wir wissen, dass der Open-Preis von EURUSD war 1,28940 bei der Eröffnung des neuen 1-Minute-Balkens.

Ebenso wissen wir nicht die Sekunde, wenn der Tick kamm, der den entsprechenden Schlusskurs dieses 1-Minute-Balkens entspricht. Wir wissen nur das es der letzte Preis auf diesem 1-Minute-Balken ist, der als Close-Preis aufgezeichnet wurde. Für diesen Minuten war der Preis 1,28958. Die Zeit des Auftretens von High- und Low-Preisen ist ebenfalls unbekannt, aber wir wissen, dass die maximalen und minimalen Preisen die Ebenen von 1,28958 und 1,28940 beziehungsweise besucht haben.

Um die Handelsstrategie zu testen, brauchen wir eine Folge von Ticks, auf der die Arbeit des Expert Advisors simuliert wird. Damit für jeden 1-Minute-Balken wissen wir die vier **Kontrollpunkte**, wo der Preis auf jeden Fall gewesen ist. Wenn der Balken nur 4 Ticks hat, dann sind diese Informationen ausreichend für Testen, aber in der Regel ist Tick-Volumen mehr als 4. Daher ist es notwendig, zusätzliche Kontrollpunkte für Ticks zu generieren, die zwischen den Preisen Open, High, Low und Close kam. Das Prinzip der Generierung von Ticks in der "Jeder Tick"-Modus wird in dem Artikel The Algorithm of Ticks' Generation within the Strategy Tester of the MetaTrader 5 Terminal, beschrieben, eine Abbildung von denen unten gezeigt ist.

© 2000-2025, MetaQuotes Ltd.



Beim Testen im "Jeder Tick"-Modus, wird die Funktion OnTick() von EA **an jedem Kontrollpunkt aufgerufen**. Jeder Kontrollpunkt ist ein Tick vom generierten Sequenz. Der Expert Advisor erhält die Zeit un den Preis des simulierten Ticks, so wie es wäre, wenn Sie online arbeiten.

> **Wichtig:** "Jeder Tick" ist der genauesten Test- Modus, aber zur gleichen Zeit nimmt er am meisten Zeit in Anspruch. Für die erste Auswertung der meisten Handelsstrategien ist in der Regel ausreichend, eine der beiden anderen Test-Modi zu verwenden.

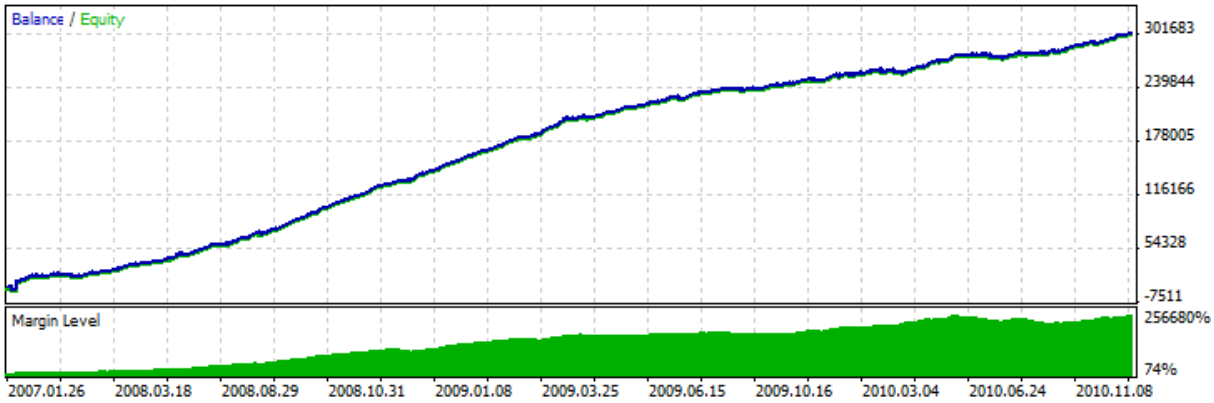
1 minute OHLC

"Jeder Tick" ist ist der genaueste der drei Modi, aber zur gleichen Zeit, ist er der langsamste. Der Lauf der OnTick()-Handler erfolgt bei jedem Tick, während Tick-Volumen ziemlich groß sein können. Für eine Strategie, bei der die Ticksfolge von Kursbewegungen während des Balkens keine Rolle spielt, gibt es ein schneller und rauer Simulationsmodus - "1 Minute OHLC".

Im "1 minute OHLC"-Modus wird Tick-Sequenz nur auf der Basis von **OHLC-Preisen von 1-Minute-Balken** gebaut, wird die Anzahl der generierten Kontrollpunkte deutlich reduziert - damit die Testzeit auch reduziert. Der Start von OnTick()-Funktion wird auf alle Kontrollpunke gemacht, die auf die Preise OHLC von 1-Minute_Balken gebaut werden.

Die Weigerung, zusätzliche Ticks zwischen den Preisen von Open, High, Low und Close zu generieren, führt zu einer starren Determinismus in der Entwicklung des Preises vom Moment, wenn der Open-Preis bestimmt wird. Dies macht es möglich, einen "Gral des Tests" zu erstellen, der einen schönen aufsteigenden Bilanz-Chart für Tests zeigt. Ein Beispiel von solchem Gral ist in Code Base verfügbar - Grr-al.

© 2000-2025, MetaQuotes Ltd.



Die Abbildung zeigt einen sehr attraktiven Chart von Testen dieses Expert Advisors. Wie wird es aufgenommen? Wir wissen 4 Preise für den 1-Minute-Balken, und wir wissen auch, dass der erste Preis ist Open, und der letzte ist Close. Zwischen ihnen sind High und Low, ist die Reihenfolge ihres Auftretens nicht bekannt, aber es ist bekannt, dass der Preis High größer als oder gleich zu Open ist (Low ist kleiner oder gleich dem Preis Open).

Ist ausreichend genug, die Zeit von Erhalt des Open-Preises zu ermitteln, und dann den nächsten Tick zu analysieren, um festzustellen, ob es High oder Low ist. Wenn der Preis liegt unter dem Open-Preis, dann haben wir den Low-Preis - kaufen wir auf diesem Tick, wird der nächste Tick dem High-Preis entsprechen, an denen wir schließen Buy und öffnen Sell. Der nächste Tick ist der letzte, er ist der Close-Preis, und wir schließen den Verkauf (Sell) auf ihn.

Wenn nach dem Preis wir einen Tick mit einem Preis größer als der Open-Preis erhalten, dann ist die Reihenfolge der Trades umgekehrt. Verarbeiten nun einen 1-Minute-Balken in diesem "Cheat"-Modus, und warten auf den nächsten. Bei der Prüfung eines solchen Expert Advisors auf die Geschichte läuft alles gut, aber wenn er online lauft, sehen wir die Wahrheit - die Linie der Balance ist immer noch glatt, aber es geht nach unten. Um diesen Trick zu entlarven, müssen wir einfach den EA in der "Jeder Tick"-Modus ausführen.

> **Wichtig:** Wenn die Testergebnisse von EA in den rauen Test-Modi ("1 Minute OHLC" und "Nur Eröffnungspreise") zu gut scheinen, testen Sie ihn im "Jeder Tick"-Modus.

Nur Eröffnungspreise

In diesem Modus werden Ticks basierend auf den OHLC Preise vom Zeitraum, der für Testen ausgewählt ist, generiert. Die OnTick()-Funktion des Expert Advisors läuft nur am Anfang des Balkens auf dem Open-Preis. Aufgrund dieser Funktion Stopp-Ebenen und erledigten Aufträge können zu nicht angegebenem Preis aktiviert werden (vor allem beim Testen auf höheren Zeitrahmen). Im Gegenzug dafür sind wir in der Lage, einen Expert Advisor schnell zu testen.

Eine Ausnahme bei der Generierung von Ticks im "Nur Eröffnungspreise"-Modus ist die Perioden W1 und MN1: Zeitrahmen für diese Ticks sind für den OHLC Preise jeden Tages generiert, sondern nicht für OHLC Preise der Woche oder des Monats, jeweils.

Zum Beispiel testen wir einen Expert Advisor auf EURUSD H1 im "Nur Eröffnungspreise"-Modus. In diesem Fall ist die Gesamtzahl von Ticks (Kontrollpunkte) nicht mehr als 4*Anzahl der 1-Stunde-Balken, die in der Testenintervall sind. Aber gleichzeitig **wird OnTick()-Handler nur an der Öffnung** von 1-Stunde-**Balken** aufgerufen. Auf der anderen (""versteckt" aus dem Expert Advisor) Ticks laufen Prüfungen, die für korrekten Testen notwendig sind:

© 2000-2025, MetaQuotes Ltd.

- Berechnung der Margin-Anforderungen;
- Auslösen von Stop Loss und Take Profit Ebenen;
- Auslösen von erledigten Aufträge;
- Löschen von abgelaufenen erledigten Aufträge.

Wenn es keine offenen Positionen oder erledigten Aufträge gibt, brauchen wir nicht diese Prüfungen auf versteckten Ticks, und Performance-Gewinn kann erheblich sein. Dieser Modus "Nur Eröffnungspreise" ist gut zum Testen von Strategien, die Deals nur auf der Eröffnung von Balken machen und erledigten Aufträge nicht verwenden, und auch verwenden sie nicht StopLoss, TakeProfit. Für die Klasse solcher Strategien bleibt alle notwendigen Genauigkeit von Tests.

Verwenden wir Moving Average Expert Advisor aus dem Standard-Paket als ein Beispiel für einen EA, die in jedem Modus getestet werden kann. Die Logik dieses EAs ist so, dass alle Entscheidungen bei der Eröffnung des Balkens und Deals werden sofort ausgeführt, ohne die Verwendung von erledigten Aufträge gemacht. Starten wir Testen auf EURUSD H1 im Intervall von 2010.01.09 bis 2010.31.12 un vergleichen wir die Charts. Die Abbildung zeigt Bilanz-Charts aus dem Tester-Bericht für alle drei Modi.



Wie Sie sehen können Charts auf den verschiedenen Modi des Tests sind die gleichen für den EA Moving Average.

Es gibt mehrere Einschränkungen des Modes "Nur Eröffnungspreise":

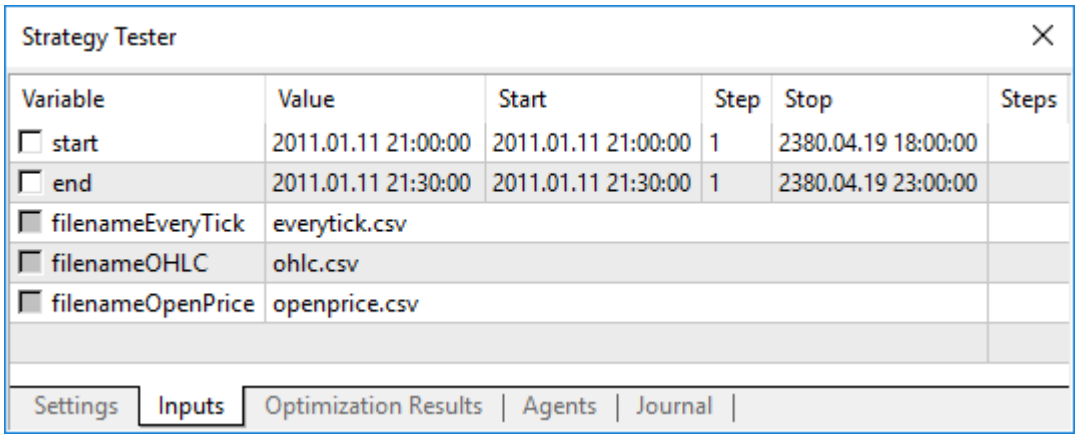
- Sie können nicht Handelsregime "zufällige Verzögerung " verwenden;
- Im EA können Sie die Daten am Zeitrahmen, der niedriger als der für die Tests/Optimierung eingesetzte Zeitrahmen ist, nicht verwenden. Zum Beispiel, wenn der Test / Optimierung auf Zeitraum H1 durchgeführt wird, können Sie Zugriff auf die Daten von H2, H3, H4, etc. haben, nicht aber auf die Daten M30, M20, M10, etc. Darüber hinaus älteren Zeiträumen, die aufgerufen sind, müssen ein Vielfaches des Test-Zeitrahmens sein. Zum Beispiel, wenn beim Testen auf M20 können Sie nict M30 zugreifen, aber die Daten aus H1 sind verfügbar. Diese Einschränkungen sind aufgrund der Unfähigkeit, Daten von unteren und nicht multiplen Zeitrahmen von die Balken, die während des Test/der Optimierung generiert werden, zu erhalten.

© 2000-2025, MetaQuotes Ltd.

- Einschränkungen beim Zugriff auf Daten anderer Zeitrahmen auch auf andere Symbole, deren Daten im Expert Advisor verwendet werden, gelten. In diesem Fall ist die Begrenzung für jedes Symbol von dem ersten Zeitrahmen abhängig, der während des Tests/der Optimierung aufgetreten ist. Z. B. die wir testen auf das Symbol und die Periode EURUSD H1, der EA greift Daten von GBPUSD M20 für das erste Mal. In diesem Fall ist der EA in der Lage, weitere Daten EURUSD H1, H2, usw. zu verwenden, sowie GBPUSD M20, H1, H2 usw.

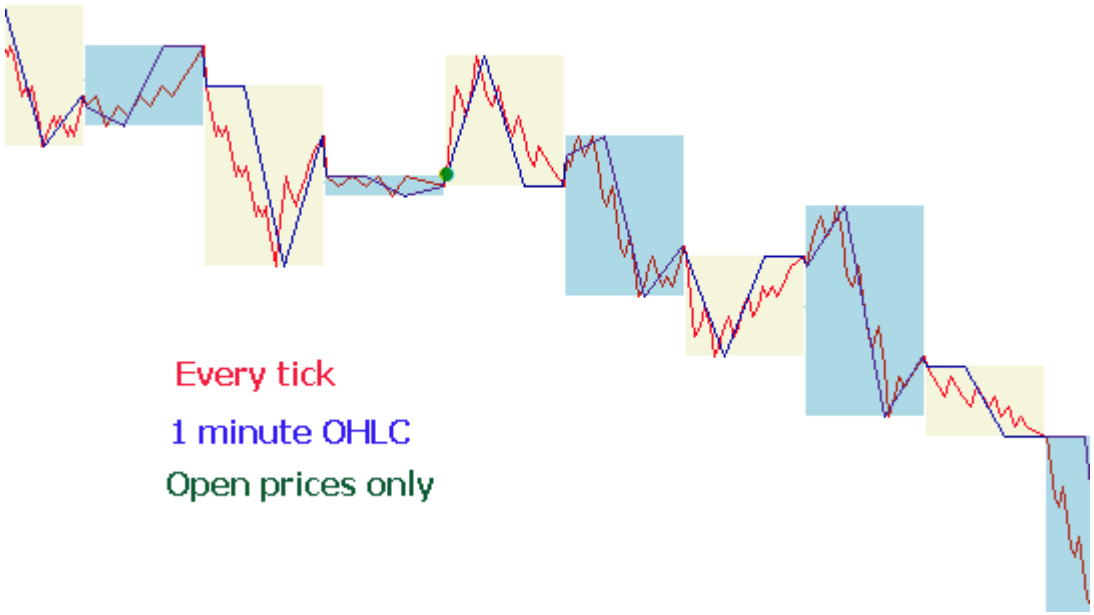
Wichtig: "Nur Eröffnungspreise"-Modus ist der schnellste, aber er kann nicht für alle Trading-Strategien verwendet sein. Wählen Sie den gewünschten Test-Modus basiert auf die Eigenschaften des Handelssystems.

Am Ende des Abschnitts über die Modellierung Modi geben wir einen visuellen Vergleich der verschiedenen Arten der Generierung von Ticks für EURUSD für zwei Balken M15 auf dem Intervall 2011.01.11 21:00:00 - 2011.01.11 21.30.00. Die Ticks wurden in verschiedene Dateien mit Hilfe vom Expert Advisor WriteTicksFromTester.mq5 gespeichert, und das Ende der Namen dieser Dateien sind in input-Parametern filenamEveryTick, filenameOHLC und filenameOpenPrice angegeben.



Um drei Dateien mit drei Tick-Sequenzen (für jeden der Modi "Jeder Tick", "1 Minute OHLC" und "Nur Eröffnungspreise") zu erhalten, wurde der Expert Advisor dreimal in den etsprechenden Modi mit der einzelnen Läufe gestartet. Dann wurden die Daten aus diesen drei Dateien mit Indikator TicksFromTester.mq5 auf dem Chart gezeigt. Indikator-Code wird an diesem Artikel angebracht.

© 2000-2025, MetaQuotes Ltd.



Standardmäßig werden alle Dateioperationen in MQL5 Sprache innerhalb von "Datei-Sandbox" gemacht, und beim Testen kann der Expert Advisor nur eigene "Dateisystem-Sandbox" zugreifen. Um der Indikator und der EA beim Test mit Dateien von einem Ordner arbeiten könnten, verwendeten wir die Flagge FILE_COMMON. Ein Beispiel-Code aus dem EA:

```
//--- Datei öffnen
   file=FileOpen(filename,FILE_WRITE|FILE_CSV|FILE_COMMON,";");
//--- Den Erfolg der Operation überprüfen
   if(file==INVALID_HANDLE)
     {
      PrintFormat("Datei %s konnte nicht für Schreiben geöffnet werden. Fehlercode=%d"
      return;
     }
   else
     {
      //--- Informieren wir über das Schreiben in den Ordner von allen Client-Terminal
      PrintFormat("Die Datei wird im Ordner %s geschrieben",TerminalInfoString(TERMINA
     }
```

Im Indikator beim Datenlesen haben wir auch Flagge FILE_COMMON verwendet. Dies erlaubte uns, manuelle Übertragung der notwendigen Dateien von einem Ordner zum anderen zu vermeiden.

```
//--- Datei öffnen
   int file=FileOpen(fname,FILE_READ|FILE_CSV|FILE_COMMON,";");
//--- Den Erfolg der Operation überprüfen
   if(file==INVALID_HANDLE)
     {
      PrintFormat("Datei %s konnte nicht für Lesen geöffnet werden. Fehlercode=%d",fna
      return;
     }
   else
     {
      //--- Informieren über den Speicherort des freigegebenen Ordners von allen Clien
      PrintFormat("Die Datei wird aus dem Ordner %s gelesen werden",TerminalInfoStrin
     }
```

© 2000-2025, MetaQuotes Ltd.

Simulation von Spread

Die Differenz zwischen den Preisen Bid und Ask wird Spread genannt. Beim Testen wird der Spread nicht modelliert, sondern wird er aus historischen Daten genommen. If the spread is less than or equal to zero in the historical data, then the last known (at the moment of generation) spread of is used by testing agent.

Im Tester ist Spread immer schwimmend. D.h. SymbolInfoInteger(symbol, SYMBOL_SPREAD_FLOAT) gibt immer true zurück.

Darüber hinaus werden in historische Daten Werte von Tick- und Handelsvolumen gespeichert. Für das Speichern und Abrufen von Daten verwenden wir eine spezielle MqlRates:

```
struct MqlRates
  {
   datetime time;         // Balken Öffnungszeit
   double   open;         // Open-Preis
   double   high;         // High-Preis
   double   low;          // Low-Preis
   double   close;        // Close-Preis
   long     tick_volume;  // Tick-Volumen
   int      spread;       // Spread
   long     real_volume;  // Markt-Volumen
  };
```

Verwendung realer Ticks beim Testen

Das Testen und die Optimierung anhand realer Ticks sind besonders nah an den realen Bedingungen. Statt der anhand Minutendaten generierten Ticks werden reale Ticks verwendet, die vom Broker gespeichert wurden. Dies sind die Ticks von Börsen und Liquiditätsanbietern.

Um maximale Genauigkeit beim Testen zu gewährleisten, werden auch Minutenbalken im Modus "Anhand realer Ticks" verwendet. Anhand der dieser Balken werden die Tickdaten überprüft und korrigiert. Darüber hinaus kann man auf solche Weise Abweichungen der Charts im Tester und im Kundenterminal voneinander vermeiden.

Der Tester überprüft die Übereinstimmung der Tickdaten mit den Parametern eines Minutenbalkens: der Tick darf nicht die High/Low Levels des Balkens überschreiten; der Tick, der eine Minute eröffnet und schließt, muss mit den Open/Close Preisen des Balkens übereinstimmen. Die Volumina werden auch miteinander verglichen. Wenn eine Unstimmigkeit festgestellt wird, werden alle Ticks, die diesem Minutenbalken entsprechen, fortgelassen. Statt dieser Ticks werden erzeugte Ticks verwendet (wie im "Alle Ticks" Modus).

Wenn in der Historie eines Symbols ein Minutenbalken vorhanden ist, die Tickdaten für diese Minute aber nicht vorliegen, generiert der Strategietester die Ticks im Modus "Alle Ticks". Dies erlaubt es, einen korrekten Chart im Falle unvollständiger Tickdaten zu zeichnen.

Wenn in der Historie eines Symbols kein Minutenbalken vorhanden ist, die Tickdaten für diese Minute aber vorliegen, können sie im Strategietester verwendet werden. Die Balken von Börsensymbolen werden zum Beispiel nach Last Preisen gebildet. Wenn vom Server nur Ticks mit Bid/Ask Preisen ohne Last Preis eintreffen, wird der Balken nicht gebildet. Der Strategietester wird diese Tickdaten verwenden, weil sie den Minutendaten nicht widersprechen.

© 2000-2025, MetaQuotes Ltd.

Die Tickdaten können aus unterschiedlichen Gründen nicht mit den Minutenbalken übereinstimmen. Zum Beispiel infolge einer Unterbrechung der Verbindung oder anderer Störungen bei der Übertragung der Daten von einer Quelle ins Kundenterminal. Beim Testen gelten Minutenbalken als genauer und verlässlicher.

Beim Testen anhand realer Ticks sind die folgenden Besonderheiten zu berücksichtigen:

- Beim Starten eines Tests werden nicht nur Tickdaten, sondern auch Minutendaten eines Symbols synchronisiert.
- Die Ticks werden im Cache des Symobols im Tester gespeichert. Die Cache-Größe beträgt max. 128 000 Ticks. Beim Eintreffen neuer Ticks werden die ältesten Ticks aus dem Cache entfernt. Allerdings kann man mithilfe der CopyTicks Funktions auch die Ticks erhalten, die außerhalb des Cache liegen (nur beim Testen anhand realer Ticks). In diesem Fall werden die Daten aus der Datenbank von Ticks im Strategietester abgerufen, die mit der jeweiligen Datenbank im Kundenterminal übereinstimmt. In der Datenbank werden keine Korrekturen anhand der Minutenbalken vorgenommen. Aus diesem Grund können sich diese Ticks von den Ticks im Cache unterscheiden.

Globale Variablen des Client-Terminals

Beim Testen werden globale Variablen, des Client-Terminals auch emuliert, aber sie werden nicht mit echten globalen Variablen des Terminals, die können im Terminal auf der Taste F3 gesehen werden, verbunden. Dies bedeutet, dass alle Operationen mit globalen Variablen beim Testen ausserhalb des Terminals hergestellt werden (in der Test-Agent).

Berechnung von Indikatoren beim Testen

Im Echtzeitmodus werden die Indikatorwerte bei jedem Tick berechnet.

Im Strategietester werden die Indikatoren nur dann berechnet, wenn auf die Daten zugegriffen wird, d.h. wenn Indikatorpufferwerte angefordert werden. Die einzigen Ausnahmen sind nutzerdefinierte Indikatoren mit der Direktive #property tester_everytick_calculate. In diesem Fall wird bei jedem Tick eine Neuberechnung durchgeführt.

Im visuellen Testmodus werden alle Indikatoren beim Eintreffen eines neuen Ticks bedingungslos neu berechnet, um im visuellen Testchart korrekt angezeigt zu werden.

Der Indikator wird einmal pro Tick berechnet. Alle nachfolgenden Anfragen nach Indikatordaten führen erst dann zu einer Neuberechnung, wenn ein neuer Tick eintrifft. Wenn also der Timer in einem EA über die Funktion EventSetTimer() aktiviert ist, werden die Indikatordaten ab dem letzten Tick vor jedem Aufruf von OnTimer() angefordert. Wenn der Indikator zum letzten Tick noch nicht berechnet wurde, werden die Berechnungen der Indikatorwerte gestartet. Wurden die Daten bereits vorbereitet, werden sie ohne Neuberechnung bereitgestellt.

Somit werden alle Indikatorberechnungen auf die ressourcenschonendste Art und Weise durchgeführt — wenn der Indikator zu einem bestimmten Tick bereits berechnet wurde, werden seine Daten "wie sie sind" bereitgestellt. Es wird keine Neuberechnung gestartet.

Herunterladen von Geschichte beim Testen

© 2000-2025, MetaQuotes Ltd.

Die Geschichte eines getesteten Symbols wird vor dem Testprozess synchronisiert und durch das Terminal aus dem Handel-Server geladen. In der ersten Zeit, lädt das Terminal alle verfügbaren Geschichte eines Symbols. Weiterhin werden nur die neuen Daten geladen.

Test-Agent erhält Geschichte des getesteten Symbols vom Terminal unmittelbar nach dem Start von Testen. Wenn Daten von anderen Instrumenten in den Testprozess verwendet werden (zum Beispiel, es ist ein Multi-Currency-Expert Advisor), dann erfordert der Test-Agent Geschichte aus dem Terminal beim ersten Anruf zu solchen Daten. Wenn historische Daten zur Verfügung am Terminal sind, wurden sie sofort an die Test-Agenten übertragen. Wenn Daten nicht verfügbar sind, lädt das Terminal sie vom Server und dann übergibt sie an Agenten.

Daten von zusätzlichen Instrumenten werden auch zur Berechnung von Cross-Rates für den Handelsoperationen erforderlich. Zum Beispiel, wenn beim Testen einer Strategie auf EURCHF mit Depot-Währung in USD, vor der Verarbeitung des ersten Handeloperation fordert der Agent historische Daten von EURUSD und USDCHF aus dem Terminal, obwohl die Strategie enthält keine direkten Aufruf von diesen Symbolen.

Vor dem Testen einer Multi-Currency-Strategie, empfiehlt es sich, alle notwendigen historischen Daten auf dem Terminal herunter zu laden. Dies wird dazu beitragen, Verzögerungen von Tests/Optimierung auf Grund des Download der erforderlichen Daten zu vermeiden. Zum Beispiel können Sie die Geschichte durch das Öffnen des entsprechenden Charts und Scrollen ihn zu Beginn der Geschichte herunterladen. Ein Beispiel für gezwungenen Download der Geschichte in Terminal ist in der MQL5 Dokumentationabschnitt Datenzugriffsanordnung gegeben.

Test-Agenten empfangen Geschichte vom Terminal in der verpackten Form. Während des nächsten Testen, lädt Tester nicht Geschichte aus dem Terminal, weil die erforderlichen Daten seit dem letzten Lauf des Testers verfügbar sind.

- Das Terminal lädt die Geschichte aus dem Handelsserver nur einmal, wenn der Agent die Geschichte für das testierte Symbol aus dem Terminal zum ersten Mal anfordert. Die Geschichte wird in gepackter Form geladen, um den Verkehr zu reduzieren.
- Ticks werden nicht über das Netzwerk gesendet, werden sie auf Tester-Agenten generiert.

Multi-Currency-Testen

Mit dem Tester können Sie Strategien, die auf mehreren Instrumenten handeln, auf der Geschichte überprüfen. Diese Experten werden herkömmlicherweise als Multi-Währungs-EA bezeichnet, da in den vorherigen Plattformen Test für nur ein Instrument durchgeführt wurde. MetaTrader 5 Tester kann auch Handel Handel über alle verfügbaren Instrumente simulieren.

Geschichte der verwendeten Instrumente wird vom Tester aus dem **Client-Terminal** (nicht aus dem Handelsserver!) automatisch beim ersten Aufruf an das Instrument heruntergeladen.

Der Agent lädt nur fehlende Geschichte mit einer geringen Marge, um die notwendigen Daten über die Geschichte für die Berechnung von Indikatoren zu Beginn des Tests zu haben. Die minimale Geschichten beim Download aus dem Handelsserver für Zeitrahmen D1 und weniger ist ein Jahr. Wenn daher das Testen auf dem Intervall 2010.11.01-2010.12.01 (mit jeweils einem Monat als Intervall) mit der Periode von M15 (jede Bar ist gleich 15 Minuten) gestartet wird, wird die Geschichte des Instrumentes von dem Terminal für das gesamte Jahr 2010 angefordert. Für den Zeitrahmen Weekly wird die Geschichte von 100 Bars angefordert, die ungefähr zwei Jahre ist (52 Wochen pro Jahr). Für

© 2000-2025, MetaQuotes Ltd.

das Testen auf einer monatlichen Zeitrahmen Monthly wird der Agent die Geschichte für 8 Jahre (12 Monate * 8 Jahre = 96 Jahre) anfordern.

Wenn aus irgendwelchen Gründen die für das Testen notwendige Reserve der Bars vor dem Anfang des Testens nicht versorgt werden kann, wird **das Anfangsdatum des Testens** von der Vergangenheit bis zur Gegenwart automatisch geschoben, um solche Reserve zu gewährleisten.

Beim Testen wird auch "Markt Durchsicht" emuliert, in der man die Informationen über Instrumente erhalten kann. Standardmäßig gibt es am Anfang des Testens nur ein Symbol im "Markt Durchsicht" Tester - das Symbol, auf dem das Testen gestartet wurde. Alle notwendigen Symbole werden zum "Markt Durchsicht" Tester (nicht Terminal!) automatisch beim Zugriff verbunden.

> Vor dem Anfang des Testens eines Multi-Currency-Expert Advisors ist es notwendig die für das Testen erforderlichen Instrumente im " Markt Durchsicht" des Terminals auszuwählen und die erforderlichen Daten zu laden. Wenn Sie zum ersten Mal auf das fremde Symbol zugreifen, wird Daten dieses Symbols zwischen dem Test-Agenten und dem Client-Terminal automatisch synchronisiert. Das "fremde" Symbol - ist ein Symbol, das anders ist als dasjenige, auf dem das Testen gestartet wurde.

Der Zugriff auf das Daten des fremden Symbols erfolgt in den folgenden Fällen:

- Verwendung der Funktionen der technischen Indikatoren und IndicatorCreate() auf Symbol/Zeitrahmen;
- Anforderung an "Markt Durchsicht" (Market Watch) nach fremdem Symbol:
  1. SeriesInfoInteger
  2. Bars
  3. SymbolSelect
  4. SymbolIsSynchronized
  5. SymbolInfoDouble
  6. SymbolInfoInteger
  7. SymbolInfoString
  8. SymbolInfoTick
  9. SymbolInfoSessionQuote
  10. SymbolInfoSessionTrade
  11. MarketBookAdd
  12. MarketBookGet
- Anforderung an Zeitreihen nach Symbol/Zeitrahmen durch die Funktionen:
  1. CopyBuffer
  2. CopyRates
  3. CopyTime
  4. CopyOpen
  5. CopyHigh
  6. CopyLow
  7. CopyClose

© 2000-2025, MetaQuotes Ltd.

8. CopyTickVolume
9. CopyRealVolume
10. CopySpread

Wenn Sie zum ersten Mal auf das fremde Symbol zugreifen, wird der Prozess des Testens gestoppt und die Geschichte von Symbol/Zeitrahmen vom Terminal auf dem Test-Agenten heruntergeladen. Gleichzeitig wird die Generierung einer Folge von Ticks für dieses Symbol aktiviert.

Für jedes Instrument wird seine eigene Folge von Ticks in Übereinstimmung mit dem gewählten Modus der Generierung von Ticks generiert. Ausserdem können Sie auch die Geschichte für die notwendigen Symbolen durch den Aufruf von der Funktion SymbolSelect () im Handler OnInit () anfordern - Die Geschichte wird unmittelbar vor dem Testen des Expert Advisors geladen.

Um das Multi-Currency-Testen im Client-Terminal MetaTrader 5 zu durchführen, ist es nicht erforderlich zusätzliche Anstrengungen zu unternehmen. Wir müssen nur die Charts der entsprechenden Instrumente im Client-Terminal öffnen. Die Geschichte der notwendigen Symbole wird automatisch vom Hadels-Server geladen, sofern die Daten darauf sind.

Simulation der Zeit im Tester

Beim Testen ist die lokale Zeit TimeLocal() immer gleich der Server-Zeit TimeTradeServer(). Im Gegenzug ist die Server-Zeit immer gleich der Zeit, die der GMT - TimeGMT()entspricht. Somit zeigen alle diese Funktionen beim Testen die gleiche Zeit.

Das Fehlen der Unterschied zwischen GMT, lokale und Server Zeit wurde in dem Tester bewusst gemacht, weil die Verbindung zum Server nicht immer hergestellt werden kann. Und die Ergebnisse des Testens sollen identisch sein, unabhängig davon, ob es die Verbindung gibt oder nicht. Information über die Zeit der Server wird nicht lokal aufbewahren, sondern wird vom Server geholt.

Grafische Objekte beim Testen

Während des Testens / Optimierung wird die Konstruktion von grafischen Objekten nicht ausgeführt. Wenn man auf die Eigenschaften des geschaffenen Objektes während des Testens / Optimierung zugreift, erhält der Expert Advisor einen Nullwert.

> Diese Einschränkung gilt nicht für das Testen im visuellen Modus.

OnTimer()-Funktion im Tester

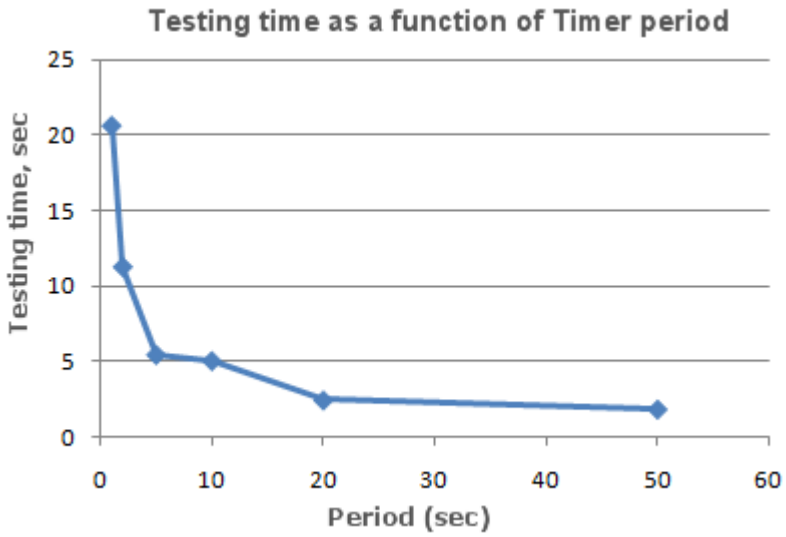
MQL5 bietet die Möglichkeit die Ereignisse des Timers zu bearbeiten. Der Handler OnTimer() wird unabhängig von Modus des Testens aufgerufen. Dies bedeutet, dass, wenn das Testen im Modus "Nur Eröffnungspreise" auf den Zeitraum H4 gestartet wird und innerhalb des Expert Advisors wird einen Timer gestellt, der jede Sekunde aufruft, dann wird der Handler onTick () auf der Eröffnung jedes H4 Balkens einmal aufgerufen und der Handler OnTimer () wird während des Balkens 14.400-mal (3600 Sekunden * 4.00 Stunden) aufgerufen. Soviel die Zeit des Testens des Expert Advisors dabei zunehmen wird, hängt von der Logik des Expert Advisors ab.

Um die Abhängigkeit der Zeit des Testens von der vorgegebenen Frequenz des Timers zu überprüfen, wurde ein einfacher Expert Advisor ohne Handelsoperationen geschrieben.

© 2000-2025, MetaQuotes Ltd.

```
//--- input parameters
input int      timer=1;              // Timerwert, Sek
input bool     timer_switch_on=true; // Timer ist aktiviert
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Timer läuft wenn  timer_switch_on==true
   if(timer_switch_on)
     {
      EventSetTimer(timer);
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- stoppen den Timer
   EventKillTimer();
  }
//+------------------------------------------------------------------+
//| Timer function                                                   |
//+------------------------------------------------------------------+
void OnTimer()
  {
//---
// unternehmen wir nichts,der Körper des Handlers ist leer
  }
//+------------------------------------------------------------------+
```

Die Zeit des Testens wurde bei verschiedenen Werten des Parameter von Timer (Periodizität des Ereignisses Timer) gemessen. Auf den erhaltenen Daten ist der Chart der Abhängigkeit der Zeit des Testens T von der Wert der Periodizität Period gebaut.



© 2000-2025, MetaQuotes Ltd.

Es ist deutlich zu erkennen, je geringer der Parameter Timer bei der Initialisierung durch Funktion EventSetTimer(Timer) ist, desto weniger ist der Zeitraum (Period) zwischen den Aufrufen der Handler OnTimer (), und desto mehr ist die Zeit des Testens T unter den gleichen anderen Bedingungen.

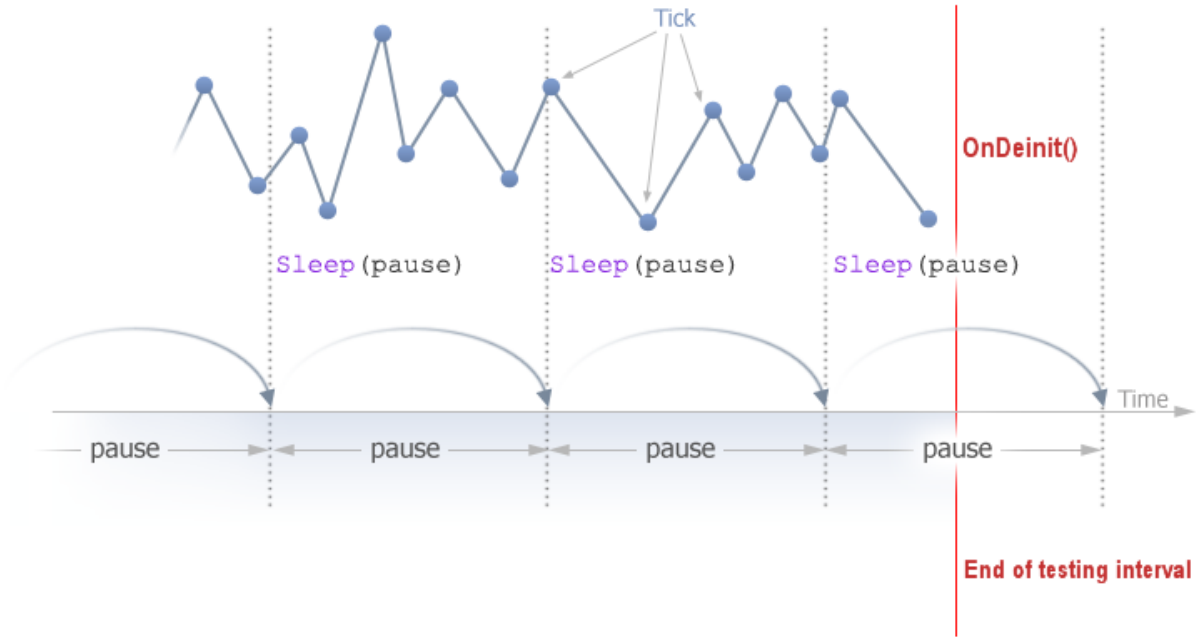
Sleep()-Funktion im Tester

Die Funktion Sleep() ermöglicht es Ihnen, im Expert Advisor oder Script die Ausführung des mql5-Programms für eine Zeitlang bei der Arbeit auf dem Chart auszusetzen. Dies kann nützlich sein wenn Sie irgendwelche Daten, die zum Zeitpunkt der Anfrage noch nicht verfügbar sind, anfordern und Sie müssen warten, bis sie bereit sind. Das ausführliche Beispiel der Nutzung der Funktion Sleep () können Sie im Abschnitt Datenzugriff organisieren finden.

Im Tester halten die Aufrufe Sleep () den Prozess des Testens nicht auf. Beim Aufruf von Sleep() werden die generierten Ticks innerhalb der angegebenen Verzögerung "gespielt", infolgedessen können offene Aufträge, Stops usw auslösen. Nach dem Aufruf Sleep () verlängert sich die im Tester simulierte Zeit in einem Intervall, das im Parameter der Funktion Sleep angegeben ist.

Wenn als Ergebnis der Ausführung der Funktion Sleep () die aktuelle Zeit im Tester über den Zeitraum des Testens hinausgeht, dann erhalten Sie eine Fehlermeldung ' Endlosschleife in Sleep '. Wenn Sie diese Fehlermeldung erhalten, die Ergebnisse des Testens nicht abgelehnt werden. Alle Berechnungen werden in ihrem vollen Umfang (die Anzahl der Deals, Drawdown, etc.) durchgeführt und die Ergebnisse dieses Tests werden auf dem Terminal weitergegeben.

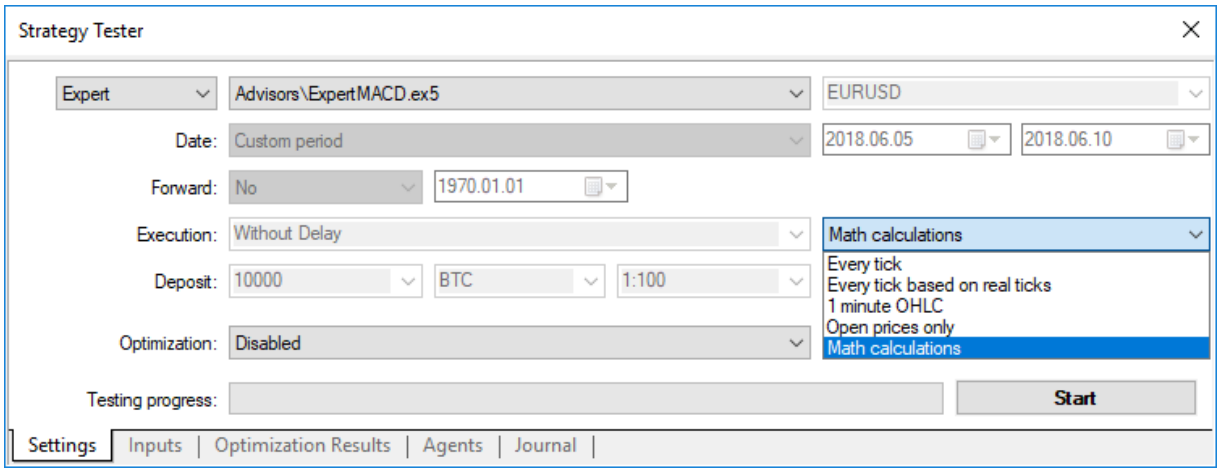
Die Funktion Sleep () wird in OnDeinit () nicht funktionieren, da nach ihrem Aufruf die Zeit des Testens das Intervall des Testens garantiert übertreffen wird .



Verwenden des Testers für Optimierungsprobleme in mathematischen Berechnungen

© 2000-2025, MetaQuotes Ltd.

Der Tester im MetaTrader 5-Terminal kann nicht nur für das Testen der Handelsstrategien, sondern auch für mathematische Berechnungen verwendet werden. Dafür muss man in den Einstellungen den entsprechenden Modus wählen:



Wenn Sie den Modus "Mathematische Berechnungen" wählen, wird der Test-Agent "leer" laufen. Der leerer Lauf bedeutet, dass Generierung der Ticks und Herunterladen der Geschichte nicht durchgeführt werden. Bei solchem Lauf werden nur drei Funktionen aufgerufen: OnInit(), OnTester(), OnDeinit().

Wenn das Enddatum des Testens kleiner oder gleich dem Anfangsdatum des Testens ist, so bedeutet es, dass die Testens im Modus "Matematische Berechnungen" durchgeführt werden.

Wenn der Tester für Lösung der mathematischen Aufgaben verwendet wird, wird die Geschichte nicht heruntergeladen und die Ticks werden nicht generiert.

Eine typische mathematische Aufgabe für die Lösung im Tester MetaTrader 5 - die Suche nach einem Extremum einer Funktion mit vielen Variablen. Um sie zu lösen, müssen Sie folgendes tun:

- Legen Sie einen Berechnungsblock der Funktionswerten von mehreren Variablen in OnTester() und geben Sie den berechneten Wert durch return(Wert_der Funktion) zurück;
- Übertragen Sie die Parameter der Funktion auf die globalen Reichweite des Programms als input-Variablen;

Kompilieren wir den Expert Advisor, öffnen wir das Fenster "Tester". Auf der Registerkarte "Eingabeparameter" wählen wir die erforderlichen Input Variablen und definieren wir für sie die Reihe von Parameterwerten und den Schritt für das Durchprobieren.

Wählen wir den Typ der Optimierung - "Langsam(vollständige Suche der Parameter)" oder - "Schnell (genetischer Algorithmus)". Für eine einfache Suche des Extremums der Funktion ist es besser die schnelle Optimierung zu wählen, aber wenn man die Werte für die ganze Reihe von Variablen ausrechnen muss, dann ist es am besten die langsame Optimierung zu verwenden.

Wählen wir den Modus "Mathematische Berechnungen" und klicken wir auf "Start" um den Optimierungsprozess zu starten. Beachten Sie, dass bei der Optimierung immer das lokale Maximum des Wertes von der Funktion OnTester gesucht wird. Für die Suche lokalen Minimums kann man das Inverse des ausgerechneten Wertes der Funktion aus der Funktion OnTester zurückgeben:

```
return(1/Wert_der Funktion);
```

© 2000-2025, MetaQuotes Ltd.

Dabei muss man selbständig überprüfen, dass der Wert_der Funktion null nicht gleich ist, sonst man kritischen Fehler der Division durch Null erhalten kann. Es gibt noch eine andere Option, die bequemer ist und verzerrt die Ergebnisse der Optimierung nicht, sie wurde von den Lesern des Artikels vorgeschlagen:

```
return(-Wert_der Funktion);
```

In dieser Option muss man der Wert_der Funktion auf Gleichheit mit Null nicht überprüfen und die Oberfläche der Optimierungsergebnisse in 3D-Darstellung selbst hat die gleiche Form, die nur von der ersten Option gespiegelt ist.

Verwenden wir die Funktion sink() als Beispiel:

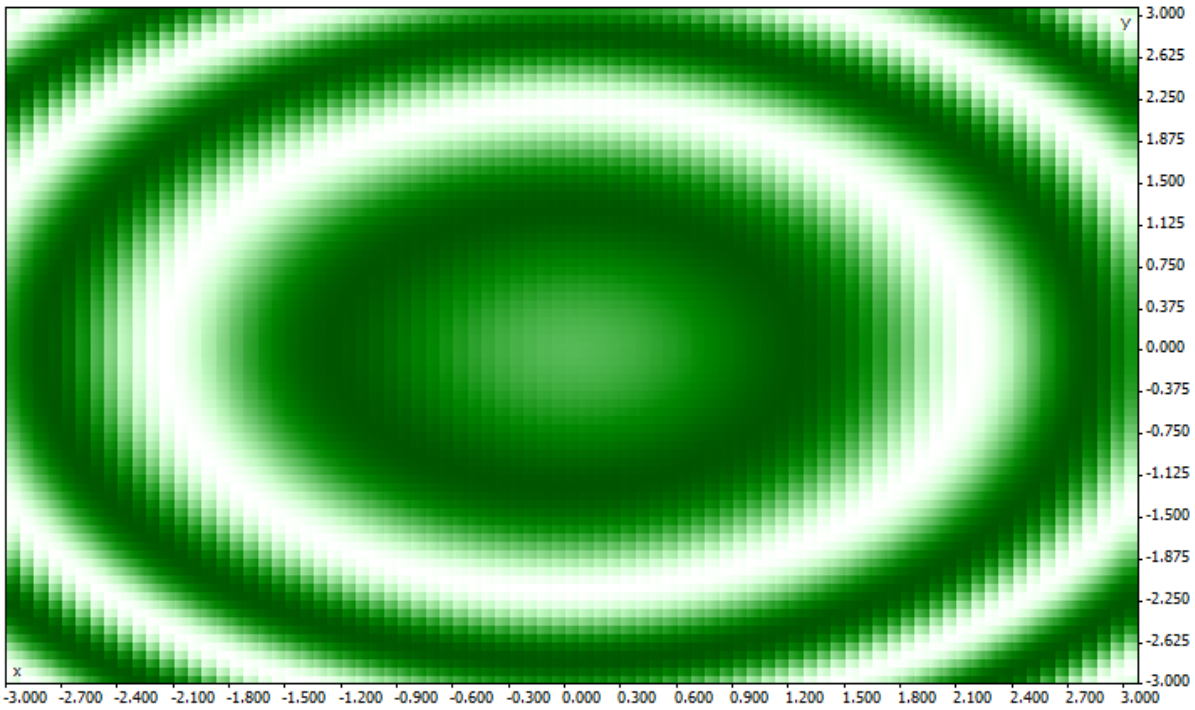
$$sink(x,y) = \sin(x^2 + y^2)$$

Stellen wir den Code des Expert Advisors für die Suche des Extremums dieser Funktion in OnTester():

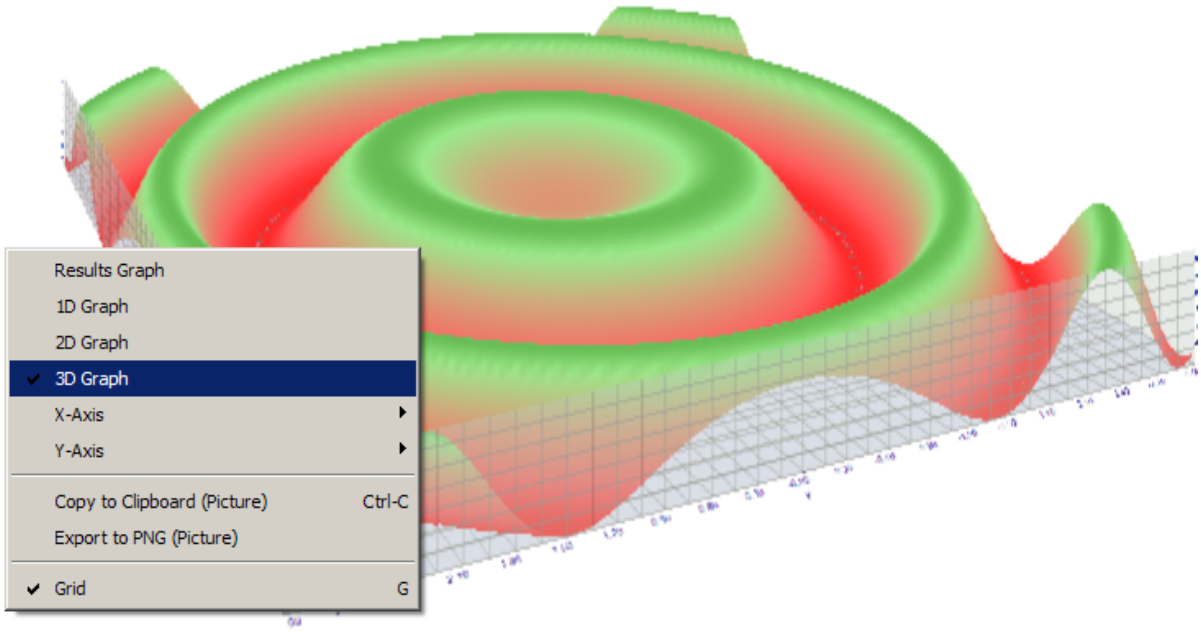
```
//+------------------------------------------------------------------+
//|                                                         Sink.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- input parameters
input double   x=-3.0; // start=-3, step=0.05, stop=3
input double   y=-3.0; // start=-3, step=0.05, stop=3
//+------------------------------------------------------------------+
//| Tester function                                                  |
//+------------------------------------------------------------------+
double OnTester()
  {
//---
   double sink=MathSin(x*x+y*y);
//---
   return(sink);
  }
//+------------------------------------------------------------------+
```

Durchführen wir eine Optimierung und stellen wir die Ergebnisse der Optimierung in Form von 2D Chart vor.

© 2000-2025, MetaQuotes Ltd.



Je besser der Wert für ein angegebenes Paar von Parametern (x, y) ist, desto gesättigter ist die Farbe. Wie es aus der Sicht der Formel der Funktion sink() erwartet wurde, deren Wert konzentrische Kreise mit Mittelpunkt bei (0,0) bildet. Für die Funktion sink () existiert kein absolutes Extremum. Es ist bei der Anzeige der Ergebnisse der Optimierung im 3D-Modus gut sichtbar:



Synchronization von Balken beim Testen in "Open Prices Only"-Modus

Der Tester im MetaTrader 5 Client-Terminal ermöglicht es Ihnen so genannte "Multi-Currency" Expert Advisors zu überprüfen. Ein Multi-Currency-Expert Advisor ist ein Expert Advisor, der auf zwei oder mehr Symbole handelt.

© 2000-2025, MetaQuotes Ltd.

Das Testen der Strategien, die auf einigen Instrumenten handeln, legt einige zusätzliche technische Anforderungen an den Tester:

- Die Generierung der Ticks für diese Instrumenten;
- Die Berechnung der Indikatorwerte für diese Instrumenten;
- Berechnung der Margin-Anforderungen für diese Instrumenten;
- Die Synchronization der generierten Ticksfolgen für alle Handelsinstrumenten.

Der Tester generiert und spielt eine Tick-Sequenz für jedes Instrument in Übereinstimmung mit dem ausgewählten Modus des Handels. Dabei wird eine neue Barfür jedes Instrument geöffnet, unabhängig davon, wie die Bar auf anderem Instrument geöffnet wurde. Dies bedeutet, dass beim Testen des Multi-Currency-Expert Advisors die Situation (meistens ist es der Fall)auftreten kann, wenn eine neue Bar für ein Instrument bereits geöffnet wurde und noch nicht für die anderen. So passiert alles beim Testen genauso wie im wirklichen Leben.

Solche authentische Simulation der Geschichte im Tester verursacht keine Probleme, solange die Modi "Jeder Tick" und "1 Minute OHLC" des Testens verwendet werden. Bei diesen Modi wird eine ausreichende Anzahl von Ticks innerhalb einer Kerze generiert, um den Moment der Synchronisation der Bars von verschiedenen Symbolen erwarten zu können. Aber wie kann man die Multi-Currency-Strategien im Modus "Nur Eröffnungspreise" testen, wenn die Bars auf Handelsinstrumente zwingend synchronisiert werden muss? In diesem Modus wird der Expert Advisor nur auf einen Tick aufgerufen, der der Öffnungszeit der Bar entspricht.

Wir erklären nun am Beispiel: Wenn wir den Expert Advisor auf das Symbol EURUSD testen, und auf EURUSD wurde eine neue stündliche Kerze geöffnet, dann erfahren wir leicht diese Tatsache - beim Testen im Modus "Nur Eröffnungspreise" entspricht das Ereignis NewTick dem Moment der Eröffnung der Bar auf den Zeitraum des Testens. Aber dabei gibt es keine Garantie, dass die neue Kerze nach dem Symbol GBPUSD geöffnet wurde, das im Expert Advisor verwendet wird.

Unter gewöhnlichen Umständen ist es ausreichend genug die Arbeit der Funktion OnTick() zu beenden und das Erscheinen einer neuen Bar auf USDJPY auf dem nächsten Tick zu überprüfen. Aber beim Testen im Modus "Nur Eröffnungspreise" wird es kein anderer Tick sein, und so mag es scheinen, dass dieses Modus für das Testen des Currency-Expert Advisors nicht passt. Aber das ist nicht so - vergessen Sie nicht, dass sich der Tester in MetaTrader 5 genauso wie im wirklichen Leben verhält. Sie können warten, bis eine neue Bar auf einem anderen Symbole mittels Funktion Sleep() geöffnet wird!

Der Code des Expert Advisors Synchronize_Bars_Use_Sleep.mq5, der ein Beispiel der Synchronization der Bars beim Testen im Modus "Nur Eröffnungspreise" zeigt.

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                  Synchronize_Bars_Use_Sleep.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- input parameters
input string   other_symbol="USDJPY";
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- nachprüfen wir das aktuelle Symbol
   if(_Symbol==other_symbol)
     {
      PrintFormat("Man muss ein anderes Symbol angeben oder starten Sie das Testen auf
      //--- stoppen wir das Testen des Expert Advisors zwangsmäßig
      return(INIT_PARAMETERS_INCORRECT);
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- statistische Variable für die Speicherung der Zeit der Öffnung der letzten Bar
   static datetime last_bar_time=0;
//---  das Merkmal der Synchronisation der Öffnungszeit der letzten Bar auf verschiede
   static bool synchonized=false;
//--- wenn statische Variable ist nicht initialisiert
   if(last_bar_time==0)
     {
      //--- es ist ein erster Aufruf, speichern wir die Öffnungszeit und verlassen ihr
      last_bar_time=(datetime)SeriesInfoInteger(_Symbol,Period(),SERIES_LASTBAR_DATE);
      PrintFormat("Variable wird mit dem Wert %s initialisiert",TimeToString(last_bar_
     }
//--- erhalten wir die Öffnungszeit der letzten Bar nach unserem Symbol
   datetime curr_time=(datetime)SeriesInfoInteger(Symbol(),Period(),SERIES_LASTBAR_DAT
//--- wenn die Zeit der Öffnung der aktuellen Bar ist nicht gleich der Zeit, die gespe
   if(curr_time!=last_bar_time)
     {
      //--- speichern wir die Öffnungszeit der neuen Bar in statistischer Variable
      last_bar_time=curr_time;
      //--- nicht synchroniesiert
      synchonized=false;
      //--- geben wir die Meldung über dieses Ereignis aus
      PrintFormat("Auf dem Symbol %s wurde eine neue Bar in %s geöffnet",_Symbol,TimeT
     }
//--- hier werden wir die Öffnungszeit der Bar auf dem fremden Symbol speichern
   datetime other_time;
//--- Schleife bis die Öffnungszeit des letzten Bars nach anderem Symbol wird gleich c
   while(!(curr_time==(other_time=(datetime)SeriesInfoInteger(other_symbol,Period(),SE
     {
      PrintFormat("Warten Sie 5 Sekunden..");
      //--- warten Sie 5 Sekunden und anfordern Sie noch einmal SeriesInfoInteger(othe
      Sleep(5000);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- die Öffnungszeit der Bar ist gleich für beide Symbole
   synchonized=true;
   PrintFormat("Die Öffnungszeit der letzten Bar auf dem Symbol %s ist: %s",_Symbol,T
   PrintFormat("Die Öffnungszeit der letzten Bar auf dem Symbol %s ist: %s",other_symb
//--- TimeCurrent() passt nicht, verwenden wir TimeTradeServer() für
   Print("Die Bars wurden synchronisiert um,TimeToString(TimeTradeServer(),TIME_DATE|T
  }
//+------------------------------------------------------------------+
```

Beachten Sie die letzte Zeile im Handler, die die aktuelle Zeit anzeigt, wenn die Tatsache der Synchronisation festgestellt wurde:

```
   Print("Die Bars wurden synchronisiert um ",TimeToString(TimeTradeServer(),TIME_SEC
```

Um die aktuelle Uhrzeit anzuzeigen, verwendeten wir die Funktion TimeTradeServer(), und nicht TimeCurrent(). Es geht darum, dass die Funktion TimeCurrent () die Zeit des letzten Ticks zurückgibt, die sich nach der Verwendung von Sleep() auf keine Weise geändert hat. Starten Sie den Handler im Modus "Nur Eröffnungspreise" und Sie erhalten die Meldungen über Synchronization der Bars.

Core 1	2010.12.01 20:00:05	The bars are synchronized at 2010.12.01 20:00:05
Core 1	2010.12.01 20:00:05	Open bar time of the chart symbol USDJPY: 2010.12.01 20:00
Core 1	2010.12.01 20:00:05	A new bar has appeared on symbol EURUSD: 2010.12.01 20:00
Core 1	2010.12.01 20:00:00	Waiting 5 seconds..
Core 1	2010.12.01 20:00:05	A new bar has appeared on symbol EURUSD: 2010.12.01 20:00
Core 1	2010.12.01 16:00:05	The bars are synchronized at 2010.12.01 16:00:05

Verwenden Sie die Funktion TimeTradeServer () anstelle von TimeCurrent (), wenn Sie die aktuelle Serverzeit und nicht die Eingangszeit des letzten Ticks erhalten möchten.

Es gibt einen anderen Weg die Bars zu synchronisieren - mit Hilfe eines Timers. Ein Beispiel solchen Expert Advisors ist Synchronize_Bars_Use_OnTimer.mq5, der dem Artikel beigefügt ist.

IndicatorRelease()-Funktion im Tester

Nach dem Abschluss des einzelnen Testens wird ein Chart des Instrumentes automatisch geöffnet, auf dem die abgeschlossenen Deals und die Indikatoren angezeigt werden, die in Expert Advisor verwendet wurden. Das hilft die Momente des Eingangs und des Ausgangs optisch zu überprüfen und sie mit den Werten der Indikatoren zu vergleichen.

> **Hinweis:** Die Indikatoren, die auf dem automatisch geöffneten Chart nach der Durchführung des Testens angezeigt werden, werden nach Beendigung des Testens neu berechnet. Auch wenn diese Indikatoren wurden in dem getesteten Expert-Advisor verwendet.

Aber in einigen Fällen benötigt der Programmierer die Informationen über die im Handelsalgorithmus mitwirkenden Indikatoren auszublenden. Zum Beispiel, wird der Code des Experten vermietet oder wird als Ausführungsdatei ohne Quellcode verkauft. Für diese Ziele ist die Funktion IndicatorRelease() geeignet.

Wenn im Terminal die Schablone mit dem Namen tester.tpl im Verzeichnis /profiles/templates des Client-Terminals angegeben wird, dann wird sie zum geöffneten Chart verwendet werden. In ihrer Abwesenheit wird die Standard-Schablone angewendet (default.tpl).

© 2000-2025, MetaQuotes Ltd.

Die Funktion IndicatorRelease () wird für die Befreiung von Berechnungen des Indikators verwendet, wenn es nicht mehr nötig ist. So können Sie sowohl Speicherplatz als auch Prozessor-Ressourcen sparen, denn jeder Tick verursacht die Berechnung des Indikators. Ihr zweiter Zweck ist das Verbot der Anzeige des Indikators auf dem Chart des Testens nach dem einzelnen Lauf.

Um die Anzeige des Indikators auf dem Chart am Ende des Testens zu verbieten, rufen Sie IndicatorRelease () mit dem Handle des Indikators im Handler OnDeinit (). Die Funktion OnDeinit() wird immer nach dem Abschluss oder vor der Anzeige des Charts des Testens aufgerufen.

```
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   bool hidden=IndicatorRelease(handle_ind);
   if(hidden) Print("IndicatorRelease() erfolgreich abgeschlossen");
   else Print("IndicatorRelease() gab false zurück. Fehlercode ",GetLastError());
 }
```

Um die Anzeige des Indikators auf dem Chart nach dem Abschluss des einzelnen Testens zu verbieten, verwenden Sie die Funktion IndicatorRelease() im Handler OnDeinit().

Behandlung von Ereignissen im Tester

Das Vorhandensein des Handlers OnTick() im Expert Advisor ist nicht obligatorisch, um ihn der Überprüfung auf die historischen Daten im Tester des MetaTrader 5 Terminals zu unterwerfen. Es ist ausreichend für den Expert Advisor wenigstens einen der folgenden Funktion-Handlers zu haben:

- OnTick() - Event-Handler des Erhaltes neuen Ticks;
- OnTrade() - Handler des Handelseregnisses;
- OnTimer() - Event-Hanlder der Ankunft des Signals aus dem Timer;
- OnChartEvent() - Handler der Benutzerereignisse.

Beim Testen kann man die Benutzerereignisse mit Hilfe der Funktion OnChartEvent() im Expert Advisor bearbeiten, aber in den Indikatoren wird diese Funktion im Tester nicht aufgerufen. Auch wenn der Indikator den Handler OnChartEvent() hat und dieser Indikator wird in dem getesteten Expert Advisor verwendet, wird der Indikator keine Benutzerereignisse bekommen.

Der Indikator kann beim Testen die Benutzerereignisse mittels Funktion EventChartCustom() generieren, und der Expert Advisor kann dieses Ereignis in OnChartEvent() bearbeiten.

Zusätzlich zu den obengenannten Ereignissen werden im Strategie-Tester die speziellen Ereignisse generiert, die mit dem Prozess des Testens und der Optimierung verbunden sind:

- Tester - dieses Ereignis wird nach dem Testen des Expert Advisors auf historische Daten generiert. Das Ereigniss Tester wird durch die Funktion OnTester() bearbeitet. Diese Funktion kann nur in Experten beim Testen verwendet werden und dient Zuallererst für die Berechnung eines Wertes, der als Kriterium Custom max bei genetischer Optimierung der Eingabeparameter verwendet wird.
- TesterInit - dieses Ereignis wird beim Start der Optimierung im Strategie-Tester vor dem ersten Durchlauf generiert. Bearbeitung des Ereignisses TesterInit wird durch die Funktion OnTesterInit() durchgeführt. Der Expert Advisor, der diesen Handler hat, wird beim Start der Optimierung automatisch auf einem gesonderten Chart des Terminals geladen mit dem Symbol und der Periode,

© 2000-2025, MetaQuotes Ltd.

die im Tester angegeben wurden, und ein Ereignis TesterInit empfängt. Die Funktion soll einen Expert Advisor vor Optimierung initialisieren um dann die Ergebnisse der Optimierung zu bearbeiten.

- TesterPass - dieses Ereignis wird beim Empfang eines neuen Datenframes generiert. Bearbeitung des Ereignisses TesterInit wird durch die Funktion OnTesterPass() durchgeführt. Der Expert Advisor mit diesen Handler wird automatisch auf einem gesonderten Chart des Terminals geladen mit dem für ein Test angegebenen Symbol und der Periode, und erhält Ereignisse TesterPass beim Erhalten von einem Frame bei der Optimierung. Die Funktion wird für die dynamische Verarbeitung von Optimierungsergebnisse direkt "on the fly" ausgelegt, ohne für die Vollendung der Optimierung zu warten. Frames werden durch Funktion FrameAdd() hinzugefügt. Diese Funktion kann nach einem Durchgang im Handler OnTester() aufgerufen werden.
- TesterDeinit - dieses Ereignis wird nach der Optimierung eines Expert Advisors generiert. Bearbeitung des Ereignisses TesterDeinit wird durch die Funktion OnTesterDeinit() durchgeführt. Der Expert Advisor mit diesem Handler wird automatisch auf dem Chart beim Start der Optimierung geladen, und erhält ein Ereignis TesterDeinit nach ihrer Fertigstellung. Die Funktion wird für die abschließende Bearbeitung aller Optimierungsergebnisse verwendet.

Testagenten

Das Testen wird mit Hilfe der Testagenten im MetaTrader 5 Client-Terminal durchgeführt. Lokalen Agenten werden automatisch erstellt und aktiviert. Die Anzahl der lokalen Agenten entspricht standardmäßig der Anzahl der Kerne auf dem Computer.

Jeder Testagent hat seine eigene Kopie von globalen Variablen, die mit dem Client-Terminal nicht verbunden ist. Das Terminal ist ein Dispatcher, der die Aufgaben an die lokalen und Remote-Agenten verteilt. Wenn Sie die nächste Aufgabe des Testens des Expert Advisors mit den angegebenen Parametern ausgeführt haben, gibt der Agent dem Terminal die Ergebnisse zurück. Beim einzelnen Testen wird nur einen Agenten verwendet.

Der Agent speichert die vom Terminal erhaltenen Geschichte in den einzelnen Ordnern mit dem Namen des Instrumentes, das heisst die Geschichte für EURUSD in einem gesonderten Ordner mit dem Namen EURUSD aufbewahrt wird. Ausserdem wird die Geschichte des Instrumentes durch die Quellen geteilt. Die Struktur für die Speicherung der Geschichte ist wie folgt:

```
Verzeichnis_des Testers\Agent-IPaddress-Port\bases\Name_der Quelle\history\Name_des Ir
```

Zum Beispiel kann die Geschichte der EURUSD von MetaQuotes-Demo-Server in einem Ordner Verzeichnis_des Testers \\ Agent-127.0.0.1-3000 \\ bases \\ MetaQuotes-Demo \\ EURUSD gespeichert werden.

Der lokale Agent befindet sich nach Beendigung des Testens im Bereitschaftsmodus und wartet auf die nächste Aufgabe im Laufe von 5 Minuten, um die Zeit mit dem Start bei den nächsten Aufrufen nicht zu verschwenden. Nur stellt nach Ablauf der Wartezeit der lokale Agent seine Arbeit ein und wird aus dem Speicher des Computers entladet.

Bei der vorzeitigen Beendigung des Testens seitens des Benutzers (Schaltfläche "Abbrechnen"), und auch bei der Schließung des Client-Terminals stellen alle lokalen Agenten ihre Arbeit sofort ein und werden aus dem Speicher entladet.

Der Datenaustausch zwischen dem Terminal und dem Agenten

© 2000-2025, MetaQuotes Ltd.

Beim Start des Testens bereitet das Terminal einige Parameter-Blöcke für das Senden an den Agenten vor:

- Die Eingabeparameter des Testens (Simulationsmodus, das Intervall des Testens, Instrumente, Optimierungskriterium usw)
- Liste der im "Markt Durchsicht" ausgewählten Instrumente
- Die Spezifikation des getesteten Instrumentes (die Kontraktgröße, die zulässigen Margen vom Markt für die Einstellung von StopLoss und Takeprofit, usw)
- Der getestete Expert-Advisor und die Werte seiner Eingabeparameter
- Informationen über weitere Dateien (Bibliotheken, Indikatoren, Daten-Dateien - # Property tester_ ...)

tester_indicator	string	Name des Benutzerindikators im Format "Name_des Indikators.ex5." Die für Testen erforderliche Indikatoren werden automatisch aus Aufruf der Funktionen iCustom() bestimmt, wenn der entsprechende Parameter von der Konstantzeile vorgegeben ist. Für andere Fälle (Verwendung der Funktion IndicatorCreate() oder Verwendung einer nicht Konstantzeile im Parameter, der den Namen des Indikator vorgibt) ist diese Eigenschaft erforderlich
tester_file	string	Dateiname für Tester mit Angabe der Erweiterung, in Doppelanführungszeichen (als Konstantzeile). Die angegebene Datei wird dem Tester für Arbeit übertragen. Eingabeparameter für Testen müssen immer angegeben werden (wenn sie erforderlich sind).
tester_library	string	Name der Bibliothek mit Erweiterung in Doppelanführungszeichen. Bibliothek kann sowohl mit der Erweiterung dll, als auch mit der Erweiterung ex5 sein kann. Die für Testen erforderliche Bibliotheken werden automatisch bestimmt. Aber wenn eine Bibliothek vom Benutzerindikator verwendet wird, muss man diese Eigenschaft verwenden.

Für jedes Parameter-Block wird ein digitaler Fingerabdruck in Form von MD5-Hash erstellt, der an den Agenten gesendet wird. MD5-Hash ist einzigartig für jeden Satz, sein Volumen ist viel kleiner als die Informationsmenge, auf deren Grundlage er berechnet wurde.

Der Agent erhält Hashes von Blöcken und vergleicht sie mit denen, die er bereits hat. Wenn der Fingerabdruck dieses Parameter-Blocks dem Agenten fehlt, oder der empfangene Hash unterscheidet sich von der bestehenden, anfordert der Agent selbst das Parameter-Block. So wird der Verkehr zwischen dem Terminal und dem Agenten reduziert.

Nach der Durchführung des Testens gibt der Agent alle Ergebnisse des Laufs zurück. Sie werden auf der Registerkarten "Testergebnisse" und "Optimierungsergebnisse" angezeigt: Der Gewinn, die Anzahl der Deals, die Sharpe Ratio, das Ergebnis der Funktion OnTester (), etc.

Bei der die Optimierung verteilt das Terminals die Aufgaben für die Durchführung des Testens in kleine Pakete an die Agenten, jedes Paket enthält mehrere Aufgaben (jede Aufgabe bedeutet einzelnes Testen mit einem Satz von Eingabeparametern). Dies reduziert die Zeit des Austausches zwischen dem Terminal und dem Agenten.

© 2000-2025, MetaQuotes Ltd.

Agenten werden die vom Terminal erhaltenen EX5-Dateien (Expert-Advisor, Indikatoren, Bibliotheken, etc.) aus Gründen der Sicherheit nie auf Ihrer Festplatte speichern, damit man die gesendeten Daten auf einem Computer mit laufendem Agenten nicht benutzen konnte. Alle übrigen Dateien, einschließlich DLL, werden in der "Sandbox" gespeichert. In der Remote-Agenten kann man die Experten unter Verwendung der DLL nicht testen.

Die Testergebnisse werden in einem speziellen Cache der Ergebnisse (der Ergebniscache) für den nachfolgenden schnellen Zugriff darauf sie, falls notwendig, aufbewahrt. Für jeden Satz von Parametern sucht das Terminals im Ergebniscache nach bereits vorliegenden Ergebnissen von den vorherigen Starts um den Neustart zu vermeiden. Wenn das Ergebnis mit einem solchen Satz von Parametern nicht gefunden wird, wird die Aufgaben für die Durchführung des Testens dem Agenten gegeben.

Der gesamte Verkehr zwischen dem Terminal und dem Agenten wird verschlüsselt.

> Ticks werden nicht über das Netzwerk gesendet, werden sie auf Tester-Agenten generiert.

Verwendung von einen freigegebenen Ordner von allen Client-Terminals

Alle Testagenten sind voneinander und von der Client-Terminal isoliert: jeder Agent hat einen eigenen Ordner, in dem seine Protokolle geschrieben werden. Ausserdem werden alle Dateioperationen beim Testen des Agenten im Ordner **Name_des Agenten/MQL5/Files** durchgeführt. Man kann jedoch die Wechselwirkung zwischen den lokalen Agenten und dem Client-Terminal über einen freigegebenen Ordner aller Client-Terminals zu verwirklichen, wenn Sie beim Öffnen der Datei das Flag FILE_COMMON angeben:

```
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- der freigegebene Ordner von allen Client-Terminals
   common_folder=TerminalInfoString(TERMINAL_COMMONDATA_PATH);
//--- geben wir den Namen dieses Ordners aus
PrintFormat("Öffnen wir die Datei in dem freigegebenen Ordner der Client-Terminals% s"
//--- öffnen wir eine Datei in dem freigegebenen Ordner (das Flag FILE_COMMON wird anc
   handle=FileOpen(filename,FILE_WRITE|FILE_READ|FILE_COMMON);
  ... weitere Aktionen
//---
   return(INIT_SUCCEEDED);
  }
```

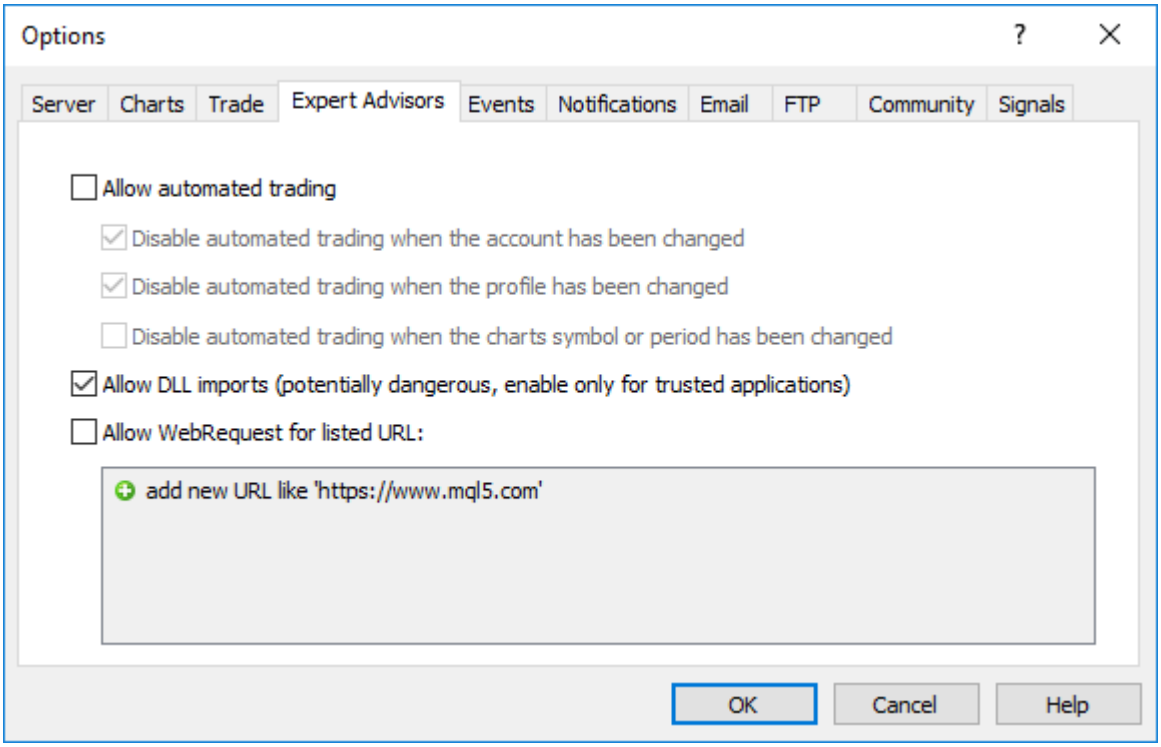
Verwendung von DLL

Zur Beschleunigung der Optimierung kann man nicht nur lokalen sondern auch Remote-Agenten verwenden. Dabei gibt es einige Beschränkungen für Remote-Agenten . Erstens können die Remote-Agenten in ihren Protokollen die Ergebnisse der Ausführung der Funktion Print(), Nachrichten über das Öffnen und Schließen von Positionen nicht ausgeben. Im Protokoll werden Mindestinformationen

© 2000-2025, MetaQuotes Ltd.

ausgegeben, so dass die falsch geschriebenen Experten mit Nachrichten die Festplatte Ihres Computers nicht überlasten können.

Die zweite Einschränkung - das Verbot der Verwendung der DLL beim Testen von Experten. DLL-Aufrufe sind auf Remote-Agenten aus Sicherheitsgründen absolut verboten. Auf der lokalen Agenten können DLL-Aufrufe nur mit der entsprechenden Erlaubnis "DLL-Import erlauben" in getesteten Experten erlaubt werden.



Hinweis: Bei der Verwendung der von Expert Advisors erhaltenen (Skripts, Indikatoren), die die DLL-Aufrufe zu erlauben erfordern, sollten Sie sich der Risiken bewusst sein, die Sie übernehmen falls Sie diese Option in den Einstellungen des Terminals erlauben. Unabhängig davon, wie der Expert-Advisor verwendet wird - für das Testen oder für den Lauf auf dem Chart.

© 2000-2025, MetaQuotes Ltd.

Vordefinierte Variablen

für jedes ausführbare mql5-Programm wird eine Reihe von vorbestimmten Variablen unterstützt, die den Zustand des laufenden Preischarts wiederspiegeln, wenn das Programm startet - Expert, Script oder Benutzeranzeiger.

Werte der vorbestimmten Variablen werden vom Client-Terminal vor dem Start des mql5-Programms eingestellt. Vorbestimmte Variablen sind konstant und können nicht aus dem mql5-Programm verändert werden. Die Ausnahme ist die Variable _LastError, die durch die Funktion ResetLastError ausgenullt werden kann.

Variable	Wert
_AppliedTo	Die Variable _AppliedTo erlaubt es, den Typ der Daten festzustellen, anhand welcher der Indikator berechnet wird
_Digits	Anzahl von Dezimalzeichen nach dem Komma
_Point	Punktgroesse des laufenden Symbols in der Quotation Währung
_LastError	Kode des letzten Fehlers
_Period	Timeframe des laufenden Charts
_RandomSeed	Der aktuelle Zustand des Generators von Pseudo-Zufallszahlen
_StopFlag	Programmstop Flagge
_Symbol	Symbolname des laufenden Charts
_UninitReason	Kode des Grundes der Programmdeinitialisierung
_IsX64	Die Variable _IsX64 erlaubt es festzustellen, in welchem Terminal ein MQL5-Programm läuft

Bibliotheken verwenden die Variablen der Programme, die sie aufgerufen hat.

© 2000-2025, MetaQuotes Ltd.

int _AppliedTo

Die Variable _AppliedTo erlaubt es, den Typ der Daten festzustellen, anhand welcher der Indikator berechnet wird:

Datentyp	Wert	Beschreibung der Daten, die für die Berechnung des Indikators verwendet werden
—	0	Der Indikator verwendet die zweite Variante des Aufrufs von OnCalculate() - die Daten für die Berechnung werden nicht durch einen bestimmten Puffer oder Datenarray angegeben
Close	1	Close-Preise
Open	2	Open-Preise
High	3	High-Preise
Low	4	Low-Preise
Median Price (HL/2)	5	Mittlerer Preis = (High+Low)/2
Typical Price (HLC/3)	6	Typischer Preis = (High+Low+Close)/3
Weighted Price (HLCC/4)	7	Gewichteter Preis = (Open+High+Low+Close)/4
Previous Indicator's Data	8	Daten des Indikators, der vor diesem Indikator im Chart gestartet wurde
First Indicator's Data	9	Daten des Indikators, der als Erster im Chart gestartet wurde
Indicator handle	10+	Daten des Indikators, der der Funktion iCustom() mithilfe des Handles des Indikators übergeben wurde. Der Wert von _AppliedTo beinhaltet den Handle des Indikators

Beispiel:

```
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
 {
//--- indicator buffers mapping
   SetIndexBuffer(0,Label1Buffer,INDICATOR_DATA);
// Typ der Daten abfragen, anhand welcher der Indikator berechnet wird
   Print("_AppliedTo=",_AppliedTo);
   Print(getIndicatorDataDescription(_AppliedTo));
//---
   return(INIT_SUCCEEDED);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Beschreibung der Daten, auf welchen der Indikator berechnet wird |
//+------------------------------------------------------------------+
string getIndicatorDataDescription(int data_id)
 {
   string descr="";
   switch(data_id)
    {
      case(0):descr="It's first type of OnCalculate() - no data buffer";
         break;
      case(1):descr="Indicator calculates on Close price";
         break;
      case(2):descr="Indicator calculates on Open price";
         break;
      case(3):descr="Indicator calculates on High price";
         break;
      case(4):descr="Indicator calculates on Low price";
         break;
      case(5):descr="Indicator calculates on Median Price (HL/2)";
         break;
      case(6):descr="Indicator calculates on Typical Price (HLC/3)";
         break;
      case(7):descr="Indicator calculates on Weighted Price (HLCC/4)";
         break;
      case(8):descr="Indicator calculates Previous Indicator's data";
         break;
      case(9):descr="Indicator calculates on First Indicator's data";
         break;
      default: descr="Indicator calculates on data of indicator with handle="+string(d
         break;
     }
//---
   return descr;
  }
```

Siehe auch

ENUM_APPLIED_PRICE

© 2000-2025, MetaQuotes Ltd.

int _Digits

Die Variable _Digits bewahrt die Anzahl von Ziffern nach Dezimalpunkt auf, die die Preisgenauigkeit des Symbols des laufenden Charts definiert.

Man kann auch die Funktion Digits() verwenden.

© 2000-2025, MetaQuotes Ltd.

double _Point

Die Variable _Point bewahrt die Punktgroesse des laufenden Symbols in Quotation Waehrung auf.

Man kann auch die Funktion Point() verwenden.

© 2000-2025, MetaQuotes Ltd.

int _LastError

Die Variable _LastError bewahrt den Kode des letzten Fehlers auf, der während der Ausführung vom mql5-Programm entstand. Durch die Funktion ResetLastError() kann der Wert auf Null gestellt werden.

Um Fehlerkode zu bekommen kann auch die Funktion GetLastError() verwendet werden.

© 2000-2025, MetaQuotes Ltd.

ENUM_TIMEFRAMES _Period

Die Variable _Period bewahrt Timeframe des laufenden Charts auf.

Man kann auch die Funktion Period() verwenden.

Sehen Sie auch

PeriodSeconds, Chartperioden, Datum und Zeit, Objektsichtbarkeit

© 2000-2025, MetaQuotes Ltd.

_RandomSeed

Eine Variable für Speicherung des aktuellen Zustandes des Generators von Pseudo-Zufallszahlen _RandomSeed ändert seinen Wert beim Aufruf von MathRand(). Um den richtigen Anfangszustand zu setzen verwenden Sie MathSrand().

Die Zufallszahl ***x***, die durch die Funktion MathRand() erhalten wird, wird bei jedem Aufruf wie folgt berechnet:

```
x=_RandomSeed*214013+2531011;
_RandomSeed=x;
x=(x>>16)&0x7FFF;
```

Sehen Sie auch

MathRand(), MathSrand(), Integer-Typen

© 2000-2025, MetaQuotes Ltd.

int _StopFlag

Die Variable _StopFlag speichert das Flag zum Anhalten des mql5-Programms. Unter normalen Bedingungen ist _StopFlag 0. Wenn das Client-Terminal versucht, das Programm anzuhalten, wird es auf einen Wert ungleich Null gesetzt.

Für Prüfung des Status der Flagge _StopFlag kann auch die Funktion IsStopped() verwendet werden.

© 2000-2025, MetaQuotes Ltd.

string _Symbol

In der Variable _Symbol wird der Symbolname des laufenden Charts aufbewahren.

Man kann auch die Funktion Symbol() verwenden.

© 2000-2025, MetaQuotes Ltd.

int _UninitReason

In der Variable _UninitReason wird der Kode des Deinitialisierungsgrundes des Programms aufbewahren.

Gewöhnlich erhält man den Kode des Initialisierungsgrundes des Programms durch die Funktion UninitializeReason().

© 2000-2025, MetaQuotes Ltd.

int _IsX64

Die Variable _IsX64 erlaubt es festzustellen, in welchem Terminal ein MQL5-Programm läuft: _IsX64=0 für das 32-Bit-Terminal und _IsX64!=0 für das 64-Bit-Terminal.

Man kann auch die Funktion TerminalInfoInteger(TERMINAL_X64) verwenden.

Beispiel:

```
// prüfen, in welchem Terminal das Programm läuft
   Print("_IsX64=",_IsX64);
   if(_IsX64)
      Print("Das Programm ",__FILE__," läuft im 64-Bit-Terminal");
   else
      Print("Das Programm ",__FILE__," läuft im 32-Bit-Terminal");
   Print("TerminalInfoInteger(TERMINAL_X64)=",TerminalInfoInteger(TERMINAL_X64));
```

Siehe auch

MQLInfoInteger, Import der Funktionen (#import)

© 2000-2025, MetaQuotes Ltd.

Allgemeine Funktionen

Allgemeine Funktionen, die keiner Gruppe angehören.

Funktion	Aktion
Alert	Gibt Meldungen in einem separaten Fenster aus
CheckPointer	Gibt den Typ eines Objekts zurück
Comment	Gibt eine Meldung in der linken oberen Ecke des Preischarts aus
CryptEncode	Wandelt Daten eines Quell-Arrays in ein Empfänger-Array anhand der angegebenen Methode um
CryptDecode	Führt eine Rückumwandlung der Daten eines Arrays aus
DebugBreak	Programmhaltepunkt beim Debugging
ExpertRemove	Stoppt den Expert Advisor und entfernt ihn vom Chart
GetPointer	Gibt den Anzeiger des Objektes zurück.
GetTickCount	Gibt die Anzahl der Millisekunden zurück, die seit dem Start des Systems vergangen sind
GetTickCount64	Gibt die Anzahl der Millisekunden zurück, die seit dem Start des Systems vergangen sind
GetMicrosecondCount	Gibt die Anzahl der Mikrosekunden zurück, die seit dem Start eines MQL5-Programms vergangen sind
MessageBox	Erzeugt und exponiert Meldungsfenster und auch verwaltet es
PeriodSeconds	Gibt Sekundenanzahl in der Periode zurück
PlaySound	Spielt Audiodatei ab
Print	Gibt die Meldung ins Magazin aus
PrintFormat	Formatiert und druckt Schriftsatz und Werte in die Log-Datei aus entsprechend dem vorgegebenen Format
ResetLastError	Stellt den Wert der vorbestimmten Variable _LastError auf Null
ResourceCreate	Erstellt eine Ressource eines Bildes basierend auf dem Datensatz
ResourceFree	Löscht dynamisch erstellte Ressource (freit den Speicher)
ResourceReadImage	Liest Daten einer Grafik-Ressource, die von der Funktion ResourceCreate() erstellt war oder in EX5-Datei beim Kompilieren gespeichert war
ResourceSave	Speichert die Ressource in eine angegebene Datei
SetUserError	Stellt die Vorbestimmte Variable _LastError in Wert ERR_USER_ERROR_FIRST + user_error

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
SetReturnError	Setzt den Fehlercode, den ein Prozess des Terminals bei seiner Beendigung zurückgibt
Sleep	Hemmt die Ausführung des laufenden Experten oder Script für bestimmte Zeit
TerminalClose	Sendet dem Terminal Adressbefehl, Arbeit zu beenden
TesterHideIndicators	Aktiviert den Modus zum Anzeigen/Ausblenden von Indikatoren, die in einem Expert Advisor verwendet werden
TesterStatistics	Es gibt den Wert der statistischen Parameter, der nach Testergebnisse berechnet ist, zurück
TesterStop	Erteilt den Befehl die Programmausführung von Tests zu beenden.
TesterDeposit	Eine spezielle Funktion, die während eines Tests Einzahlungen emuliert
TesterWithdrawal	Sonderfunktion für Emulation der Geldabhebungen beim Testen
TranslateKey	Gibt ein Unicode-Zeichen nach dem virtuellen Tastencode
ZeroMemory	Nullt die Variable aus, die durch Referenz übertragen wird. Typ der Variable kann verschieden sein, Ausnahmen bilden nur Klassen und Strukturen mit Konstrukteuren

© 2000-2025, MetaQuotes Ltd.

Alert

Zeigt ein Dialogfenster mit Nutzerdaten.

```
void  Alert(
   argument,      // der erste Wert
   ...            // weitere Werte
   );
```

Parameter

argument

[in] die Werte, getrennt durch Kommas. Für das Trennen die Informationsausgabe in mehrere Zeilen kann das Zeilenvorschubzeichen "\n" oder "\r\n" verwendet werden. Parameterzahl darf 64 nicht überschreiten.

Rückgabewert

Keinen Rückgabewert

Hinweis

Felder können nicht in die Funktion Alert() übertragen werden. Felder müssen einzeln übergeben werden. Daten des Typs double werden mit 8 Dezimalzeichen, Daten des Typs foat mit 5 Dezimalzeichen ausgegeben. für Ausgabe der reellen Zahlen mit anderer Genauigkeit oder im anderen Format muss die Funktion DoubleToString() verwendet werden.

Daten des Typs bool werden buchstäblich als "true" oder "false" ausgegeben. Zeitwerte werden als YYYY.MM.DD HH:MI:SS ausgegeben. Für eine Ausgabe des Datums im einem anderen Format muss die Funktion TimeToString() verwendet werden. Daten des Typs color werden als Zeichenkette R,G,B Zeile oder als Name einer Farbe ausgegeben, wenn diese Farbe im Farbsystem vorhanden ist.

Im Strategie-Tester wird die Alert()-Funktion nicht ausgeführt.

Beispiel:

```
//--- Enumerationen
enum ENUM_INTERSECT_DIRECTION
  {
   INTERSECT_DIRECTION_NONE= 0,  // kein Kreuzen
   INTERSECT_DIRECTION_UP  = 1,  // aufwärts kreuzen
   INTERSECT_DIRECTION_DOWN=-1,  // abwärts kreuzen
  };

//--- Eingabeparameter
input    uint                InpPeriod = 10;            // MA Periodenlänge
input    int                 InpShift  = 0;             // MA Versatz
input    ENUM_MA_METHOD      InpMethod = MODE_SMA;      // MA Methode
input    ENUM_APPLIED_PRICE  InpPrice  = PRICE_CLOSE;   // MA verwendeter Preis
```

© 2000-2025, MetaQuotes Ltd.

```
//--- globale Variable
int      ExtMaHandle;
int      ExtMaPeriod;
double   ExtData[2];
MqlRates ExtRates[2];

//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- die Periodenlänge für die Berechnung des gleitenden Durchschnitts ist gleich der
   ExtMaPeriod=int(InpPeriod<1 ? 10 : InpPeriod);
//--- Erstellen eines Handles für den Indikator Gleitender Durchschnitt mit den angege
   ExtMaHandle=iMA(Symbol(),PERIOD_CURRENT,ExtMaPeriod,InpShift,InpMethod,InpPrice);
   ResetLastError();
   if(ExtMaHandle==INVALID_HANDLE)
     {
      PrintFormat("Failed to create iMA() handle. Error code: %d",GetLastError());
      return(INIT_FAILED);
     }

//--- Abrufen des Zeitpunkts der letzten Preisaktualisierung
   datetime tick_time=TickTime();
//--- Abrufen der Daten des gleitenden Durchschnitts und der Preisdaten der letzten be
   if(GetData(ExtMaHandle,ExtData,ExtRates) && tick_time!=0)
     {
      //--- wenn der Preis über dem MA ist
      if(ExtRates[1].close>ExtData[1])
        {
         //--- Textnachricht erstellen und eine Warnung (Alert) anzeigen
         string message=StringFormat("Bar time: %s. The price is above the moving aver
         Alert(message+" at "+TimeToString(tick_time,TIME_DATE|TIME_MINUTES|TIME_SECON
         /*
        Ergebnis:
         Alert: Bar time: 2024.02.16 18:00. The price is above the moving average at 2
         */
        }
      else
        {
         //--- wenn der Preis unter dem MA ist
         if(ExtRates[1].close<ExtData[1])
           {
            //--- Textnachricht erstellen und eine Warnung (Alert) anzeigen
            string message=StringFormat("Bar time: %s. The price is below the moving a
            Alert(message+" at "+TimeToString(tick_time,TIME_DATE|TIME_MINUTES|TIME_SE
            /*
           Ergebnis:
            Alert: Bar time: 2024.02.16 19:00. The price is below the moving average a
```

© 2000-2025, MetaQuotes Ltd.

```
            */
           }
         else
           {
            //--- Textnachricht erstellen und eine Warnung (Alert) anzeigen
            string message=StringFormat("Bar time: %s. The price and moving average ar
            Alert(message+" at "+TimeToString(tick_time,TIME_DATE|TIME_MINUTES|TIME_SE
            /*
           Ergebnis:
            Alert: Bar time: 2024.02.16 20:00. The price and moving average are equal
            */
           }
        }
     }

//--- Erfolg
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
   ResetLastError();
//--- Abrufen der Daten des gleitenden Durchschnitts und der Preisdaten der letzten be
   if(!GetData(ExtMaHandle,ExtData,ExtRates))
      return;
//--- Ermitteln der Richtung, in der der Preis den gleitenden Durchschnitt auf dem akt
   ENUM_INTERSECT_DIRECTION intersect=GetIntersectDirection(ExtData,ExtRates);

//--- Variable zum Speichern der vorherigen Nachricht
   static string message_prev="";

//--- wenn der Kurs den gleitenden Durchschnitt auf dem aktuellen Balken nach oben ge
   if(intersect==INTERSECT_DIRECTION_UP)
     {
      //--- Ermitteln der Zeit des Ticks, zu der das Kreuzen stattgefunden hat
      datetime tick_time=TickTime();
      if(tick_time==0)
         return;
      //--- Erstellen einer Textnachricht
      string message=StringFormat("Bar time: %s. The price crossed the MA from bottom
      //--- wenn die vorherige Nachricht nicht mit der aktuellen übereinstimmt, wird e
      if(message!=message_prev)
        {
         Alert(message+" at "+TimeToString(tick_time,TIME_DATE|TIME_MINUTES|TIME_SECON
         message_prev=message;
         /*
         Result:\
```

© 2000-2025, MetaQuotes Ltd.

```
         Alert: Bar time: 2024.02.16 09:00. The price crossed the MA from bottom to to
         */
        }
     }

//--- wenn der Kurs den gleitenden Durchschnitt auf dem aktuellen Balken nach unten ge
   if(intersect==INTERSECT_DIRECTION_DOWN)
     {
      //--- Ermitteln der Zeit des Ticks, zu der das Kreuzen stattgefunden hat
      datetime tick_time=TickTime();
      if(tick_time==0)
         return;
      //--- Erstellen einer Textnachricht
      string message=StringFormat("Bar time: %s. The price crossed the MA from top to
      //--- wenn die vorherige Nachricht nicht mit der aktuellen übereinstimmt, wird e
      if(message!=message_prev)
        {
         Alert(message+" at "+TimeToString(tick_time,TIME_DATE|TIME_MINUTES|TIME_SECON
         message_prev=message;
         /*
         Result:\
         Alert: Bar time: 2024.02.16 10:00. The price crossed the MA from top to botto
         */
        }
     }
  }
//+------------------------------------------------------------------+
//| Kurs- und gleitende Durchschnittsdaten in Arrays kopieren        |
//+------------------------------------------------------------------+
bool GetData(int handle,double &ma_data[],MqlRates &price_data[])
  {
   ResetLastError();
//--- Abrufen der Daten des gleitenden Durchschnitts der letzten beiden Balken
   if(CopyBuffer(handle,0,0,2,ma_data)!=2)
     {
      PrintFormat("CopyBuffer() failed. Error code: %d",GetLastError());
      return(false);
     }
//--- Abrufen der Preisdaten der letzten beiden Balken
   if(CopyRates(Symbol(),PERIOD_CURRENT,0,2,price_data)!=2)
     {
      PrintFormat("CopyRates() failed. Error code: %d",GetLastError());
      return(false);
     }

   return(true);
  }
//+------------------------------------------------------------------+
//| Rückgabe der Kreuzungsrichtung von Preisen und gl. Durchschnitt  |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
ENUM_INTERSECT_DIRECTION GetIntersectDirection(double &ma_data[],MqlRates &price_data
  {
   double ma0=ma_data[1];
   double ma1=ma_data[0];
   double close0=price_data[1].close;
   double close1=price_data[0].close;

   if(close1<=ma1 && close0>ma0)
      return(INTERSECT_DIRECTION_UP);
   else
     {
      if(close1>=ma1 && close0<ma0)
         return(INTERSECT_DIRECTION_DOWN);
      else
         return(INTERSECT_DIRECTION_NONE);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Tick-Zeit in Sekunden                               |
//+------------------------------------------------------------------+
datetime TickTime()
  {
   MqlTick tick={};

   ResetLastError();
   if(!SymbolInfoTick(Symbol(),tick))
     {
      PrintFormat("SymbolInfoTick() failed. Error code: %d",GetLastError());
      return(0);
     }

   return(tick.time);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckPointer

Gibt Typ des Anzeigers des Objektes zurück.

```
ENUM_POINTER_TYPE  CheckPointer(
   object* anyobject      // Objektanzeiger
   );
```

Parameter

anyobject

[in] Objektanzeiger.

Rückgabewert

Gibt Wert aus der Enumeration ENUM_POINTER_TYPE zurück.

Hinweis

Versuch der Anwendung eines unkorrekten Anzeiger führt zum kritischen Brechen des Programms. Darum ist es notwendig, die Funktion CheckPointer zu verwenden vor dem Gebrauch des Anzeigers. Anzeiger kann in folgeneden Faellen unkorrekt sein:

- Anzeiger ist NULL gleich;
- wenn Objekt durch die Anweisung delete entfernt wurde.

Die vorliegende Funktion kann als Pruefung des Anzeigers auf Gueltigkeit verwendet werden. Nicht-Null Wert garantiert, dass der Anzeiger für Zugang verwendet werden kann.

Um einen Zeiger schnell zu überprüfen, kann man auch den Operator "!" (Beispiel) verwenden, der ihn über einen impliziten Aufruf der Funktion CheckPointer überprüft.

Beispiel:

```
//+------------------------------------------------------------------+
//| Entfernung der Liste durch Entfernung der Elementen              |
//+------------------------------------------------------------------+
void CMyList::Destroy()
  {
//--- Dienstanzeiger für Arbeit im Zyklus
   CItem* item;
//--- gehen wir durch den Zyklus und versuchen wir, dynamische Anzeiger zu entfernen
   while(CheckPointer(m_items)!=POINTER_INVALID)
     {
      item=m_items;
      m_items=m_items.Next();
      if(CheckPointer(item)==POINTER_DYNAMIC)
        {
         Print("Dynamic object ",item.Identifier()," to be deleted");
         delete (item);
        }
      else Print("Non-dynamic object ",item.Identifier()," cannot be deleted");
     }
//---
  }
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objektanzeiger, Pruefung der Objektanzeiger , Anweisung der Objektentfernung delete

© 2000-2025, MetaQuotes Ltd.

Comment

Gibt Kommentar, definiert vom Benutzer, in den oberen Chartwinkel aus.

```
void  Comment(
   argument,      // der erste Wert
   ...            // andere Werte
   );
```

Parameter

...

[in] Jede Werte, getrennt durch Kommas. Für Trennen der Informationsausgabe in mehrere Zeichen kann das Zeilenvorschubszeichen "\n" oder "\r\n" verwendet werden. Parameterzahl kann nicht mehr als 64 sein. Die allgemeine Laenge der ausgegebenen Meldung (einschliesslich die nicht dargestellten Dienstsymbole) kann nicht mehr als 2045 sein (andere Symbole werden bei der Ausgabe entfernt werden).

Rückgabewert

Keinen Rückgabewert

Hinweis

Felder können nicht in die Funktion Comment() übertragen werden. Felder muessen elementenweise gedruckt werden.

Daten des Typs double werden mit Genauigkeit bis 16 Dezimalzeichen ausgegeben entweder im traditionellen oder im wissenschaftlichen Format - abhängig davon, welches Dateneingabeformat kompakter wird. Daten des Typs float werden mit 5 Dezimalzeichen ausgegeben. Für Ausgabe der reellen Zahlen mit anderer Genauigkeit oder im vorbestimmten Format muss die Funktion DoubleToString() verwendet werden.

Daten des Typs bool werden als Zeilen "true" oder "false" ausgegeben.Daten werden als YYYY.MM.DD HH:MI:SS ausgegeben. Für Ausgabe des Datums im anderen Format muss die Funktion TimeToString() verwendet werden. Daten des Typs color werden entweder als R,G,B Zeile oder al Farbenname, wenn es diese Farbe im Farbensatz gibt.

Im Strategie-Tester im Optimierungsmodus wird die Comment()-Funktion nicht ausgeführt.

Beispiel:

```
void OnTick()
  {
//---
   double Ask,Bid;
   int Spread;
   Ask=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
   Bid=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   Spread=SymbolInfoInteger(Symbol(),SYMBOL_SPREAD);
//--- Geben wir Werte in drei Zeilen aus
   Comment(StringFormat("Ausgabe der Preise\nAsk = %G\nBid = %G\nSpread = %d",Ask,Bid,
  }
```

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

ChartSetString, ChartGetString

© 2000-2025, MetaQuotes Ltd.

CryptEncode

Wandelt Daten eines Quell-Arrays in ein Empfänger-Array anhand der angegebenen Methode um.

```
int  CryptEncode(
   ENUM_CRYPT_METHOD   method,         // Umwandlungsmethode
   const uchar&        data[],         // Quell-Array
   const uchar&        key[],          // Schlüssel
   uchar&              result[]        // Empfänger-Array
   );
```

Parameter

method

[in] Umwandlungsmethode. Kann einer der Werte der ENUM_CRYPT_METHOD sein.

data[]

[in] Quell-Array.

key[]

[in] Schlüssel.

result[]

[out] Empfänger-Array.

Gelieferter Wert

Anzahl der Bytes im Empfänger-Array oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu bekommen, muss man die Funktion GetLastError() aufrufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| ArrayToHex                                                       |
//+------------------------------------------------------------------+
string ArrayToHex(uchar &arr[],int count=-1)
  {
   string res="";
//--- Größe überprüfen
   if(count<0 || count>ArraySize(arr))
      count=ArraySize(arr);
//--- Umwandlung in eine hexadezimale Zeile
   for(int i=0; i<count; i++)
      res+=StringFormat("%.2X",arr[i]);
//---
   return(res);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   string text="The quick brown fox jumps over the lazy dog";
   string keystr="ABCDEFG";
   uchar src[],dst[],key[];
//--- Schlüssel vorbereiten
   StringToCharArray(keystr,key);
//--- Quell-Array src[] vorbereiten
   StringToCharArray(text,src);
//--- Ausgangsdaten anzeigen
   PrintFormat("Initial data: size=%d, string='%s'",ArraySize(src),CharArrayToString(s
//--- Verschlüsselung des Arrays src[] anhand DES mit einem 56-Bit-Schlüssel key[]
   int res=CryptEncode(CRYPT_DES,src,key,dst);
//--- das Ergebnis der Verschlüsselung überprüfen
   if(res>0)
     {
      //--- verschlüsselte Daten ausgeben
      PrintFormat("Encoded data: size=%d %s",res,ArrayToHex(dst));
      //--- Entschlüsselung der Daten des Arrays dst[] anhand der Methode DES mit eine
      res=CryptDecode(CRYPT_DES,dst,key,src);
     //--- Ergebnis überprüfen
      if(res>0)
        {
         //--- entschlüsselte Daten anzeigen
         PrintFormat("Decoded data: size=%d, string='%s'",ArraySize(src),CharArrayToSt
        }
      else
         Print("Fehler in CryptDecode. Fehlercode=",GetLastError());
     }
   else
      Print("Fehler in CryptEncode. Fehlercode=",GetLastError());
  }
```

Sieh auch

Operationen mit Arrays, CryptDecode()

© 2000-2025, MetaQuotes Ltd.

CryptDecode

Führt eine Entschlüsselung der mit CryptEncode() verschlüsselten Daten eines Arrays durch.

```
int  CryptDecode(
   ENUM_CRYPT_METHOD   method,        // Umwandlungsmethode
   const uchar&        data[],        // Quell-Array
   const uchar&        key[],         // Schlüssel
   uchar&              result[]       // Empfänger-Array
   );
```

Parameter

method

[in] Umwandlungsmethode. Kann einer der Werte der ENUM_CRYPT_METHOD sein.

data[]

[in] Quell-Array.

key[]

[in] Schlüssel.

result[]

[out] Empfänger-Array.

Gelieferter Wert

Anzahl der Bytes im Empfänger-Array oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu bekommen, muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

```
input string InpKey = "ABCDEFG";  // Encryptions-Schlüssel

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   string text="The quick brown fox jumps over the lazy dog";
   uchar src[],dst[],key[];

//--- Vorbereitung des Verschlüsselungsschlüssels
   StringToCharArray(InpKey,key);
//--- Vorbereitung des Quellarrays src[]
   StringToCharArray(text,src);
//--- Darstellung der Quelldaten
   PrintFormat("Initial data: size=%d, string='%s'",ArraySize(src),CharArrayToString(s

//--- Verschlüsselung des Arrays src[] mit der DES-Methode und dem 56-Bit-Schlüssel ke
```

© 2000-2025, MetaQuotes Ltd.

```
   int res=CryptEncode(CRYPT_DES,src,key,dst);

//--- Prüfung des Verschlüsselungsergebnisses
   if(res>0)
     {
      //--- Darstellung der verschlüsselten Daten
      PrintFormat("Encoded data: size=%d %s",res,ArrayToHex(dst));
      //--- Entschlüsselung des Datenarrays dst[] mit der DES-Methode und dem 56-Bit-S
      res=CryptDecode(CRYPT_DES,dst,key,src);
      //--- Ergebnisprüfung
      if(res>0)
        {
         //--- Darstellung der entschlüsselten Daten
         PrintFormat("Decoded data: size=%d, string='%s'",ArraySize(src),CharArrayToSt
        }
      else
         Print("CryptDecode failed. Error: ",GetLastError());
     }
   else
      Print("CryptEncode failed. Error: ",GetLastError());
  }
//+------------------------------------------------------------------+
//| ArrayToHex                                                       |
//+------------------------------------------------------------------+
string ArrayToHex(uchar &arr[],int count=-1)
  {
   string res="";

//--- Feststellen der Größe
   if(count<0 || count>ArraySize(arr))
      count=ArraySize(arr);

//--- in hexadezimale Zeichenfolge umwandeln
   for(int i=0; i<count; i++)
      res+=StringFormat("%.2X",arr[i]);

   return(res);
  }
```

Sieh auch

Operationen mit Arrays, CryptEncode()

© 2000-2025, MetaQuotes Ltd.

DebugBreak

Programmhaltepunkt beim Debugging .

```
void  DebugBreak();
```

Rückgabewert

Keinen Rückgabewert.

Hinweis

Die Ausführung des mql5-Programms wird nur unterbrochen, wenn das Programm im Debug-Modus ausgeführt wird. Die Funktion kann zur Anzeige von Variablenwerten und/oder zur weiteren schrittweisen Ausführung verwendet werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Kompilieren der Datei mit F5
//--- im Debugging-Modus wird, wenn i == j, durch DebugBreak() das Programm angehalter
   for(int i=0,j=20; i<20; i++,j--)
     {
      if(i==j)
         DebugBreak();
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

ExpertRemove

Stopt die Arbeit des Experten und laedt ihn aus dem Chart aus.

```
void  ExpertRemove();
```

Rückgabewert

Keinen Rückgabewert.

Hinweis

Expert wird nicht sofort beim Aufuf der Funktion ExpertRemove() gestoppt, es wird erst die Flagge für Brechen der Expertarbeit eingestellt. D.h. jedes folgendes Ereignis wird vom Experten schon nicht verarbeitet, OnDeinit() wird aufgerufen werden und Expert wird ausgeladen werden und vom Chart entfernt werden.

Der Aufruf von ExpertRemove() im Strategietester aus der Funktion OnInit() bricht den Test des aktuellen Parametersatzes ab. Ein solches Ende gilt als Initialisierungsfehler.

Wenn ExpertRemove() im Strategietester nach erfolgreicher Initialisierung eines EA aufgerufen wird, wird der Test normal mit dem Aufruf von OnDeinit() beendet und mit OnTester() abgeschlossen. In diesem Fall bleiben die gesamte Handelsstatistik und das Optimierungskriterium erhalten.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                            Test_ExpertRemove.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
input int ticks_to_close=20;// Tickzahl vor dem Expertausladen
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Print(TimeCurrent(),": " ,__FUNCTION__," reason code = ",reason);
//--- "clear" comment
   Comment("");
//---
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
   static int tick_counter=0;
//---
   tick_counter++;
   Comment("\Vor dem Expertausladen ",__FILE__," bleiben",
           (ticks_to_close-tick_counter),"Ticks");
//--- bis
   if(tick_counter>=ticks_to_close)
     {
      ExpertRemove();
      Print(TimeCurrent(),": ",__FUNCTION__," Expert wird ausgeladen werden");
     }
   Print("tick_counter = ",tick_counter);
//---
  }
//+------------------------------------------------------------------+
```

Sehen Sie auch

Programmausführung, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

GetPointer

Gibt Anzeiger des Objektes zurück.

```
void*  GetPointer(
   any_class anyobject      // Objekt jeder Klasse
   );
```

Parameter

anyobject

[in] Objekt jeder Klasse.

Rückgabewert

Gibt Objektanzeiger zurück.

Hinweis

Nur Klassenobjekte haben Anzeiger. Instanzen der Strukturen und Variablen einfacher Typen haben keine Anzeiger. Klassenobjekt, das nicht nurch die Anweisung new(), sondern ZB. automatisch im Feld der Objekte erzeugt wurde, hat trotzdem einen Anzeiger.Dieser Anzeiger wird aber des automatischen Typs POINTER_AUTOMATIC sein, darum kann die Anweisung delete() nicht angewendet werden. Sonst unterscheidet sich die Anweisung dieses Typs nicht von den Anzeigern des Typs POINTER_DYNAMIC.

Da Variablen des Typs Strukturen und einfacher Typen keine Anzeiger haben, ist es verboten, die Funktion GetPointer() für sie anzuwenden. Man darf auch nicht Anzeiger als Argument der Funktion zu übertragen. In allen genannten Faellen meldet der Compiler den Fehler.

Versuch der Anwendung zum unkorrekten Anzeiger führt zur kritischen Beendigung des Programms. Darum besteht es die Notwendigkeit, die Funktion CheckPointer() vor dem Gebrauch des Anzeigers zu verwenden. Anzeiger kann in folgenden Faellen unkorrekt sein:

- Anzeiger ist NULL gleich;
- wenn Objekt durch die Anweisung delete entfernt wurde.

Diese Funktion kann als Pruefung des Anzeigers auf Validitaet verwendet werden. Nicht-Null Wert garantiert, dass Anzeiger für Zugang verwendet werden kann.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                              Check_GetPointer.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"

//+------------------------------------------------------------------+
//| Klasse, implementierte Element der Liste                         |
//+------------------------------------------------------------------+
class CItem
  {
   int               m_id;
   string            m_comment;
   CItem*            m_next;
public:
                     CItem() { m_id=0; m_comment=NULL; m_next=NULL; }
                    ~CItem() { Print("Destructor of ",m_id,
                                     (CheckPointer(GetPointer(this))==POINTER_DYNAMIC)
                                     "dynamic":"non-dynamic"); }
   void              Initialize(int id,string comm) { m_id=id; m_comment=comm; }
   void              PrintMe() { Print(__FUNCTION__,":",m_id,m_comment); }
   int               Identifier() { return(m_id); }
   CItem*            Next() {return(m_next); }
   void              Next(CItem *item) { m_next=item; }
  };
//+------------------------------------------------------------------+
//| Die einfachste Klasse der Liste                                  |
//+------------------------------------------------------------------+
class CMyList
  {
   CItem*            m_items;
public:
                     CMyList() { m_items=NULL; }
                    ~CMyList() { Destroy(); }
    bool             InsertToBegin(CItem* item);
    void             Destroy();
  };
//+------------------------------------------------------------------+
//| Einfuegung des Elementes der Liste am Anfang                     |
//+------------------------------------------------------------------+
bool CMyList::InsertToBegin(CItem* item)
  {
   if(CheckPointer(item)==POINTER_INVALID) return(false);
//---
   item.Next(m_items);
   m_items=item;
//---
   return(true);
  }
//+------------------------------------------------------------------+
//| Entfernung der Liste durch Entfernung der Elementen              |
//+------------------------------------------------------------------+
void CMyList::Destroy()
  {
//--- Dienstanzeiger für Arbeit im Zyklus
   CItem* item;
//--- gehen wir durch den Zyklus und versuchen wir, dynamische Anzeiger zu entfernen
   while(CheckPointer(m_items)!=POINTER_INVALID)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       item=m_items;
       m_items=m_items.Next();
       if(CheckPointer(item)==POINTER_DYNAMIC)
         {
          Print("Dynamyc object ",item.Identifier()," to be deleted");
          delete (item);
         }
       else Print("Non-dynamic object ",item.Identifier()," cannot be deleted");
      }
//---
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   CMyList list;
   CItem   items[10];
   CItem*  item;
//--- erzeugen wir und fuegen in die Liste das dynamische Anzeiger des Objektes hinzu
   item=new CItem;
   if(item!=NULL)
     {
      item.Initialize(100,"dynamic");
      item.PrintMe();
      list.InsertToBegin(item);
     }
//--- fuegen wir automatische Anzeiger in die Liste hinzu
   for(int i=0; i<10; i++)
     {
      items[i].Initialize(i,"automatic");
      items[i].PrintMe();
      item=GetPointer(items[i]);
      if(CheckPointer(item)!=POINTER_INVALID)
         list.InsertToBegin(item);
     }
//--- fuegen wir noch einen dynamischen Anzeiger des Objektes am Anfang der Liste hinz
   item=new CItem;
   if(item!=NULL)
     {
      item.Initialize(200,"dynamic");
      item.PrintMe();
      list.InsertToBegin(item);
     }
//--- entfernen wir Elementen der Liste
   list.Destroy();
//--- Alle Elementen der Liste werden entfernt werden,
//--- sieh im Terminal Registerblatt Experts
  }
```

Sehen Sie auch

Objektanzeiger, Pruefung des Objektanzeigers, Anweisung der Objektentfernung delete

© 2000-2025, MetaQuotes Ltd.

GetTickCount

Funktion GetTickCount() gibt die Zahl der Millisekunden, die seit Systemstart vorbei sind.

```
uint  GetTickCount();
```

Rückgabewert

Wert des Typs uint.

Hinweis

Counter ist durch Auflösungsvermögen des Systemtimers begrenzt. Da die Zeit als das zeichenlose Ganze aufbewahren wird, wird er jede 49.7 Tage bei ununterbrochener Computerarbeit überfuellt.

Beispiel:

```
#define MAX_SIZE 40
//+------------------------------------------------------------------+
//| Skript für Messung der Rechenzeit von 40 Fibonacci-Zahlen        |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Merken wir uns den Initialwert
   uint start=GetTickCount();
//--- Variable für Erhaltung der nächsten Zahl in der Fibonacci-Reihe
   long fib=0;
//--- Zyklus, in dem wir eine bestimmte Anzahl von Zahlen in der Fibonacci-Reihe  bere
   for(int i=0;i<MAX_SIZE;i++) fib=TestFibo(i);
//--- Verstrichene Zeit in Millisekunden erhalten
   uint time=GetTickCount()-start;
//--- Eine Meldung für Journal "Experten"
   PrintFormat("Berechnung der %d ersten Fibonacci-Zahlen dauerte %d ms",MAX_SIZE,time
//--- Das Skript ist fertig
   return;
  }
//+------------------------------------------------------------------+
//| Funktion für Erhaltung der Fibonacci-Zahle durch ihr Folgenummer |
//+------------------------------------------------------------------+
long TestFibo(long n)
  {
//--- Das erste Glied der Fibonacci-Reihe
   if(n<2) return(1);
//--- Alle nachfolgenden Glieder sind nach der Formel berechnet
   return(TestFibo(n-2)+TestFibo(n-1));
  }
```

Sehen Sie auch

Datum und Zeit, EventSetMillisecondTimer, GetTickCount64, GetMicrosecondCount

© 2000-2025, MetaQuotes Ltd.

GetTickCount64

Die Funktion GetTickCount64() gibt die Anzahl der Millisekunden seit dem Start des Systems zurück.

```
ulong  GetTickCount64();
```

Rückgabewert

Ein Wert vom Typ ulong

Hinweis

Der Wert ist auf die Genauigkeit der Systemuhr beschränkt, der normalerweise ein Ergebnis mit einer Genauigkeit von 10-16 Millisekunden liefert. Im Gegensatz zu GetTickCount, dessen Wert vom Typ uint ist, und dessen Werte im Falle eines fortgesetzten Computerbetriebs alle 49,7 Tage überläuft, kann GetTickCount64() für die unbegrenzte Computerbetriebszeit verwendet werden und unterliegt keinem Überlauf.

Beispiel:

```
#define MAX_SIZE 40

//+------------------------------------------------------------------+
//| Skript zur Messung der Zeit für die Berechnung von 40 Fibo-Zahlen|
//+------------------------------------------------------------------+
void OnStart()
  {
   long fib_array[MAX_SIZE];

//--- speichern der Anfangswerte
   ulong start=GetTickCount64();
//--- eine Schleife, in der wir eine bestimmte Anzahl von Zahlen aus der Fibo-Reihe be
   for(int i=0;i<MAX_SIZE;i++)
      fib_array[i]=TestFibo(i);
//--- Ermitteln der verbrauchten Zeit in Millisekunden
   ulong time=GetTickCount64()-start;

//--- Anzeige der Fehlermeldung im Expertenjournal
   ArrayPrint(fib_array);
   PrintFormat("Calculating the first %d Fibonacci numbers took %I64u ms",MAX_SIZE,tir
  }
//+------------------------------------------------------------------+
//| Funktion zum Ermitteln einer Fibo-Zahl anhand ihrer Seriennummer |
//+------------------------------------------------------------------+
long TestFibo(long n)
  {
//--- erstes Zahl der Fibo-Reihe
   if(n<2)
      return(1);
//--- alle nachfolgenden Mitglieder werden nach dieser Formel berechnet
```

© 2000-2025, MetaQuotes Ltd.

```
   return(TestFibo(n-2)+TestFibo(n-1));
  }
```

Sehen Sia auch

Datum und Zeit, EventSetMillisecondTimer, GetTickCount, GetMicrosecondCount

© 2000-2025, MetaQuotes Ltd.

GetMicrosecondCount

Die GetMicrosecondCount() Funktion gibt die Anzahl der Mikrosekunden zurück, die seit dem Anfang des MQL5-Programms vergangen sind.

```
ulong  GetMicrosecondCount();
```

Rückgabewert

Wert des Typs ulong.

Beispiel:

```
//+------------------------------------------------------------------+
//| Der zu testende Code                                             |
//+------------------------------------------------------------------+
void Test()
  {
   int    res_int=0;
   double res_double=0;
//---
   for(int i=0;i<10000;i++)
     {
      res_int+=i*i;
      res_int++;
      res_double+=i*i;
      res_double++;
     }
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   uint   ui=0,ui_max=0,ui_min=INT_MAX;
   ulong  ul=0,ul_max=0,ul_min=INT_MAX;
//--- Anzahl der Tests
   for(int count=0;count<1000;count++)
     {
      uint  ui_res=0;
      ulong ul_res=0;
      //---
      for(int n=0;n<2;n++)
        {
         //--- Typ der Zählung auswählen
         if(n==0) ui=GetTickCount();
         else     ul=GetMicrosecondCount();
         //--- der zu testende Code
         Test();
         //--- das Ergebnis der Dimension speichern (je nach der Methode)
```

© 2000-2025, MetaQuotes Ltd.

```
         if(n==0) ui_res+=GetTickCount()-ui;
         else     ul_res+=GetMicrosecondCount()-ul;
        }
      //--- sammeln wir die minimale und die maximale Zeit der Ausführung des Codes de
      if(ui_min>ui_res) ui_min=ui_res;
      if(ui_max<ui_res) ui_max=ui_res;
      if(ul_min>ul_res) ul_min=ul_res;
      if(ul_max<ul_res) ul_max=ul_res;
     }
//---
   Print("GetTickCount error(msec): ",ui_max-ui_min);
   Print("GetMicrosecondCount error(msec): ",DoubleToString((ul_max-ul_min)/1000.0,2))
  }
```

Sehen Sie auch

Datum und Zeit, GetTickCount, GetTickCount64

© 2000-2025, MetaQuotes Ltd.

MessageBox (Nachrichtenfeld)

Erzeugt und zeigt das Meldungsfenster und verwaltet es. Das Meldungsfenster enthält eine Meldung und einen Titel, eine beliebige Kombination von vordefinierten Symbolen und Befehlsschaltflächen.

```
int  MessageBox(
   string  text,              // Meldungstext
   string  caption=NULL,      // Meldungskopf
   int     flags=0            // definiert Schaltflächensatz im Fenster
   );
```

Parameter

text

[in] Text mit der Meldung für Ausgabe.

caption=NULL

[in] Freier Text für Ausgabe im Kopf des Meldungsfensters. Wenn dieser Parameter leer ist, wird der Name des Expert Advisors im Fenstertitel angezeigt .

flags=0

[in] Optionale Flags, die das Aussehen und das Verhalten der Dialogbox bestimmen. Flags können eine Kombination aus einer speziellen Gruppe von Flags sein.

Rückgabewert

Wenn die Funktion erfolgreich ausgeführt wird, ist der Rückgabewert-einer der Werte der Rückgabekodes der MessageBox().

Hinweis

Die Funktion kann nicht in nutzerdefinierten Indikatoren verwendet werden, da durch den Aufruf von MessageBox() der gerade ausgeführte Thread für die gesamte Zeit, während der auf die Antwort des Nutzers gewartet wird, unterbrochen wird. Da alle Indikatoren für jedes Symbol in einem einzigen Thread ausgeführt werden, macht eine solche Unterbrechung den Arbeitsablauf aller Diagramme in allen Zeitrahmen für dieses Symbol unmöglich.

Beim Anwenden im Strategie-Tester wird die MessageBox()-Funktion nicht ausgeführt.

Beispiel:

```
//+------------------------------------------------------------------+
//| Die MessageBox des EAs mit einer Aufforderung zur Weiterarbeit.  |
//| Bei Erreichen einer bestimmten Anzahl einer Verlustserie         |
//| warte für die angegebene Balkenzahl und zeige MessageBox mit     |
//| einer Aufforderung zur Weiterarbeit.                             |
//| Zur Prüfung müssen wir Positionen manuell öffnen und mit         |
//| mit Verlusten schließen, da der EA seine "Positionen" der        |
//| der Einfachheit halber nicht über seine magische Zahl steuert.   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Eingabeparameter
input uint InpMaxLossDeals      =  3;    // Deals, Maximalverlust
input uint InpInactivityNumBars =  5;    // Balkenzahl ohne Aktivität des EAs

//--- globale Variable
bool     ExtFirstStart=true;             // Flag für Erststart
bool     ExtFlag=true;                   // Flag der "Arbeitserlaubnis" für den EA
uint     ExtNumLoss;                     // Anzahl aufeinanderfolgender Handelsgeschäf
datetime ExtTimeLastLoss;                // Schließzeit der letzten Position mit einer

//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Ermitteln der Anzahl aufeinanderfolgender Verlustgeschäfte und des Zeitpunkts de
   ExtNumLoss=GetNumLosingTradesInRow(ExtTimeLastLoss);

   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   Comment("");
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- Feststellen, wie viele Balken seit der letzten geschlossenen Verlustposition in
   int bars_remaining=iBarShift(Symbol(),PERIOD_CURRENT,ExtTimeLastLoss);

//--- Wenn das der erste Start ist
   if(ExtFirstStart)
     {
      //--- Wenn nach einer Reihe unrentabler Positionen bereits eine bestimmte Anzahl
      if(bars_remaining>(int)InpInactivityNumBars)
         ExtFlag=true;
      ExtFirstStart=false;
     }

//--- wenn das EA-Operations-Flag deaktiviert ist
   if(!ExtFlag)
     {
      Comment(StringFormat("The advisor is stopped for %d bars. Num Loss positions: %u
                           (InpInactivityNumBars-bars_remaining),ExtNumLoss,TimeToStrin
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- wenn nach einer Reihe von unrentablen Positionen eine bestimmte Anzahl vor
      if(bars_remaining>(int)InpInactivityNumBars)
        {
         //--- Fenster der MessageBox mit angegebenen Text und Fenstertitel
         //--- das Abfragefenster hat zwei Schaltflächen Yes/No (Ja/Nein) und ein Symb
         //--- die Schaltfläche Yes (Ja) ist standardmäßig ausgewählt.
         string mb_text="The specified number of bars of EA inactivity have passed.\n
         string mb_caption="Please note";
         int    mb_id=MessageBox(mb_text,mb_caption,MB_YESNO|MB_ICONQUESTION|MB_DEFBUT
         //--- wenn der Rückgabewert von MessageBox die gedrückte Ja-Taste ist, setze
         if(mb_id==IDYES)
           {
            ExtFlag=true;
            return;
           }
        }
      //--- ist das EA-Operations-Flag deaktiviert, verlasse OnTick()
      return;
     }

//---  das EA-Operations-Flag ist gesetzt - der EA funktioniert wie im folgenden Code
   Comment(StringFormat("The advisor is working. Num Loss positions: %u, Time last los
           ExtNumLoss,TimeToString(ExtTimeLastLoss,TIME_DATE|TIME_MINUTES|TIME_SECONDS
  }
//+------------------------------------------------------------------+
//| TradeTransaction Funktion                                        |
//+------------------------------------------------------------------+
void OnTradeTransaction(const MqlTradeTransaction& trans,
                        const MqlTradeRequest& request,
                        const MqlTradeResult& result)
  {
//--- wenn der Transaktionstyp das Hinzufügen einer Transaktion zur Historie ist
   if(trans.type==TRADE_TRANSACTION_DEAL_ADD)
     {
      //--- das Deal-Ticket abrufen und ein Deal aus der Liste mit dem Ticket auswähle
      ulong deal_ticket=trans.deal;
      if(HistoryDealSelect(deal_ticket))
        {
         //--- wenn eine Position geschlossen wurde, ermittle die Anzahl der Verlustge
         ENUM_DEAL_ENTRY entry=(ENUM_DEAL_ENTRY)HistoryDealGetInteger(deal_ticket,DEAL
         if(entry==DEAL_ENTRY_OUT || entry==DEAL_ENTRY_INOUT || entry==DEAL_ENTRY_OUT_
            ExtNumLoss=GetNumLosingTradesInRow(ExtTimeLastLoss);
        }
     }

//--- wenn die Anzahl der Verlustgeschäfte in einer Reihe größer ist als der angegeber
   if(ExtNumLoss>=InpMaxLossDeals && ExtFlag)
     {
      //--- Fenster der MessageBox mit angegebenen Text und Fenstertitel
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- das Abfragefenster hat zwei Schaltflächen Yes/No (Ja/Nein) und ein Symbol
      //--- die Schaltfläche No (Nein) ist standardmäßig ausgewählt.
      string mb_text="The number of losing trades has reached the specified maximum. T
      string mb_caption="Attention!";
      int    mb_id=MessageBox(mb_text,mb_caption,MB_YESNO|MB_ICONQUESTION|MB_DEFBUTTON
      //--- wenn in der MessageBox die Schaltfläche No (Nein) gedrückt und zurückgegeb
      if(mb_id==IDNO)
         ExtFlag=false;
     }
  }
//+------------------------------------------------------------------+
//| Gib die Anzahl der fortlaufenden Verlust-Deals zurück            |
//| und den Zeitpunkt der letzten, schließenden Position             |
//+------------------------------------------------------------------+
uint GetNumLosingTradesInRow(datetime &time_last_deal)
  {
//--- Auswahl der ganzen Historie
   if(!HistorySelect(0,TimeCurrent()))
      return(0);

//--- Ermitteln des nächsten Handelstickets anhand der Liste der historischen Deals in
   uint res=0;
   uint total=HistoryDealsTotal();
   for(int i=(int)total-1; i>=0; i--)
     {
      ulong deal_ticket=HistoryDealGetTicket(i);
      if(deal_ticket>0)
        {
         //--- wenn der Deal nicht zum Schließen der Position führt, wechsele zum näch
         ENUM_DEAL_ENTRY entry=(ENUM_DEAL_ENTRY)HistoryDealGetInteger(deal_ticket,DEAL
         if(entry!=DEAL_ENTRY_OUT && entry!=DEAL_ENTRY_OUT_BY && entry!=DEAL_ENTRY_INO
            continue;
         //--- wenn das Ergebnis des Schließens einer Position einen Gewinn ergibt, un
         if(!IsClosePositionWithLoss(deal_ticket))
            break;
         //--- den Zähler der fortlaufenden Verlust-Deals erhöhen
         res++;
         //--- die maximale Handelszeit einer Variablen zuweisen (suche die letzte)
         datetime deal_time=(datetime)HistoryDealGetInteger(deal_ticket,DEAL_TIME);
         if(deal_time>time_last_deal)
            time_last_deal=deal_time;
        }
     }

//--- Rückgabe der Anzahl der fortlaufenden Verluste
   return(res);
  }
//+------------------------------------------------------------------+
//| Rückgabe des Flags für das Schließen einer Position mit Verlust  |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool IsClosePositionWithLoss(const ulong deal_ticket)
  {
//--- Ermitteln der Eigenschaften, die den Handelsgewinn beeinflussen
   double profit=HistoryDealGetDouble(deal_ticket,DEAL_PROFIT);
   double comission=HistoryDealGetDouble(deal_ticket,DEAL_COMMISSION);
   double swap=HistoryDealGetDouble(deal_ticket,DEAL_SWAP);
   double fee=HistoryDealGetDouble(deal_ticket,DEAL_FEE);

//--- Rückgabe des Flags, das anzeigt, dass der Gesamtwert der erhaltenen Eigenschafte
   return(profit+comission+swap+fee<0);
  }
```

© 2000-2025, MetaQuotes Ltd.

PeriodSeconds

Gibt die Zahl der Sekunden in der Periode zurück.

```
int  PeriodSeconds(
   ENUM_TIMEFRAMES  period=PERIOD_CURRENT      // Chartperiode
   );
```

Parameter

period=PERIOD_CURRENT

[in] Wert der Chartperiode aus der Enumeration ENUM_TIMEFRAMES. Wenn der Parameter nicht angegeben wird, wird die Sekundenzahl des Charts der laufenden Periode, wo das Programm gestartet ist, zurückgegeben.

Rückgabewert

Zahl der Sekunden in der angegebenen Periode.

Beispiel:

```
//--- Eingabeparameter
input ENUM_TIMEFRAMES InpPeriod1 = PERIOD_CURRENT;    // Erster Zeitrahmen
input ENUM_TIMEFRAMES InpPeriod2 = PERIOD_M1;         // Zweiter Zeitrahmen

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln der Anzahl der Sekunden in den Chart-Perioden InpPeriod1 und InpPerioc
   int sec1=PeriodSeconds(InpPeriod1);
   int sec2=PeriodSeconds(InpPeriod2);
//--- Anzeige der empfangenen Werte im Protokoll
   PrintFormat("Seconds in period %s: %lu, in period %s: %lu",TimeframeDescription(Inp
//--- Berechnen, wie viele Balken der Chart-Periode InpPeriod2 in einem Balken mit der
   int res=sec1/sec2;
   if(res==0)
      res=1;
//--- Anzeige des erhaltenen Wertes im Protokoll
   PrintFormat("One bar %s contains %d bars %s",TimeframeDescription(InpPeriod1),res,T
   /*
  Ergebnis:
   Seconds in period M5: 300, in period M1: 60
   One bar M5 contains 5 bars M1
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe des Namens des Zeitrahmens                              |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
string TimeframeDescription(const ENUM_TIMEFRAMES period)
  {
   return(StringSubstr(EnumToString(period==PERIOD_CURRENT ? Period() : period), 7));
  }
```

Sehen Sia auch

_Period, Chartperioden, Datum und Zeit, Objektsichbarkeit

© 2000-2025, MetaQuotes Ltd.

PlaySound

Spielt Audiodatei ab.

```
bool  PlaySound(
   string  filename      // Dateiname
   );
```

Parameter

filename

[in] Pfad zur Audiodatei. Wenn filename=NULL ist, wird die Audiowiedergabe gestoppt.

Rückgabewert

true - wenn Audiodatei nicht gefunden wird, ansonsten false.

Hinweis

Datei muss sich im Katalog Katalog_des Terminals\Sounds oder in seinem Subkatalog befinden. Es werden nur Audiodateien im Format WAV abgespiegelt.

Aufruf von Playsound() mit einem Parameter NULL stoppt Audiowiedergabe.

Beim Anwenden im Strategie-Tester wird die PlaySound()-Funktion nicht ausgeführt.

Beispiel:

```
#include <Trade\Trade.mqh>
#define MAGIC (123)

//--- Eingabeparameter
input string          InpFileNameOK  = "ok.wav";               // Audiodatei für Erfol
input string          InpFileNameErr = "timeout.wav";          // Audiodatei für einen
input ENUM_ORDER_TYPE InpOrderType   = ORDER_TYPE_BUY_LIMIT;   // Auftragstyp
input double          InpLots        = 0.1;                    // Losgröße

//--- globale Variable
CTrade ExtTrade;

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Setzen der MagicNumber und die Auftragsart bei der Ausführung entsprechend den S
   ExtTrade.SetExpertMagicNumber(MAGIC);
   ExtTrade.SetTypeFillingBySymbol(Symbol());
//--- Aufruf der Funktion zum Erteilen eines Auftrags oder zum Eröffnen einer Position
   OrderSendWithAudio();
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Die Funktion platziert einen Auftrag oder eröffnet eine Position |
//| und spielt den Sound für Erfolg oder für einen Fehler ab         |
//+------------------------------------------------------------------+
void OrderSendWithAudio(void)
  {
   bool    res=true;
   MqlTick tick= {};

   ResetLastError();
   if(!SymbolInfoTick(Symbol(),tick))
     {
      Print("SymbolInfoTick() failed. Error code: ",GetLastError());
      PlaySound(InpFileNameErr);
      return;
     }
//--- Senden einer Anfrage an den Server
   switch(InpOrderType)
     {
      case ORDER_TYPE_BUY :
         res=ExtTrade.Buy(InpLots);
         break;
      case ORDER_TYPE_BUY_LIMIT :
         res=ExtTrade.BuyLimit(InpLots,NormalizeDouble(tick.ask-100*Point(),Digits()))
         break;
      case ORDER_TYPE_BUY_STOP :
         res=ExtTrade.BuyStop(InpLots,NormalizeDouble(tick.ask+100*Point(),Digits()));
         break;
      case ORDER_TYPE_SELL :
         res=ExtTrade.Sell(InpLots);
         break;
      case ORDER_TYPE_SELL_LIMIT :
         res=ExtTrade.SellLimit(InpLots,NormalizeDouble(tick.bid+100*Point(),Digits())
         break;
      case ORDER_TYPE_SELL_STOP :
         res=ExtTrade.SellStop(InpLots,NormalizeDouble(tick.bid-100*Point(),Digits()))
         break;
      default :
         res=false;
     }
   if(!res)
      Print("Error ",GetLastError());
   Print(ExtTrade.ResultRetcodeDescription());

//--- wenn die Anfrage akzeptiert wird, wird der Sound ok.wav abgespielt
   if(ExtTrade.ResultRetcode()==TRADE_RETCODE_DONE)
      PlaySound(InpFileNameOK);
   else
      PlaySound(InpFileNameErr);
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Sehen Sia auch

Ressourcen

© 2000-2025, MetaQuotes Ltd.

Print

Druckt eine Nachricht im Expertenjournal. Parameter können verschiedener Typs sein.

```
void  Print(
   argument,      // der erste Wert
   ...            // folgende Werte
   );
```

Parameter

...

[in] Jede Werte, getrennt durch Kommas. Parameterzahl kann nicht mehr als 64 sein.

Hinweis

Parameter können nicht in die Funktion Print() übertragen werden. Felder müssen elementenweise gedruckt werden.

Daten des Typs double werden mit 16 Dezimalzeichen ausgegeben., dabei können Daten im traditionellen oder im wissenschaftlichen Format ausgegeben werden - abhängig davon, welches Ausgabeformat am kompaktesten ist. Daten des Typs float werden mit 5 Dezimalzeichen ausgegeben. Für Ausgabe der reellen Zahlen mit anderer Genauigkeit oder im vorbestimmten Format muss die Funktion PrintFormat() verwendet werden.

Daten des Typs bool werden als Zeilen des Typs "true" oder "false" ausgegeben. Daten werden als YYYY.MM.DD HH:MI:SS ausgegeben . Für Ausgabe des Datums im anderen Format muss die Funktion TimeToString() verwendet werden. Daten des Typs color werden entweder als die Zeile R,G,B oder als Farbenname ausgegeben, wenn diese Farbe im Farbenbestand gibt.

Im Strategie-Tester im Optimierungsmodus wird die Print()-Funktion nicht ausgeführt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- Ausgeben DBL_MAX mit Print(), ist dies gleichbedeutend PrintFormat(%%.16G,DBL_MA
   Print("---- Wie DBL_MAX sieht aus -----");
   Print("Print(DBL_MAX)=",DBL_MAX);
//--- Jetzt geben wird DBL_MAX mit PrintFormat() aus
   PrintFormat("PrintFormat(%%.16G,DBL_MAX)=%.16G",DBL_MAX);
//--- Ausgabe in Expertenjournal
// Print(DBL_MAX)=1.797693134862316e+308
// PrintFormat(%.16G,DBL_MAX)=1.797693134862316E+308

//--- Sehen wir wie Typ float ausgegeben wird
   float c=(float)M_PI; // Wir müssen explizit auf das Zieltype umwandeln
   Print("c=",c, "    Pi=",M_PI, "    (float)M_PI=",(float)M_PI);
// c=3.14159    Pi=3.141592653589793    (float)M_PI=3.14159

//--- Wir zeigen, was während den arithmetischen Operationen mit echten passieren kann
   double a=7,b=200;
   Print("---- vor den arithmetischen Operationen");
   Print("a=",a,"   b=",b);
   Print("Print(DoubleToString(b,16))=",DoubleToString(b,16));
//--- Wir dividieren a durch b (7/200)
   a=a/b;
//--- Wiederherstellung des Werts von b
   b=7.0/a; // Es wird erwartet, dass b=7.0/(7.0/200.0)=>7.0/7.0*200.0=200 - aber es i
//--- Wir geben den neu berechneten Wert von b
   Print("----- nach den arithmetischen Operationen");
   Print("Print(b)=",b);
   Print("Print(DoubleToString(b,16))=",DoubleToString(b,16));
//--- Ausgabe in Expertenjournal
// Print(b)=200.0
// Print(DoubleToString(b,16))=199.9999999999999716 (Wir sehen dass b nicht gleich 200

//--- Erstellen wir einen sehr kleinen Wert epsilon=1E-013
   double epsilon=1e-13;
   Print("---- Erstellen wir einen sehr kleinen Wert");
   Print("epsilon=",epsilon); // Erhalten epsilon=1E-013
//--- Nun b weniger Epsilon - geben den Wert in Expertenjournal aus
   b=b-epsilon;
//--- Ausgeben durch zwei Methoden
   Print("---- Nach Subtraktion von epsilon aus b");
   Print("Print(b)=",b);
   Print("Print(DoubleToString(b,16))=",DoubleToString(b,16));
//--- Ausgabe in Expertenjournal
// Print(b)=199.9999999999999  (Nun kann der Wert von b nach Subtraktion von epsilon n
// Print(DoubleToString(b,16))=199.9999999999998578
//    (Nun kann der Wert von b nach Subtraktion von epsilon nicht bis zu 200 gerundet
  }
```

Sehen Sie auch

DoubleToString, StringFormat

© 2000-2025, MetaQuotes Ltd.

PrintFormat

Formatiert und druckt Symbolvorrat im Expertenjournal laut dem vorgegebenen Format.

```
void  PrintFormat(
   string format_string,    // Formatzeile
   ...                      // Werte einfacher Typen
   );
```

Parameter

format_string

[in] Formatzeile bestehl aus den normalen Symbolen und auch Formatspezifikation (wenn Argumente nach der Formatzeile stehen).

...

[in] Jede Werte einfacher Typen, die durch Kommas getrennt werden. Die gesamte Anzahl der Parameter kann nicht mehr als 64 sein, einschliesslich die Formatzeile.

Rückgabewert

Zeile.

Hinweis

Im Strategie-Tester im Optimierungsmodus wird die PrintFormat()-Funktion nicht ausgeführt.

Anzahl, Reihenfolge und Art der Parameter muss exakt der Zusammensetzung der Qualifikanten entsprechen, sonst das Druckergebnis ist undefiniert. Statt der Funktion PrintFormat() kann die Funktion printf() verwendet werden.

Wenn nach der Zeile Parameter stehen, muss diese Zeile Formatspezifikationen enthalten, die Ausgabeformat dieser Parameter bestimmen. Formatspezifikation beginnt mit Symbol (%).

Formatzeile wird von links nach rechts gelesen. Wenn die erste Formatspezifikation auftritt (wenn es gibt), wird der Wert des ersten Parameters nach der Formatzeile umgewandelt und laut der vorgegebenen Spezifikation ausgegeben. Die zweite Spezifikation ruft Umwandlung und Ausgabe des zweiten Parameters auf usw bis zum Ende der Formatzeile.

Formatspezifikation sieht so aus:

%[flags][width][.precision][{h | l | ll | I32 | I64}]type

Jedes Feld der Formatspezifikation ist entweder ein einfaches Symbol oder eine Zahl, die eine normale Formatoption bezeichnet. Die einfachste Formatspezifikation enthält nur Prozentzeichen (%) und Symbol, das den Typ des Eingabeparameters (ZB %s) bestimmt. Wenn das Prozentzeichen in der Formatzeile ausgegeben werden muss, muss die Formatspezifikation %%. verwendet werden.

flags

© 2000-2025, MetaQuotes Ltd.

Flagge	Beschreibung	Default-Verhalten
- (minus)	Linksbuendige Ausrichtung innerhalb der vorgegebenen Breite	Rechtsbuendige Ausrichtung
+ (plus)	Ausgabe des Zeichens + oder - für Werte der Zeichentypen	Zeichen wird ausgegeben, nur wenn der Wert negativ ist.
0 (Null)	Vor dem Ausgabewert werden Nullen innerhalb der vorgegebenen Breite zugesetzt. Wenn die Flagge **0** mit dem ganzzahligen Format (**i**, **u**, **x**, **X**, **o**, **d**) angegeben wird und Spezifikation der Genauigkeit vorgegeben wird (ZB, %04.d),wird 0 ignoriert.	Nichts wird eingegeben
Space	Vor dem Ausgabewert wird Space gestellt, wenn der Wert ein positiver Zeichenwert ist.	Spaces werden nicht eingegeben
#	Beim Joint-Verwenden mit **o**, **x** oder **X** wird vor dem Ausgabewert 0, 0x oder 0X zugesetzt.	Nichts wird eingegeben
	Beim Joint-Verwenden mit Format **e**, **E**, **a** oder **A** wird der Wert immer mit Dezimalpunkt ausgegeben.	Dezimalpunkt wird erst dann ausgegeben, wenn es einen Nicht-Null Bruchanteil gibt.
	Beim Joint-Verwenden mit Format **g** oder **G**, bestimmt Flagge das Vorhandensein des Dezimalpunktes und beugt die Reduzierung der führenden Nullen vor. Flagge # wird beim Joint-Verwenden mit Formaten **c**, **d**, **i**, **u**, **s** ignoriert.	Dezimalpunkt wird erst dann ausgegeben, wenn es einen Nicht-Null Bruchanteil gibt. Fuehrende Nullen werden reduziert.

width

Eine nichtnegative Dezimalzahl, die minimale Anzahl der Ausgabesymbole des formatierten Wertes vorgibt. Wenn die Anzahl der Ausgabesymbole weniger als angegebene Breite ist, wird die entsprechende Anzahl der Spaces links und rechts abhängig von Ausrichtung zugesetzt (Flagge -). Beim Vorhandensein der Flagge Null, wird vor dem Ausgabewert die entsprechende Anzahl von Nullen zugesetzt. Wenn die Anzahl der Ausgabesymbole mehr als die vorgegebene Breite ist, wird der Ausgabewert nie reduziert.

Wenn als Breite Sternchen (*) angegeben wird, muss in der Liste der übertragenen Parameter der Wert des Typs int an der entsprechenden Stelle sein, der für Breiteangabe des Ausgabewertes verwendet wird.

precision

Eine nichtnegative Dezimalzahl, die Genauigkeit der Ausgabe bestimmt - Anzahl der Dezimalzeichen. Zum Unterschied von Spezifikation der Länge kann Spezifikation der Genauigkeit einen Teil des Bruchwertes mit Aufrundung oder ohne Aufrundung reduzieren.

© 2000-2025, MetaQuotes Ltd.

für verschiedene Typen (**type**) des Formats wird Spezifikation der Genauigkeit verschieden verwendet.

Typen	Beschreibung	Default-Verhalten
a, A	Spezifikation der Genauigkeit zeigt Anzahl der Dezimalzeichen	Default-Genauigkeit - 6.
c, C	nicht verwendet	
d, i, u, o, x, X	Zeigt minimale Zahl der ausgegebenen Ziffern. Wenn die Anzahl der Ziffern im vorgegebenen Parameter weniger als die angegebene Genauigkeit ist, wird der Ausgabewert links durch Nullen ergaenzt, Der Ausgabewert wird nicht reduziert, wenn die Anzahl der Ausgabeziffern mehr als die angegebene Genauigkeit ist.	Default-Genauigkeit - 1.
e, E, f	Gibt die Anzahl der Dezimalzeichen an, die letzte Ausgabeziffer wird ausgerundet.	Default-Genauigkeit - 6. Wenn die Genauigkeit 0 angegeben wird oder der Bruchteil fehlt, wird der Dezimalpunkt nicht ausgegeben
g, G	Gibt die maximale Anzahl der bedeutsamen Ziffern an	Es wird 6 bedeutsame Ziffern ausgegeben.
s	Gibt die Anzahl der Ausgabezeilensymbole an. Wenn die Zeilenlaenge mehr als der Wert der Genauigkeit ist, wird sie bei der Ausgabe reduziert.	die ganze Zeile wird ausgegeben

```
PrintFormat("1. %s", _Symbol);
PrintFormat("2. %.3s", _Symbol);
int length=4;
PrintFormat("3. %.*s", length, _Symbol);
/*
1. EURUSD
2. EUR
3. EURU
/
```

h | l | ll | I32 | I64

Spezifikation der Laengedimensionen, die als Parameter übertragen werden.

© 2000-2025, MetaQuotes Ltd.

Typ des Parameters	Das verwendete Praefix	Joint-Spezifikator des Typs
int	l (kleiner L)	d, i, o, x, or X
uint	l (kleiner L)	o, u, x, or X
long	ll (zwei kleine L)	d, i, o, x, or X
short	h	d, i, o, x, or X
ushort	h	o, u, x, or X
int	I32	d, i, o, x, or X
uint	I32	o, u, x, or X
long	I64	d, i, o, x, or X
ulong	I64	o, u, x, or X

type

Spezifikator des Typs ist ein einziges obligatorische Feld für formatierte Ausgabe.

Symbol	Typ	Ausgabeformat
c	int	Symbol des Typs short (Unicode)
C	int	Symbol des Typs char (ANSI)
d	int	Zeichendezimalganzzahl
i	int	Zeichendezimalganzzahl
o	int	Zeichenlose Oktalganzzahl
u	int	Zeichenlose Dezimalganzzahl
x	int	Zeichenlose hexadezimalganzzahl mit "abcdef" Verwendung
X	int	Zeichenlose hexadezimalganzzahl mit "ABCDEF" Verwendung
e	double	Realwert in Format [-]d.dddd **e** [sign]ddd, wo d - eine Dezimalzahl, dddd - eine oder mehrere Dezimalzahlen, ddd - dreistellige Zahl, die Exponentengroesse bestimmt, Zeichen - plus oder minus
E	double	Dem Format e, gleich unter Ausschluss, dass Exponentenzeichen mit einem Grossbuchstaben anfaengt (**E** statt e)
f	double	Realwert in Format [-]dddd.dddd, wo dddd - eine oder mehrere Dezimalziffern. Anzahl der ausgegebenen Dezimalzeichen hängt von der Größe der Zahl ab. Anzahl der Dezimalzeichen hängt von der erforderlichen Genauigkeit ab.

© 2000-2025, MetaQuotes Ltd.

Symbol	Typ	Ausgabeformat
g	double	Realwert, der im Format **f** oder **e** ausgegeben wird, abhängig davon, welches Format kompakter ist.
G	double	Realwert der im Format f oder e ausgegeben wird, abhängig davon, welche Ausgabe kompakter ist.
a	double	Realwert in Format [-]0xh.hhhh **p**±dd, wo h.hhhh - mantissa in Form der hexadezimaler Ziffern mit Verwendung von "abcdef", dd - eine oder mehrere Ziffern einer Exponente. Anzahl der Dezimalzeichen wird durch Spezifikation der Genauigkeit bestimmt.
A	double	Realwert in Format [-]0xh.hhhh **P**±dd, wo h.hhhh - mantissa in Form von hexadezimaler Ziffern mit Verwendung von "ABCDEF", dd - eine oder mehrere Ziffern einer Exponente. Anzahl der Dezimalzeichen wird durch Spezifikation der Genauigkeit bestimmt.
s	string	Zeilenausgabe

Anstatt der Funktion PrintFormat() kann die Funktion printf() verwendet werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- Name des Handelservers
   string server=AccountInfoString(ACCOUNT_SERVER);
//--- Kontonummer
   int login=(int)AccountInfoInteger(ACCOUNT_LOGIN);
//--- Ausgabe von long
   long leverage=AccountInfoInteger(ACCOUNT_LEVERAGE);
   PrintFormat("%s %d: Hebel = 1:%I64d",
               server,login,leverage);
//--- Währung der Einlage
   string currency=AccountInfoString(ACCOUNT_CURRENCY);
//--- Ausgabe von double mit 2 Ziffern nach dem Dezimalkomma
   double equity=AccountInfoDouble(ACCOUNT_EQUITY);
   PrintFormat("%s %d: eigene Mittel auf dem Konto = %.2f %s",
               server,login,equity,currency);
//--- Ausgabe von double mit dem Vorzeichen +/-
   double profit=AccountInfoDouble(ACCOUNT_PROFIT);
   PrintFormat("%s %d: aktuelles Ergebnis nach offenen Positionen = %+.2f %s",
               server,login,profit,currency);
//--- Ausgabe von double mit einer variablen Anzahl von Ziffern nach dem Dezimalkomma
   double point_value=SymbolInfoDouble(_Symbol,SYMBOL_POINT);
   string format_string=StringFormat("%%s: der Wert eines Punktes  = %%.%df",_Digits);
   PrintFormat(format_string,_Symbol,point_value);
//--- Ausgabe von int
   int spread=(int)SymbolInfoInteger(_Symbol,SYMBOL_SPREAD);
   PrintFormat("%s: aktueller Spread in Punkten = %d ",
               _Symbol,spread);
//--- Ausgabe von double im wissenschaftlichen Format mit einem Gleitkomma und 17 Stel
   PrintFormat("DBL_MAX = %.17e",DBL_MAX);
//--- Ausgabe von double im wissenschaftlichen Format mit einem Gleitkomma und 17 Stel
   PrintFormat("EMPTY_VALUE = %.17e",EMPTY_VALUE);
//--- Ausgabe mithilfe von PrintFormat() mit standardmäßiger Genauigkeit
   PrintFormat("PrintFormat(EMPTY_VALUE) = %e",EMPTY_VALUE);
//--- Einfache Ausgabe via Print()
   Print("Print(EMPTY_VALUE) = ",EMPTY_VALUE);
/* Ergebnis der Ausführung
   MetaQuotes-Demo 1889998: Hebel = 1:100
   MetaQuotes-Demo 1889998: eigene Mittel auf dem Konto = 22139.86 USD
   MetaQuotes-Demo 1889998: aktuelles Ergebniss nach offenen Postionen = +174.00 USD
   EURUSD: Wert eines Punktes  = 0.00001
   EURUSD: aktueller Spread in Punkten = 12
   DBL_MAX = 1.79769313486231570e+308
   EMPTY_VALUE = 1.79769313486231570e+308
   PrintFormat(EMPTY_VALUE) = 1.797693e+308
   Print(EMPTY_VALUE) = 1.797693134862316e+308
*/
  }
```

Sehen Sie auch

StringFormat, DoubleToString, Realtypen (double, float)

© 2000-2025, MetaQuotes Ltd.

ResetLastError

Stellt den Wert der vorbestimmten Variable _LastError auf Null ein.

```
void  ResetLastError();
```

Rückgabewert

Keinen Rückgabewert.

Hinweis

Es muss erwähnt werden, dass die Funktion GetLastError() setzt die Variable _LastError nicht auf Null. Gewöhnlich wird die Funktion ResetLastError() vor dem Aufruf der Funktion aufgerufen, nach der die Entstehung des Fehlers geprüft wird.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- den letzten Fehlercode vor Aufruf der Funktion zurücksetzen,
//--- andernfalls kann GetLastError() den vorherigen Fehlercode zurückgeben
   long lres=SymbolInfoInteger("123456",SYMBOL_DIGITS);
   PrintFormat("lres=%d  error=%d",lres,GetLastError());
   lres=SymbolInfoInteger(_Symbol,SYMBOL_DIGITS);
   PrintFormat("lres=%d  error=%d",lres,GetLastError());
   ResetLastError();
   lres=SymbolInfoInteger(_Symbol,SYMBOL_DIGITS);
   PrintFormat("lres=%d  error=%d",lres,GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

ResourceCreate

Erstellt die Ressource eines Bildes basierend auf dem Datensatz. Es gibt zwei Versionen der Funktion:

Erstellen einer Datei-basierte Ressource

```
bool  ResourceCreate(
   const string      resource_name,       // Ressourcenname
   const string      path                 // Ein relativer Pfad zur Datei
   );
```

Erstellen einer Ressource basierend auf einem Array von Pixeln

```
bool  ResourceCreate(
   const string      resource_name,       // Name der Ressource
   const uint&       data[],              // Datensatz im Form eines Arrays
   uint              img_width,           // Breite der erstellten Bildressource
   uint              img_height,          // Höhe der erstellten Bildressource
   uint              data_xoffset,        // Verschiebung der linken oberen Ecke des e
   uint              data_yoffset,        // vertikale Abwärtsverschiebung der linken
   uint              data_width,          // Gesamtbreite des Bildes basierend auf der
   ENUM_COLOR_FORMAT color_format         // Farbverarbeitungsmethode
   );
```

Parameter

resource_name

[in] Name der Ressource.

data[][]

[in] Eindimensionales oder zweidimensionales Array um ein vollständiges Bild zu erstellen.

img_width

[in] Breite der rechteckigen Bereich des Bildes in Pixel um in der Ressource in der Form von Bildern zu speichern. Es kann nicht mehr als *data_width sein.*

img_height

[in] Höhe der rechteckigen Bereich des Bildes in Pixel um in der Ressource in der Form von Bildern zu speichern.

data_xoffset

[in] Horizontaler Offset der rechteckigen Bereich des Bildes in Pixel nach rechts.

data_yoffset

[in] Vertikaler Offset der rechteckigen Bereich des Bildes in Pixel nach unten.

data_width

[in] Nur für eindimensionale Arrays erforderlich und impliziert die volle Breite des aus dem Datensatz zu erstellenden Bildes. Wenn *data_width*=0 ist, wird angenommen, dass er gleich *img_width angenommen*. Für zweidimensionale Arrays, wird dieser Parameter ignoriert, und es wird angenommen, dass er der zweite Dimension des Arrays *data[]* gleich ist.

color_format

[in] Farbe Verarbeitungsmethode aus der Enumeration ENUM_COLOR_FORMAT.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError(). Mögliche Fehler:

- 4015 - ERR_RESOURCE_NAME_DUPLICATED (die Namen der dynamischen und statischen Ressourcen sind gleich)
- 4016 - ERR_RESOURCE_NOT_FOUND (Ressource nicht gefunden)
- 4017 - ERR_RESOURCE_UNSUPPORTED_TYPE (Ressource-Typ wird nicht unterstützt)
- 4018 - ERR_RESOURCE_NAME_IS_TOO_LONG (Name der Ressource ist zu lang).

Hinweis

Wenn die zweite Version der Funktion aufgerufen wird, um die gleiche Ressource mit unterschiedlichen Parametern von Breite, Höhe und Offset zu erstellen, die neue Ressource nicht erstellt, sondern einfach eine vorhandene Ressource aktualisiert wird.

Die erste Version der Funktion erlaubt Ihnen, Bilder und Sounds aus Dateien zu laden, wird die zweite Version nur für die dynamische Erstellung von Bildern verwendet.

Die Bilder müssen im BMP-Format mit einer Farbtiefe von 24 oder 32 Bit sein, Sounds können nur im Format WAV sein. Die Größe der Ressource sollte nicht mehr als 16 Mb sein.

ENUM_COLOR_FORMAT

Bezeichnung	Beschreibung
COLOR_FORMAT_XRGB_NOALPHA	Bestandteil der Alpha-Kanal wird ignoriert
COLOR_FORMAT_ARGB_RAW	Die Farbkomponenten werden durch das Terminal nicht verarbeitet (müssen korrekt durch den Nutzer angegeben werden)
COLOR_FORMAT_ARGB_NORMALIZE	Die Farbkomponenten werden durch das Terminal verarbeitet

Beispiel:

```
//+------------------------------------------------------------------+
//| Grafische Ressourcendaten aktualisieren                          |
//+------------------------------------------------------------------+
void Update(const string res_name,const uint &pixel_data[],const uint width,const uint
  {
//--- Verlassen, wenn Dimensionen von 0 übergeben wurden
   if(width==0 || height==0)
      return;
//--- Ressourcendaten aktualisieren und den Chart neu zeichnen
   if(ResourceCreate(res_name,pixel_data,width,height,0,0,0,COLOR_FORMAT_ARGB_NORMALI
      ChartRedraw();
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklaration der Parameter der grafischen Ressource
   string rc_name="Resource";
   uint   rc_width=100;
   uint   rc_height=100;
   uint   rc_data[];
   uint   rc_size=rc_width*rc_height;

   ResetLastError();
//--- Festlegen der Größe des Pixelarrays
   if(ArrayResize(rc_data,rc_size)!=rc_size)
     {
      Print("ArrayResize() failed. Error code: ",GetLastError());
      return;
     }
//--- Füllen des Pixelarrays mit einer transparenten Farbe und Erstellen einer grafis
   ArrayInitialize(rc_data,0x00FFFFFF);
   if(!ResourceCreate(rc_name,rc_data,rc_width,rc_height,0,0,0,COLOR_FORMAT_ARGB_NORMA
     {
      Print("ResourceCreate() failed. Error code: ",GetLastError());
      return;
     }
   Print("Size of created recource array: ",rc_data.Size());

//--- Überprüfung der erstellten grafischen Ressource.
//--- Ermitteln der Zeit- und Preisdaten des aktuellen Balkens
   MqlTick tick={};
   if(!SymbolInfoTick(Symbol(),tick))
     {
      Print("SymbolInfoTick() failed. Error code: ",GetLastError());
      return;
     }
//--- Erstellen des Bitmap-Objekts unter Verwendung der Koordinaten des letzten Tickpr
   string obj_name="Bitmap";
   if(!ObjectCreate(0,obj_name,OBJ_BITMAP,0,tick.time,tick.bid))
     {
      Print("ObjectCreate() failed. Error code: ",GetLastError());
      return;
     }
//--- die Breite und Höhe des erstellten Bitmap-Objekts auf die Breite und Höhe der gr
//--- Setzen des Ankerpunkts des Objekts auf seine Mitte.
   ObjectSetInteger(0,obj_name,OBJPROP_XSIZE,rc_width);
   ObjectSetInteger(0,obj_name,OBJPROP_YSIZE,rc_height);
   ObjectSetInteger(0,obj_name,OBJPROP_ANCHOR,ANCHOR_CENTER);
//--- Angabe der zuvor erstellten grafischen Ressource für das Bitmap-Objekt als Bildq
//--- in diesem Fall müssen wir, um den Namen der verwendeten grafischen Ressource anz
   ObjectSetString(0,obj_name,OBJPROP_BMPFILE,"::"+rc_name);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Einstellen der Farbe DodgerBlue mit einer Transparenz von 200
   uint clr=ColorToARGB(clrDodgerBlue,200);
//--- Füllen des gesamten Pixelarrays der grafischen Ressource mit der eingestellten F
   ArrayInitialize(rc_data,clr);
//--- Aktualisierung der grafischen Ressourcendaten
   Update(rc_name,rc_data,rc_width,rc_height,true);
  }
```

Siehe auch

Ressourcen, ObjectCreate(), ObjectSetString(), OBJPROP_BMPFILE

© 2000-2025, MetaQuotes Ltd.

ResourceFree

Löscht dynamisch erstellte Ressource (gibt den von der Ressource belegten Speicher frei).

```
bool  ResourceFree(
   const string  resource_name      // Ressource-Name
   );
```

Optionen

resource_name

[in] Name der Ressource, muss beginnen mit "::".

Rückgabewert

true - wenn erfolgreich, andernfalls false. Um Informationen über den Fehler zu erhalten, rufen Sie GetLastError() an.

Hinweis

Funktion ResourceFree() erlaubt es dem Entwickler von MQL5-Programm Speicherbedarf beim Arbeiten mit Ressourcen zu kontrollieren. Grafische Objekte, die an aus dem Speicher gelöschten Ressource gebunden sind, werden richtig nach ihrer Löschung angezeigt werden. Aber die neu erstellten graphischen Objekte (OBJ_BITMAP und OBJ_BITMAP_LABEL) können nicht die gelöschte Ressource nutzen.

Die Funktion löscht nur die dynamischen Ressourcen, die durch das Programm erstellt werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklaration der Parameter der grafischen Ressource
   string rc_name="Resource";
   uint   rc_width=100;
   uint   rc_height=100;
   uint   rc_data[];
   uint   rc_size=rc_width*rc_height;

   ResetLastError();
//--- Festlegen der Größe des Pixelarrays
   if(ArrayResize(rc_data,rc_size)!=rc_size)
     {
      Print("ArrayResize() failed. Error code: ",GetLastError());
      return;
     }
//--- Füllen des Pixelarrays mit einer transparenten Farbe und Erstellen einer grafis
   ArrayInitialize(rc_data,0x00FFFFFF);
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!ResourceCreate(rc_name,rc_data,rc_width,rc_height,0,0,0,COLOR_FORMAT_ARGB_NORMA
     {
      Print("ResourceCreate() failed. Error code: ",GetLastError());
      return;
     }
   Print("Size of created recource array: ",rc_data.Size());

//--- Überprüfung der erstellten grafischen Ressource.
//--- Ermitteln der Zeit- und Preisdaten des aktuellen Balkens
   MqlTick tick={};
   if(!SymbolInfoTick(Symbol(),tick))
     {
      Print("SymbolInfoTick() failed. Error code: ",GetLastError());
      return;
     }
//--- Erstellen des Bitmap-Objekts unter Verwendung der Koordinaten des letzten Tickpr
   string obj_name="Bitmap";
   if(!ObjectCreate(0,obj_name,OBJ_BITMAP,0,tick.time,tick.bid))
     {
      Print("ObjectCreate() failed. Error code: ",GetLastError());
      return;
     }
//--- die Breite und Höhe des erstellten Bitmap-Objekts auf die Breite und Höhe der gr
//--- Setzen des Ankerpunkts des Objekts auf seine Mitte.
   ObjectSetInteger(0,obj_name,OBJPROP_XSIZE,rc_width);
   ObjectSetInteger(0,obj_name,OBJPROP_YSIZE,rc_height);
   ObjectSetInteger(0,obj_name,OBJPROP_ANCHOR,ANCHOR_CENTER);
//--- Angabe der zuvor erstellten grafischen Ressource für das Bitmap-Objekt als Bildq
//--- in diesem Fall müssen wir, um den Namen der verwendeten grafischen Ressource anz
   ObjectSetString(0,obj_name,OBJPROP_BMPFILE,"::"+rc_name);

//--- Einstellen der Farbe DodgerBlue mit einer Transparenz von 200
   uint clr=ColorToARGB(clrDodgerBlue,200);
//--- Füllen des gesamten Pixelarrays der grafischen Ressource mit der eingestellten F
   ArrayInitialize(rc_data,clr);
//--- Aktualisierung der grafischen Ressourcendaten
   Update(rc_name,rc_data,rc_width,rc_height,true);

//--- drei Sekunden warten und dann die Bildfarbe ändern
   Print("Wait 3 seconds before changing color");
   Sleep(3000);
//--- Einstellen der Farbe OrangeRed mit einer Transparenz von 200
   Print("Change color");
   clr=ColorToARGB(clrOrangeRed,200);
//--- Füllen des gesamten Pixelarrays der grafischen Ressource mit der eingestellten F
   ArrayInitialize(rc_data,clr);
//--- Aktualisierung der grafischen Ressourcendaten
   Update(rc_name,rc_data,rc_width,rc_height,true);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- drei Sekunden warten und dann grafische Ressource freigeben
   Print("Wait 3 seconds before ResourceFree()");
   Sleep(3000);
   bool res=ResourceFree("::"+rc_name);
   Print("ResourceFree: ",res);

//--- Versuch, die Farbe nach Freigabe der Ressource zu ändern
   Print("Trying to change color to GreenYellow after ResourceFree()");
//--- Einstellen der Farbe GreenYellow mit einer Transparenz von 200
   clr=ColorToARGB(clrGreenYellow,200);
//--- Füllen des gesamten Pixelarrays der grafischen Ressource mit der eingestellten F
   ArrayInitialize(rc_data,clr);
//--- Aktualisierung der grafischen Ressourcendaten (das Bild bleibt erhalten, aber di
   Update(rc_name,rc_data,rc_width,rc_height,true);
   Print("The color has not changed because the resource has been released");

//--- drei Sekunden warten und dann das Bitmap-Objekt löschen
   Print("Wait 3 seconds before deleting the Bitmap object");
   Sleep(3000);
   Print("Delete Bitmap object");
   ObjectDelete(0,obj_name);
  }
//+------------------------------------------------------------------+
//| Grafische Ressourcendaten aktualisieren                          |
//+------------------------------------------------------------------+
void Update(const string res_name,const uint &pixel_data[],const uint width,const uint
  {
//--- Verlassen, wenn Dimensionen von 0 übergeben wurden
   if(width==0 || height==0)
      return;
//--- Ressourcendaten aktualisieren und den Chart neu zeichnen
   if(ResourceCreate(res_name,pixel_data,width,height,0,0,0,COLOR_FORMAT_ARGB_NORMALI
      ChartRedraw();
  }
```

Siehe auch

Ressourcen, ObjectCreate(), PlaySound(), ObjectSetString(), OBJPROP_BMPFILE

© 2000-2025, MetaQuotes Ltd.

ResourceReadImage

Liest Daten einer Grafik-Ressource, die von der Funktion ResourceCreate() erstellt war oder in EX5-Datei beim Kompilieren gespeichert war.

```
bool  ResourceReadImage(
   const string      resource_name,       // Name der Grafik-Ressource zum Lesen
   uint&             data[],              // Array, um die Daten aus der Ressource zu
   uint&             width,               // Um die Breite des Bildes in der Ressource
   uint&             height,              // Um die Bildhöhe in der Ressource zu erhal
   );
```

Optionen

resource_name

[in] Name der grafischen Ressource, die das Bild enthält. Um auf die eigenen Ressourcen zuzugreifen, wird der Name in der Kurzform von "::resourcename" angegeben. Wenn Sie die Ressource aus der kompilierten EX5-Datei laden müssen, muss der volle Name mit dem relativen Pfad zu dem Ordner MQL5, den Dateinamen und Ressource-Name sein - "path\\filename.ex5::resourcename".

data[][]

[in] Ein eindimensionales oder zweidimensionales Array in dem die Daten eines graphischen Ressource erhalten werden.

img_width

[out] Bildbreite der Ressource in Pixels.

img_height

[out] Bildhöhe der Ressource in Pixels.

Rückgabewert

true - wenn erfolgreich, andernfalls false. Um Informationen über den Fehler zu erhalten, rufen Sie GetLastError() an.

Hinweis

Wenn das Array *data[]* dann für die Erstellung einer eine Grafik-Ressource verwendet werden soll, sollten die Farbe Format COLOR_FORMAT_ARGB_NORMALIZE oder COLOR_FORMAT_XRGB_NOALPHA verwendet werden.

Wenn das Array *data[]* zweidimensional ist, und die zweite Dimension kleiner ist als X (width) der grafischen Ressource, gibt die Funktion ResourceReadImage() false zurück und das Lesen wird nicht vorgenommen. Ist die Ressource jedoch vorhanden, wird die tatsächliche Bildgröße in den Parametern width und height zurückgegeben. Dadurch kann ein weiterer Versuch unternommen werden, Daten von der Ressource zu erhalten.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklaration der Parameter der grafischen Ressource
   string rc_name="Resource";
   uint   rc_width=100;
   uint   rc_height=100;
   uint   rc_data[];
   uint   rc_size=rc_width*rc_height;

   ResetLastError();
//--- Festlegen der Größe des Pixelarrays
   if(ArrayResize(rc_data,rc_size)!=rc_size)
     {
      Print("ArrayResize() failed. Error code: ",GetLastError());
      return;
     }
//--- Füllen des Pixelarrays mit einer transparenten Farbe und Erstellen einer grafis
   ArrayInitialize(rc_data,0x00FFFFFF);
   if(!ResourceCreate(rc_name,rc_data,rc_width,rc_height,0,0,0,COLOR_FORMAT_ARGB_NORMA
     {
      Print("ResourceCreate() failed. Error code: ",GetLastError());
      return;
     }
   Print("Size of created recource array: ",rc_data.Size());

//--- Überprüfung der erstellten grafischen Ressource.
//--- Ermitteln der Zeit- und Preisdaten des aktuellen Balkens
   MqlTick tick={};
   if(!SymbolInfoTick(Symbol(),tick))
     {
      Print("SymbolInfoTick() failed. Error code: ",GetLastError());
      return;
     }
//--- Erstellen des Bitmap-Objekts unter Verwendung der Koordinaten des letzten Tickpr
   string obj_name="Bitmap";
   if(!ObjectCreate(0,obj_name,OBJ_BITMAP,0,tick.time,tick.bid))
     {
      Print("ObjectCreate() failed. Error code: ",GetLastError());
      return;
     }
//--- die Breite und Höhe des erstellten Bitmap-Objekts auf die Breite und Höhe der gr
//--- Setzen des Ankerpunkts des Objekts auf seine Mitte.
   ObjectSetInteger(0,obj_name,OBJPROP_XSIZE,rc_width);
   ObjectSetInteger(0,obj_name,OBJPROP_YSIZE,rc_height);
   ObjectSetInteger(0,obj_name,OBJPROP_ANCHOR,ANCHOR_CENTER);
//--- Angabe der zuvor erstellten grafischen Ressource für das Bitmap-Objekt als Bildq
//--- In diesem Fall müssen wir, um den Namen der verwendeten grafischen Ressource anz
   ObjectSetString(0,obj_name,OBJPROP_BMPFILE,"::"+rc_name);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Einstellen der Farbe DodgerBlue mit einer Transparenz von 200
   uint clr=ColorToARGB(clrDodgerBlue,200);
//--- Füllen des gesamten Pixelarrays der grafischen Ressource mit der eingestellten F
   ArrayInitialize(rc_data,clr);
//--- Aktualisierung der grafischen Ressourcendaten
   Update(rc_name,rc_data,rc_width,rc_height,true);

//--- 3 Sekunden warten, bevor die grafischen Ressourcendaten gelesen werden
   Print("Wait 3 seconds before ResourceReadImage()");
   Sleep(3000);
//--- das Bild aus der Ressource in ein neues Array von Pixeln einlesen
   uint rc_data_copy[];
   uint w=0,h=0;
   ResetLastError();
   if(!ResourceReadImage("::"+rc_name,rc_data_copy,w,h))
     {
      Print("ResourceReadImage() failed. Error code: ",GetLastError());
      return;
     }

//--- Einstellen der Farbe OrangeRed mit einer Transparenz von 200
   clr=ColorToARGB(clrOrangeRed,200);
//--- Füllen der gesamten Pixelmatrix der grafischen Ressource mit der eingestellten F
   ArrayInitialize(rc_data_copy,clr);
   if(!ResourceCreate(rc_name+"Copy",rc_data_copy,rc_width,rc_height,0,0,0,COLOR_FORMA
     {
      Print("New ResourceCreate() failed. Error code: ",GetLastError());
      return;
     }
   Print("Size of created new recource array: ",rc_data_copy.Size());

//--- Erstellen des Objekts "Graphical label" unter Verwendung der Koordinaten des let
   string obj_name2="BitmapLabel";
   if(!ObjectCreate(0,obj_name2,OBJ_BITMAP_LABEL,0,0,0))
     {
      Print("ObjectCreate() failed. Error code: ",GetLastError());
      return;
     }
//--- Ermitteln der Bildschirmkoordinaten anhand des zuvor erhaltenen Preises und der
   int x=0,y=0;
   if(!ChartTimePriceToXY(0,0,tick.time,tick.bid,x,y))
     {
      Print("New ChartTimePriceToXY() failed. Error code: ",GetLastError());
      return;
     }
//--- Setzen der Breite und Höhe des erstellten grafischen Label-Objekts mit der Breit
//--- Setzen des Ankerpunkts des Objekts auf seine Mitte.
   ObjectSetInteger(0,obj_name2,OBJPROP_XSIZE,rc_width);
```

© 2000-2025, MetaQuotes Ltd.

```
   ObjectSetInteger(0,obj_name2,OBJPROP_YSIZE,rc_height);
   ObjectSetInteger(0,obj_name2,OBJPROP_ANCHOR,ANCHOR_LEFT_UPPER);
   ObjectSetInteger(0,obj_name2,OBJPROP_XDISTANCE,x);
   ObjectSetInteger(0,obj_name2,OBJPROP_YDISTANCE,y);
//--- kopierte grafische Ressource als Bilddatei für das grafische Label-Objekt festle
//--- in diesem Fall müssen wir, um den Namen der verwendeten grafischen Ressource anz
   ObjectSetString(0,obj_name2,OBJPROP_BMPFILE,"::"+rc_name+"Copy");

//--- Ändern der Farbe des neuen grafischen Etikettenobjekts
   Print("Wait 3 seconds before changing color to GreenYellow");
   Sleep(3000);
//--- Einstellen der Farbe GreenYellow mit einer Transparenz von 200
   clr=ColorToARGB(clrGreenYellow,200);
//--- Füllen des gesamten Pixelarrays der neuen grafischen Ressource mit der eingestel
   ArrayInitialize(rc_data_copy,clr);
//--- Aktualisierung der grafischen Ressourcendaten
   Update(rc_name+"Copy",rc_data_copy,rc_width,rc_height,true);

//--- drei Sekunden warten und dann Ressourcen und beide Objekte löschen
   Print("Wait 3 seconds before deleting both objects");
   Sleep(3000);
   Print("Deleting Resource and all Bitmap objects");
   ResourceFree("::"+rc_name);
   ResourceFree("::"+rc_name+"Copy");
   ObjectDelete(0,obj_name);
   ObjectDelete(0,obj_name2);
  }
//+------------------------------------------------------------------+
//| Grafische Ressourcendaten aktualisieren                          |
//+------------------------------------------------------------------+
void Update(const string res_name,const uint &pixel_data[],const uint width,const uint
  {
//--- Verlassen, wenn Dimensionen von 0 übergeben wurden
   if(width==0 || height==0)
      return;
//--- Ressourcendaten aktualisieren und den Chart neu zeichnen
   if(ResourceCreate(res_name,pixel_data,width,height,0,0,0,COLOR_FORMAT_ARGB_NORMALIZ
      ChartRedraw();
  }
```

Siehe auch

Ressourcen, ObjectCreate(), ObjectSetString(), OBJPROP_BMPFILE

© 2000-2025, MetaQuotes Ltd.

ResourceSave

Speichert die Ressource in eine angegebene Datei.

```
bool  ResourceSave(
   const string  resource_name      // Ressourcenname
   const string  file_name          // Dateiname
   );
```

Parameter

resource_name

[in] Ressourcenname muss mit "::" anfangen.

file_name

[in] Dateiname relativ zu MQL5\Files.

Rückgabewert

Gibt true beim Erfolgsfall zurück, anderenfalls false. Für Erhaltung des Fehlerkodes muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Funktion überschreibt immer die Datei und erstellt, falls erforderlich, alle dazwischen liegenden Unterverzeichnisse im Dateinamen, falls sie nicht existieren.

Beispiel:

```
//--- die Parameter der grafische Ressource
string ExtResName="Resource";
int    ExtResWidth=100;
int    ExtResHeight=100;
uint   ExtResData[];
int    ExtResSize=ExtResWidth*ExtResHeight;

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   ResetLastError();
//--- Festlegen der Größe des Pixelarrays
   if(ArrayResize(ExtResData,ExtResSize)!=ExtResSize)
     {
      Print("ArrayResize() failed. Error code: ",GetLastError());
      return;
     }
//--- Füllen des Pixelarrays mit einer transparenten Farbe und Erstellen einer grafis
   ArrayInitialize(ExtResData,0x00FFFFFF);
   if(!ResourceCreate(ExtResName,ExtResData,ExtResWidth,ExtResHeight,0,0,0,COLOR_FORM
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("ResourceCreate() failed. Error code: ",GetLastError());
       return;
      }
   Print("Size of created recource array: ",ExtResData.Size());

//--- Überprüfung der erstellten grafischen Ressource.
//--- Ermitteln der Zeit- und Preisdaten des aktuellen Balkens
   MqlTick tick={};
   if(!SymbolInfoTick(Symbol(),tick))
      {
       Print("SymbolInfoTick() failed. Error code: ",GetLastError());
       return;
      }
//--- Erstellen des Bitmap-Objekts unter Verwendung der Koordinaten des letzten Tickpr
   string obj_name="Bitmap";
   if(!ObjectCreate(0,obj_name,OBJ_BITMAP,0,tick.time,tick.bid))
      {
       Print("ObjectCreate() failed. Error code: ",GetLastError());
       return;
      }
//--- die Breite und Höhe des erstellten Bitmap-Objekts auf die Breite und Höhe der gr
//--- Setzen des Ankerpunkts des Objekts auf seine Mitte.
   ObjectSetInteger(0,obj_name,OBJPROP_XSIZE,ExtResWidth);
   ObjectSetInteger(0,obj_name,OBJPROP_YSIZE,ExtResHeight);
   ObjectSetInteger(0,obj_name,OBJPROP_ANCHOR,ANCHOR_CENTER);
//--- Angabe der zuvor erstellten grafischen Ressource für das Bitmap-Objekt als Bildq
//--- in diesem Fall müssen wir, um den Namen der verwendeten grafischen Ressource anz
   ObjectSetString(0,obj_name,OBJPROP_BMPFILE,"::"+ExtResName);

//--- Einstellen der Farbe GreenYellow mit einer Transparenz von 200
   uint clr=ColorToARGB(clrGreenYellow,200);
//--- Füllen des gesamten Pixelarrays der grafischen Ressource mit der eingestellten F
   ArrayInitialize(ExtResData,clr);
//--- das Gitter mit der Farbe DodgerBlue zeichnen
   for(int x=0;x<ExtResWidth;x+=9)
      LineVertical(x,0,ExtResHeight,ColorToARGB(clrDodgerBlue,200));
   for(int y=0;y<ExtResHeight;y+=9)
      LineHorizontal(0,ExtResWidth,y,ColorToARGB(clrDodgerBlue,200));
//--- Aktualisierung der grafischen Ressourcendaten
   Update(ExtResName,ExtResData,ExtResWidth,ExtResHeight,true);

//--- 3 Sekunden warten, bevor die Ressource und das grafische Objekt gelöscht wird
   Print("Wait 3 seconds before deleting the resource and graphic object");
   Sleep(3000);

//--- Speichern der Ressource in der Datei
   ResetLastError();
   if(!ResourceSave("::"+ExtResName,"ResourceSave\\"+ExtResName+".bmp"))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("ResourceSave() failed. Error code: ",GetLastError());
       return;
      }
//--- die Ressource und das grafische Objektbild löschen
   if(!ResourceFree("::"+ExtResName))
      {
       Print("ResourceFree() failed. Error code: ",GetLastError());
       return;
      }
   if(!ObjectDelete(0,obj_name))
      {
       Print("ObjectDelete() failed. Error code: ",GetLastError());
       return;
      }

//--- jetzt wird das Objekt "Graphical label" unter Verwendung der Koordinaten des Pre
//--- Abrufen der Bildschirmkoordinaten unter Verwendung des zuvor erhaltenen Preises
   int x=0,y=0;
   if(!ChartTimePriceToXY(0,0,tick.time,tick.bid,x,y))
      {
       Print("ChartTimePriceToXY() failed. Error code: ",GetLastError());
       return;
      }
   obj_name="BitmapLabel";
   if(!ObjectCreate(0,obj_name,OBJ_BITMAP_LABEL,0,0,0))
      {
       Print("ObjectCreate() failed. Error code: ",GetLastError());
       return;
      }
//--- Setzen des Ankerpunkts des Objekts auf seine Mitte.
   ObjectSetInteger(0,obj_name,OBJPROP_ANCHOR,ANCHOR_CENTER);
//--- bestimmen der Koordinaten des grafischen Label-Objekt des Preises und der Zeit d
   ObjectSetInteger(0,obj_name,OBJPROP_XDISTANCE,x-ExtResWidth);
   ObjectSetInteger(0,obj_name,OBJPROP_YDISTANCE,y);
//--- die Ressource, die zuvor als bmp-Bild gespeichert wurde, für das grafische Etike
   ObjectSetString(0,obj_name,OBJPROP_BMPFILE,"\\Files\\ResourceSave\\"+ExtResName+".b
//--- den Chart aktualisieren
   ChartRedraw();

//--- 3 Sekunden warten, bevor das grafische Label-Objekt gelöscht wird
   Print("Wait 3 seconds before deleting the new graphic object");
   Sleep(3000);
   if(!ObjectDelete(0,obj_name))
      Print("ObjectDelete() failed. Error code: ",GetLastError());
  }
//+------------------------------------------------------------------+
//| Grafische Ressourcendaten aktualisieren                          |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void Update(const string res_name,const uint &pixel_data[],const uint width,const uint
  {
//--- Verlassen, wenn Dimensionen von 0 übergeben wurden
   if(width==0 || height==0)
      return;
//--- Ressourcendaten aktualisieren und den Chart neu zeichnen
   if(ResourceCreate(res_name,pixel_data,width,height,0,0,0,COLOR_FORMAT_ARGB_NORMALIZ
      ChartRedraw();
  }
//+------------------------------------------------------------------+
//| Eine Vertikale ziehen                                            |
//+------------------------------------------------------------------+
void LineVertical(int x,int y1,int y2,const uint clr)
  {
   int tmp;
//--- nach Y sortieren
   if(y1>y2)
     {
      tmp=y1;
      y1 =y2;
      y2 =tmp;
     }
//--- Linie außerhalb der Bildgrenzen
   if(y2<0 || y1>=ExtResHeight || x<0 || x>=ExtResWidth)
      return;
//--- innerhalb der Bildgrenzen bleiben
   if(y1<0)
      y1=0;
   if(y2>=ExtResHeight)
      y2=ExtResHeight-1;
//--- zeichnen der Linie
   int index=y1*ExtResWidth+x;
   for(int i=y1; i<=y2; i++,index+=ExtResWidth)
      ExtResData[index]=clr;
  }
//+------------------------------------------------------------------+
//| Eine Horizontale ziehen                                          |
//+------------------------------------------------------------------+
void LineHorizontal(int x1,int x2,int y,const uint clr)
  {
   int tmp;
//--- nach X sortieren
   if(x1>x2)
     {
      tmp=x1;
      x1 =x2;
      x2 =tmp;
     }
//--- Linie außerhalb der Bildgrenzen
```

© 2000-2025, MetaQuotes Ltd.

```
   if(x2<0 || x1>=ExtResWidth || y<0 || y>=ExtResHeight)
      return;
//--- innerhalb der Bildgrenzen bleiben
   if(x1<0)
      x1=0;
   if(x2>=ExtResWidth)
      x2=ExtResWidth-1;
//--- zeichnen der Linie
   ArrayFill(ExtResData,y*ExtResWidth+x1,(x2-x1)+1,clr);
  }
```

Siehe auch

Ressourcen, ObjectCreate(), PlaySound(), ObjectSetString(), OBJPROP_BMPFILE

© 2000-2025, MetaQuotes Ltd.

SetReturnError

Setzt den Fehlercode, den ein Prozess des Terminals bei seiner Beendigung zurückgibt.

```
void  SetReturnError(
   int ret_code      // Klienten-Terminal Fehlercode
   );
```

Parameter

ret_code

[in] Der Fehlercode, der vom Prozess des Klienten-Terminals bei seiner Beendigung zurückgegeben wird.

Rückgabewert

Kein Rückgabewert.

Hinweis

Das Setzen des Rückgabewertes *ret_code* mit der Funktion SetReturnError() hilft, die Gründe für die Beendigung des programmtechnischen Betriebs zu analysieren, wenn das Terminal über die Befehlszeile gestartet worden war.

Im Gegensatz zu TerminalClose() beendet die Funktion SetReturnError() den Terminalbetrieb nicht. Stattdessen wird nur der Fehlercode gesetzt, den das Terminal nach seiner Beendigung zurückgibt.

Wenn die Funktion SetReturnError() mehrfach und/oder aus verschiedenen MQL5-Programmen aufgerufen wird, gibt das Terminal den letzten eingestellten Rückgabewert zurück.

Der eingestellte Fehlercode wird nach Abschluss des Terminalprozesses zurückgegeben, mit Ausnahme der folgenden Fälle:

- Ein kritischer Fehler ist während der Ausführung aufgetreten;
- Die Funktion TerminalClose(int ret_code), die den Befehl zum Abschluss des Terminalbetriebs mit einem bestimmten Fehlercode ausgibt, wurde aufgerufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   matrix matrix_a =
     {
        {-3.474589, 1.106384, -9.091977,-3.925227 },
        {-5.522139, 2.366887,-15.162351,-6.357512 },
        { 8.394926,-2.960067, 22.292115, 9.524129 },
        { 7.803242,-2.080287, 19.217706, 8.186645 }
     };
   matrix matrix_l(4,4);
   matrix matrix_u(4,4);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- LU Zerlegung
   matrix_a.LU(matrix_l,matrix_u);

//--- Test, ob A = L * U
   matrix matrix_lu=matrix_l.MatMul(matrix_u);
   int    compare_errors=(int)matrix_a.Compare(matrix_lu,1e-29);
   Print("MatrixCompare errors=",compare_errors);

//--- nach Beendigung gibt das Client-Terminal die Anzahl der Fehler beim Vergleich de
   SetReturnError(compare_errors);
  }
```

Siehe auch

Programmausführung, Ausführungsfehler,Deinitialisierungsgruende , TerminalClose

© 2000-2025, MetaQuotes Ltd.

SetUserError

Stellt die Vorbestimmte Variable _LastError in Wert ERR_USER_ERROR_FIRST + user_error

```
void  SetUserError(
   ushort user_error,    // Fehlernummer
   );
```

Parameter

user_error

[in] Nummer des Fehlers, festgelegt vom Benutzer.

Rückgabewert

Keinen Rückgabewert.

Hinweis

Nach Einstellung des Fehlers durch die Funktion SetUserError(user_error), gibt die Funktion GetLastError()den den Wert, gleich ERR_USER_ERROR_FIRST + user_error.

Beispiel:

```
void OnStart()
  {
//--- legen wir Fehlernummer 65537=(ERR_USER_ERROR_FIRST +1)fest
   SetUserError(1);
//--- erhalten wir Kode des letzten Fehlers
   Print("GetLastError = ",GetLastError());
/*
   Ergebnis
   GetLastError = 65537
*/
  }
```

© 2000-2025, MetaQuotes Ltd.

Sleep

Funktion verzögert die Ausführung des laufenden Experten oder des Scripts für bestimmtes Zeitintervall.

```
void  Sleep(
   int  milliseconds      // Intervall
   );
```

Parameter

milliseconds

[in] Verzögerungsintervall in Milisekunden.

Rückgabewert

Keinen Rückgabewert.

Hinweis

Die Funktion Sleep() kann aus Nutzerindikatoren nicht aufgerufen werden, denn Intervalle werden im Interface Thread ausgeführt und müssen es nicht hemmen. Die Funktion verfügt über eine integrierte Prüfung der Stopp-Flag des Experten alle 0.1 Sekunden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige eines Countdowns von 10 bis 1 in den Kommentaren des Charts
   for(int i=10; i>0 && !IsStopped(); i--)
     {
      Comment(StringFormat("Wait %u seconds",i));
      Sleep(1000);
     }
//--- Schreiben eines Textes in den "ankommenden" Kommentar, der den Zweck des Skripts
   string text="This was a test showing how the Sleep() function works";
   string mess="";
   for(int i=0; i<(int)text.Length(); i++)
     {
      mess+=ShortToString(text.GetChar(i));
      Sleep(100);
      Comment(mess);
     }
//--- Auf Wiedersehen...
   Sleep(1000);
   for(int i=0; i<6; i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      mess=(i % 2 == 0 ? "" : "  Bye!");
      Comment(mess);
      Sleep(300);
     }
//--- Löschen des Textes vom Chart
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

TerminalClose

Sendet dem Teminal den Befehl, Operation zu beenden.

```
bool  TerminalClose(
   int ret_code      // Abschlusskode des Client-Terminals
   );
```

Parameter

ret_code

[in] Rückgabewert, der vom Prozess des Client-Terminals beim Operationsabschluss zurückgegeben wird.

Rückgabewert

Gibt true beim Erfolg zurück, sonst false.

Hinweis

Funktion TerminalClose() stoppt die Terminalsoperation nicht sofort, sie sendet dem Terminal den Befehl, Arbeit abzuschliessen.

Im Ratgeberkode, der TerminalClose() aufgerufen hat,muessen alle Vorbereitungen gemacht werden, um Operation sofort zu beenden (ZB alle früher geoeffnete Dateien muessen routinemäßig gesclossen werden). Sofort nach Aufruf dieser Funktion muss die Anweisung return .

Parameter *ret_code* ermöglicht den notwendigen Rückgabekode anzugeben, um Gruende des Programmabschlusses der Terminalsoperation zunanalysieren, wenn das Teminal aus der Befehlszeile gestartet wird.

Beispiel:

```
//--- input parameters
input int  tiks_before=500; // Zahl der Ticks bis Abschluss
input int  pips_to_go=15;   // Distanz in Pips
input int  seconds_st=50;   // Zahl der Sekunden gegeben dem Experten
//--- globals
datetime   launch_time;
int        tick_counter=0;
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Print(__FUNCTION__," reason code = ",reason);
   Comment("");
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   static double first_bid=0.0;
   MqlTick       tick;
   double        distance;
//---
   SymbolInfoTick(_Symbol,tick);
   tick_counter++;
   if(first_bid==0.0)
     {
      launch_time=tick.time;
      first_bid=tick.bid;
      Print("first_bid = ",first_bid);
      return;
     }
//--- Preisgang in Punkten
   distance=(tick.bid-first_bid)/_Point;
//--- Gebenwir die Meldung aus, um Arbeit des Experten zu befolgen
   string comm="Vom Start:\r\n\x25CF vergang es Sekunden: "+
               IntegerToString(tick.time-launch_time)+" ;"+
               "\r\n\x25CF erhaltene Ticks: "+(string)tick_counter+" ;"+
               "\r\n\x25CF Preis ging in Punkten: "+StringFormat("%G",distance);
   Comment(comm);
//--- Abschnitt für Pruefung der Bedingungen für Terminalabschluss
   if(tick_counter>=tiks_before)
      TerminalClose(0);    // Ende nach Tickcounter
   if(distance>pips_to_go)
      TerminalClose(1);    // gehen oben um Pipszahl  pips_to_go
   if(distance<-pips_to_go)
      TerminalClose(-1);   // gehen unten um Pipszahl pips_to_go
   if(tick.time-launch_time>seconds_st)
      TerminalClose(100);  // Beenden laut Timeout
//---
  }
```

Sehen Sie auch

Programmausführung, Ausführungsfehler, Deinitialisierungsguende

© 2000-2025, MetaQuotes Ltd.

TesterHideIndicators

Aktiviert den Modus zum Anzeigen/Ausblenden von Indikatoren, die in einem Expert Advisor verwendet werden. Die Funktion dient zur Verwaltung der Sichtbarkeit der verwendeten Indikatoren nur beim Testen.

```
void  TesterHideIndicators(
   bool      hide     // Flag
   );
```

Parameter

hide

[in] Das Flag zum Ausblenden der Indikatoren beim Testen. Geben Sie true an, wenn man die erstellten Indikatoren ausblenden muss, andernfalls false.

Rückgabewert

Kein

Hinweis

Standardmäßig werden alle Indikatoren, die im zu testenden Expert Advisor erstellt wurden, im Chart des visuellen Testens angezeigt. Darüber hinaus werden diese Indikatoren im Chart angezeigt, der sich automatisch öffnet, sobald das Testen abgeschlossen ist. Die Funktion TesterHideIndicators() erlaubt es dem Entwickler, die Anzeige der verwendeten Indikatoren im Codes des Expert Advisors zu deaktivieren.

Um die Anzeige eines verwendeten Indikators beim Testen eines Expert Advisors zu deaktivieren, muss man vor der Erstellung seines Handles TesterHideIndicators() mit dem Parameterfalse aufrufen. Alle Indikatoren, die danach erstellt werden, werden mit dem Flag zum Ausblenden markiert. Indikatoren, die mit dem Flag zum Ausblenden markiert wurden, werden während eines visuellen Tests und auf dem Chart, der sich nach dem Ende des Testens automatisch öffnet, nicht angezeigt.

Um den Modus zu deaktivieren, in welchem die erstellten Indikatoren ausgeblendet werden, muss man wieder TesterHideIndicators() aufrufen, aber mit dem Parameter true. Im Chart des Testens können nur die Indikatoren angezeigt werden, die im zu testenden Expert Advisor erstellt werden. Die Regel gilt nur für die Fälle, wenn es kein einziges Template im Ordner <data_folder>MQL5\Profiles\Templates gibt.

Wenn ein spezielles Template <expert_name>.tpl im Ordner <data_folder>MQL5\Profiles\Templates vorhanden ist, werden nur Indikatoren aus diesem Template während eines visuellen Tests und im Testchart angezeigt. In diesem Fall werden die Indikatoren, die im zu testenden Expert Advisor verwendet werden, nicht angezeigt. Auch wenn die Funktion TesterHideIndicators() mit dem Parameter true im Code des Expert Advisors aufgerufen wurde.

Wenn es kein spezielles Template <expert_name>.tpl im Ordner <data_folder>MQL5\Profiles\Templates gibt, werden beim visuellen Testen und im Chart des Testens Indikatoren aus dem Template tester.tpl und Indikatoren aus dem Expert Advisor angezeigt, die nicht durch die Funktion TesterHideIndicators() deaktiviert wurden. Wenn das Template tester.tpl nicht vorhanden ist, werden Indikatoren aus dem Template default.tpl verwendet.

© 2000-2025, MetaQuotes Ltd.

Wenn der Strategietester kein passendes Template findet (<expert_name>.tpl, tester.tpl oder default.tpl), verwaltet die Funktion TesterHideIndicators() die Anzeige der im Expert Advisor verwendeten Indikatoren.

Beispiel:

```
bool CSampleExpert::InitIndicators(void)
 {
   TesterHideIndicators(true);
//--- create MACD indicator
   if(m_handle_macd==INVALID_HANDLE)
      if((m_handle_macd=iMACD(NULL,0,12,26,9,PRICE_CLOSE))==INVALID_HANDLE)
       {
         printf("Error creating MACD indicator");
         return(false);
        }
   TesterHideIndicators(false);
//--- create EMA indicator and add it to collection
   if(m_handle_ema==INVALID_HANDLE)
      if((m_handle_ema=iMA(NULL,0,InpMATrendPeriod,0,MODE_EMA,PRICE_CLOSE))==INVALID_H
       {
         printf("Error creating EMA indicator");
         return(false);
        }
//--- succeed
   return(true);
  }
```

Siehe auch

IndicatorRelease

© 2000-2025, MetaQuotes Ltd.

TesterStatistics

Die Funktion gibt den Wert des statistischen Parameters zurück, der aus den Testergebnissen berechnet wurde.

```
double  TesterStatistics(
   ENUM_STATISTICS statistic_id      // ID
   );
```

Parameter

statistic_id

[in] Identifikator des statistischen Parameters aus der Enumeration ENUM_STATISTICS.

Rückgabewert

Der Wert des statistischen Parameters des Testergebnisses.

Hinweis

Die Funktion kann innerhalb von OnTester() oder OnDeinit() im Tester aufgerufen werden. In anderen Fällen ist das Ergebnis undefiniert.

Beispiel:

```
// Der EA, basierend auf der Standarddatei "MACD Sample.mq5",
// zeigt das Ergebnis von TesterStatistics() in der Ereignisbehandlung des Testers
#define MACD_MAGIC 1234502
//---
#include <Trade\Trade.mqh>
#include <Trade\SymbolInfo.mqh>
#include <Trade\PositionInfo.mqh>
#include <Trade\AccountInfo.mqh>
//---
input double InpLots          =0.1; // Lots
input int    InpTakeProfit    =50;  // Take-Profit (in Pips)
input int    InpTrailingStop  =30;  // Trailing Stop Level (in Pips)
input int    InpMACDOpenLevel =3;   // MACD Eröffnungslevel (in Pips)
input int    InpMACDCloseLevel=2;   // MACD-Level für das Schließen (in Pips)
input int    InpMATrendPeriod =26;  // MA Periodenlänge des Trends
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit(void)
  {
//--- Erstellen von allen benötigten Objekten
   if(!ExtExpert.Init())
      return(INIT_FAILED);
//--- erfolgreiche Initialisierung
   return(INIT_SUCCEEDED);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Expert new tick handling function                                |
//+------------------------------------------------------------------+
void OnTick(void)
  {
   static datetime limit_time=0; // letzte Aufrufzeit unter Berücksichtigung von 'time
//--- wenn die Zeit den angegebenen Wert von limit_time überschreitet
   if(TimeCurrent()>=limit_time)
     {
 //--- Überprüfung der Daten
      if(Bars(Symbol(),Period())>2*InpMATrendPeriod)
        {
         //--- wenn erfolgreich, erhöhe limit_time um 'timeout' Sekunden
         if(ExtExpert.Processing())
            limit_time=TimeCurrent()+ExtTimeOut;
        }
     }
  }
//+------------------------------------------------------------------+
//| Expert tester handling function                                  |
//+------------------------------------------------------------------+
double OnTester(void)
  {
   double ret=TesterStatistics(STAT_PROFIT_FACTOR);
   double profit=TesterStatistics(STAT_PROFIT);
   int    trades_total=(int)TesterStatistics(STAT_TRADES);
   int    profit_total=(int)TesterStatistics(STAT_PROFIT_TRADES);
   int    loss_total=(int)TesterStatistics(STAT_LOSS_TRADES);
   PrintFormat("%s: Profit = %.2f, trades total: %lu, profit trades total: %lu, loss t
   return(ret);
   /*
  Ergebnis:
   OnTester: Profit = 209.84, trades total: 13, profit trades total: 11, loss trades t
   final balance 10209.84 USD
   OnTester result 3.020606644198363
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

TesterStop

Erteilt den Befehl die Programmausführung von Tests zu beenden.

```
void  TesterStop();
```

Rückgabewert

Kein Rückgabewert.

Hinweis

Mit der Funktion TesterStop() kann eine normale, vorzeitige Beendigung eines Eas in einem Testagenten erzwungen werden - zum Beispiel, wenn eine bestimmte Anzahl von Verlustpositionen oder ein voreingestellter Wert für den Drawdown erreicht wurden.

TesterStop() führt zu einem normalen Ende eines Tests, daher wird die Funktion OnTester() aufgerufen, und die gesamte entstandene Handelsstatistik und der Wert des Optimierungskriteriums werden dem Strategietester übergeben.

Der Aufruf von ExpertRemove() im Strategietester führt auch zu einem normalen Testende und ermöglicht den Erhalt der Handelsstatistik, aber der EA wird aus dem Speicher des Testagenten entfernt. In diesem Fall führt ein erneuter Testlauf mit den nächsten Parametereinstellungen zu einer Zeitverzögerung wegen des erneuten Ladens des Programms. Daher wäre TesterStop() die vorzuziehende Option für die vorzeitige Beendigung eines Tests.

Beispiel:

```
//--- defines
#define BALANCE_LOSS_STOP  100.0       // Wert des Salden-Drawdowns, bei dem die Prüfu
#define EQUITY_LOSS_STOP   100.0       // der Wert des Kapital-Drawdown, bei dem der T

//--- Eingabeparameter
input  double  InpLots        =  0.1;  // Lots
input  uint    InpStopLoss    =  50;   // Stop-Loss in Points
input  uint    InpTakeProfit  =  150;  // Take-Profit in Points
sinput ulong   InpMagic       =  123;  // MagicNumber
sinput ulong   InpDeviation   =  5;    // Abweichung
//--- globale Variable
CTrade      trade;                     // Klasseninstanz von trade
CSymbolInfo symb;                      // Klasseninstanz des Symbols
CAccountInfo account;                  // Klasseninstanz des Kontos
...
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
   ...
//--- erfolgreiche Initialisierung
   return(INIT_SUCCEEDED);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- update der aktuellen Kurse
   if(!symb.RefreshRates())
      return;
   ...

//--- wenn der Saldo oder das Kapital stärker gesunken sind als in den Makrosubstituti
//--- gilt der Test als nicht erfolgreich und die Funktion TesterStop() wird aufgerufe
//--- Prüfung auf einen Saldenverlust um mehr als BALANCE_LOSS_STOP
   if(balance_prev!=account.Balance())
     {
      if(account.Balance()<balance_prev-BALANCE_LOSS_STOP)
        {
         PrintFormat("The initial balance of %.2f %s decreased by %.2f %s, and now has
         TesterStop();
         /*
        Ergebnis:
         The initial balance of 10000.00 USD decreased by 100.10 USD, and now has a va
         TesterStop() called on 9% of testing interval
         */
        }
     }
//--- Prüfung des Kapitalverlusts um mehr als EQUITY_LOSS_STOP
   if(equity_prev!=account.Equity())
     {
      if(account.Equity()<equity_prev-EQUITY_LOSS_STOP)
        {
         PrintFormat("The initial equity of %.2f %s decreased by %.2f %s, and now has
         TesterStop();
         /*
        Ergebnis:
         The initial equity of 10000.00 USD decreased by 100.10 USD, and now has a val
         TesterStop() called on 9% of testing interval
         */
        }
     }
  }
```

Siehe auch

Programmausführung, Testen von Handelsstrategien, ExpertRemove, SetReturnError

© 2000-2025, MetaQuotes Ltd.

TesterDeposit

Eine spezielle Funktion, die während eines Tests Einzahlungen emuliert. Sie kann von Geldmanagementsystemen verwendet werden.

```
bool  TesterDeposit(
   double money      // der dem Konto gutzuschreibender Betrag
   );
```

Parameter

money

[in] Geldbetrag, der in der Kontowährung dem Konto gutgeschrieben wird.

Rückgabewert

Liefert bei Erfolg true, ansonsten false.

Beispiel:

```
//--- defines
#define BALANCE_LOSS_DEPOSIT  100.0    // Wert des Drawdowns des Saldos, zu dem die Ge

//--- Eingabeparameter
input  double  InpLots        =  0.1;  // Lots
input  uint    InpStopLoss    =  50;   // Stop-Loss in Points
input  uint    InpTakeProfit  =  150;  // Take-Profit in Points
sinput ulong   InpMagic       =  123;  // Magic-Mummer
sinput ulong   InpDeviation   =  5;    // Abweichung
//--- globale Variable
CTrade      trade;                     // Klasseninstanz von trade
CSymbolInfo symb;                      // Klasseninstanz des Symbols
CAccountInfo account;                  // Klasseninstanz des Kontos
...
double      balance_dep_summ;          // Gesamtbetrag der Kontostanderhöhungen
uint        balance_dep_total;         // Anzahl der Kontostanderhöhungen
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
   ...
//--- Sichern der Anfangswerte des Saldos
   balance_prev=account.Balance();
   balance_dep_summ=0;
   balance_dep_total=0;
//--- erfolgreiche Initialisierung
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- update der aktuellen Kurse
   if(!symb.RefreshRates())
      return;
   ...

//--- wenn der Saldo stärker gesunken ist, als in der Makrosubstitution BALANCE_LOSS_I
//--- ist es notwendig, das Konto aufzuladen und die Funktion TesterDeposit() aufzurui
//--- Prüfung, ob der Saldo um mehr als BALANCE_LOSS_DEPOSIT gesunken ist
   if(balance_prev!=account.Balance())
     {
      if(account.Balance()<balance_prev-BALANCE_LOSS_DEPOSIT)
        {
         double loss=balance_prev-account.Balance();
         PrintFormat("The initial balance of %.2f %s decreased by %.2f %s. It is neces
         if(TesterDeposit(loss))
           {
            balance_dep_total++;
            balance_dep_summ+=loss;
            balance_prev=account.Balance();
            PrintFormat("Funds have been deposited into the account. Account balance:
            PrintFormat("Total deposits: %lu. Amount of deposits: %.2f %s.",balance_de
           }
         /*
        Ergebnis:
         The initial balance of 10000.00 USD decreased by 116.00 USD. It is necessary
         deal #45 balance 116.00 [deposit] done
         Funds have been deposited into the account. Account balance: 10000.00 USD.
         Total deposits: 1. Amount of deposits: 116.00 USD.
         */
        }
     }
  }
//+------------------------------------------------------------------+
//| Tester Funktion                                                  |
//+------------------------------------------------------------------+
double OnTester()
  {
//--- der maximale Drawdown des Saldos in Geldwerten für den Output-Handler festlegen
   double ret=TesterStatistics(STAT_BALANCE_DD);
//--- Anzeige einer Meldung über den Drawdown, die Anzahl der Einzahlungen und deren C
   PrintFormat("%s: Maximum balance drawdown in money: %.2f %s. Total deposits: %lu. A
//--- Ergebnisrückgabe
   return(ret);
   /*
  Ergebnis:
```

© 2000-2025, MetaQuotes Ltd.

```
   OnTester: Maximum balance drawdown in money: 5188.50 USD. Total deposits: 46. Amour
   final balance 4867.50 USD
   OnTester result 5188.5
   */
  }
```

Siehe auch

TesterWithdrawal

© 2000-2025, MetaQuotes Ltd.

TesterWithdrawal

Eine spezielle Funktion, die während eines Tests Abhebungen emuliert. Sie kann von Geldmanagementsystemen verwendet werden.

```
bool  TesterWithdrawal(
   double money      // abzuhebende Summe
   );
```

Parameter

money

[in] Geldbetrag in der Kontowährung, der vom Konto abgebucht wird.

Rückgabewert

Liefert bei Erfolg true, ansonsten false.

Beispiel:

```
//--- defines
#define BALANCE_PROFIT_WITHDRAWAL   5  // der Wert des Saldengewinns, bei dem Mittel v

//--- Eingabeparameter
input  double  InpLots        =  0.1;  // Lots
input  uint    InpStopLoss    =  50;   // Stop-Loss in Points
input  uint    InpTakeProfit  =  150;  // Take-Profit in Points
sinput ulong   InpMagic       =  123;  // Magic-Mummer
sinput ulong   InpDeviation   =  5;    // Abweichung
//--- globale Variable
CTrade      trade;                     // Klasseninstanz von trade
CSymbolInfo symb;                      // Klasseninstanz des Symbols
CAccountInfo account;                  // Klasseninstanz des Kontos
...
double      balance_op_sum;            // Gesamtbetrag der Saldenoperationen
uint        balance_op_total;          // Anzahl der Saldenoperationen
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
   ...
//--- Sichern der Anfangswerte des Saldos
   balance_prev=account.Balance();
   balance_op_sum=0;
   balance_op_total=0;
//--- erfolgreiche Initialisierung
   return(INIT_SUCCEEDED);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- update der aktuellen Kurse
   if(!symb.RefreshRates())
      return;
   ...

//--- wenn der Saldengewinn den aktuellen Saldo um den in der Makrosubstitution BALANC
//--- ist es notwendig, diese Mittel vom Konto abzuheben. Die Funktion TesterWithdrawa
//--- Den Saldengewinn auf Überschreitung von BALANCE_PROFIT_WITHDRAWAL prüfen
   if(balance_prev!=account.Balance())
     {
      if(account.Balance()>balance_prev+BALANCE_PROFIT_WITHDRAWAL)
        {
         double profit=account.Balance()-balance_prev;
         PrintFormat("The account balance has been increased by %.2f %s. Need to withd
         if(TesterWithdrawal(profit))
           {
            balance_op_total++;
            balance_op_summ+=profit;
            balance_prev=account.Balance();
            PrintFormat("Funds have been withdrawn from the account. Account balance:
            PrintFormat("Total withdrawals: %lu. Amount of withdrawals: %.2f %s.",bala
           }
         /*
        Ergebnis:
         The account balance has been increased by 21.00 USD. Need to withdraw these f
         deal #13 balance -21.00 [withdrawal] done
         Funds have been withdrawn from the account. Account balance: 10000.00 USD.
         Total withdrawals: 1. Amount of withdrawals: 21.00 USD.
         */
        }
     }
  }
//+------------------------------------------------------------------+
//| Tester Funktion                                                  |
//+------------------------------------------------------------------+
double OnTester()
  {
//--- der maximale Drawdown des Saldos in Geldwerten für den Output-Handler festlegen
   double ret=TesterStatistics(STAT_BALANCE_DD);
//--- Anzeige einer Meldung über den Drawdown, die Anzahl der Abhebungen und deren Ges
   PrintFormat("%s: Maximum balance drawdown in money: %.2f %s. Total withdrawals: %lu
//--- Ergebnisrückgabe
   return(ret);
   /*
```

© 2000-2025, MetaQuotes Ltd.

```
  Ergebnis:
   OnTester: Maximum balance drawdown in money: 5188.50 USD. Total withdrawals: 2. Amo
   final balance 4867.50 USD
   OnTester result 5188.5
   */
  }
```

Siehe auch

TesterDeposit

© 2000-2025, MetaQuotes Ltd.

TranslateKey

Gibt ein Unicode-Zeichen nach dem virtuellen Tastencode unter Berücksichtigung der aktuellen Sprache und des Status der Steuerungstasten zurück.

```
short  TranslateKey(
   int  key_code      // Tastencode für das Erhalten des Unicode-Zeichens
   );
```

Parameter

key_code

[in] Tastencode.

Rückgabewert

Unicode-Zeichen, wenn die Umwandlung erfolgreich war. Im Fehlerfall gibt die Funktionen -1 zurück.

Hinweis

Die Funktion verwendet ToUnicodeEx für die Umwandlung der vom Nutzer gedrückten Tasten in Unicode-Zeichen. Ein Fehler kann nur in dem Fall auftreten, wenn ToUnicodeEx nicht ausgelöst wurde - z.B., beim Versuch ein Zeichen für die SHIFT-Taste zu erhalten.

Beispiel:

```
void OnChartEvent(const int id,const long& lparam,const double& dparam,const string& s
  {
   if(id==CHARTEVENT_KEYDOWN)
     {
      short sym=TranslateKey((int)lparam);
      //--- wenn das eingegebene Zeichen erfolgreich in Unicode umgewandelt wurde
      if(sym>0)
         Print(sym,"'",ShortToString(sym),"'");
      else
         Print("Error in TranslateKey for key=",lparam);
     }
  }
```

Siehe auch

Ereignisse des Kundenterminals, OnChartEvent

© 2000-2025, MetaQuotes Ltd.

ZeroMemory

Die Funktion setzt die Variable auf Null, die ihr per Referenz übertragen wird.

```
void  ZeroMemory(
   void & variable      // Variable, der Null zugewiesen wird
   );
```

Parameter

Variable

[in] [out] Als Referenz übergebene Variable, die auf Null gesetzt bzw. mit Null initialisiert werden soll.

Rückgabewert

Keinen Rückgabewert.

Hinweis

Wenn der Funktionsparameter eine Zeichenkette ist, ist dieser Aufruf gleichbedeutend mit der Angabe eines NULL-Wertes für sie.
Für einfache Typen und ihre Arrays und Strukturen/Klassen, die aus solchen Typen bestehen, ist dies eine einfache Nullsetzung.
Für Objekte, die Strings und dynamische Arrays enthalten, wird ZeroMemory() für jedes Mitglied aufgerufen.
Bei allen Arrays, die nicht durch den Modifikator const geschützt sind, werden alle Elemente auf Null gesetzt.
Bei Arrays von komplexen Objekten wird ZeroMemory() für jedes Element aufgerufen.

ZeroMemory() kann nicht auf Klassen mit geschützten Mitgliedern oder bei Klassenableitungen angewendet werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklaration und Initialisierung der Zeichenkette
   string str="Test ZeroMemory func";
//--- die Zeile an das Protokoll senden, bevor ZeroMemory() auf sie angewendet wird
   PrintFormat("The line before applying ZeroMemory() to it: '%s'",str);
//--- Rücksetzen der Zeichenkette und Senden des Ergebnisses an das Protokoll
   ZeroMemory(str);
   Print("The same line after applying ZeroMemory() to it: '",str,"'");
  /*
  Ergebnis:
   The line before applying ZeroMemory() to it: 'Test ZeroMemory func'
   The same line after applying ZeroMemory() to it: ''
```

© 2000-2025, MetaQuotes Ltd.

```
  */

//--- Deklarieren und Initialisieren der Variablen vom Typ int
   int var=123;
//--- die Zeile an das Protokoll senden, bevor ZeroMemory() auf sie angewendet wird
   PrintFormat("\nThe integer variable before applying ZeroMemory() to it: %d",var);
//--- die Variable zurücksetzen und das Ergebnis an das Protokoll senden
   ZeroMemory(var);
   PrintFormat("The same variable after applying ZeroMemory() to it: %d",var);
  /*
  Ergebnis:
   The integer variable before applying ZeroMemory() to it: 123
   The same variable after applying ZeroMemory() to it: 0
  */

//--- das Array vom Typ int deklarieren und initialisieren
   int arr[]={0,1,2,3,4,5,6,7,8,9};
//--- das Array an das Protokoll senden, bevor ZeroMemory() darauf angewendet wird
   Print("\nThe integer array before applying ZeroMemory() to it:");
   ArrayPrint(arr);
//--- das Array zurücksetzen und das Ergebnis an das Protokoll senden
   ZeroMemory(arr);
   Print("The same array after applying ZeroMemory() to it:");
   ArrayPrint(arr);
  /*
  Ergebnis:
   The integer array before applying ZeroMemory() to it:
   0 1 2 3 4 5 6 7 8 9
   The same array after applying ZeroMemory() to it:
   0 0 0 0 0 0 0 0 0 0
  */

//--- eine Struktur mit zwei Feldern - Zeichenkette und Ganzzahl - deklarieren
   struct STest
     {
      string   var_string;
      long     var_long;
     };
//--- das Array STest vom Typ einer Struktur deklarieren und initialisieren
   STest arr_struct[]={ {"0",0}, {"1",1}, {"2",2}, {"3",3} };
//--- das Array an das Protokoll senden, bevor ZeroMemory() darauf angewendet wird
   Print("\nThe array struct before applying ZeroMemory() to it:");
   ArrayPrint(arr_struct);
//--- das Struktur-Array zurücksetzen und das Ergebnis an das Protokoll senden
   ZeroMemory(arr_struct);
   Print("The same array struct after applying ZeroMemory() to it:");
   ArrayPrint(arr_struct);
  /*
  Ergebnis:
```

© 2000-2025, MetaQuotes Ltd.

```
   The array struct before applying ZeroMemory() to it:
       [var_string] [var_long]
   [0] "0"                  0
   [1] "1"                  1
   [2] "2"                  2
   [3] "3"                  3
   The same array struct after applying ZeroMemory() to it:
       [var_string] [var_long]
   [0] null                 0
   [1] null                 0
   [2] null                 0
   [3] null                 0
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

Gruppe der Funktionen für Arbeit mit Arrays

Arrays können nicht mehr als vierdimensional sein. Jede Dimension wird von Null bis *Dimensons_Größe-1* indiziert. Im Einzelfall des eindimensionalen Arrays aus 50 Elementen wird die Anwendung zum ersten Element als array[0], zum letzten - array[49] aussehen.

Funktion	Handlung
ArrayBsearch	Gibt Index des ersten gefundenen Elementes in der ersten Dimension des Arrays
ArrayCopy	Kopiert ein Array in das andere Array
ArrayCompare	Gibt das Ergebnis eines Vergleichs um zwei Arrays von einfachen Typen oder angepassten Strukturen ohne komplexe Objekte zurück
ArrayFree	Befreit Puffer jedes dynamischen Arrays und stellt die Größe der ersten Null-Dimension auf 0.
ArrayGetAsSeries	Prüft die Richtung des Arrayindizierens
ArrayInitialize	Stellt alle Elementen des nummerischen Arrays in eine Größe ein
ArrayFill	Füllt das numerische Array mit den angegeben Wert
ArrayIsSeries	Prüft - ob das Array Timeserie ist
ArrayIsDynamic	Prüft - ob das Array dynamisches Array ist
ArrayMaximum	Suche des Elementes mit maximalem Wert
ArrayMinimum	Suche des Elementes mit minimalem Wert
ArrayPrint	Gibt Arrays eines einfachen Typs oder einer einfachen Struktur im Journal aus
ArrayRange	Gibt die Zahl der Elementen in der gegebenen Dimension des Arrays zurück
ArrayResize	Stellt die neue Größe in der ersten Dimension des Arrays ein
ArrayInsert	Einfügen einer Anzahl von Elementen eines Quellarrays in ein Zielarray, beginnend mit dem angegebenen Index
ArrayRemove	Entfernt die angegebene Anzahl von Elementen im Array, beginnend mit dem angegebenen Index
ArrayReverse	Kehrt die angegebene Anzahl von Elementen im Array um, beginnend mit dem angegebenen Index
ArraySetAsSeries	Stellt die Richtung der Indizierung im Array ein
ArraySize	Gibt die Zahl von Elementen im Array zurück
ArraySort	Sortieren der numerischen Arrays nach der ersten Dimension
ArraySwap	Vertauscht die Inhalte von zwei dynamischen Arrays eines Typs

© 2000-2025, MetaQuotes Ltd.

Funktion	Handlung
ArrayToFP16	Kopiert ein Array vom Typ float oder double in ein Array vom Typ ushort mit dem angegebenen Format
ArrayToFP8	Kopiert ein Array vom Typ float oder double in ein Array vom Typ uchar mit dem angegebenen Format
ArrayFromFP16	Kopiert ein Array vom Typ ushort in ein Array vom Typ float oder double mit dem angegebenen Format
ArrayFromFP8	Kopiert ein Array vom Typ uchar in ein Array vom Typ float oder double mit dem angegebenen Format

© 2000-2025, MetaQuotes Ltd.

ArrayBsearch

Sucht nach dem angegebenen Wert in einem aufsteigend sortierten mehrdimensionalen Array. Es wird unter Elementen der ersten Dimension ermittelt.

für Suchen im Array des Typs double

```
int ArrayBsearch(
   const double&    array[],   // das Array für die Suche
   double           value      // was wird gesucht
   );
```

für Suche im Array des Typs float

```
int ArrayBsearch(
   const float&    array[],   // das Array für die Suche
   float           value      // was wird gesucht
   );
```

für Suche im Array des Typs long

```
int ArrayBsearch(
   const long&    array[],   // das Array für die Suche
   long           value      // was wird gesucht
   );
```

für Suche im Array des Typs int

```
int ArrayBsearch(
   const int&    array[],   // Array für die Suche
   int           value      // was wird gesucht
   );
```

für Suche im Array des Typs short

```
int ArrayBsearch(
   const short&    array[],   // Array für die Suche
   short           value      // was wird gesucht
   );
```

für Suche im Array des Typs char

```
int ArrayBsearch(
   const char&    array[],   // Array für die Suche
   char           value      // was wird gesucht
   );
```

Parameter

array[]

[in] Numerisches Array für die Suche.

value

© 2000-2025, MetaQuotes Ltd.

[in] Wert für die Suche.

Rückgabewert

Gibt Index des gefundenen Elementes zurück. Wenn der gesuchte Wert nicht gefunden wird, wird Index des Elementes, der den nächsten Wert hat.

Hinweis

Binäre Suche verarbeitet sortierte Arrays nur. Für Sortieren des numerischen Arrays wird die Funktion ArraySort() verwendet.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Skript auf der RSI Indikator wird im Chart Fenster angezeigt"
#property description "die Daten wie oft der Markt in eine überkauft"
#property description "und überverkauft Zonen in den angegebenen Zeitraum war."
//--- Beim Starten des Skripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input int                InpMAPeriod=14;                    // Periode des gleitenden
input ENUM_APPLIED_PRICE InpAppliedPrice=PRICE_CLOSE;       // Preistyp
input double             InpOversoldValue=30.0;             // Überverkauft Ebene
input double             InpOverboughtValue=70.0;           // Überkauft Ebene
input datetime           InpDateStart=D'2012.01.01 00:00';  // Das Anfangsdatum der An
input datetime           InpDateFinish=D'2013.01.01 00:00'; // das Abschlußdatum der A
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   double rsi_buff[]; // das Array der Indikatorwerte
   int    size=0;     // die Größe des Arrays
//--- erhalten Sie die Handle des RSI Indikators
   ResetLastError();
   int rsi_handle=iRSI(Symbol(),Period(),InpMAPeriod,InpAppliedPrice);
   if(rsi_handle==INVALID_HANDLE)
     {
      //--- Es misslang, Handle des Indikators zu bekommen
      PrintFormat("Fehler beim Erhalten die Handle des Indikators. Fehlercode = %d",Ge
      return;
     }
//--- Wir befinden uns im Zyklus, bis der Indikator alle Werten berechnen wird
   while(BarsCalculated(rsi_handle)==-1)
     {
      //--- Wir gehen hinaus, wenn der Benutzer die Arbeit des Skripts zwangsläufig be
      if(IsStopped())
         return;
      //--- die Verzögerungszeit, damit der Indikator geschafft, seine Werte zu berech
      Sleep(10);
     }
//--- kopieren Sie die Indikatorwerte für eine angegebene Period
   ResetLastError();
   if(CopyBuffer(rsi_handle,0,InpDateStart,InpDateFinish,rsi_buff)==-1)
     {
      PrintFormat("Es misslang, die Indikatorwerte zu kopieren. Fehlercode = %d",GetLa
      return;
     }
//--- erhalten Sie die Größe des Arrays
   size=ArraySize(rsi_buff);
//--- sortieren Sie das Array
   ArraySort(rsi_buff);
//--- erkennen Sie welches Prozent der Zeit sich den Markt in der überverkauft Zone be
   double ovs=(double)ArrayBsearch(rsi_buff,InpOversoldValue)*100/(double)size;
//--- erkennen Sie welches Prozent der Zeit sich den Markt in der überkauft Zone befan
   double ovb=(double)(size-ArrayBsearch(rsi_buff,InpOverboughtValue))*100/(double)siz
//--- bilden Sie die Zeilen für die Schlussfolgerung der Daten
   string str="Seit "+TimeToString(InpDateStart,TIME_DATE)+" bis "
              +TimeToString(InpDateFinish,TIME_DATE)+" der Markt ist in:";
   string str_ovb="den überkauft Zone "+DoubleToString(ovb,2)+"% Zeit";
   string str_ovs="den überverkauft Zone "+DoubleToString(ovs,2)+"% Zeit";
//--- zeigen Sie die Daten in der Chart
   CreateLabel("top",5,60,str,clrDodgerBlue);
   CreateLabel("overbought",5,35,str_ovb,clrDodgerBlue);
   CreateLabel("oversold",5,10,str_ovs,clrDodgerBlue);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- zeichnen Sie die Chart
   ChartRedraw(0);
//--- die Verzögerung
   Sleep(10000);
  }
//+------------------------------------------------------------------+
//| Zeigt den Kommentar in der unteren linken Ecke der Chart         |
//+------------------------------------------------------------------+
void CreateLabel(const string name,const int x,const int y,
                 const string str,const color clr)
  {
//--- Label erstellen
   ObjectCreate(0,name,OBJ_LABEL,0,0,0);
//--- Bindung von Label zu dem linken unteren Ecke
   ObjectSetInteger(0,name,OBJPROP_CORNER,CORNER_LEFT_LOWER);
//--- ändern Sie die Position der Ankerpunkt
   ObjectSetInteger(0,name,OBJPROP_ANCHOR,ANCHOR_LEFT_LOWER);
//--- die Länge der X-Achse von der Ankerpunkt
   ObjectSetInteger(0,name,OBJPROP_XDISTANCE,x);
//--- die Länge der Y-Achse von der Ankerpunkt
   ObjectSetInteger(0,name,OBJPROP_YDISTANCE,y);
//--- der Text-Label
   ObjectSetString(0,name,OBJPROP_TEXT,str);
//--- die Textfarbe
   ObjectSetInteger(0,name,OBJPROP_COLOR,clr);
//--- die Textgröße
   ObjectSetInteger(0,name,OBJPROP_FONTSIZE,12);
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayCopy

Kopiert ein Array in das andere Array.

```
int  ArrayCopy(
   void&        dst_array[],          // Zielarray für Kopieren
   const void&  src_array[],          // Das Array von dem kopiert wird
   int          dst_start=0,          // Index, mit dem in Rezipienten geschrieben wird
   int          src_start=0,          // Index, mit dem Kopieren von der Quelle beginnt
   int          count=WHOLE_ARRAY     // Anzahl der Elemente
   );
```

Parameter

dst_array[]

[out] Array-Rezipient.

src_array[]

[in] Ausgangs-Array.

dst_start=0

[in] Anfangsindex für Array-Rezipienten. Standardwert des Anfangsindexes ist - 0.

src_start=0

[in] Anfangsindex für Ausgangs-Array. Standardwert des Anfangsindexes ist - 0.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für Kopieren. Als Voreinstellung wird das ganze Array kopiert. (count=WHOLE_ARRAY).

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

Hinweis

Wenn count<0 oder count>src_size-src_start, wird das Rest des Arrays kopiert. Arrays werden von links nach rechts kopiert. Für serielle Arrays wird die Anfangsposition unter Berücksichtigung des Kopierens von links nach rechts korrekt neubestimmt.

Wenn die Arrays verschiedener Typen angehören, wird beim Kopieren der Versuch, jedes Element des Ausgangsarrays zum Typ des Array-Rezipienten zu reduzieren. Ein Zeilenarray kann nur in ein Zeilenarray kopiert werden. Arrays der Klassen und Strukturen, die Objekte enthalten, die initialisiert werden müssen, werden nicht kopiert. Array der Strukturen kann nur ins Array desselben Typs kopiert werden.

Für dynamische Arrays mit Indexierung wie in Timeseries wird das Empfänger-Array automatisch bis die Menge der kopierenden Daten vergößert (wenn die Menge der kopierenden Daten größer als das Array ist). Empfängener-Array wird nicht automatisch verkleinert.

Beispiel:

```
#property description "Der Indikator hervorhebt die Kerzen, die lokal sind"
#property description "Maximum und Minimum. Die Länge des Intervalls für Aufsuchen"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "die Extremums kann man durch die Eingabeparameters einstellt we
//--- Indikator Einstellungen
#property indicator_chart_window
#property indicator_buffers 5
#property indicator_plots   1
//---- plot
#property indicator_label1  "Extremums"
#property indicator_type1   DRAW_COLOR_CANDLES
#property indicator_color1  clrLightSteelBlue,clrRed,clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- vordefinierte Konstante
#define INDICATOR_EMPTY_VALUE 0.0
//--- Eingabeparameters
input int InpNum=4; // Die Länge des Halb-Intervalls
// --- Indikator Puffers
double ExtOpen[];
double ExtHigh[];
double ExtLow[];
double ExtClose[];
double ExtColor[];
//--- globale Variablen
int    ExtStart=0; // der Index der ersten Kerze, die kein Extremum ist
int    ExtCount=0; // Anzahl der Kerzen, nicht den Extremums in diesen Intervall
//+------------------------------------------------------------------+
//| Zeichnen die Kerzen, nicht Extremums                             |
//+------------------------------------------------------------------+
void FillCandles(const double &open[],const double &high[],
                 const double &low[],const double &close[])
  {
//--- Zeichnen Sie die Kerzen
   ArrayCopy(ExtOpen,open,ExtStart,ExtStart,ExtCount);
   ArrayCopy(ExtHigh,high,ExtStart,ExtStart,ExtCount);
   ArrayCopy(ExtLow,low,ExtStart,ExtStart,ExtCount);
   ArrayCopy(ExtClose,close,ExtStart,ExtStart,ExtCount);
  }
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator Puffer
   SetIndexBuffer(0,ExtOpen);
   SetIndexBuffer(1,ExtHigh);
   SetIndexBuffer(2,ExtLow);
   SetIndexBuffer(3,ExtClose);
   SetIndexBuffer(4,ExtColor,INDICATOR_COLOR_INDEX);
//--- erstellen Sie die Wert, die nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,INDICATOR_EMPTY_VALUE);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- erstellen Sie die Name von Indikator-Puffers für die Anzeige im Daten-Fenster
   PlotIndexSetString(0,PLOT_LABEL,"Open;High;Low;Close");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- stellen Sie die gerade Richtung der Indexierung in Zeitreihen ein
   ArraySetAsSeries(open,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
   ArraySetAsSeries(close,false);
//--- Variable zur Berechnung den Anfang des Bars
   int start=prev_calculated;
//--- für die ersten InpNum*2 Bars wird die Berechnung nicht durchgeführt
   if(start==0)
     {
      start+=InpNum*2;
      ExtStart=0;
      ExtCount=0;
     }
//--- Wenn sich nur eine Bar gebildet hat, so werden wir den nächsten potentiellen Ext
   if(rates_total-start==1)
      start--;
//--- der Index der Bar, den wir auf Extremum prüfen werden
   int ext;
//--- die Zyklus von die Berechnung der Indikatorwerte
   for(int i=start;i<rates_total-1;i++)
     {
      //--- vornherein an der i-ten Bar ohne Zeichnung
      ExtOpen[i]=0;
      ExtHigh[i]=0;
      ExtLow[i]=0;
      ExtClose[i]=0;
      //--- der Extremum Index für die Überprüfung
      ext=i-InpNum;
      //--- die Überprüfung für das lokale Maximum
```

© 2000-2025, MetaQuotes Ltd.

```
      if(IsMax(high,ext))
        {
         //--- bezeichnen Sie die Extremum Kerze
         ExtOpen[ext]=open[ext];
         ExtHigh[ext]=high[ext];
         ExtLow[ext]=low[ext];
         ExtClose[ext]=close[ext];
         ExtColor[ext]=1;
         //--- die übrigen Kerzen werden wir bis zu Extremum mit der neutralen Farbe k
         FillCandles(open,high,low,close);
         //--- ändern Sie die Variablenwerte
         ExtStart=ext+1;
         ExtCount=0;
         //--- gehen Sie zum nächsten Iteration
         continue;
        }
      //--- Überprüfung für das lokale Minimum
      if(IsMin(low,ext))
        {
         //--- bezeichnen Sie die Extremum Kerze
         ExtOpen[ext]=open[ext];
         ExtHigh[ext]=high[ext];
         ExtLow[ext]=low[ext];
         ExtClose[ext]=close[ext];
         ExtColor[ext]=2;
         //--- die übrigen Kerzen werden wir bis zu Extremum mit der neutralen Farbe k
         FillCandles(open,high,low,close);
         //--- ändern Sie die Variablenwerte
         ExtStart=ext+1;
         ExtCount=0;
         //--- gehen Sie zum nächsten Iteration
         continue;
        }
      // vergrößern Sie die Anzahl nicht den Extremums in diesem Intervall
      ExtCount++;
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ob das aktuelle Element des Arrays ein lokales Maximum ist       |
//+------------------------------------------------------------------+
bool IsMax(const double &price[],const int ind)
  {
//--- Variable von dem Intervall-Anfang
   int i=ind-InpNum;
//--- Variable von dem Intervall-Abschluß
   int finish=ind+InpNum+1;
//--- die Überprüfung für die erste Hälfte des Intervalls
```

© 2000-2025, MetaQuotes Ltd.

```
   for(;i<ind;i++)
     {
      if(price[ind]<=price[i])
         return(false);
     }
//--- die Überprüfung für die zweite Hälfte des Intervalls
   for(i=ind+1;i<finish;i++)
     {
      if(price[ind]<=price[i])
         return(false);
     }
//--- es ist Extremum
   return(true);
  }
//+------------------------------------------------------------------+
//| Ob das aktuell Element des Arrays ein lokales Minimum ist        |
//+------------------------------------------------------------------+
bool IsMin(const double &price[],const int ind)
  {
//--- Variable von dem Intervall-Anfang
   int i=ind-InpNum;
//--- Variable von dem Intervall-Abschluß
   int finish=ind+InpNum+1;
//--- die Überprüfung für die erste Hälfte des Intervalls
   for(;i<ind;i++)
     {
      if(price[ind]>=price[i])
         return(false);
     }
//--- die Überprüfung für die zweite Hälfte des Intervalls
   for(i=ind+1;i<finish;i++)
     {
      if(price[ind]>=price[i])
         return(false);
     }
//--- es ist Extremum
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayCompare

Gibt das Ergebnis eines Vergleichs von zwei Arrays des gleichen Typs zurück. Es kann verwendet werden, um Arrays von einfachen Typen oder angepassten Strukturen ohne komplexe Objekte zu vergleichen - d.h. die keine Zeichen, dynamische Arrays von Klassen oder anderen Strukturen mit komplexe Objekten enthalten.

```
int  ArrayCompare(
   const void&  array1[],            // das erste Array
   const void&  array2[],            // das zweite Array
   int          start1=0,            // Anfangsversatz im ersten Array
   int          start2=0,            // Anfangsversatz im zweiten Array
   int          count=WHOLE_ARRAY    // Anzahl der Elemente für den Vergleich
   );
```

Parameter

array1[]

[in] Das erste Array.

array2[]

[in] Das zweite Array.

start1=0

[In] Der Ausgangsindex des Elementes im ersten Array, mit dem der Vergleich beginnen wird. Als Voreinstellung der Startindex - 0.

start2=0

[In] Der Ausgangsindex des Elementes im zweiten Array, mit dem der Vergleich beginnen wird. Als Voreinstellung der Startindex - 0.

count=WHOLE_ARRAY

[in] Die Anzahl der Elemente, die man vergleichen muss. Als Voreinstellung beteiligt alle Elemente beider Arrays in dem Vergleich (count=WHOLE_ARRAY).

Der Rückgabewert

- -1, wenn array1[] weniger als array2[]
- 0, wenn array1[] und array2[] sind gleich
- 1, wenn array1[] mehr als array2[]
- 2, Beim Auftreten des Fehlers wegen der Inkompatibilität der Typen der verglichenen Arrays, oder wenn es start1, start2 oder count Werte gibt, die zum Ausgang aus dem Grenzen von Array bringen.

Hinweis

Bei unterschiedlichen Größen der zu vergleichenden Arrays und dem angegebenen Wert count=WHOLE_ARRAY für den Fall, dass ein Array eine exakte Teilmenge eines anderen ist, wird die Funktion nicht 0 zurückgeben (die Arrays werden al ungleich angesehen). In diesem Fall wird als Ergebnis des Vergleichs eine Art Größenvergleich der beiden Arrays zurückgegeben: -1, wenn die Größe von array1[] kleiner ist als die Größe von array2[], ansonsten 1.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
//--- globale Variable
double   ExtArrayFirst[];
double   ExtArraySecond[];

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Array-Größe festlegen
   if(ArrayResize(ExtArrayFirst,10)!=10)
     {
      Print("ArrayResize() failed for ExtArrayFirst. Error code: ",GetLastError());
      return;
     }
   if(ArrayResize(ExtArraySecond,10)!=10)
     {
      Print("ArrayResize() failed for ExtArraySecond. Error code: ",GetLastError());
      return;
     }

//--- Füllen des Arrays mit den Indexwerten von i und j in einer Schleife
   int total=ArraySize(ExtArrayFirst);
   for(int i=0, j=total-1; i<total; i++,j--)
     {
      //--- Füllen des Arrays ExtArrayFirst von links nach rechts
      //--- Füllen des Arrays ExtArraySecond von rechts nach links
      ExtArrayFirst[i]=i;
      ExtArraySecond[i]=j;
     }
//--- Arrays vergleichen und der Ausgabe des Ergebnisses im Log
   ArrayComparePrint(ExtArrayFirst,ExtArraySecond);
   /*
  Ergebnis:
   ExtArrayFirst:
   0.00000 1.00000 2.00000 3.00000 4.00000 5.00000 6.00000 7.00000 8.00000 9.00000
   ExtArraySecond:
   9.00000 8.00000 7.00000 6.00000 5.00000 4.00000 3.00000 2.00000 1.00000 0.00000
   Result ArrayCompare(): ExtArrayFirst is smaller than ExtArraySecond (result = -1)
   */

//--- jetzt spiegeln wir die Arrays
//--- Füllen des Arrays mit den Indexwerten von i und j in einer Schleife
   for(int i=0, j=total-1; i<total; i++,j--)
     {
      //--- Füllen des Arrays ExtArrayFirst von rechts nach links
      //--- Füllen des Arrays ExtArraySecond von links nach rechts
```

© 2000-2025, MetaQuotes Ltd.

```
      ExtArrayFirst[i]=j;
      ExtArraySecond[i]=i;
     }
//--- Arrays vergleichen und der Ausgabe des Ergebnisses im Log
   ArrayComparePrint(ExtArrayFirst,ExtArraySecond);
   /*
  Ergebnis:
   ExtArrayFirst:
   9.00000 8.00000 7.00000 6.00000 5.00000 4.00000 3.00000 2.00000 1.00000 0.00000
   ExtArraySecond:
   0.00000 1.00000 2.00000 3.00000 4.00000 5.00000 6.00000 7.00000 8.00000 9.00000
   Result ArrayCompare(): ExtArrayFirst is larger than ExtArraySecond (result = 1)
   */

//--- Füllen wir nun die Felder in eine Richtung
//--- Füllen der Arrays mit den Indexwerten i in einer Schleife
   for(int i=0; i<total; i++)
     {
      //--- Füllen beider Arrays von links nach rechts
      ExtArrayFirst[i]=i;
      ExtArraySecond[i]=i;
     }
//--- Arrays vergleichen und der Ausgabe des Ergebnisses im Log
   ArrayComparePrint(ExtArrayFirst,ExtArraySecond);
   /*
  Ergebnis:
   ExtArrayFirst:
   0.00000 1.00000 2.00000 3.00000 4.00000 5.00000 6.00000 7.00000 8.00000 9.00000
   ExtArraySecond:
   0.00000 1.00000 2.00000 3.00000 4.00000 5.00000 6.00000 7.00000 8.00000 9.00000
   Result ArrayCompare(): ExtArrayFirst and ExtArraySecond are equal (result = 0)
   */
  }
//+------------------------------------------------------------------+
//| Vergleichen und Anzeigen des Ergebnisses                         |
//+------------------------------------------------------------------+
void ArrayComparePrint(const double &array1[], const double &array2[])
  {
   //--- Ausdruck der Kopfzeile und des Arrayinhalts
   Print("ExtArrayFirst:");
   ArrayPrint(array1);
   Print("ExtArraySecond:");
   ArrayPrint(array2);
   //--- die Arrays vergleichen und das Vergleichsergebnis ausgeben
   int    res=ArrayCompare(array1,array2);
   string res_str=(res>0 ? "ExtArrayFirst is larger than ExtArraySecond" : res<0 ? "Ex
   PrintFormat("Result ArrayCompare(): %s (result = %d)\n",res_str,res);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

ArrayFree

Befreit Puffer jedes dynamischen Arrays und stellt die Größe der Null-Dimension auf 0 ein.

```
void  ArrayFree(
   void&  array[]      // Array
   );
```

Parameter

array[]

[in] Dynamisches Array.

Rückgabewert

keinen Rückgabewert.

Anmerkung

Bei der Schreibung des Skripts und der Indikatoren können eine Notwendigkeit im Einführung der ArrayFree() Funktion nicht so oft entstehen, denn am Ende des Skripts der gesamte verwendete Speicher sofort befreit wird und in den benutzereigen Indikatoren stellt die Grundarbeit mit den Arrays den Zugriff auf die Indikator Puffers, deren Größe wird von der bearbeiten Subsystem des Terminals automatisch schaffen.

Wenn das Programm braucht, um Speicher in den komplizierten dynamischen Bedingungen zu verwalten, so wird die ArrayFree () Funktion offensichtlich und sofort selbständig der Speicher zu befreien, der mit dem schon nutzlos dynamischen Array beschäftigt ist.

Beispiel:

```
#include <Controls\Dialog.mqh>
#include <Controls\Button.mqh>
#include <Controls\Label.mqh>
#include <Controls\ComboBox.mqh>
// --- vorbestimmte Konstanten
#define X_START 0
#define Y_START 0
#define X_SIZE 280
#define Y_SIZE 300
//+------------------------------------------------------------------+
//| Die Klasse des Dialoges für Arbeit mit dem Speichern             |
//+------------------------------------------------------------------+
class CMemoryControl : public CAppDialog
  {
private:
   //--- die Größe des Arrays
   int               m_arr_size;
   // --- Arrays
   char              m_arr_char[];
   int               m_arr_int[];
   float             m_arr_float[];
```

© 2000-2025, MetaQuotes Ltd.

```
   double            m_arr_double[];
   long              m_arr_long[];
   //--- Beschriftungen
   CLabel            m_lbl_memory_physical;
   CLabel            m_lbl_memory_total;
   CLabel            m_lbl_memory_available;
   CLabel            m_lbl_memory_used;
   CLabel            m_lbl_array_size;
   CLabel            m_lbl_array_type;
   CLabel            m_lbl_error;
   CLabel            m_lbl_change_type;
   CLabel            m_lbl_add_size;
   //--- Schaltflächen
   CButton           m_button_add;
   CButton           m_button_free;
   // --- Listen
   CComboBox         m_combo_box_step;
   CComboBox         m_combo_box_type;
   // --- der aktuelle Wert des Array-Typ aus der Liste
   int               m_combo_box_type_value;

public:
                     CMemoryControl(void);
                    ~CMemoryControl(void);
   //--- Methode der Erzeugung eines Objekts der Klasse
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- der Bearbeiter des Ereignisses von Chart
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- erstellen Sie der Beschriftung
   bool              CreateLabel(CLabel &lbl,const string name,const int x,const int y
   //--- erstellen Sie die Steuerelemente
   bool              CreateButton(CButton &button,const string name,const int x,const
   bool              CreateComboBoxStep(void);
   bool              CreateComboBoxType(void);
   //--- der Bearbeiter des Ereignisses
   void              OnClickButtonAdd(void);
   void              OnClickButtonFree(void);
   void              OnChangeComboBoxType(void);
   //--- die Methoden der Arbeit mit dem aktuellen Array
   void              CurrentArrayFree(void);
   bool              CurrentArrayAdd(void);
  };
//+------------------------------------------------------------------+
//| Die Befreiung den Speichern des aktuellen Arrays                 |
//+------------------------------------------------------------------+
void CMemoryControl::CurrentArrayFree(void)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Rückstellung der Größe des Arrays
   m_arr_size=0;
//--- die Befreiung des Arrays
   if(m_combo_box_type_value==0)
      ArrayFree(m_arr_char);
   if(m_combo_box_type_value==1)
      ArrayFree(m_arr_int);
   if(m_combo_box_type_value==2)
      ArrayFree(m_arr_float);
   if(m_combo_box_type_value==3)
      ArrayFree(m_arr_double);
   if(m_combo_box_type_value==4)
      ArrayFree(m_arr_long);
  }
//+------------------------------------------------------------------+
//| Die Unternehmung den Speichern für den aktuelle Arrays hinzufügen|
//+------------------------------------------------------------------+
bool CMemoryControl::CurrentArrayAdd(void)
  {
//--- wenn die Größe der verwendeten Speicher ist mehr als die Größe des physikalische
   if(TerminalInfoInteger(TERMINAL_MEMORY_PHYSICAL)/TerminalInfoInteger(TERMINAL_MEMOR
      return(false);
//--- die Unternehmung den Speichern abhängig von dem aktuellen Typ zu wählen
   if(m_combo_box_type_value==0 && ArrayResize(m_arr_char,m_arr_size)==-1)
      return(false);
   if(m_combo_box_type_value==1 && ArrayResize(m_arr_int,m_arr_size)==-1)
      return(false);
   if(m_combo_box_type_value==2 && ArrayResize(m_arr_float,m_arr_size)==-1)
      return(false);
   if(m_combo_box_type_value==3 && ArrayResize(m_arr_double,m_arr_size)==-1)
      return(false);
   if(m_combo_box_type_value==4 && ArrayResize(m_arr_long,m_arr_size)==-1)
      return(false);
//--- der Speicher ist gewählt
   return(true);
  }
//+------------------------------------------------------------------+
//| Der Bearbeitung des Ereignisses                                  |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CMemoryControl)
ON_EVENT(ON_CLICK,m_button_add,OnClickButtonAdd)
ON_EVENT(ON_CLICK,m_button_free,OnClickButtonFree)
ON_EVENT(ON_CHANGE,m_combo_box_type,OnChangeComboBoxType)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Konstrukteur                                                     |
//+------------------------------------------------------------------+
CMemoryControl::CMemoryControl(void)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Destrukteur                                                      |
//+------------------------------------------------------------------+
CMemoryControl::~CMemoryControl(void)
  {
  }
//+------------------------------------------------------------------+
//| Methode der Erzeugung eines Objekts der Klasse                   |
//+------------------------------------------------------------------+
bool CMemoryControl::Create(const long chart,const string name,const int subwin,
                            const int x1,const int y1,const int x2,const int y2)
  {
//--- der Erzeugung eines Objekts der grundlegende Klasse
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- die Vorbereitung die Zeilen für die Beschriftungen
   string str_physical="Memory physical = "+(string)TerminalInfoInteger(TERMINAL_MEMOR
   string str_total="Memory total = "+(string)TerminalInfoInteger(TERMINAL_MEMORY_TOTA
   string str_available="Memory available = "+(string)TerminalInfoInteger(TERMINAL_MEM
   string str_used="Memory used = "+(string)TerminalInfoInteger(TERMINAL_MEMORY_USED)+
//--- erstellen Sie die Beschriftungen
   if(!CreateLabel(m_lbl_memory_physical,"physical_label",X_START+10,Y_START+5,str_phy
      return(false);
   if(!CreateLabel(m_lbl_memory_total,"total_label",X_START+10,Y_START+30,str_total,12
      return(false);
   if(!CreateLabel(m_lbl_memory_available,"available_label",X_START+10,Y_START+55,str_
      return(false);
   if(!CreateLabel(m_lbl_memory_used,"used_label",X_START+10,Y_START+80,str_used,12,cl
      return(false);
   if(!CreateLabel(m_lbl_array_type,"type_label",X_START+10,Y_START+105,"Array type =
      return(false);
   if(!CreateLabel(m_lbl_array_size,"size_label",X_START+10,Y_START+130,"Array size =
      return(false);
   if(!CreateLabel(m_lbl_error,"error_label",X_START+10,Y_START+155,"",12,clrRed))
      return(false);
   if(!CreateLabel(m_lbl_change_type,"change_type_label",X_START+10,Y_START+185,"Chang
      return(false);
   if(!CreateLabel(m_lbl_add_size,"add_size_label",X_START+10,Y_START+210,"Add to arra
      return(false);
//--- erstellen Sie das Steuerelement
   if(!CreateButton(m_button_add,"add_button",X_START+15,Y_START+245,"Add",12,clrBlue)
      return(false);
   if(!CreateButton(m_button_free,"free_button",X_START+75,Y_START+245,"Free",12,clrBl
      return(false);
   if(!CreateComboBoxType())
      return(false);
   if(!CreateComboBoxStep())
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die Initialisierung der Variablen
   m_arr_size=0;
//--- die erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Erstellen Sie die Schaltfläche                                   |
//+------------------------------------------------------------------+
bool CMemoryControl::CreateButton(CButton &button,const string name,const int x,
                                  const int y,const string str,const int font_size,
                                  const int clr)
  {
//--- erstellen Sie die Schaltfläche
   if(!button.Create(m_chart_id,name,m_subwin,x,y,x+50,y+20))
      return(false);
//--- der Text
   if(!button.Text(str))
      return(false);
//--- der Schriftgröße
   if(!button.FontSize(font_size))
      return(false);
//--- die Farbe der Beschriftung
   if(!button.Color(clr))
      return(false);
//--- fügen Sie die Schaltfläche in den Kontrolelementen hinzu
   if(!Add(button))
      return(false);
//--- die erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Erstellen Sie die Liste für die Größe des Arrays                 |
//+------------------------------------------------------------------+
bool CMemoryControl::CreateComboBoxStep(void)
  {
//--- erstellen Sie die Liste
   if(!m_combo_box_step.Create(m_chart_id,"step_combobox",m_subwin,X_START+100,Y_START
      return(false);
//--- fügen Sie die Elemente in die Liste hinzu
   if(!m_combo_box_step.ItemAdd("100 000",100000))
      return(false);
   if(!m_combo_box_step.ItemAdd("1 000 000",1000000))
      return(false);
   if(!m_combo_box_step.ItemAdd("10 000 000",10000000))
      return(false);
   if(!m_combo_box_step.ItemAdd("100 000 000",100000000))
      return(false);
//--- stellen Sie das aktuelle Element der Liste ein
   if(!m_combo_box_step.SelectByValue(1000000))
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
//--- fügen Sie die Liste in die Kontrolelemente hinzu
   if(!Add(m_combo_box_step))
      return(false);
//--- die erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Erstellen Sie die Liste für den Typ des Arrays                   |
//+------------------------------------------------------------------+
bool CMemoryControl::CreateComboBoxType(void)
  {
//--- erstellen Sie die Liste
   if(!m_combo_box_type.Create(m_chart_id,"type_combobox",m_subwin,X_START+100,Y_START
      return(false);
//--- fügen Sie die Elemente in die Liste hinzu
   if(!m_combo_box_type.ItemAdd("char",0))
      return(false);
   if(!m_combo_box_type.ItemAdd("int",1))
      return(false);
   if(!m_combo_box_type.ItemAdd("float",2))
      return(false);
   if(!m_combo_box_type.ItemAdd("double",3))
      return(false);
   if(!m_combo_box_type.ItemAdd("long",4))
      return(false);
//--- stellen Sie das aktuelle Element der Liste ein
   if(!m_combo_box_type.SelectByValue(3))
      return(false);
//--- behalten Sie das aktuelle Element der Liste
   m_combo_box_type_value=3;
//--- fügen Sie die Liste in die Kontrolelemente hinzu
   if(!Add(m_combo_box_type))
      return(false);
//--- die erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Erstellen Sie der Beschriftung                                   |
//+------------------------------------------------------------------+
bool CMemoryControl::CreateLabel(CLabel &lbl,const string name,const int x,
                                 const int y,const string str,const int font_size,
                                 const int clr)
  {
//--- erstellen Sie der Beschriftung
   if(!lbl.Create(m_chart_id,name,m_subwin,x,y,0,0))
      return(false);
//--- der Text
   if(!lbl.Text(str))
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
//--- der Schriftgröße
   if(!lbl.FontSize(font_size))
      return(false);
//--- die Farbe
   if(!lbl.Color(clr))
      return(false);
//--- fügen Sie der Beschriftung in die Kontrolelemente hinzu
   if(!Add(lbl))
      return(false);
//--- erfolgreichs sein
   return(true);
  }
//+------------------------------------------------------------------+
//| Der Bearbeiter des Ereignisses auf die "Add" Schaltfläche zu     |
//| klicken                                                          |
//+------------------------------------------------------------------+
void CMemoryControl::OnClickButtonAdd(void)
  {
//--- erhöhen Sie die Größe des Arrays
   m_arr_size+=(int)m_combo_box_step.Value();
//--- versuchen Sie der Speicher für den aktuellen Array zu wählen
   if(CurrentArrayAdd())
     {
      //--- der Speicher ist gewählt, führen wir den aktuellen Zustand an dem Chart he
      m_lbl_memory_available.Text("Memory available = "+(string)TerminalInfoInteger(TE
      m_lbl_memory_used.Text("Memory used = "+(string)TerminalInfoInteger(TERMINAL_MEM
      m_lbl_array_size.Text("Array size = "+IntegerToString(m_arr_size));
      m_lbl_error.Text("");
     }
   else
     {
      //--- es misslang, der Speicher zu wählen, führen Sie die Nachtrichtung vom Fehl
      m_lbl_error.Text("Array is too large, error!");
      //--- geben Sie die vorherige Größe des Arrays zurück
      m_arr_size-=(int)m_combo_box_step.Value();
     }
  }
//+------------------------------------------------------------------+
//| Der Bearbeiter des Ereignisses auf die "Free" Schaltfläche zu    |
//| klicken                                                          |
//+------------------------------------------------------------------+
void CMemoryControl::OnClickButtonFree(void)
  {
//--- befreien Sie den Speichern des aktuellen Arrays
   CurrentArrayFree();
//--- führen wir den aktuellen Zustand an dem Chart heraus
   m_lbl_memory_available.Text("Memory available = "+(string)TerminalInfoInteger(TERMI
   m_lbl_memory_used.Text("Memory used = "+(string)TerminalInfoInteger(TERMINAL_MEMORY
```

© 2000-2025, MetaQuotes Ltd.

```
   m_lbl_array_size.Text("Array size = 0");
   m_lbl_error.Text("");
  }
//+------------------------------------------------------------------+
//| Der Bearbeiter des Ereignisses der Änderungsliste                |
//+------------------------------------------------------------------+
void CMemoryControl::OnChangeComboBoxType(void)
  {
//--- prüfen Sie, ob sich der Typ von Array geändert hat
   if(m_combo_box_type.Value()!=m_combo_box_type_value)
     {
      //--- befreien Sie den Speicher des aktuellen Arrays
      OnClickButtonFree();
      //--- arbeiten sie mit dem anderen Typ von Array
      m_combo_box_type_value=(int)m_combo_box_type.Value();
      //--- führen wir den neuen Typ des Arrays an dem Chart heraus
      if(m_combo_box_type_value==0)
         m_lbl_array_type.Text("Array type = char");
      if(m_combo_box_type_value==1)
         m_lbl_array_type.Text("Array type = int");
      if(m_combo_box_type_value==2)
         m_lbl_array_type.Text("Array type = float");
      if(m_combo_box_type_value==3)
         m_lbl_array_type.Text("Array type = double");
      if(m_combo_box_type_value==4)
         m_lbl_array_type.Text("Array type = long");
     }
  }
//--- das Objekt der Klasse von CMemoryControl
CMemoryControl ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- erstellen Sie das Dialog
   if(!ExtDialog.Create(0,"MemoryControl",0,X_START,Y_START,X_SIZE,Y_SIZE))
      return(INIT_FAILED);
//--- der Start
   ExtDialog.Run();
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayGetAsSeries

Prüft Richtung des Arrayindizierens.

```
bool  ArrayGetAsSeries(
   const void&  array[]    // Array für Prüfung
   );
```

Parameter

array

[in] Das geprüfte Array.

Rückgabewert

Gibt true zurück, wenn die Flagge AS_SERIES für das angegebene Array eingestellt wird, d.h. der Arrayzugang wird umgekehrt wie in einer Zeitreihe durchgeführt. Zeitreihe unterscheidet sich vom normalen Array dadurch, dass Indizieren der Elemente der Zeitreihe vom Arrayende zum Arrayanfang durchgeführt (von den neuesten Daten zu den ältesten).

Hinweis

Für Prüfung des Arrays auf Zugehörigkeit muss die Funktion ArrayIsSeries() verwendet werden. Arrays der Preisdaten, die als Eingabeparameter in die Funktion OnCalculate() übertragen werden, nicht haben Richtung des Indizierens wie in Zeitreihen. Die notwendige Richtung des Indizierens kann durch die Funktion ArraySetAsSeries() eingestellt werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "der Indikator rechnet die absoluten Werten des Unterschieds zwi
#property description "Open und Close oder High und Low, und zeichnet sie im abgesonde
#property description "in Form vom Histogramm."
//--- der Indikator Einstellungen
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_style1  STYLE_SOLID
#property indicator_width1  3
//--- Eingabeparameters
input bool InpAsSeries=true; // Die Richtung des Indizierens in der Indikator Puffer
input bool InpPrices=true;   // Preise für die Berechnung (true - Open,Close; false -
//--- Indikator Puffer
double ExtBuffer[];
//+------------------------------------------------------------------+
//| Die Berechnung der Indikatorwerte                                |
//+------------------------------------------------------------------+
void CandleSizeOnBuffer(const int rates_total,const int prev_calculated,
                        const double &first[],const double &second[],double &buffer[])
  {
//---Variable zur Berechnung den Anfang des Bars
   int start=prev_calculated;
//--- wen die Indikatorwerte auf vorhergehend Tick schon berechnet waren, so ist auf
   if(prev_calculated>0)
      start--;
//--- bestimmen Sie die Richtung der Indizierung in Arrays
   bool as_series_first=ArrayGetAsSeries(first);
   bool as_series_second=ArrayGetAsSeries(second);
   bool as_series_buffer=ArrayGetAsSeries(buffer);
//--- ändern Sie die Richtung der Indizierung auf die direkte, wenn nötig
   if(as_series_first)
      ArraySetAsSeries(first,false);
   if(as_series_second)
      ArraySetAsSeries(second,false);
   if(as_series_buffer)
      ArraySetAsSeries(buffer,false);
//--- berechnen Sie die Indikatorwerte
   for(int i=start;i<rates_total;i++)
      buffer[i]=MathAbs(first[i]-second[i]);
  }
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Indikator Puffers
   SetIndexBuffer(0,ExtBuffer);
//--- stellen Sie die Richtung der Indizierung in dem Indikator Puffer ein
   ArraySetAsSeries(ExtBuffer,InpAsSeries);
//--- prüfen wir für welche Preise es wird der Indikator berechnet
   if(InpPrices)
     {
      //--- Open und Close Preise
      PlotIndexSetString(0,PLOT_LABEL,"BodySize");
      //--- erstellen Sie die Farbe des Indikators
      PlotIndexSetInteger(0,PLOT_LINE_COLOR,clrOrange);
     }
   else
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- High und Low Preise
      PlotIndexSetString(0,PLOT_LABEL,"ShadowSize");
      //--- erstellen Sie die Farbe des Indikators
      PlotIndexSetInteger(0,PLOT_LINE_COLOR,clrDodgerBlue);
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- die Berechnung des Indikators je nach der Wert der Fahne
   if(InpPrices)
      CandleSizeOnBuffer(rates_total,prev_calculated,open,close,ExtBuffer);
   else
      CandleSizeOnBuffer(rates_total,prev_calculated,high,low,ExtBuffer);
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Zugang zu Zeitreihen, ArraySetAsSeries

© 2000-2025, MetaQuotes Ltd.

ArrayInitialize

Initialisiert das numerische Array durch den vorgegebenen Wert.

For initialization of an array of char type

```
int  ArrayInitialize(
   char    array[],     // initialisiertes array
   char    value        // der zu gestellte wert
   );
```

For initialization of an array of short type

```
int  ArrayInitialize(
   short   array[],     // initialisiertes array
   short   value        // der zu gestellte wert
   );
```

For initialization of an array of int type

```
int  ArrayInitialize(
   int     array[],     // initialisiertes array
   int     value        // der zu gestellte wert
   );
```

For initialization of an array of long type

```
int  ArrayInitialize(
   long    array[],     // initialisiertes array
   long    value        // der zu gestellte wert
   );
```

For initialization of an array of float type

```
int  ArrayInitialize(
   float   array[],     // initialisiertes array
   float   value        // der zu gestellte wert
   );
```

For initialization of an array of double type

```
int  ArrayInitialize(
   double  array[],     // initialisiertes array
   double  value        // der zu gestellte wert
   );
```

For initialization of an array of bool type

```
int  ArrayInitialize(
   bool    array[],     // initialisiertes array
   bool    value        // der zu gestellte wert
   );
```

© 2000-2025, MetaQuotes Ltd.

For initialization of an array of uint type

```
int  ArrayInitialize(
   uint    array[],     // initialisiertes array
   uint    value        // der zu gestellte wert
   );
```

Parameter

array[]

[out] Das numerische Array, das initialisiert werden muss.

value

[in] Neuer Wert, der für alle Elemente des Arrays eingestellt werden muss.

Rückgabewert

Anzahl der Elemente.

Hinweis

Die Funktion ArrayResize() erlaubt die Größe für ein Array mit einem Reserve vorzugeben, damit es größer wird ohne physische Umverteilung des Speichers. Das wird für Operationsgeschwindigkeit gemacht, denn Operationen der Speicherverteilung sind langsam genug.

Die Initialisierung des Arrays vom Ausdruck ArrayInitialize(array, init_val) bedeutet nicht die Initialisierung durch denselben Wert der Elemente der Reserve, die für dieses Array gewählt ist. Bei weiteren Vergrößerungen des Arrays *array* durch die Funktion ArrayResize() innerhalb des laufenden Reserves, am Ende des Arrays werden die Elemente einfügt, deren Werte nicht bestimmt sind und am öftesten sind sie nicht *init_val* gleich.

Beispiel:

```
void OnStart()
  {
//--- dynamisches Array
   double array[];
//--- geben wir die Größe des Arrays für 100 Elemente vor und reservieren wir noch ein
   ArrayResize(array,100,10);
//--- initialisieren wir die Elemente des Arrays durch den Wert EMPTY_VALUE=DBL_MAX
   ArrayInitialize(array,EMPTY_VALUE);
   Print("Werte der letzten 10 Elemente des Arrays nach der Initialisierung");
   for(int i=90;i<100;i++) printf("array[%d]=%G",i,array[i]);
//--- vergrößern wir das Array um 5 Elemente
   ArrayResize(array,105);
   Print("Werte der letzten 10 Elemente des Arrays nach ArrayResize(array,105)");
//--- Werte der letzten 5 Elemente wurden aus Reservepuffer erhalten
   for(int i=95;i<105;i++) printf("array[%d] = %G",i,array[i]);
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayFill

Füllt das numerische Array mit dem angegebenen Wert.

```
void  ArrayFill(
   void&  array[],        // Array
   int    start,          // Index des Anfangselements
   int    count,          // Anzahl der Elemente
   void   value           // Wert, der mit Array gefüllt wird
   );
```

Parameter

array[]

[out] Array des einfachen Typs (char, uchar, short, ushort, int, uint, long, ulong, bool, color, datetime, float, double).

start

[in] Index des Anfangselements (von welchem Element auszufüllen). Anbei wird die eingestellte Seriengröße Flagge nicht berücksichtigt.

count

[in] Die Anzahl der Elemente, die gefüllt werden muss.

value

[in] Wert, der mit Array gefüllt wird.

Rückgabewert

Keinen Rückgabewert.

Anmerkung

Beim Aufruf der ArrayFill() Funktion immer wird die gewöhnliche Richtung der Indexierung - von links nach rechts verstanden, d.h. die Veränderung den Zugang auf die Elemente des Arrays mit Hilfe der ArraySetAsSeries() Funktion wird nicht beachtet.

Das mehrdimensionale Array bei Bearbeitung von der Funktion ArrayFill werden eindimensional, zum Beispiel, wird das Array array[2][4] wie array[8], deshalb bei der Arbeit mit diesem Array zulässig vorgestellt, den Index des Anfangselementes gleich 5 zu bezeichnen. So wird der Aufruf von ArrayFill(array, 5, 2, 3.14) für das Array array[2][4] der Wert die 3.14 Elemente des Arrays array[1][1] und array[1][2] ausfüllen.

Beispiel:

```
void OnStart()
  {
//--- deklarieren wir ein dynamisches Array
   int a[];
//--- stellen wir die Größe ein
   ArrayResize(a,10);
//--- füllen wir die ersten 5 Elementen mit dem Wert 123
   ArrayFill(a,0,5,123);
//--- füllen wir 5 Elementen (seit 5.) mit dem Wert 456
   ArrayFill(a,5,5,456);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Die Werte aller Elemente ausgeben
   for(int i=0;i<ArraySize(a);i++) printf("a[%d] = %d",i,a[i]);
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayIsDynamic

Prüft, ob das Array dynamisch ist.

```
bool  ArrayIsDynamic(
   const void&  array[]    // das geprüfte Array
   );
```

Parameter

array[]

[in] Das geprüfte Array.

Rückgabewert

Gibt true zurück, wenn das angegebene Array dynamisch ist, anderenfalls false.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Dieser Indikator berechnet die Werte nicht, und einmal versucht
#property description "Der Anruf der Funktion ArrayFree() zu drei Arrays: dynamisch, s
#property description "dem Indikator Puffer. Die Ergebnisse werden in die Zeitschrift
//--- der Indikator Einstellungen
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- globale Variablen
double ExtDynamic[];   // das dynamische Array
double ExtStatic[100]; // das statische Array
bool   ExtFlag=true;   // die Flagge
double ExtBuff[];      // der Indikator Puffer
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- trennen Sie den Speicher für den Array ab
   ArrayResize(ExtDynamic,100);
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,ExtBuff);
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const int begin,
                const double &price[])
  {
//--- führen Sie die Analyse einmal durch
   if(ExtFlag)
     {
      //--- versuchen Sie den Speicher für die Arrays zu befreien
      //--- 1. Das dynamische Array
      Print("+============================+");
      Print("1. Überprüfung des dynamischen Arrays:");
      Print("Die Größe bis zur Befreiung des Speichers = ",ArraySize(ExtDynamic));
      Print("Es ist das dynamische Array = ",ArrayIsDynamic(ExtDynamic) ? "Ja" : "Nein
      //--- versuchen Sie den Speicher des Arrays zu befreien
      ArrayFree(ExtDynamic);
      Print("Die Größe nach der Befreiung des Speichers = ",ArraySize(ExtDynamic));
      //--- 2. Das statistisches Array
      Print("2. Überprüfung des statischen Arrays:");
      Print("Die Größe bis zur Befreiung des Speichers = ",ArraySize(ExtStatic));
      Print("Es ist das dynamische Array = ",ArrayIsDynamic(ExtStatic) ? "Ja" : "Nein"
      //--- versuchen Sie den Speicher des Arrays zu befreien
      ArrayFree(ExtStatic);
      Print("Die Größe nach der Befreiung des Speichers = ",ArraySize(ExtStatic));
      //--- 3. Indikator Puffer
      Print("3. Überprüfung des Indikator Puffers:");
      Print("Die Größe bis zur Befreiung des Speichers = ",ArraySize(ExtBuff));
      Print("Es ist das dynamische Array = ",ArrayIsDynamic(ExtBuff) ? "Ja" : "Nein");
      //--- versuchen Sie den Speicher des Arrays zu befreien
      ArrayFree(ExtBuff);
      Print("Die Größe nach der Befreiung des Speichers = ",ArraySize(ExtBuff));
      //--- ändern Sie den Wert der Flagge
      ExtFlag=false;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

ArrayIsSeries

Prüft, ob das Array Zeitreihe ist.

```
bool  ArrayIsSeries(
   const void&  array[]    // das geprüfte Array
   );
```

Parameter

array[]

[in] Das geprüfte Array.

Rückgabewert

Gibt true zurück, wenn das geprüfte Array ein Array-Zeitreihe ist, anderenfalls false. Arrays, die als Parameter der Funktion OnCalculate(übertragen werden, müssen auf Zugangsordnung zu Elementen des Arrays durch die Funktion ArrayGetAsSeries() geprüft werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Label1"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Indikator Puffers
double         Label1Buffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,Label1Buffer,INDICATOR_DATA);
//---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---
   if(ArrayIsSeries(open))
      Print("open[] is timeseries");
   else
      Print("open[] is not timeseries!!!");
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

ArrayMaximum

Sucht im eindimensionalen numerischen Array das maximale Element.

```
int  ArrayMaximum(
   const void&   array[],             // Array für die Suche
   int           start=0,             // Anfangsindex für die Suche
   int           count=WHOLE_ARRAY    // Anzahl der geprüften Elemente
   );
```

Parameter

array[]

[in] Numerisches Array, in dem Suche durchgeführt wird.

start=0

[in] Anfangsindex für die Suche.

count=WHOLE_ARRAY

[in] Anzahl der Elementen für die Suche. Als Voreinstellung wird das ganze Array kopiert (count=WHOLE_ARRAY).

Rückgabewert

Funktion gibt Index des gefundenen Elementes unter Beachtung der Seriengröße des Arrays. Beim Misserfolg gibt die Funktion -1 zurück.

Hinweis

Die Suche nach dem maximalen Element erfolgt unter Berücksichtigung des Wertes des AS_SERIES Parameters.

Die ArrayMaximum und ArrayMinimum Funktionen nehmen als Parameter das Array mit beliebig vielen Dimensionen an, dabei wird nur in der ersten (zero) Dimension gesucht.

Beispiel:

```
#property description "Der Indikator zeigt die Kerzen mit dem älteren auf die aktuelle
//--- Indikator Einstellungen
#property indicator_chart_window
#property indicator_buffers 16
#property indicator_plots   8
//---- plot 1
#property indicator_label1  "BearBody"
#property indicator_color1  clrSeaGreen,clrSeaGreen
//---- plot 2
#property indicator_label2  "BearBodyEnd"
#property indicator_color2  clrSeaGreen,clrSeaGreen
//---- plot 3
#property indicator_label3  "BearShadow"
#property indicator_color3  clrSalmon,clrSalmon
//---- plot 4
#property indicator_label4  "BearShadowEnd"
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_color4  clrSalmon,clrSalmon
//---- plot 5
#property indicator_label5  "BullBody"
#property indicator_color5  clrOlive,clrOlive
//---- plot 6
#property indicator_label6  "BullBodyEnd"
#property indicator_color6  clrOlive,clrOlive
//---- plot 7
#property indicator_label7  "BullShadow"
#property indicator_color7  clrSkyBlue,clrSkyBlue
//---- plot 8
#property indicator_label8  "BullShadowEnd"
#property indicator_color8  clrSkyBlue,clrSkyBlue
//--- vordefinierte Konstante
#define INDICATOR_EMPTY_VALUE 0.0
//--- Eingabeparameters
input ENUM_TIMEFRAMES InpPeriod=PERIOD_H4;              // Der Zeitrahmen für die Bere
input datetime        InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum der Analys
//--- die Indikator Puffers für die Bären Kerzen
double   ExtBearBodyFirst[];
double   ExtBearBodySecond[];
double   ExtBearBodyEndFirst[];
double   ExtBearBodyEndSecond[];
double   ExtBearShadowFirst[];
double   ExtBearShadowSecond[];
double   ExtBearShadowEndFirst[];
double   ExtBearShadowEndSecond[];
//--- die Indikator Puffers für die Bullen Kerzen
double   ExtBullBodyFirst[];
double   ExtBullBodySecond[];
double   ExtBullBodyEndFirst[];
double   ExtBullBodyEndSecond[];
double   ExtBullShadowFirst[];
double   ExtBullShadowSecond[];
double   ExtBullShadowEndFirst[];
double   ExtBullShadowEndSecond[];
//--- globale Variablen
datetime ExtTimeBuff[];      // Puffer für Zeit mit der höheren Zeitrahmen
int      ExtSize=0;          // die Größe des Zeitpuffers
int      ExtCount=0;         // der Index innerhalb des Zeitpuffers
int      ExtStartPos=0;      // die Anfangsposition für die Berechnung des Indikators
bool     ExtStartFlag=true;  // Die Hilfsflagge für das Erhalten der Anfangsposition
datetime ExtCurrentTime[1];  // die letzte Zeit wenn der Bar auf dem älteren Zeitrahme
datetime ExtLastTime;        // die letzte Zeit auf dem älteren Zeitrahmen, für den di
bool     ExtBearFlag=true;   // die Flagge für die Bestimmung der Ordnung der Speiche
bool     ExtBullFlag=true;   // die Flagge für die Bestimmung der Ordnung der Speiche
int      ExtIndexMax=0;      // der Index des maximalen Elementes in dem Array
int      ExtIndexMin=0;      // der Index des minimalen Elementes in dem Array
int      ExtDirectionFlag=0; // die Richtung Die Bewegungen des Preises in der aktuel
```

© 2000-2025, MetaQuotes Ltd.

```
//--- der Einzug zwischen dem Eröffnungspreis und Schlusspreis der Kerze für die richt
const double ExtEmptyBodySize=0.2*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//+------------------------------------------------------------------+
//| Zeichnung der Hauptteil der Kerze                                |
//+------------------------------------------------------------------+
void FillCandleMain(const double &open[],const double &close[],
                    const double &high[],const double &low[],
                    const int start,const int last,const int fill_index,
                    int &index_max,int &index_min)
  {
//--- finden Sie die Indexe der maximalen und minimalen Elemente in den Arrays
   index_max=ArrayMaximum(high,ExtStartPos,last-start+1); // Maximum in Hoch
   index_min=ArrayMinimum(low,ExtStartPos,last-start+1);  // Minimum in Tief
//--- bestimmen Sie, wie viel Bars auf den aktuellen Zeitrahmen wir werden zeichnen
   int count=fill_index-start+1;
//--- wenn der Schlusspreis am ersten Bar mehr als der Schlusspreis am letzten Bar - e
   if(open[start]>close[last])
     {
      //--- wenn früher die Kerze Bullen war, so freimachen wir die Werte den Bären In
      if(ExtDirectionFlag!=-1)
         ClearCandle(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShac
      //--- Bären Kerze
      ExtDirectionFlag=-1;
      //--- bilden Sie die Kerze
      FormCandleMain(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShac
                     close[last],high[index_max],low[index_min],start,count,ExtBearFla
      //--- Funktionsausgang
      return;
     }
//--- wenn der Schlusspreis am ersten Bar weniger als der Schlusspreis am letzten - es
   if(open[start]<close[last])
     {
      //--- wenn früher die Kerze Bären war, so freimachen wir die Werte den Bären Ind
      if(ExtDirectionFlag!=1)
         ClearCandle(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShac
      //--- Bullen Kerze
      ExtDirectionFlag=1;
      //--- bilden Sie die Kerze
      FormCandleMain(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShac
                     open[start],high[index_max],low[index_min],start,count,ExtBullFla
      //--- Funktionsausgang
      return;
     }
//--- wenn sie in diesem Teil der Funktion, so bedeutet es, so stellt sich der Eröffnu
//--- dem Schlusspreis am letzten Bar; wir werden solche Kerze Bären halten
//--- wenn früher die Kerze Bullen war, so freimachen wir die Werte der Bullen Indikat
   if(ExtDirectionFlag!=-1)
      ClearCandle(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShadowS
//--- Bären Kerze
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtDirectionFlag=-1;
//--- wenn Schlusspreise und Eröffnungspreise gleich sind, dann verschoben Sie für die
   if(high[index_max]!=low[index_min])
      FormCandleMain(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShad
                     open[start]-ExtEmptyBodySize,high[index_max],low[index_min],start
   else
      FormCandleMain(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShad
                     open[start],open[start]-ExtEmptyBodySize,high[index_max],
                     high[index_max]-ExtEmptyBodySize,start,count,ExtBearFlag);
  }
//+------------------------------------------------------------------+
//| Zeichnung das Ende der Kerze                                     |
//+------------------------------------------------------------------+
void FillCandleEnd(const double &open[],const double &close[],
                   const double &high[],const double &low[],
                   const int start,const int last,const int fill_index,
                   const int index_max,const int index_min)
  {
//--- wenn es nur eine Bar gibt, dann wir nicht zeichnen
   if(last-start==0)
      return;
//--- wenn der Schlusspreis am ersten Bar mehr als der Schlusspreis am letzten Bar - e
   if(open[start]>close[last])
     {
      //--- bilden Sie das Ende der Kerze
      FormCandleEnd(ExtBearBodyEndFirst,ExtBearBodyEndSecond,ExtBearShadowEndFirst,Ext
                    open[start],close[last],high[index_max],low[index_min],fill_index,
      //--- Funktionsausgang
      return;
     }
//--- wenn der Schlusspreis am ersten Bar weniger als der Schlusspreis am letzten - es
   if(open[start]<close[last])
     {
      //--- bilden Sie das Ende der Kerze
      FormCandleEnd(ExtBullBodyEndFirst,ExtBullBodyEndSecond,ExtBullShadowEndFirst,Ext
                    close[last],open[start],high[index_max],low[index_min],fill_index,
      //--- Funktionsausgang
      return;
     }
//--- wenn sie in diesem Teil der Funktion, so bedeutet es, so stellt sich der Eröffnu
//--- dem Schlusspreis am letzten Bar; wir werden solche Kerze Bären halten
//--- bilden Sie das Ende der Kerze
   if(high[index_max]!=low[index_min])
      FormCandleEnd(ExtBearBodyEndFirst,ExtBearBodyEndSecond,ExtBearShadowEndFirst,Ext
                    open[start]-ExtEmptyBodySize,high[index_max],low[index_min],fill_i
   else
      FormCandleEnd(ExtBearBodyEndFirst,ExtBearBodyEndSecond,ExtBearShadowEndFirst,Ext
                    open[start]-ExtEmptyBodySize,high[index_max],high[index_max]-ExtEr
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Überprüfen Sie die Indikator Periode
   if(!CheckPeriod((int)Period(),(int)InpPeriod))
      return(INIT_PARAMETERS_INCORRECT);
//--- die Anzeige der Preisdaten auf dem Vordergrund
   ChartSetInteger(0,CHART_FOREGROUND,0,1);
//--- Bindung von Indikator Puffers
   SetIndexBuffer(0,ExtBearBodyFirst);
   SetIndexBuffer(1,ExtBearBodySecond);
   SetIndexBuffer(2,ExtBearBodyEndFirst);
   SetIndexBuffer(3,ExtBearBodyEndSecond);
   SetIndexBuffer(4,ExtBearShadowFirst);
   SetIndexBuffer(5,ExtBearShadowSecond);
   SetIndexBuffer(6,ExtBearShadowEndFirst);
   SetIndexBuffer(7,ExtBearShadowEndSecond);
   SetIndexBuffer(8,ExtBullBodyFirst);
   SetIndexBuffer(9,ExtBullBodySecond);
   SetIndexBuffer(10,ExtBullBodyEndFirst);
   SetIndexBuffer(11,ExtBullBodyEndSecond);
   SetIndexBuffer(12,ExtBullShadowFirst);
   SetIndexBuffer(13,ExtBullShadowSecond);
   SetIndexBuffer(14,ExtBullShadowEndFirst);
   SetIndexBuffer(15,ExtBullShadowEndSecond);
//--- stellen Sie einige Werte der Eigenschaften für die Konstruktion des Indikators e
   for(int i=0;i<8;i++)
     {
      PlotIndexSetInteger(i,PLOT_DRAW_TYPE,DRAW_FILLING); // der Typ der graphischen P
      PlotIndexSetInteger(i,PLOT_LINE_STYLE,STYLE_SOLID); // der Stil der Linie auf di
      PlotIndexSetInteger(i,PLOT_LINE_WIDTH,1);           // die Dicke der Linie auf d
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const int &spread[])
  {
//--- wenn es noch keine berechnete Bars gibt
   if(prev_calculated==0)
     {
      //--- bekommen Sie die Zeit wenn der Bar auf dem älteren Zeitrahmen erschienen w
      if(!GetTimeData())
         return(0);
     }
//--- stellen Sie die gerade Richtung der Indexierung ein
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
   ArraySetAsSeries(open,false);
   ArraySetAsSeries(close,false);
//--- Variable zur Berechnung den Anfang des Bars
   int start=prev_calculated;
//--- wenn der Bar gebildet ist, so berechnen wir den Indikatorwert darauf
   if(start!=0 && start==rates_total)
      start--;
//--- die Zyklus von die Berechnung der Indikatorwerte
   for(int i=start;i<rates_total;i++)
     {
      //--- füllen Sie die i-ten Elemente des Indikator Puffers mit dem leeren Werten
      FillIndicatorBuffers(i);
      //--- führen Sie die Berechnung für die Bars seit dem Datum InpDateStart durch
      if(time[i]>=InpDateStart)
        {
         //--- zum ersten Mal bestimmen Sie eine Position mit der wir werden die Werte
         if(ExtStartFlag)
           {
            //--- speichern Sie der Index des Anfangsbar
            ExtStartPos=i;
            //--- bestimmen Sie das erste Datum auf dem älteren Zeitrahmen, das mehr a
            while(time[i]>=ExtTimeBuff[ExtCount])
               if(ExtCount<ExtSize-1)
                  ExtCount++;
            //--- ändern Sie die Flaggewert, um in diesen Block nicht mehr zu geraten
            ExtStartFlag=false;
           }
         //--- prüfen Sie, ob es noch Elemente in das Array sind
         if(ExtCount<ExtSize)
           {
            //--- warten Sie, wenn der Zeitwert auf dem aktuellen Zeitrahmen der Wert
            if(time[i]>=ExtTimeBuff[ExtCount])
              {
               //--- zeichnen Sie den Hauptteil der Kerze (ohne Zeichnung zwischen der
               FillCandleMain(open,close,high,low,ExtStartPos,i-1,i-2,ExtIndexMax,ExtI
               //--- zeichnen Sie das Ende der Kerze (das Gebiet zwischen der letzten
```

© 2000-2025, MetaQuotes Ltd.

```
                FillCandleEnd(open,close,high,low,ExtStartPos,i-1,i-1,ExtIndexMax,ExtIn
                //--- bewegen Sie die Anfangsposition für die Zeichnung der nächsten Ke
                ExtStartPos=i;
                //--- erhöhen Sie der Array Zähler
                ExtCount++;
               }
             else
                continue;
            }
          else
            {
             //--- der Fehlerwert zu stürzen
             ResetLastError();
             //--- bekommen Sie die letzte Datum auf dem älteren Zeitrahmen
             if(CopyTime(Symbol(),InpPeriod,0,1,ExtCurrentTime)==-1)
               {
                Print("Fehler beim Kopieren die Daten, Kode = ",GetLastError());
                return(0);
              ;}
             //--- wenn das neue Datum mehr ist, so beenden wir die Bildung der Kerze
             if(ExtCurrentTime[0]>ExtLastTime)
               {
                //--- freimachen Sie das Gebiet zwischen der letzten und vorletzten Bar
                ClearEndOfBodyMain(i-1);
                //--- zeichnen Sie dieses Gebiet mit dem Hilfs-Indikator Puffers
                FillCandleEnd(open,close,high,low,ExtStartPos,i-1,i-1,ExtIndexMax,ExtIn
                //--- bewegen Sie die Anfangsposition für die Zeichnung der nächsten Ke
                ExtStartPos=i;
                //--- stürzen Sie die Fahne der Preisrichtung
                ExtDirectionFlag=0;
                //--- speichern Sie das neue letzte Datum
                ExtLastTime=ExtCurrentTime[0];
               }
             else
               {
                //--- bilden Sie die Kerze
                FillCandleMain(open,close,high,low,ExtStartPos,i,i,ExtIndexMax,ExtIndex
               }
            }
         }
      }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Überprüfen Sie die Indikator Periode auf die Korrektheit         |
//+------------------------------------------------------------------+
bool CheckPeriod(int current_period,int high_period)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die Periode des Indikators soll mehr als der Zeitrahmen sein, auf dem er dargest
   if(current_period>=high_period)
     {
      Print("Fehler! Der Wert von der Indikator Periode soll mehr als der aktuellen Ze
      return(false);
     }
//--- wenn die Periode des Indikators eine Woche oder einer Monat ist, so ist die Peri
   if(high_period>32768)
      return(true);
//--- bringen Sie die Werte der Perioden bis zu Minuten
   if(high_period>30)
      high_period=(high_period-16384)*60;
   if(current_period>30)
      current_period=(current_period-16384)*60;
//--- die Periode des Indikators soll Fach der Zeitrahmen sein, auf dem er dargestellt
   if(high_period%current_period!=0)
     {
      Print("Fehler! Der Wert der Periode des Indikators soll Fach der aktuellen Zeitr
      return(false);
     }
//--- die Periode des Indikators soll der Wert der Zeitrahmen übertreten, auf dem er i
   if(high_period/current_period<3)
     {
      Print("Fehler! Der Wert der Periode des Indikators soll der Wert der aktuellen Z
      return(false);
     }
//--- die Periode des Indikators ist für den aktuellen Zeitrahmen korrekt
   return(true);
  }
//+------------------------------------------------------------------+
//| Bekommen Sie die Daten der Zeit auf dem älteren Zeitrahmen       |
//+------------------------------------------------------------------+
bool GetTimeData(void)
  {
//--- der Fehlerwert zu stürzen
   ResetLastError();
//--- kopieren Sie alle Daten auf die aktuelle Zeit
   if(CopyTime(Symbol(),InpPeriod,InpDateStart,TimeCurrent(),ExtTimeBuff)==-1)
     {
      //--- der Fehlerwert zu stürzen
      int code=GetLastError();
      //--- drucken Sie der Fehlertext
      PrintFormat("Fehler beim Kopieren die Daten! %s",code==4401
                  ? "Die Geschichte wird noch beladen!"
                  : "Kode = "+IntegerToString(code));
      //--- geben Sie false für den nochmaligen Versuch der Beladung der Daten zurück
      return(false);
     }
//--- bekommen Sie die Größe des Arrays
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtSize=ArraySize(ExtTimeBuff);
//--- stellen Sie der Index des Zyklus für das Array gleich der Null ein
   ExtCount=0;
//--- stellen Sie die Position der aktuellen Kerze in diesem Zeitrahmen gleich der Nul
   ExtStartPos=0;
   ExtStartFlag=true;
//--- bekommen Sie der letzten Wert der Zeit auf dem älteren Zeitrahmen
   ExtLastTime=ExtTimeBuff[ExtSize-1];
//--- die erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion bildet den Hauptteil der Kerze. Je nach dem Wert    |
//| der Flagge, die Funktion bestimmt, welche Daten sich in welche   |
//| Arrays für die korrekte Abbildung einschreiben sollen.           |
//+------------------------------------------------------------------+
void FormCandleMain(double &body_fst[],double &body_snd[],
                    double &shadow_fst[],double &shadow_snd[],
                    const double fst_value,const double snd_value,
                    const double fst_extremum,const double snd_extremum,
                    const int start,const int count,const bool flag)
  {
//--- prüfen Sie der Wert der Flagge
   if(flag)
     {
      //--- bilden Sie der Körper der Kerze
      FormMain(body_fst,body_snd,fst_value,snd_value,start,count);
      //--- bilden Sie der Schatten der Kerze
      FormMain(shadow_fst,shadow_snd,fst_extremum,snd_extremum,start,count);
    ;}
   else
     {
      //--- bilden Sie der Körper der Kerze
      FormMain(body_fst,body_snd,snd_value,fst_value,start,count);
      //--- bilden Sie der Schatten der Kerze
      FormMain(shadow_fst,shadow_snd,snd_extremum,fst_extremum,start,count);
     }
  }
//+------------------------------------------------------------------+
//| Die Funktion bildet das Ende der Kerze. Je nach dem Wert der     |
//| Flagge, die Funktion bestimmt, welche Daten sich in welche Arrays|
//| für die korrekte Abbildung einschreiben sollen.                  |
//+------------------------------------------------------------------+
void FormCandleEnd(double &body_fst[],double &body_snd[],
                   double &shadow_fst[],double &shadow_snd[],
                   const double fst_value,const double snd_value,
                   const double fst_extremum,const double snd_extremum,
                   const int end,bool &flag)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- prüfen Sie der Wert der Flagge
   if(flag)
     {
      //--- bilden Sie das Ende der Körper der Kerze
      FormEnd(body_fst,body_snd,fst_value,snd_value,end);
      //--- bilden Sie das Ende der Schatten der Kerze
      FormEnd(shadow_fst,shadow_snd,fst_extremum,snd_extremum,end);
      //--- ändern Sie der Wert der Flagge gegen das Entgegengesetzte
      flag=false;
     }
   else
     {
      //--- bilden Sie das Ende der Körper der Kerze
      FormEnd(body_fst,body_snd,snd_value,fst_value,end);
      //--- bilden Sie das Ende der Schatten der Kerze
      FormEnd(shadow_fst,shadow_snd,snd_extremum,fst_extremum,end);
      //--- ändern Sie der Wert der Flagge gegen das Entgegengesetzte
      flag=true;
     }
  }
//+------------------------------------------------------------------+
//| Die Entleerung das Ende der Kerze (das Gebiet zwischen der       |
//| letzten und vorletzten Bar)                                      |
//+------------------------------------------------------------------+
void ClearEndOfBodyMain(const int ind)
  {
   ClearCandle(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShadowSecc
   ClearCandle(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShadowSecc
  }
//+------------------------------------------------------------------+
//| Die Entleerung der Kerze                                         |
//+------------------------------------------------------------------+
void ClearCandle(double &body_fst[],double &body_snd[],double &shadow_fst[],
                 double &shadow_snd[],const int start,const int count)
  {
//--- die Überprüfung
   if(count!=0)
     {
      //--- füllen Sie der Indikator Puffers mit dem leeren Wert aus
      ArrayFill(body_fst,start,count,INDICATOR_EMPTY_VALUE);
      ArrayFill(body_snd,start,count,INDICATOR_EMPTY_VALUE);
      ArrayFill(shadow_fst,start,count,INDICATOR_EMPTY_VALUE);
      ArrayFill(shadow_snd,start,count,INDICATOR_EMPTY_VALUE);
     }
  }
//+------------------------------------------------------------------+
//| Die Bildung des Hauptteiles der Kerze                            |
//+------------------------------------------------------------------+
void FormMain(double &fst[],double &snd[],const double fst_value,
```

© 2000-2025, MetaQuotes Ltd.

```
                const double snd_value,const int start,const int count)
  {
//--- die Überprüfung
   if(count!=0)
     {
      //--- füllen Sie die Indikator Puffers Werte aus
      ArrayFill(fst,start,count,fst_value);
      ArrayFill(snd,start,count,snd_value);
     }
  }
//+------------------------------------------------------------------+
//| Die Bildung das Ende der Kerze                                   |
//+------------------------------------------------------------------+
void FormEnd(double &fst[],double &snd[],const double fst_value,
             const double snd_value,const int last)
  {
//--- füllen Sie die Indikator Puffers Werte aus
   ArrayFill(fst,last-1,2,fst_value);
   ArrayFill(snd,last-1,2,snd_value);
  }
//+------------------------------------------------------------------+
//| Füllen Sie die i-ten Elemente des Indikator Puffers mit dem      |
//| leeren Werte aus                                                 |
//+------------------------------------------------------------------+
void FillIndicatorBuffers(const int i)
  {
//--- stellen Sie der leer Wert in die Zelle der Indikator Puffers ein
   ExtBearBodyFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearBodySecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBearBodyEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearBodyEndSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowEndSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodyFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodySecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodyEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodyEndSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowEndSecond[i]=INDICATOR_EMPTY_VALUE;
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayMinimum

Sucht nach dem minimalen Element in der ersten Dimension eines mehrdimensionalen Arrays.

```
int  ArrayMinimum(
   const void&   array[],             // das Array für die Suche
   int           start=0,             // Anfangsindex für die Suche
   int           count=WHOLE_ARRAY    // Zahl der geprüften Elementen
   );
```

Parameter

array[]

[in] Numerisches Array, in dem die Suche durchgeführt wird.

start=0

[in] Anfangsindex für die Prüfung.

count=WHOLE_ARRAY

[in] Zahl der Elementen für Suche. Als Voreinstellung sucht im ganzen Array (count=WHOLE_ARRAY).

Rückgabewert

Funktion gibt Index des gefundenen Elementes unter Beachtung der Seriengröße des Arrays. Beim Misserfolg gibt die Funktion -1 zurück.

Hinweis

Suche nach dem minimalen Element unter Berücksichtigung des Wertes des AS_SERIES Parameters.

Die ArrayMaximum und ArrayMinimum Funktionen nehmen als Parameter das Array mit beliebig vielen Dimensionen an, dabei wird nur in der ersten (zero) Dimension gesucht.

Beispiel:

```
#property description "Der Indikator zeigt die Kerzen mit dem älteren auf die aktuelle
//--- Indikator Einstellungen
#property indicator_chart_window
#property indicator_buffers 16
#property indicator_plots   8
//---- plot 1
#property indicator_label1  "BearBody"
#property indicator_color1  clrSeaGreen,clrSeaGreen
//---- plot 2
#property indicator_label2  "BearBodyEnd"
#property indicator_color2  clrSeaGreen,clrSeaGreen
//---- plot 3
#property indicator_label3  "BearShadow"
#property indicator_color3  clrSalmon,clrSalmon
//---- plot 4
#property indicator_label4  "BearShadowEnd"
#property indicator_color4  clrSalmon,clrSalmon
```

© 2000-2025, MetaQuotes Ltd.

```
//---- plot 5
#property indicator_label5  "BullBody"
#property indicator_color5  clrOlive,clrOlive
//---- plot 6
#property indicator_label6  "BullBodyEnd"
#property indicator_color6  clrOlive,clrOlive
//---- plot 7
#property indicator_label7  "BullShadow"
#property indicator_color7  clrSkyBlue,clrSkyBlue
//---- plot 8
#property indicator_label8  "BullShadowEnd"
#property indicator_color8  clrSkyBlue,clrSkyBlue
//--- vordefinierte Konstante
#define INDICATOR_EMPTY_VALUE 0.0
//--- Eingabeparameters
input ENUM_TIMEFRAMES InpPeriod=PERIOD_H4;              // Der Zeitrahmen für die Bere
input datetime        InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum der Analys
//--- die Indikator Puffers für die Bären Kerzen
double   ExtBearBodyFirst[];
double   ExtBearBodySecond[];
double   ExtBearBodyEndFirst[];
double   ExtBearBodyEndSecond[];
double   ExtBearShadowFirst[];
double   ExtBearShadowSecond[];
double   ExtBearShadowEndFirst[];
double   ExtBearShadowEndSecond[];
//--- die Indikator Puffers für die Bullen Kerzen
double   ExtBullBodyFirst[];
double   ExtBullBodySecond[];
double   ExtBullBodyEndFirst[];
double   ExtBullBodyEndSecond[];
double   ExtBullShadowFirst[];
double   ExtBullShadowSecond[];
double   ExtBullShadowEndFirst[];
double   ExtBullShadowEndSecond[];
//--- globale Variablen
datetime ExtTimeBuff[];      // Puffer für Zeit mit der höheren Zeitrahmen
int      ExtSize=0;          // die Größe des Zeitpuffers
int      ExtCount=0;         // der Index innerhalb des Zeitpuffers
int      ExtStartPos=0;      // die Anfangsposition für die Berechnung des Indikators
bool     ExtStartFlag=true;  // Die Hilfsflagge für das Erhalten der Anfangsposition
datetime ExtCurrentTime[1];  // die letzte Zeit wenn der Bar auf dem älteren Zeitrahme
datetime ExtLastTime;        // die letzte Zeit auf dem älteren Zeitrahmen, für den di
bool     ExtBearFlag=true;   // die Flagge für die Bestimmung der Ordnung der Speicher
bool     ExtBullFlag=true;   // die Flagge für die Bestimmung der Ordnung der Speicher
int      ExtIndexMax=0;      // der Index des maximalen Elementes in dem Array
int      ExtIndexMin=0;      // der Index des minimalen Elementes in dem Array
int      ExtDirectionFlag=0; // die Richtung Die Bewegungen des Preises in der aktuell
//--- der Einzug zwischen dem Eröffnungspreis und Schlusspreis der Kerze für die richt
```

© 2000-2025, MetaQuotes Ltd.

```
const double ExtEmptyBodySize=0.2*SymbolInfoDouble(Symbol(),SYMBOL_POINT);
//+------------------------------------------------------------------+
//| Zeichnung der Hauptteil der Kerze                                |
//+------------------------------------------------------------------+
void FillCandleMain(const double &open[],const double &close[],
                    const double &high[],const double &low[],
                    const int start,const int last,const int fill_index,
                    int &index_max,int &index_min)
  {
//--- finden Sie die Indexe der maximalen und minimalen Elemente in den Arrays
   index_max=ArrayMaximum(high,ExtStartPos,last-start+1); // Maximum in Hoch
   index_min=ArrayMinimum(low,ExtStartPos,last-start+1);  // Minimum in Tief
//--- bestimmen Sie, wie viel Bars auf den aktuellen Zeitrahmen wir werden zeichnen
   int count=fill_index-start+1;
//--- wenn der Schlusspreis am ersten Bar mehr als der Schlusspreis am letzten Bar - e
   if(open[start]>close[last])
     {
      //--- wenn früher die Kerze Bullen war, so freimachen wir die Werte den Bären In
      if(ExtDirectionFlag!=-1)
         ClearCandle(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShac
      //--- Bären Kerze
      ExtDirectionFlag=-1;
      //--- bilden Sie die Kerze
      FormCandleMain(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShac
                     close[last],high[index_max],low[index_min],start,count,ExtBearFla
      //--- Funktionsausgang
      return;
     }
//--- wenn der Schlusspreis am ersten Bar weniger als der Schlusspreis am letzten - es
   if(open[start]<close[last])
     {
      //--- wenn früher die Kerze Bären war, so freimachen wir die Werte den Bären Inc
      if(ExtDirectionFlag!=1)
         ClearCandle(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShac
      //--- Bullen Kerze
      ExtDirectionFlag=1;
      //--- bilden Sie die Kerze
      FormCandleMain(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShac
                     open[start],high[index_max],low[index_min],start,count,ExtBullFla
      //--- Funktionsausgang
      return;
     }
//--- wenn sie in diesem Teil der Funktion, so bedeutet es, so stellt sich der Eröffnu
//--- dem Schlusspreis am letzten Bar; wir werden solche Kerze Bären halten
//--- wenn früher die Kerze Bullen war, so freimachen wir die Werte der Bullen Indikat
   if(ExtDirectionFlag!=-1)
      ClearCandle(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShadowS
//--- Bären Kerze
   ExtDirectionFlag=-1;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn Schlusspreise und Eröffnungspreise gleich sind, dann verschoben Sie für die
   if(high[index_max]!=low[index_min])
      FormCandleMain(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShad
                     open[start]-ExtEmptyBodySize,high[index_max],low[index_min],start
   else
      FormCandleMain(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShad
                     open[start],open[start]-ExtEmptyBodySize,high[index_max],
                     high[index_max]-ExtEmptyBodySize,start,count,ExtBearFlag);
  }
//+------------------------------------------------------------------+
//| Zeichnung das Ende der Kerze                                     |
//+------------------------------------------------------------------+
void FillCandleEnd(const double &open[],const double &close[],
                   const double &high[],const double &low[],
                   const int start,const int last,const int fill_index,
                   const int index_max,const int index_min)
  {
//--- wenn es nur eine Bar gibt, dann wir nicht zeichnen
   if(last-start==0)
      return;
//--- wenn der Schlusspreis am ersten Bar mehr als der Schlusspreis am letzten Bar - e
   if(open[start]>close[last])
     {
      //--- bilden Sie das Ende der Kerze
      FormCandleEnd(ExtBearBodyEndFirst,ExtBearBodyEndSecond,ExtBearShadowEndFirst,Ext
                    open[start],close[last],high[index_max],low[index_min],fill_index,
      //--- Funktionsausgang
      return;
     }
//--- wenn der Schlusspreis am ersten Bar weniger als der Schlusspreis am letzten - es
   if(open[start]<close[last])
     {
      //--- bilden Sie das Ende der Kerze
      FormCandleEnd(ExtBullBodyEndFirst,ExtBullBodyEndSecond,ExtBullShadowEndFirst,Ext
                    close[last],open[start],high[index_max],low[index_min],fill_index,
      //--- Funktionsausgang
      return;
     }
//--- wenn sie in diesem Teil der Funktion, so bedeutet es, so stellt sich der Eröffnu
//--- dem Schlusspreis am letzten Bar; wir werden solche Kerze Bären halten
//--- bilden Sie das Ende der Kerze
   if(high[index_max]!=low[index_min])
      FormCandleEnd(ExtBearBodyEndFirst,ExtBearBodyEndSecond,ExtBearShadowEndFirst,Ext
                    open[start]-ExtEmptyBodySize,high[index_max],low[index_min],fill_i
   else
      FormCandleEnd(ExtBearBodyEndFirst,ExtBearBodyEndSecond,ExtBearShadowEndFirst,Ext
                    open[start]-ExtEmptyBodySize,high[index_max],high[index_max]-ExtEr
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Überprüfen Sie die Indikator Periode
   if(!CheckPeriod((int)Period(),(int)InpPeriod))
      return(INIT_PARAMETERS_INCORRECT);
//--- die Anzeige der Preisdaten auf dem Vordergrund
   ChartSetInteger(0,CHART_FOREGROUND,0,1);
//--- Bindung von Indikator Puffers
   SetIndexBuffer(0,ExtBearBodyFirst);
   SetIndexBuffer(1,ExtBearBodySecond);
   SetIndexBuffer(2,ExtBearBodyEndFirst);
   SetIndexBuffer(3,ExtBearBodyEndSecond);
   SetIndexBuffer(4,ExtBearShadowFirst);
   SetIndexBuffer(5,ExtBearShadowSecond);
   SetIndexBuffer(6,ExtBearShadowEndFirst);
   SetIndexBuffer(7,ExtBearShadowEndSecond);
   SetIndexBuffer(8,ExtBullBodyFirst);
   SetIndexBuffer(9,ExtBullBodySecond);
   SetIndexBuffer(10,ExtBullBodyEndFirst);
   SetIndexBuffer(11,ExtBullBodyEndSecond);
   SetIndexBuffer(12,ExtBullShadowFirst);
   SetIndexBuffer(13,ExtBullShadowSecond);
   SetIndexBuffer(14,ExtBullShadowEndFirst);
   SetIndexBuffer(15,ExtBullShadowEndSecond);
//--- stellen Sie einige Werte der Eigenschaften für die Konstruktion des Indikators e
   for(int i=0;i<8;i++)
     {
      PlotIndexSetInteger(i,PLOT_DRAW_TYPE,DRAW_FILLING); // der Typ der graphischen P
      PlotIndexSetInteger(i,PLOT_LINE_STYLE,STYLE_SOLID); // der Stil der Linie auf di
      PlotIndexSetInteger(i,PLOT_LINE_WIDTH,1);           // die Dicke der Linie auf d
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- wenn es noch keine berechnete Bars gibt
   if(prev_calculated==0)
     {
      //--- bekommen Sie die Zeit wenn der Bar auf dem älteren Zeitrahmen erschienen w
      if(!GetTimeData())
         return(0);
     }
//--- stellen Sie die gerade Richtung der Indexierung ein
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
   ArraySetAsSeries(open,false);
   ArraySetAsSeries(close,false);
//--- Variable zur Berechnung den Anfang des Bars
   int start=prev_calculated;
//--- wenn der Bar gebildet ist, so berechnen wir den Indikatorwert darauf
   if(start!=0 && start==rates_total)
      start--;
//--- die Zyklus von die Berechnung der Indikatorwerte
   for(int i=start;i<rates_total;i++)
     {
      //--- füllen Sie die i-ten Elemente des Indikator Puffers mit dem leeren Werten
      FillIndicatorBuffers(i);
      //--- führen Sie die Berechnung für die Bars seit dem Datum InpDateStart durch
      if(time[i]>=InpDateStart)
        {
         //--- zum ersten Mal bestimmen Sie eine Position mit der wir werden die Werte
         if(ExtStartFlag)
           {
            //--- speichern Sie der Index des Anfangsbar
            ExtStartPos=i;
            //--- bestimmen Sie das erste Datum auf dem älteren Zeitrahmen, das mehr a
            while(time[i]>=ExtTimeBuff[ExtCount])
               if(ExtCount<ExtSize-1)
                  ExtCount++;
            //--- ändern Sie die Flaggewert, um in diesen Block nicht mehr zu geraten
            ExtStartFlag=false;
           }
         //--- prüfen Sie, ob es noch Elemente in das Array sind
         if(ExtCount<ExtSize)
           {
            //--- warten Sie, wenn der Zeitwert auf dem aktuellen Zeitrahmen der Wert
            if(time[i]>=ExtTimeBuff[ExtCount])
              {
               //--- zeichnen Sie den Hauptteil der Kerze (ohne Zeichnung zwischen der
               FillCandleMain(open,close,high,low,ExtStartPos,i-1,i-2,ExtIndexMax,ExtI
               //--- zeichnen Sie das Ende der Kerze (das Gebiet zwischen der letzten
               FillCandleEnd(open,close,high,low,ExtStartPos,i-1,i-1,ExtIndexMax,ExtIn
```

© 2000-2025, MetaQuotes Ltd.

```
                //--- bewegen Sie die Anfangsposition für die Zeichnung der nächsten Ke
                ExtStartPos=i;
                //--- erhöhen Sie der Array Zähler
                ExtCount++;
               }
             else
                continue;
            }
          else
            {
             //--- der Fehlerwert zu stürzen
             ResetLastError();
             //--- bekommen Sie die letzte Datum auf dem älteren Zeitrahmen
             if(CopyTime(Symbol(),InpPeriod,0,1,ExtCurrentTime)==-1)
               {
                Print("Fehler beim Kopieren die Daten, Kode = ",GetLastError());
                return(0);
               }
             //--- wenn das neue Datum mehr ist, so beenden wir die Bildung der Kerze
             if(ExtCurrentTime[0]>ExtLastTime)
               {
                //--- freimachen Sie das Gebiet zwischen der letzten und vorletzten Bar
                ClearEndOfBodyMain(i-1);
                //--- zeichnen Sie dieses Gebiet mit dem Hilfs-Indikator Puffers
                FillCandleEnd(open,close,high,low,ExtStartPos,i-1,i-1,ExtIndexMax,ExtIn
                //--- bewegen Sie die Anfangsposition für die Zeichnung der nächsten Ke
                ExtStartPos=i;
                //--- stürzen Sie die Flagge der Preisrichtung
                ExtDirectionFlag=0;
                //--- speichern Sie das neue letzte Datum
                ExtLastTime=ExtCurrentTime[0];
               }
             else
               {
                //--- bilden Sie die Kerze
                FillCandleMain(open,close,high,low,ExtStartPos,i,i,ExtIndexMax,ExtIndex
               }
            }
         }
      }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Überprüfen Sie die Indikator Periode auf die Korrektheit         |
//+------------------------------------------------------------------+
bool CheckPeriod(int current_period,int high_period)
  {
//--- die Periode des Indikators soll mehr als der Zeitrahmen sein, auf dem er dargest
```

© 2000-2025, MetaQuotes Ltd.

```
   if(current_period>=high_period)
     {
      Print("Fehler! Der Wert von der Indikator Periode soll mehr als der aktuellen Ze
      return(false);
     }
//--- wenn die Periode des Indikators eine Woche oder einer Monat ist, so ist die Peri
   if(high_period>32768)
      return(true);
//--- bringen Sie die Werte der Perioden bis zu Minuten
   if(high_period>30)
      high_period=(high_period-16384)*60;
   if(current_period>30)
      current_period=(current_period-16384)*60;
//--- die Periode des Indikators soll Fach der Zeitrahmen sein, auf dem er dargestellt
   if(high_period%current_period!=0)
     {
      Print("Fehler! Der Wert der Periode des Indikators soll Fach der aktuellen Zeitr
      return(false);
     }
//--- die Periode des Indikators soll der Wert der Zeitrahmen übertreten, auf dem er
   if(high_period/current_period<3)
     {
      Print("Fehler! Der Wert der Periode des Indikators soll der Wert der aktuellen Z
      return(false);
     }
//--- die Periode des Indikators ist für den aktuellen Zeitrahmen korrekt
   return(true);
  }
//+------------------------------------------------------------------+
//| Bekommen Sie die Daten der Zeit auf dem älteren Zeitrahmen       |
//+------------------------------------------------------------------+
bool GetTimeData(void)
  {
//--- der Fehlerwert zu stürzen
   ResetLastError();
//--- kopieren Sie alle Daten auf die aktuelle Zeit
   if(CopyTime(Symbol(),InpPeriod,InpDateStart,TimeCurrent(),ExtTimeBuff)==-1)
     {
      //--- der Fehlerwert zu stürzen
      int code=GetLastError();
      //--- drucken Sie der Fehlertext
      PrintFormat("Fehler beim Kopieren die Daten! %s",code==4401
                  ? "Die Geschichte wird noch beladen!"
                  : "Kode = "+IntegerToString(code));
      //--- geben Sie false für den nochmaligen Versuch der Beladung der Daten zurück
      return(false);
     }
//--- bekommen Sie die Größe des Arrays
   ExtSize=ArraySize(ExtTimeBuff);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- stellen Sie der Index des Zyklus für das Array gleich der Null ein
   ExtCount=0;
//--- stellen Sie die Position der aktuellen Kerze in diesem Zeitrahmen gleich der Nul
   ExtStartPos=0;
   ExtStartFlag=true;
//--- bekommen Sie der letzten Wert der Zeit auf dem älteren Zeitrahmen
   ExtLastTime=ExtTimeBuff[ExtSize-1];
//--- die erfolgreiche Ausführung
   return(true);
  }
//+------------------------------------------------------------------+
//| Die Funktion bildet den Hauptteil der Kerze. Je nach dem Wert    |
//| der Flagge, die Funktion bestimmt, welche Daten sich in welche   |
//| Arrays für die korrekte Abbildung einschreiben sollen.           |
//+------------------------------------------------------------------+
void FormCandleMain(double &body_fst[],double &body_snd[],
                    double &shadow_fst[],double &shadow_snd[],
                    const double fst_value,const double snd_value,
                    const double fst_extremum,const double snd_extremum,
                    const int start,const int count,const bool flag)
  {
//--- prüfen Sie der Wert der Flagge
   if(flag)
     {
      //--- bilden Sie der Körper der Kerze
      FormMain(body_fst,body_snd,fst_value,snd_value,start,count);
      //--- bilden Sie der Schatten der Kerze
      FormMain(shadow_fst,shadow_snd,fst_extremum,snd_extremum,start,count);
     }
   else
     {
      //--- bilden Sie der Körper der Kerze
      FormMain(body_fst,body_snd,snd_value,fst_value,start,count);
      //--- bilden Sie der Schatten der Kerze
      FormMain(shadow_fst,shadow_snd,snd_extremum,fst_extremum,start,count);
     }
  }
//+------------------------------------------------------------------+
//| Die Funktion bildet das Ende der Kerze. Je nach dem Wert der     |
//| Flagge, die Funktion bestimmt, welche Daten sich in welche Arrays|
//| für die korrekte Abbildung einschreiben sollen.                  |
//+------------------------------------------------------------------+
void FormCandleEnd(double &body_fst[],double &body_snd[],
                   double &shadow_fst[],double &shadow_snd[],
                   const double fst_value,const double snd_value,
                   const double fst_extremum,const double snd_extremum,
                   const int end,bool &flag)
  {
//--- prüfen Sie der Wert der Flagge
```

© 2000-2025, MetaQuotes Ltd.

```
   if(flag)
     {
      //--- bilden Sie das Ende der Körper der Kerze
      FormEnd(body_fst,body_snd,fst_value,snd_value,end);
      //--- bilden Sie das Ende der Schatten der Kerze
      FormEnd(shadow_fst,shadow_snd,fst_extremum,snd_extremum,end);
      //--- ändern Sie der Wert der Flagge gegen das Entgegengesetzte
      flag=false;
     }
   else
     {
      //--- bilden Sie das Ende der Körper der Kerze
      FormEnd(body_fst,body_snd,snd_value,fst_value,end);
      //--- bilden Sie das Ende der Schatten der Kerze
      FormEnd(shadow_fst,shadow_snd,snd_extremum,fst_extremum,end);
      //--- ändern Sie der Wert der Flagge gegen das Entgegengesetzte
      flag=true;
     }
  }
//+------------------------------------------------------------------+
//| Die Entleerung das Ende der Kerze (das Gebiet zwischen der       |
//| letzten und vorletzten Bar)                                      |
//+------------------------------------------------------------------+
void ClearEndOfBodyMain(const int ind)
  {
   ClearCandle(ExtBearBodyFirst,ExtBearBodySecond,ExtBearShadowFirst,ExtBearShadowSecc
   ClearCandle(ExtBullBodyFirst,ExtBullBodySecond,ExtBullShadowFirst,ExtBullShadowSecc
  }
//+------------------------------------------------------------------+
//| Die Entleerung der Kerze                                         |
//+------------------------------------------------------------------+
void ClearCandle(double &body_fst[],double &body_snd[],double &shadow_fst[],
                 double &shadow_snd[],const int start,const int count)
  {
//--- die Überprüfung
   if(count!=0)
     {
      //--- füllen Sie der Indikator Puffers mit dem leeren Wert aus
      ArrayFill(body_fst,start,count,INDICATOR_EMPTY_VALUE);
      ArrayFill(body_snd,start,count,INDICATOR_EMPTY_VALUE);
      ArrayFill(shadow_fst,start,count,INDICATOR_EMPTY_VALUE);
      ArrayFill(shadow_snd,start,count,INDICATOR_EMPTY_VALUE);
     }
  }
//+------------------------------------------------------------------+
//| Die Bildung des Hauptteiles der Kerze                            |
//+------------------------------------------------------------------+
void FormMain(double &fst[],double &snd[],const double fst_value,
              const double snd_value,const int start,const int count)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- die Überprüfung
   if(count!=0)
     {
      //--- füllen Sie die Indikator Puffers Werte aus
      ArrayFill(fst,start,count,fst_value);
      ArrayFill(snd,start,count,snd_value);
     }
  }
//+------------------------------------------------------------------+
//| Die Bildung das Ende der Kerze                                   |
//+------------------------------------------------------------------+
void FormEnd(double &fst[],double &snd[],const double fst_value,
             const double snd_value,const int last)
  {
//--- füllen Sie die Indikator Puffers Werte aus
   ArrayFill(fst,last-1,2,fst_value);
   ArrayFill(snd,last-1,2,snd_value);
  }
//+------------------------------------------------------------------+
//| Füllen Sie die i-ten Elemente des Indikator Puffers mit dem      |
//| leeren Werte aus                                                 |
//+------------------------------------------------------------------+
void FillIndicatorBuffers(const int i)
  {
//--- stellen Sie der leer Wert in die Zelle der Indikator Puffers ein
   ExtBearBodyFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearBodySecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBearBodyEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearBodyEndSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBearShadowEndSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodyFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodySecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodyEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullBodyEndSecond[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowEndFirst[i]=INDICATOR_EMPTY_VALUE;
   ExtBullShadowEndSecond[i]=INDICATOR_EMPTY_VALUE;
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayPrint

Gibt Arrays eines einfachen Typs oder einer einfachen Struktur im Journal aus.

```
void  ArrayPrint(
   const void&   array[],             // ausgegebenes Array
   uint          digits=_Digits,      // Anzahl von Dezimalstellen nach dem Komma
   const string  separator=NULL,      // Trennzeichen zwischen den Werten der Felder e
   ulong         start=0,             // Index des ersten ausgegebenen Elements
   ulong         count=WHOLE_ARRAY,   // Anzahl von ausgegebenen Elementen
   ulong         flags=ARRAYPRINT_HEADER|ARRAYPRINT_INDEX|ARRAYPRINT_LIMIT|ARRAYPRINT_
   );
```

Parameter

array[]

[in] Array eines einfachen Typs oder einer einfachen Struktur.

digits=_Digits

[in] Anzahl der Stellen nach dem Komma für echte Typen. Standardmäßig ist gleich _Digits.

separator=NULL

[in] Trennzeichen zwischen den Werten der Felder eines Strukturelements. Der standardmäßige Wert NULL bezeichnet eine leere Zeile; in diesem Fall gilt eine Leerstelle als Trennzeichen.

start=0

[in] Index des ersten ausgegebenen Elements des Arrays. Standardmäßig wird ab dem Null-Index ausgegeben.

count=WHOLE_ARRAY

[in] Anzahl der Elemente des Arrays, die ausgegeben werden müssen. Standardmäßig wird das ganze Array ausgegeben (count=WHOLE_ARRAY).

flags=ARRAYPRINT_HEADER|ARRAYPRINT_INDEX|ARRAYPRINT_LIMIT|ARRAYPRINT_ALIGN

[in] Kombination von Flags, die den Ausgabemodus setzt. Standardmäßig sind alle Flags aktiviert:

- ARRAYPRINT_HEADER - Überschriften für das Struktur-Array ausgeben
- ARRAYPRINT_INDEX - Indexnummer links ausgeben
- ARRAYPRINT_LIMIT - nur die ersten 100 und die letzten 100 Elemente des Arrays ausgeben. Wird verwendet, wenn nur ein Teil eines großen Arrays ausgegeben werden muss.
- ARRAYPRINT_ALIGN - die Ausrichtung der Werte aktivieren - Zahlen werden nach rechts ausgerichtet, Zeilen - nach links.
- ARRAYPRINT_DATE - bei der Ausgabe von datetime Datum als dd.mm.yyyy anzeigen
- ARRAYPRINT_MINUTES - bei der Ausgabe von datetime Zeit als HH:MM anzeigen
- ARRAYPRINT_SECONDS - bei der Ausgabe von datetime Zeit als HH:MM:SS anzeigen

Rückgabewert

Nein

Hinweis

ArrayPrint() zeigt nicht alle Felder eines Struktur-Arrays im Journal aus - die Felder Arrays und Objektbezeichner werden ausgelassen. Diese Spalten werden zwecks einer einfachen Darstellung

© 2000-2025, MetaQuotes Ltd.

nicht angezeigt. Wenn Sie die Ausgabe aller Felder einer solchen Struktur benötigen, brauchen Sie eine eigene Funktion mit der gewünschten Formatierung zu schreiben.

Beispiel:

```
//--- gibt die Werte der letzten zehn Balken aus
   MqlRates rates[];
   if(CopyRates(_Symbol,_Period,1,10,rates))
     {
      ArrayPrint(rates);
      Print("Überprüfung\n[time]\t[open]\t[high]\t[low]\t[close]\t[tick_volume]\t[spre
      for(int i=0;i<10;i++)
        {
         PrintFormat("[%d]\t%s\t%G\t%G\t%G\t%G\t%G\t%G\t%I64d\t",i,
         TimeToString(rates[i].time,TIME_DATE|TIME_MINUTES|TIME_SECONDS),
         rates[i].open,rates[i].high,rates[i].low,rates[i].close,
         rates[i].tick_volume,rates[i].spread,rates[i].real_volume);
        }
     }
   else
      PrintFormat("CopyRates failed, error code=%d",GetLastError());
//--- Beispiel
/*
                    [time]  [open]  [high]   [low] [close] [tick_volume] [spread] [rea
   [0] 2016.11.09 04:00:00 1.11242 1.12314 1.11187 1.12295         18110       10   17
   [1] 2016.11.09 05:00:00 1.12296 1.12825 1.11930 1.12747         17829        9   15
   [2] 2016.11.09 06:00:00 1.12747 1.12991 1.12586 1.12744         13458       10    9
   [3] 2016.11.09 07:00:00 1.12743 1.12763 1.11988 1.12194         15362        9   12
   [4] 2016.11.09 08:00:00 1.12194 1.12262 1.11058 1.11172         16833        9   12
   [5] 2016.11.09 09:00:00 1.11173 1.11348 1.10803 1.11052         15933        8   10
   [6] 2016.11.09 10:00:00 1.11052 1.11065 1.10289 1.10528         11888        9    8
   [7] 2016.11.09 11:00:00 1.10512 1.11041 1.10472 1.10915          7284       10    5
   [8] 2016.11.09 12:00:00 1.10915 1.11079 1.10892 1.10904          8710        9    6
   [9] 2016.11.09 13:00:00 1.10904 1.10913 1.10223 1.10263          8956        7    7
   Überprüfung
   [time] [open] [high] [low] [close] [tick_volume] [spread] [real_volume]
   [0] 2016.11.09 04:00:00 1.11242 1.12314 1.11187 1.12295 18110 10 17300175000
   [1] 2016.11.09 05:00:00 1.12296 1.12825 1.1193 1.12747 17829 9 15632176000
   [2] 2016.11.09 06:00:00 1.12747 1.12991 1.12586 1.12744 13458 10 9593492000
   [3] 2016.11.09 07:00:00 1.12743 1.12763 1.11988 1.12194 15362 9 12352245000
   [4] 2016.11.09 08:00:00 1.12194 1.12262 1.11058 1.11172 16833 9 12961333000
   [5] 2016.11.09 09:00:00 1.11173 1.11348 1.10803 1.11052 15933 8 10720384000
   [6] 2016.11.09 10:00:00 1.11052 1.11065 1.10289 1.10528 11888 9 8084811000
   [7] 2016.11.09 11:00:00 1.10512 1.11041 1.10472 1.10915 7284 10 5087113000
   [8] 2016.11.09 12:00:00 1.10915 1.11079 1.10892 1.10904 8710 9 6769629000
   [9] 2016.11.09 13:00:00 1.10904 1.10913 1.10223 1.10263 8956 7 7192138000
*/
```

Sieh auch

© 2000-2025, MetaQuotes Ltd.

FileSave, FileLoad

© 2000-2025, MetaQuotes Ltd.

ArrayRange

Gibt die Anzahl der Elemente in der angegebenen Arraydimension.

```
int  ArrayRange(
   const void&   array[],      // Array für die Prüfung
   int           rank_index    // Nummer der Dimension
   );
```

Parameter

array[]

[in] Das geprüfte Array.

rank_index

[in] Index der Dimension.

Rückgabewert

Anzahl der Elemente in der angegebenen Arraydimension

Hinweis

Da die Indizes bei Null beginnen, ist die Anzahl der Dimensionen des Arrays um eins größer als der Index der letzten Dimension.

Beispiel:

```
void OnStart()
  {
//--- erstellen Sie vierdimensionalen Array
   double array[][5][2][4];
//--- stellen Sie die Größe der Nullmessung
   ArrayResize(array,10,10);
//--- drucken Sie die Dimensionen der Messungen
   int temp;
   for(int i=0;i<4;i++)
     {
      //--- bekommen Sie die Größe der i-ten Messung
      temp=ArrayRange(array,i);
      //--- drucken
      PrintFormat("dim = %d, range = %d",i,temp);
     }
//--- Ergebnis
// dim = 0, range = 10
// dim = 1, range = 5
// dim = 2, range = 2
// dim = 3, range = 4
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrayResize

Verändert die Größe der ersten Dimension des Arrays

```
int  ArrayResize(
   void&  array[],              // Array, das durch Referenz übertragen wurde
   int    new_size,             // die neue Größe des Arrays
   int    reserve_size=0        // Reserve des Arrays (redundant)
   );
```

Parameter

array[]

[out] das Array, dessen Größe verändert werden soll

new_size

[in] Die neue Größe der ersten Dimension

reserve_size=0

[in] Die Größe für Zusatzreserve

Rückgabewert

Bei erfolgreicher Durchführung gibt die Funktion die Anzahl aller Elemente zurück, die das Array nach der Größenveränderung enthält; anderenfalls gibt -1 zurück und das Array verändert seine Größen nicht.

Wenn ArrayResize() auf ein static Array, eine Zeitreihe oder ein Indikatorpuffer angewendet wird, bleibt die Arraygröße gleich - diese Arrays werden nicht verändert. In diesem Fall, wenn *new_size*<=ArraySize(*array*), wird die Funktion *new_size* zurückgeben, andernfalls den Wert -1.

Hinweis

Die Funktion kann nur auf dynamischen Arrays angewendet werden. Es sollte beachtet werden, dass die Größe dynamischer Arrays, die als Indikatorpuffer mit der Funktion SetIndexBuffer() definiert wurden, nicht verändert werden können. Für Indikatorpuffer werden alle Änderungen der Größe von Laufzeit-Subsystem des Terminals durchgeführt.

Die Gesamtzahl der Elemente im Array darf nicht größer als 2147483647 sein.

Bei häufigen Speicherveränderungen wird empfohlen, den dritten Parameter zu verwenden, der eine Reserve festlegt, um die Anzahl der physikalischen Speicherzuweisungen zu reduzieren. Alle nachfolgenden Aufrufe der Funktion ArrayResize führen dann nicht mehr zu einer neuen physikalischer Speicherzuweisung, in diesen Fällen ändert sich nur die Größe der ersten Dimension des Arrays im reservierten Speicherbereich. Es sei daran erinnert, dass der dritte Parameter nur dann verwendet wird, wenn die physikalische Speicherzuweisung stattfindet, zum Beispiel:

```
ArrayResize(arr,1000,1000);
for(int i=1;i<3000;i++)
   ArrayResize(arr,i,1000);
```

In diesem Fall gibt es nur zwei Speicherzuweisungen, einmal vor der Schleife mit 3000 Iterationen (dabei wird die Dimension des Arrays auf 1000 festgelegt) und das zweite Mal wenn i gleich 2000 ist.

© 2000-2025, MetaQuotes Ltd.

Wenn der dritte Parameter weggelassen worden wäre, würden es 2000 einzelne physikalische Speicherzuweisungen geben, welches die Ausführung des Programms verzögern würde.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Zähler
   ulong start=GetTickCount();
   ulong now;
   int   count=0;
//--- Array demonstriert die schnelle Version
   double arr[];
   ArrayResize(arr,100000,100000);
//--- Prüfen wir, wie schnell die Version mit Reservierung von Speicher arbeitet
   Print("--- Test Fast: ArrayResize(arr,100000,100000)");
   for(int i=1;i<=300000;i++)
     {
      //--- Geben wir eine neue Arraygröße ein mit einer Reserve von 100000 Elementen
      ArrayResize(arr,i,100000);
      //--- Wenn eine runde Zahl erreicht wird, wird die Größe des Arrays und die benö
      if(ArraySize(arr)%100000==0)
        {
         now=GetTickCount();
         count++;
         PrintFormat("%d. ArraySize(arr)=%d Time=%d ms",count,ArraySize(arr),(now-star
         start=now;
        }
     }
//--- Jetzt zeigen wir wie langsam die Version ohne reserviertem Speicher ist
   double slow[];
   ArrayResize(slow,100000,100000);
//---
   count=0;
   start=GetTickCount();
   Print("---- Test Slow: ArrayResize(slow,100000)");
//---
   for(int i=1;i<=300000;i++)
     {
      //--- Geben wir eine neue Arraygröße ohne die zusätzliche Reserve
      ArrayResize(slow,i);
      //--- Wenn eine runde Zahl erreicht wird, wird die Größe des Arrays und die benö
      if(ArraySize(slow)%100000==0)
        {
         now=GetTickCount();
         count++;
         PrintFormat("%d. ArraySize(slow)=%d Time=%d ms",count,ArraySize(slow),(now-st
         start=now;
        }
     }
  }
//--- Ergebnis des Beispiel-Skripts
/*
   Test_ArrayResize (EURUSD,H1)   --- Test Fast: ArrayResize(arr,100000,100000)
   Test_ArrayResize (EURUSD,H1)   1. ArraySize(arr)=100000 Time=0 ms
   Test_ArrayResize (EURUSD,H1)   2. ArraySize(arr)=200000 Time=0 ms
   Test_ArrayResize (EURUSD,H1)   3. ArraySize(arr)=300000 Time=0 ms
   Test_ArrayResize (EURUSD,H1)   ---- Test Slow: ArrayResize(slow,100000)
   Test_ArrayResize (EURUSD,H1)   1. ArraySize(slow)=100000 Time=0 ms
   Test_ArrayResize (EURUSD,H1)   2. ArraySize(slow)=200000 Time=0 ms
   Test_ArrayResize (EURUSD,H1)   3. ArraySize(slow)=300000 Time=228511 ms
*/
```

© 2000-2025, MetaQuotes Ltd.

Siehe auch

ArrayInitialize

© 2000-2025, MetaQuotes Ltd.

ArrayInsert

Einfügen einer Anzahl von Elementen eines Quellarrays in ein Zielarray, beginnend mit dem angegebenen Index.

```
bool  ArrayInsert(
   void&        dst_array[],          // Zielarray
   const void&  src_array[],          // Quellarray
   uint         dst_start,            // Index des Zielarrays
   uint         src_start=0,          // Index des Quellarrays
   uint         count=WHOLE_ARRAY     // Anzahl der einzufügenden Elemente
   );
```

Parameter

dst_array[]

[in][out] Das Zielarray, dem die Elemente hinzugefügt werden sollen.

src_array[]

[in] Quellarray, aus dem die Elemente kopiert werden.

dst_start

[in] Index des Zielarrays, ab dem die Elemente eingefügt werden sollen.

src_start=0

[in] Index des Quellarrays, ab dem die Elemente aus dem Quellarray kopiert werden sollen.

count

[in] Anzahl der Elemente, die aus dem Quellarray kopiert werden sollen. WHOLE_ARRAY bedeutet, dass alle Elemente ab dem angegebenen Index des Quellarrays bis zu seinem Ende kopiert werden.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 5052 - ERR_SMALL_ARRAY (die Parameter *start* und/oder *count* sind falsch eingestellt worden oder das Quellarray *src_array[]* ist leer),
- 5056 - ERR_SERIES_ARRAY (das Array kann nicht geändert werden, es ist ein Indikatorpuffer),
- 4006 - ERR_INVALID_ARRAY (eine Kopie auf sich selbst ist nicht erlaubt, die Arrays sind unterschiedlichen Typs oder es gibt ein Array mit fester Größe, das Klassenobjekte oder Destruktorstrukturen enthält),
- 4005 - ERR_STRUCT_WITHOBJECTS_ORCLASS (das Array enthält keine POD-Strukturen, was bedeutet, dass ein einfaches Kopieren unmöglich ist),
- Fehler, die beim Ändern der Größe des Zielarrays *dst_array[]* auftreten, finden Sie in der Funktionsbeschreibung von ArrayRemove().

Hinweis

Wenn die Funktion für ein Array mit fester Größe verwendet wird, ändert sich die Größe des Zielarrays *dst_array[]* selbst nicht. Ausgehend von der Position *dst_start* werden die Elemente des

© 2000-2025, MetaQuotes Ltd.

Zielarrays nach rechts verschoben (die *überzähligen* Elemente "gehen verloren"), während die vom Quellarray kopierten Elemente ihren Platz einnehmen.

Sie können die Elemente nicht in die dynamischen Arrays einfügen, die von der Funktion SetIndexBuffer() als Indikatorpuffer definiert wurden. Bei Indikatorpuffern werden alle Größenänderungsvorgänge vom ausführenden Subsystem des Terminals durchgeführt.

Im Quellarray werden die Elemente ab dem Index *src_start* kopiert. Die Größe des Quellarrays bleibt unverändert. Die Elemente, die dem Zielarray hinzugefügt werden sollen, sind keine Links zu den Elementen des Quellarrays. Das bedeutet, dass nachfolgende Änderungen der Elemente in einem der beiden Arrays nicht in dem anderen Array berücksichtigt werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklarieren des Arrays mit fixer Größe und der Wertezuweisung
   int array_dest[10];
   for(int i=0;i<10;i++)
     {
      array_dest[i]=i;
     }
   //--- Quellarray
   int array_source[10];
   for(int i=0;i<10;i++)
     {
      array_source[i]=10+i;
     }
//--- Anzeigen des Arrays vor dem Einfügen der Elemente
   Print("Vor dem Aufruf von ArrayInsert()");
   ArrayPrint(array_dest);
   ArrayPrint(array_source);
//--- Einfügen von 3 Elementen in das Array und dessen Darstellung danach
   ArrayInsert(array_dest,array_source,4,0,3);
   Print("Nach dem Aufruf von ArrayInsert()");
   ArrayPrint(array_dest);
/*
  Ausführungsergebnis:
   Vor dem Aufruf von ArrayInsert()
   0 1 2 3 4 5 6 7 8 9
   Nach dem Aufruf von ArrayInsert()
   0 1 2 3 10 11 12 7 8 9
*/
```

Siehe auch

ArrayRemove, ArrayCopy, ArrayResize, ArrayFree

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

ArrayRemove

Entfernt die angegebene Anzahl von Elementen im Array, beginnend mit dem angegebenen Index.

```
bool  ArrayRemove(
   void&         array[],            // Array irgendeines Typs
   uint          start,              // Startindex für das Entfernen
   uint          count=WHOLE_ARRAY   // Anzahl der Elemente
   );
```

Parameter

array[]

[in][out] Array.

Start

[in] Index, ab dem die Elemente entfernt werden sollen.

count=WHOLE_ARRAY

[in] Anzahl der zu entfernenden Elemente. WHOLE_ARRAY bedeutet, dass alle Elemente ab dem angegebenen Index bis zum Ende des Array entfernt werden.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 5052 - ERR_SMALL_ARRAY (Zu großer Wert für *start*),
- 5056 - ERR_SERIES_ARRAY (das Array kann nicht geändert werden, es ist ein Indikatorpuffer),
- 4003 - ERR_INVALID_PARAMETER (Zu großer Wert für *count*),
- 4005 - ERR_STRUCT_WITHOBJECTS_ORCLASS (Array mit fixer Größe, das komplexe Objekte mit einem Destruktor enthält),
- 4006 - ERR_INVALID_ARRAY (Array mit fixer Größe, das Strukturen oder Klassenobjekte mit einem Destruktor enthält).

Hinweis

Wenn die Funktion für ein Array mit fester Größe verwendet wird, ändert sich die Array-Größe nicht: Der verbleibende "Rest" wird physisch an die Position *start* kopiert. Zum genauen Verständnis der Funktionsweise der Funktion betrachten Sie das folgende Beispiel. "Physisches" Kopieren bedeutet, dass die kopierten Objekte nicht durch Aufruf des Konstruktors oder des Kopieroperators erzeugt werden. Stattdessen wird die binäre Darstellung eines Objekts kopiert. Aus diesem Grund können Sie die Funktion ArrayRemove() nicht auf das Array mit fester Größe anwenden, das Objekte mit dem Destruktor enthält (die Fehler ERR_INVALID_ARRAY oder ERR_STRUCT_WITHOBJECTS_ORCLASS werden gesetzt). Beim Entfernen eines solchen Objekts würde der Destruktor zweimal aufgerufen werden - für das Originalobjekt und seine Kopie.

Sie können keine Elemente aus dynamischen Arrays entfernen, die von der Funktion SetIndexBuffer() als Indikatorpuffer definiert wurden. Dies führt zum Fehler ERR_SERIES_ARRAY. Bei Indikatorpuffern werden alle Größenänderungsvorgänge vom ausführenden Subsystem des Terminals durchgeführt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklarieren des Arrays mit fixer Größe und der Wertezuweisung
   int array[10];
   for(int i=0;i<10;i++)
     {
      array[i]=i;
     }
//--- Anzeigen des Arrays vor dem Entfernen der Elemente
   Print("Vor dem Aufruf von ArrayRemove()");
   ArrayPrint(array);
//--- Entfernen von 2 Elementen im Array und dessen Darstellung danach
   ArrayRemove(array,4,2);
   Print("Nach dem Aufruf von ArrayRemove()");
   ArrayPrint(array);
/*
  Ausführungsergebnis:
  Vor dem Aufruf von ArrayRemove()
   0 1 2 3 4 5 6 7 8 9
  Nach dem Aufruf von ArrayRemove()
   0 1 2 3 6 7 8 9 8 9
*/
```

Siehe auch

ArrayInsert, ArrayCopy, ArrayResize, ArrayFree

© 2000-2025, MetaQuotes Ltd.

ArrayReverse

Kehrt die angegebene Anzahl von Elementen im Array um, beginnend mit dem angegebenen Index.

```
bool  ArrayReverse(
   void&         array[],            // Array irgendeines Typs
   uint          start=0,            // Startindex der Umkehrung
   uint          count=WHOLE_ARRAY   // Anzahl der Elemente
   );
```

Parameter

array[]

[in][out] Array.

start=0

[in] Startindex der Umkehrung.

count=WHOLE_ARRAY

[in] Anzahl der umzukehrenden Elemente. Wenn WHOLE_ARRAY angegeben ist, werden alle Arrayelemente, ab dem im Parameter *start* angegebenen Index bis zum Ende, in umgekehrter Reihenfolge eingeordnet.

Rückgabewert

Liefert bei Erfolg true, ansonsten false.

Hinweis

Die Funktion ArraySetAsSeries() verschiebt die Arrayelemente nicht physikalisch. Stattdessen wird nur die Indexierungsrichtung für den Zugriff auf die Elemente wie in der Zeitreihe umgekehrt. Die Funktion ArrayReverse() bewegt die Arrayelemente physikalisch so, dass das Array "umgekehrt" wird.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklarieren des Arrays mit fixer Größe und der Wertezuweisung
   int array[10];
   for(int i=0;i<10;i++)
     {
      array[i]=i;
     }
//--- Anzeigen des Arrays vor der Neuordnung der Elemente
   Print("Vor dem Aufruf von ArrayReverse()");
   ArrayPrint(array);
//--- Umkehren von 3 Elementen im Array und dessen Darstellung danach
   ArrayReverse(array,4,3);
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("Nach dem Aufruf von ArrayReverse()");
   ArrayPrint(array);
/*
  Ausführungsergebnis:
  Vor dem Aufruf von ArrayReverse()
   0 1 2 3 4 5 6 7 8 9
  Nach dem Aufruf von ArrayReverse()
   0 1 2 3 6 5 4 7 8 9
*/
```

Siehe auch

ArrayInsert, ArrayRemove, ArrayCopy, ArrayResize, ArrayFree, ArrayGetAsSeries, ArraySetAsSeries

© 2000-2025, MetaQuotes Ltd.

ArraySetAsSeries

Stellt Flagge AS_SERIES für das angegebene Objekt des dynamischen Arrays ein, Indizieren der Elemente wird wie in Zeitreihen durchgeführt werden.

```
bool  ArraySetAsSeries(
   const void&  array[],     // Array durch Referenz
   bool         flag         // true bedeutet umgekehrte Reihenfolge des Indizierens
   );
```

Parameter

array[]

[in][out] Numerisches Array für Einstellung.

flag

[in] Richtung des Arrayindizierens.

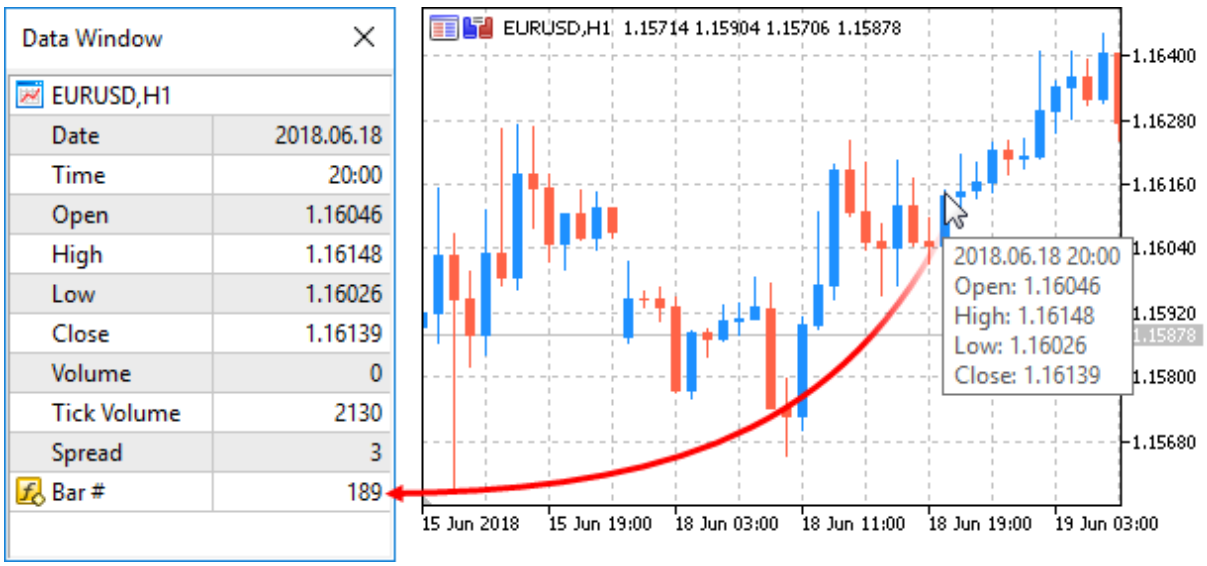
Rückgabewert

Gibt true beim Erfolg zurück, sonst false.

Hinweis

Flagge AS_SERIES kann nicht für die vieldimensionale Arrays und für die statische Arrays (d.h. Arrays, deren Größe noch während der Kompilierung in eckigen Klammern angegeben wird) eingestellt werden. Indizieren einer Zeitreihe unterscheidet sich von normalen Feld dadurch, dass Indizieren der Elemente einer Zeitreihe vom Feldende zum Feldanfang durchgeführt wird (von den neuesten Daten zu den ältesten Daten).

Beispiel: Indikator, der Barnummer zeigt



© 2000-2025, MetaQuotes Ltd.

```
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Numeration
#property indicator_label1  "Numeration"
#property indicator_type1   DRAW_LINE
#property indicator_color1  CLR_NONE
//--- Indikator Puffers
double         NumerationBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,NumerationBuffer,INDICATOR_DATA);
//--- stellen wir Pufferindizieren wie in einer Zeitreihe ein
   ArraySetAsSeries(NumerationBuffer,true);
//--- stellen wir Genauigkeit der Darstellung in DataWindow ein
   IndicatorSetInteger(INDICATOR_DIGITS,0);
//--- wie der Name des Indikatorfeldes in DataWindow aussehen wird
   PlotIndexSetString(0,PLOT_LABEL,"Bar #");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---  wir werden die Eröffnungszeit der laufenden Nullbar aufbewahren
   static datetime currentBarTimeOpen=0;
//--- kehren wir die Größe des Arrays time[] um - machen wir es wie in einer Zeitreihe
   ArraySetAsSeries(time,true);
//--- wenn sich die Zeit der Nullbar davon unterscheidet, was wir aufbewahren
   if(currentBarTimeOpen!=time[0])
     {
     //--- numerieren wir alle Bars von der laufenden Position in die Charttiefe
      for(int i=rates_total-1;i>=0;i--) NumerationBuffer[i]=i;
      currentBarTimeOpen=time[0];
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Zugang zu Zeitreihen, ArrayGetAsSeries

© 2000-2025, MetaQuotes Ltd.

ArraySize

Gibt die Anzahl von Elementen des angegebenen Arrays zurück.

```
int  ArraySize(
   const void&  array[]    // das geprüfte Array
   );
```

Parameter

array[]

[in] Array jedes Typs.

Rückgabewert

Wert des Typs int.

Hinweis

für das eindimensionale Array ist der Wert, der durch die Funktion ArraySize, zurückgegeben wird dem Wert ArrayRange(array,0) gleich.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- erstellen Sie die Arrays
   double one_dim[];
   double four_dim[][10][5][2];
//--- Größe
   int one_dim_size=25;
   int reserve=20;
   int four_dim_size=5;
//--- Hilfsvariable
   int size;
//--- trennen Sie den Speichern ohne der Reservierung
   ArrayResize(one_dim,one_dim_size);
   ArrayResize(four_dim,four_dim_size);
//--- 1. das eindimensionale Array
   Print("+============================================================+");
   Print("die Größe des Arrays:");
   Print("1. Das eindimensionale Array");
   size=ArraySize(one_dim);
   PrintFormat("Die Größe der Nullmessung = %d, Die Größe des Arrays = %d",one_dim_siz
//--- 2. das vieldimensionale Array
   Print("2. Das vieldimensionale Array");
   size=ArraySize(four_dim);
   PrintFormat("Die Größe der Nullmessung = %d, Die Größe des Arrays = %d",four_dim_si
//--- die Größe der Messungen
   int d_1=ArrayRange(four_dim,1);
   int d_2=ArrayRange(four_dim,2);
   int d_3=ArrayRange(four_dim,3);
   Print("Uberprüfung:");
   Print("Die Nullmessung = Die Größe des Arrays / (Die erste Messung * Die zweite Mes
   PrintFormat("%d = %d / (%d * %d * %d)",size/(d_1*d_2*d_3),size,d_1,d_2,d_3);
//--- 3. das eindimensionale Array mit der Speicherreservierung
   Print("3. Das eindimensionale Array mit der Speicherreservierung");
//--- erhöhen Sie den Wert zweimal
   one_dim_size*=2;
//--- trennen Sie den Speichern mit der Reservierung
   ArrayResize(one_dim,one_dim_size,reserve);
//--- drucken Sie die Größe
   size=ArraySize(one_dim);
   PrintFormat("Die Größe mit der Reservierung = %d, Die wirkliche Größe des Arrays =
  }
```

© 2000-2025, MetaQuotes Ltd.

ArraySort

Sortiert ein mehrdimensionales Array aufsteigend in der ersten Dimension.

```
bool  ArraySort(
   void&  array[]      // Array für Sortieren
   );
```

Parameter

array[]

[in][out] Numerischer Wert für Sortieren.

Rückgabewert

Gibt true beim Erfolg zurück, sonst false.

Hinweis

Das Array wird immer aufsteigend sortiert, unabhängig vom Wert des AS_SERIES Parameters.

Die ArraySort und ArrayBSearch Funktionen nehmen als Parameter ein Array mit beliebig vielen Dimensionen an, dabei erfolgt die Sortierung und die Suche nur in der ersten (zero) Dimension.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator analysiert die Daten für den letzten Monat und ze
#property description "und größen Tick Volumen aus. Für die Bestimmung solcher Kerzen
#property description "das Tick Volumen Array erzeugt. Die Kerzen, deren Volumen die e
#property description "der Prozente des Arrays bilden, gelten für die Kleinen. Die Ker
#property description "die letzte InpBigVolume der Prozente des Arrays bilden, gelten
//--- der Indikator Einstellungen
#property indicator_chart_window
#property indicator_buffers 5
#property indicator_plots   1
//--- plot
#property indicator_label1  "VolumeFactor"
#property indicator_type1   DRAW_COLOR_CANDLES
#property indicator_color1  clrDodgerBlue,clrOrange
#property indicator_style1  STYLE_SOLID
#property indicator_width1  2
//--- vordefinierte Konstante
#define INDICATOR_EMPTY_VALUE 0.0
//--- Eingabeparameters
input int InpSmallVolume=15; // Das Prozent der kleinen Volumen (<50)
input int InpBigVolume=20;   // Das Prozent der größen Volumen (<50)
//--- die Anfangszeit der Analyse (wird sich verschieben)
datetime ExtStartTime;
//--- die Indikator Puffers
double   ExtOpenBuff[];
double   ExtHighBuff[];
double   ExtLowBuff[];
double   ExtCloseBuff[];
double   ExtColorBuff[];
//--- die Grenzwerte der Volumen für die Abbildung der Kerzen
long     ExtLeftBorder=0;
long     ExtRightBorder=0;
//+------------------------------------------------------------------+
//| Erhalten Sie die Grenzwerte für Tick Volumen                     |
//+------------------------------------------------------------------+
bool GetVolumeBorders(void)
  {
//--- Variablen
   datetime stop_time;  // die Abschlusszeit von Kopieren
   long     buff[];     // Der Puffer, wohin wir kopieren werden
//--- Abschlusszeit - aktuelle Zeit
   stop_time=TimeCurrent();
//--- die Anfangszeit - einen Monat früher als die aktuelle
   ExtStartTime=GetStartTime(stop_time);
//--- erhalten Sie die Werte der Tick Volumen
   ResetLastError();
   if(CopyTickVolume(Symbol(),Period(),ExtStartTime,stop_time,buff)==-1)
     {
      //--- Es misslang, die Daten zu bekommen, geben Sie false für den Start der Mann
      PrintFormat("Es misslang, die Werte von Tick Volumen zu bekommen. Fehlercode = %
      return(false);
     }
//--- berechnen Sie die Größe des Arrays
   int size=ArraySize(buff);
//--- sortieren Sie das Array
   ArraySort(buff);
//--- bestimmen Sie die Werte der linken und der rechten Grenzen für Tick Volumen
   ExtLeftBorder=buff[size*InpSmallVolume/100];
   ExtRightBorder=buff[(size-1)*(100-InpBigVolume)/100];
//--- die erfolgreiche Ausführung
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Erhalten Sie dem Datum, einen Monat weniger als der übertragenen |
//+------------------------------------------------------------------+
datetime GetStartTime(const datetime stop_time)
  {
//--- konvertieren Sie die Abschlusszeit zu die Variable von die MqlDateTime-Struktur
   MqlDateTime temp;
   TimeToStruct(stop_time,temp);
//--- erhalten Sie dem Datum, einen Monat weniger
   if(temp.mon>1)
      temp.mon-=1;  // der aktuelle Monat ist nicht das erste des Jahres, d.h. die Nur
   else
     {
      temp.mon=12;  // der aktuelle Monat ist das erste des Jahres, d.h. die Nummer de
      temp.year-=1; // und die Nummer des Jahres ist einer Nummer weniger
     }
//--- die Nummer des tages wird nicht mehr als 28
   if(temp.day>28)
      temp.day=28;
//--- geben das bekommene Datum zurück
   return(StructToTime(temp));
  }
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Prüfen Sie, ob die Eingabeparameters den Bedingungen zu Zufriedenheit ausfallen
   if(InpSmallVolume<0 || InpSmallVolume>=50 || InpBigVolume<0 || InpBigVolume>=50)
     {
      Print("Inkorrekt Eingabeparameters");
      return(INIT_PARAMETERS_INCORRECT);
     }
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,ExtOpenBuff);
   SetIndexBuffer(1,ExtHighBuff);
   SetIndexBuffer(2,ExtLowBuff);
   SetIndexBuffer(3,ExtCloseBuff);
   SetIndexBuffer(4,ExtColorBuff,INDICATOR_COLOR_INDEX);
//--- erstellen Sie die Wert, die nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,INDICATOR_EMPTY_VALUE);
//--- stellen Sie die Beschriftungen für die Indikator Puffers
   PlotIndexSetString(0,PLOT_LABEL,"Open;High;Low;Close");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- prüfen Sie, ob es noch die unbearbeiteten Bars sind
```

© 2000-2025, MetaQuotes Ltd.

```
   if(prev_calculated<rates_total)
     {
      //--- bekommen Sie die neue Werte der linken und rechten Grenzen für Volumen
      if(!GetVolumeBorders())
         return(0);
     }
//---Variable zur Berechnung den Anfang des Bars
   int start=prev_calculated;
//--- wen die Indikatorwerte auf vorhergehend Tick schon berechnet waren, so ist auf d
   if(start>0)
      start--;
//--- stellen Sie die direkte Richtung der Indizierung in Zeitspanne
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(open,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
   ArraySetAsSeries(close,false);
   ArraySetAsSeries(tick_volume,false);
//--- der Zyklus für Berechnung der Indikatorwerte
   for(int i=start;i<rates_total;i++)
     {
      //--- zeichnen Sie die Kerzen seit dem Anfangsdatum
      if(ExtStartTime<=time[i])
        {
         //--- wenn der Wert ist nicht weniger als die rechte Grenze, dann zeichnen Si
         if(tick_volume[i]>=ExtRightBorder)
           {
            //--- bekommen Sie die Daten um die Kerze zu zeichnen
            ExtOpenBuff[i]=open[i];
            ExtHighBuff[i]=high[i];
            ExtLowBuff[i]=low[i];
            ExtCloseBuff[i]=close[i];
            //--- die Farbe DodgerBlue
            ExtColorBuff[i]=0;
            //--- fahren Sie die Zyklus fort weiter
            continue;
           }
         //--- wenn dir Wert ist nicht mehr als die linke Grenze, dann zeichnen Sie di
         if(tick_volume[i]<=ExtLeftBorder)
           {
            //--- bekommen Sie die Daten um die Kerze zu zeichnen
            ExtOpenBuff[i]=open[i];
            ExtHighBuff[i]=high[i];
            ExtLowBuff[i]=low[i];
            ExtCloseBuff[i]=close[i];
            //--- die Farbe Orange
            ExtColorBuff[i]=1;
            //--- fahren Sie die Zyklus fort weiter
            continue;
           }
        }
      //--- für die Bars, die nicht in die Berechnung gerieten, stellen Sie der leere
      ExtOpenBuff[i]=INDICATOR_EMPTY_VALUE;
      ExtHighBuff[i]=INDICATOR_EMPTY_VALUE;
      ExtLowBuff[i]=INDICATOR_EMPTY_VALUE;
      ExtCloseBuff[i]=INDICATOR_EMPTY_VALUE;
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

ArrayBsearch

© 2000-2025, MetaQuotes Ltd.

ArraySwap

Vertauscht die Inhalte von zwei dynamischen Arrays eines Typs. Für mehrdimensionale Arrays muss die Anzahl der Elemente in allen Dimensionen außer der ersten gleich sein.

```
bool  ArraySwap(
   void&  array1[],      // erstes Array
   void&  array2[]       // zweites Array
   );
```

Parameter

array1[]

[in][out] Array numerischen Typs.

array2[]

[in][out] Array numerischen Typs.

Rückgabewert

Wenn erfolgreich true, andernfalls false. In diesem Fall gibt GetLastError() den Fehlercode ERR_INVALID_ARRAY zurück.

Hinweis

Die Funktion akzeptiert dynamische Arrays von einem Typ und einer Dimension, außer der ersten. Für Integer-Typen wird das Vorzeichen ignoriert, d.h. char==uchar

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Arrays für das Speichern von Kursen
   double source_array[][8];
   double   dest_array[][8];
   MqlRates rates[];
//--- Daten der letzten 20 Kerzen im aktuellen Zeitrahmen erhalten
   int copied=CopyRates(NULL,0,0,20,rates);
   if(copied<=0)
     {
      PrintFormat("CopyRates(%s,0,0,20,rates) failed, error=%d",
                  Symbol(),GetLastError());
      return;
     }
//--- die Größe des Arrays entsprechend der Anzahl der kopierten Daten setzen
   ArrayResize(source_array,copied);
//--- das Array rate_array_1[] mit Daten aus rates[] füllen
   for(int i=0;i<copied;i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      source_array[i][0]=(double)rates[i].time;
      source_array[i][1]=rates[i].open;
      source_array[i][2]=rates[i].high;
      source_array[i][3]=rates[i].low;
      source_array[i][4]=rates[i].close;
      source_array[i][5]=(double)rates[i].tick_volume;
      source_array[i][6]=(double)rates[i].spread;
      source_array[i][7]=(double)rates[i].real_volume;
     }
//--- Daten zwischen source_array[] und dest_array[] vertauschen
   if(!ArraySwap(source_array,dest_array))
     {
      PrintFormat("ArraySwap(source_array,rate_array_2) failed, error code=%d",GetLast
      return;
     }
//--- sicherstellen, dass die Größe des Quellarrays nach dem Vertauschen gleich Null
   PrintFormat("ArraySwap() done: ArraySize(source_array)=%d",ArraySize(source_array))
//--- Daten des Empfänger-Arrays dest_array[] ausgeben
   ArrayPrint(dest_array);
  }
```

Siehe auch

ArrayCopy, ArrayFill, ArrayRange, ArrayIsDynamic

© 2000-2025, MetaQuotes Ltd.

ArrayToFP16

Kopiert ein Array vom Typ float oder double in ein Array vom Typ ushort mit dem angegebenen Format.

```
bool   ArrayToFP16(
   const ushort&         dst_array[],        // copy to
   const float&          src_array[],        // copy from
   ENUM_FLOAT16_FORMAT   fmt                 // format
   );
```

Überladung für den Typ double

```
bool   ArrayToFP16(
   const ushort&         dst_array[],        // copy to
   const double&         src_array[],        // copy from
   ENUM_FLOAT16_FORMAT   fmt                 // format
   );
```

Parameter

dst_array[]

[out] Empfänger-Array vom Typ ushort.

src_array[]

[in] Quell-Array vom Typ float oder double.

fmt

[in] Format der Kopie aus der Enumeration ENUM_FLOAT16_FORMAT.

Rückgabewert

Gibt bei Erfolg „true“ zurück, andernfalls „false“.

Hinweis

Die Formate FLOAT16 und BFLOAT16 sind in der Enumeration ENUM_FLOAT16_FORMAT definiert und werden in MQL5 nur für Operationen mit ONNX-Modelle verwendet.

Die Funktion konvertiert die Eingabeparameter vom Typ float oder double in den Typ FLOAT16 und BFLOAT16. Diese Eingabeparameter werden dann in der Funktion OnnxRun verwendet.

FLOAT16, auch bekannt als half-precision float (Halbgenaues-Float), verwendet 16 Bit zur Darstellung von Gleitkommazahlen. Dieses Format bietet ein Gleichgewicht zwischen Genauigkeit und Recheneffizienz. FLOAT16 wird häufig in Deep-Learning-Algorithmen und neuronalen Netzen verwendet, die eine Hochleistungsverarbeitung großer Datensätze erfordern. Dieses Format beschleunigt Berechnungen durch die Reduzierung der Zahlengröße, was besonders wichtig ist, wenn tiefe neuronale Netze auf GPUs trainiert werden.

BFLOAT16 (oder Brain Floating Point 16) verwendet ebenfalls 16 Bit, unterscheidet sich jedoch von FLOAT16 im Ansatz der Formatdarstellung. In diesem Format werden 8 Bits zur Darstellung des

© 2000-2025, MetaQuotes Ltd.

Exponenten zugewiesen, während die restlichen 7 Bits zur Darstellung der Mantisse verwendet werden. Dieses Format wurde für den Einsatz im Deep Learning und der künstlichen Intelligenz entwickelt, insbesondere in der Tensor Processing Unit (TPU) von Google. BFLOAT16 zeigt eine hervorragende Leistung beim Training neuronaler Netzwerke und kann Berechnungen effektiv beschleunigen.

Beispiel: Die Funktion aus dem Artikel „Arbeiten mit ONNX-Modellen in den Formaten float16 und float8“

```
//+------------------------------------------------------------------+
//| RunCastFloat16ToDouble                                           |
//+------------------------------------------------------------------+
bool RunCastFloat16ToDouble(long model_handle)
  {
   PrintFormat("test=%s",__FUNCTION__);
   double test_data[12]= {1,2,3,4,5,6,7,8,9,10,11,12};
   ushort data_uint16[12];
   if(!ArrayToFP16(data_uint16,test_data,FLOAT_FP16))
     {
      Print("error in ArrayToFP16. error code=",GetLastError());
      return(false);
     }
   Print("test array:");
   ArrayPrint(test_data);
   Print("ArrayToFP16:");
   ArrayPrint(data_uint16);
   U<ushort> input_float16_values[3*4];
   U<double> output_double_values[3*4];
   float test_data_float[];
   if(!ArrayFromFP16(test_data_float,data_uint16,FLOAT_FP16))
     {
      Print("error in ArrayFromFP16. error code=",GetLastError());
      return(false);
     }
   for(int i=0; i<12; i++)
     {
      input_float16_values[i].value=data_uint16[i];
      PrintFormat("%d input value =%f  Hex float16 = %s  ushort value=%d",i,test_data_
     }
   Print("ONNX input array:");
   ArrayPrint(input_float16_values);
   bool res=OnnxRun(model_handle,ONNX_NO_CONVERSION,input_float16_values,output_double
   if(!res)
     {
      PrintFormat("error in OnnxRun. error code=%d",GetLastError());
      return(false);
     }
   Print("ONNX output array:");
```

© 2000-2025, MetaQuotes Ltd.

```
   ArrayPrint(output_double_values);
//---
   double sum_error=0.0;
   for(int i=0; i<12; i++)
     {
      double delta=test_data[i]-output_double_values[i].value;
      sum_error+=MathAbs(delta);
      PrintFormat("%d output double %f = %s  difference=%f",i,output_double_values[i].
     }
//---
   PrintFormat("test=%s   sum_error=%f",__FUNCTION__,sum_error);
//---
   return(true);
  }
```

Siehe auch

ArrayFromFP16, ArrayCopy

© 2000-2025, MetaQuotes Ltd.

ArrayToFP8

Kopiert ein Array vom Typ float oder double in ein Array vom Typ uchar mit dem angegebenen Format.

```
bool   ArrayToFP8(
   const uchar&          dst_array[],        // copy to
   const float&          src_array[],        // copy from
   ENUM_FLOAT8_FORMAT    fmt                 // format
   );
```

Überladung für den Typ double

```
bool   ArrayToFP8(
   const uchar&          dst_array[],        // copy to
   const double&         src_array[],        // copy from
   ENUM_FLOAT8_FORMAT    fmt                 // format
   );
```

Parameter

dst_array[]

[out] Empfänger-Array des Typs uchar.

src_array[]

[in] Quell-Array vom Typ float oder double.

fmt

[in] Format der Kopie aus der Enumeration ENUM_FLOAT8_FORMAT.

Rückgabewert

Gibt bei Erfolg „true“ zurück, andernfalls „false“.

Hinweis

Alle Arten von FP8-Formaten sind in der Enumeration ENUM_FLOAT8_FORMAT definiert und werden in MQL5 nur für Operationen mit ONNX-Modellen verwendet.

Die Funktion konvertiert Eingabeparameter vom Typ float oder double in einen der FP8-Typen. Diese Eingabeparameter werden dann in der Funktion OnnxRun verwendet.

FP8 (8-Bit-floating point) ist einer der Datentypen, die zur Darstellung von Gleitkommazahlen verwendet werden. In FP8 wird jede Zahl durch 8 Datenbits dargestellt, die typischerweise in drei Komponenten unterteilt sind: Vorzeichen, Exponent und Mantisse. Dieses Format bietet ein Gleichgewicht zwischen Genauigkeit und Speichereffizienz und ist daher attraktiv für Anwendungen, die Speicher und Recheneffizienz erfordern.

Durch die Verwendung einer kompakten Zahlendarstellung reduziert FP8 den Speicherbedarf und beschleunigt die Berechnungen. Darüber hinaus kann FP8 für die Implementierung von Operationen auf niedriger Ebene wie arithmetische Berechnungen und Signalverarbeitung nützlich sein.

© 2000-2025, MetaQuotes Ltd.

Beispiel: Die Funktion aus dem Artikel „Arbeiten mit ONNX-Modellen in den Formaten float16 und float8“

```
//+------------------------------------------------------------------+
//| RunCastFloat8Float                                               |
//+------------------------------------------------------------------+
bool RunCastFloat8ToFloat(long model_handle,const ENUM_FLOAT8_FORMAT fmt)
  {
   PrintFormat("TEST: %s(%s)",__FUNCTION__,EnumToString(fmt));
//---
   float test_data[15]   = {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15};
   uchar data_float8[15] = {};
   if(!ArrayToFP8(data_float8,test_data,fmt))
     {
      Print("error in ArrayToFP8. error code=",GetLastError());
      OnnxRelease(model_handle);
      return(false);
     }
   U<uchar> input_float8_values[3*5];
   U<float> output_float_values[3*5];
   float    test_data_float[];
//--- convert float8 to float
   if(!ArrayFromFP8(test_data_float,data_float8,fmt))
     {
      Print("error in ArrayFromFP8. error code=",GetLastError());
      OnnxRelease(model_handle);
      return(false);
     }
   for(uint i=0; i<data_float8.Size(); i++)
     {
      input_float8_values[i].value=data_float8[i];
      PrintFormat("%d input value =%f  Hex float8 = %s  ushort value=%d",i,test_data_f
     }
   Print("ONNX input array: ",ArrayToString(input_float8_values));
//--- execute model (convert float8 to float using ONNX)
   if(!OnnxRun(model_handle,ONNX_NO_CONVERSION,input_float8_values,output_float_values
     {
      PrintFormat("error in OnnxRun. error code=%d",GetLastError());
      OnnxRelease(model_handle);
      return(false);
     }
   Print("ONNX output array: ",ArrayToString(output_float_values));
//--- calculate error (compare ONNX and ArrayFromFP8 results)
   double sum_error=0.0;
   for(uint i=0; i<test_data.Size(); i++)
     {
      double delta=test_data_float[i]-(double)output_float_values[i].value;
```

© 2000-2025, MetaQuotes Ltd.

```
      sum_error+=MathAbs(delta);
      PrintFormat("%d output float %f = %s difference=%f",i,output_float_values[i].val
     }
//---
   PrintFormat("%s(%s): sum_error=%f\n",__FUNCTION__,EnumToString(fmt),sum_error);
   return(true);
  }
```

Siehe auch

ArrayFromFP8, ArrayCopy

© 2000-2025, MetaQuotes Ltd.

ArrayFromFP16

Kopiert ein Array vom Typ ushort in ein Array vom Typ float oder double mit dem angegebenen Format.

```
bool   ArrayFromFP16(
   const float&          dst_array[],        // copy to
   const ushort&         src_array[],        // copy from
   ENUM_FLOAT16_FORMAT   fmt                 // format
   );
```

Überladung für den Typ double

```
bool   ArrayFromFP16(
   const double&         dst_array[],        // copy to
   const ushort&         src_array[],        // copy from
   ENUM_FLOAT16_FORMAT   fmt                 // format
   );
```

Parameter

dst_array[]

[out] Empfänger-Array des Typs float oder double.

src_array[]

[in] Quellarray vom Typ ushort.

fmt

[in] Format der Kopie aus der Enumeration ENUM_FLOAT16_FORMAT.

Rückgabewert

Gibt bei Erfolg „true“ zurück, andernfalls „false“.

Hinweis

Die Formate FLOAT16 und BFLOAT16 sind in der Enumeration ENUM_FLOAT16_FORMAT definiert und werden in MQL5 nur für Operationen mit ONNX-Modelle verwendet.

Wenn die aus der Ausführung der Funktion OnnxRun erhaltenen Ausgabeparameter vom Typ FLOAT16 und BFLOAT16 sind, können Sie diese Funktion verwenden, um das Ergebnis in Float- oder Double-Arrays zu konvertieren.

FLOAT16, auch bekannt als half-precision float (Halbgenaues-Float), verwendet 16 Bit zur Darstellung von Gleitkommazahlen. Dieses Format bietet ein Gleichgewicht zwischen Genauigkeit und Recheneffizienz. FLOAT16 wird häufig in Deep-Learning-Algorithmen und neuronalen Netzen verwendet, die eine Hochleistungsverarbeitung großer Datensätze erfordern. Dieses Format beschleunigt Berechnungen durch die Reduzierung der Zahlengröße, was besonders wichtig ist, wenn tiefe neuronale Netze auf GPUs trainiert werden.

© 2000-2025, MetaQuotes Ltd.

BFLOAT16 (oder Brain Floating Point 16) verwendet ebenfalls 16 Bit, unterscheidet sich jedoch von FLOAT16 im Ansatz der Formatdarstellung. In diesem Format werden 8 Bits zur Darstellung des Exponenten zugewiesen, während die restlichen 7 Bits zur Darstellung der Mantisse verwendet werden. Dieses Format wurde für den Einsatz im Deep Learning und der künstlichen Intelligenz entwickelt, insbesondere in der Tensor Processing Unit (TPU) von Google. BFLOAT16 zeigt eine hervorragende Leistung beim Training neuronaler Netzwerke und kann Berechnungen effektiv beschleunigen.

Beispiel: Die Funktion aus dem Artikel „Arbeiten mit ONNX-Modellen in den Formaten float16 und float8“

```
//+------------------------------------------------------------------+
//| RunCastFloat16ToDouble                                           |
//+------------------------------------------------------------------+
bool RunCastFloat16ToDouble(long model_handle)
  {
   PrintFormat("test=%s",__FUNCTION__);
   double test_data[12]= {1,2,3,4,5,6,7,8,9,10,11,12};
   ushort data_uint16[12];
   if(!ArrayToFP16(data_uint16,test_data,FLOAT_FP16))
     {
      Print("error in ArrayToFP16. error code=",GetLastError());
      return(false);
     }
   Print("test array:");
   ArrayPrint(test_data);
   Print("ArrayToFP16:");
   ArrayPrint(data_uint16);
   U<ushort> input_float16_values[3*4];
   U<double> output_double_values[3*4];
   float test_data_float[];
   if(!ArrayFromFP16(test_data_float,data_uint16,FLOAT_FP16))
     {
      Print("error in ArrayFromFP16. error code=",GetLastError());
      return(false);
     }
   for(int i=0; i<12; i++)
     {
      input_float16_values[i].value=data_uint16[i];
      PrintFormat("%d input value =%f  Hex float16 = %s  ushort value=%d",i,test_data_
     }
   Print("ONNX input array:");
   ArrayPrint(input_float16_values);
   bool res=OnnxRun(model_handle,ONNX_NO_CONVERSION,input_float16_values,output_double
   if(!res)
     {
      PrintFormat("error in OnnxRun. error code=%d",GetLastError());
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   Print("ONNX output array:");
   ArrayPrint(output_double_values);
//---
   double sum_error=0.0;
   for(int i=0; i<12; i++)
     {
      double delta=test_data[i]-output_double_values[i].value;
      sum_error+=MathAbs(delta);
      PrintFormat("%d output double %f = %s  difference=%f",i,output_double_values[i].
     }
//---
   PrintFormat("test=%s   sum_error=%f",__FUNCTION__,sum_error);
//---
   return(true);
  }
```

Siehe auch

ArrayToFP16, ArrayCopy

© 2000-2025, MetaQuotes Ltd.

ArrayFromFP8

Kopiert ein Array vom Typ uchar in ein Array vom Typ float oder double mit dem angegebenen Format.

```
bool   ArrayFromFP8(
   const float&          dst_array[],        // copy to
   const uchar&          src_array[],        // copy from
   ENUM_FLOAT8_FORMAT    fmt                 // format
   );
```

Überladung für den Typ double

```
bool   ArrayFromFP8(
   const double&         dst_array[],        // copy to
   const uchar&          src_array[],        // copy from
   ENUM_FLOAT8_FORMAT    fmt                 // format
   );
```

Parameter

dst_array[]

[out] Empfänger-Array des Typs float oder double.

src_array[]

[in] Quellarray vom Typ uchar.

fmt

[in] Format der Kopie aus der Enumeration ENUM_FLOAT8_FORMAT.

Rückgabewert

Gibt bei Erfolg „true“ zurück, andernfalls „false“.

Hinweis

Alle Arten von FP8-Formaten sind in der Enumeration ENUM_FLOAT8_FORMAT definiert und werden in MQL5 nur für Operationen mit ONNX-Modellen verwendet.

Wenn die aus der Ausführung der Funktion OnnxRun erhaltenen Ausgabeparameter FP8 aus der Enumeration ENUM_FLOAT8_FORMAT sind, können Sie diese Funktion verwenden, um das Ergebnis in Float- oder Double-Arrays umzuwandeln.

FP8 (8-Bit-floating point) ist einer der Datentypen, die zur Darstellung von Gleitkommazahlen verwendet werden. In FP8 wird jede Zahl durch 8 Datenbits dargestellt, die typischerweise in drei Komponenten unterteilt sind: Vorzeichen, Exponent und Mantisse. Dieses Format bietet ein Gleichgewicht zwischen Genauigkeit und Speichereffizienz und ist daher attraktiv für Anwendungen, die Speicher und Recheneffizienz erfordern.

Durch die Verwendung einer kompakten Zahlendarstellung reduziert FP8 den Speicherbedarf und beschleunigt die Berechnungen. Darüber hinaus kann FP8 für die Implementierung von Operationen auf niedriger Ebene wie arithmetische Berechnungen und Signalverarbeitung nützlich sein.

© 2000-2025, MetaQuotes Ltd.

Beispiel: Die Funktion aus dem Artikel „Arbeiten mit ONNX-Modellen in den Formaten float16 und float8“

```
//+------------------------------------------------------------------+
//| RunCastFloat8Float                                               |
//+------------------------------------------------------------------+
bool RunCastFloat8ToFloat(long model_handle,const ENUM_FLOAT8_FORMAT fmt)
  {
   PrintFormat("TEST: %s(%s)",__FUNCTION__,EnumToString(fmt));
//---
   float test_data[15]   = {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15};
   uchar data_float8[15] = {};
   if(!ArrayToFP8(data_float8,test_data,fmt))
     {
      Print("error in ArrayToFP8. error code=",GetLastError());
      OnnxRelease(model_handle);
      return(false);
     }
   U<uchar> input_float8_values[3*5];
   U<float> output_float_values[3*5];
   float    test_data_float[];
//--- convert float8 to float
   if(!ArrayFromFP8(test_data_float,data_float8,fmt))
     {
      Print("error in ArrayFromFP8. error code=",GetLastError());
      OnnxRelease(model_handle);
      return(false);
     }
   for(uint i=0; i<data_float8.Size(); i++)
     {
      input_float8_values[i].value=data_float8[i];
      PrintFormat("%d input value =%f  Hex float8 = %s  ushort value=%d",i,test_data_f
     }
   Print("ONNX input array: ",ArrayToString(input_float8_values));
//--- execute model (convert float8 to float using ONNX)
   if(!OnnxRun(model_handle,ONNX_NO_CONVERSION,input_float8_values,output_float_values
     {
      PrintFormat("error in OnnxRun. error code=%d",GetLastError());
      OnnxRelease(model_handle);
      return(false);
     }
   Print("ONNX output array: ",ArrayToString(output_float_values));
//--- calculate error (compare ONNX and ArrayFromFP8 results)
   double sum_error=0.0;
   for(uint i=0; i<test_data.Size(); i++)
     {
      double delta=test_data_float[i]-(double)output_float_values[i].value;
```

© 2000-2025, MetaQuotes Ltd.

```
      sum_error+=MathAbs(delta);
      PrintFormat("%d output float %f = %s difference=%f",i,output_float_values[i].val
     }
//---
   PrintFormat("%s(%s): sum_error=%f\n",__FUNCTION__,EnumToString(fmt),sum_error);
   return(true);
  }
```

Siehe auch

ArrayToFP8, ArrayCopy

© 2000-2025, MetaQuotes Ltd.

Matrizen und Vektoren

Eine Matrix ist eine zweidimensionale Anordnung von Double-, Float- oder komplexen Zahlen.

Ein Vektor ist eine eindimensionale Anordnung von Double-, Float- oder komplexen Zahlen. Der Vektor hat keinen Hinweis darauf, ob er vertikal oder horizontal ist. Dies wird durch den Verwendungskontext bestimmt. So wird beispielsweise bei der Vektoroperation Dot davon ausgegangen, dass der linke Vektor horizontal und der rechte vertikal ist. Wenn die Angabe des Typs erforderlich ist, können einzeilige oder einspaltige Matrizen verwendet werden. Im Allgemeinen ist dies jedoch nicht erforderlich.

Matrizen und Vektoren weisen den Speicher für Daten dynamisch zu. Matrizen und Vektoren sind Objekte, die bestimmte Eigenschaften haben, z. B. die Art der Daten, die sie enthalten, und die Dimensionen. Die Eigenschaften von Matrizen und Vektoren können mit Methoden wie vector_a.Size(), matrix_b.Rows(), vector_c.Norm(), matrix_d.Cond() und anderen ermittelt werden. Jede Dimension kann geändert werden.

Bei der Erstellung und Initialisierung von Matrizen werden sogenannte statische Methoden verwendet (diese sind wie statische Methoden einer Klasse). Zum Beispiel: matrix::Eye(), matrix::Identity(), matrix::Ones(), vector::Ones(), matrix: :Zeros(), vector::Zeros(), matrix::Full(), vector::Full(), matrix::Tri().

Matrix- und Vektoroperationen implizieren im Moment nicht die Verwendung des komplexen Datentyps, da diese Entwicklungsrichtung noch nicht abgeschlossen ist.

MQL5 unterstützt die Übergabe von Matrizen und Vektoren an DLLs. Dies ermöglicht den Import von Funktionen, die die entsprechenden Typen verwenden, aus externen Variablen.

Matrizen und Vektoren werden als Zeiger auf einen Puffer an eine DLL übergeben. Um z. B. eine Matrix vom Typ float zu übergeben, muss der entsprechende Parameter der aus der DLL exportierten Funktion einen Pufferzeiger vom Typ float aufnehmen.

MQL5

```
#import "mmlib.dll"
bool sgemm(uint flags, matrix<float> &C, const matrix<float> &A, const matrix<float> &
#import
```

C++

```
extern "C" __declspec(dllexport) bool sgemm(UINT flags, float *C, const float *A, cons
```

Zusätzlich zu den Puffern sollten Sie Matrix- und Vektorgrößen für eine korrekte Verarbeitung übergeben.

Alle Matrix- und Vektormethoden sind unten in alphabetischer Reihenfolge aufgeführt.

Funktion	Aktion	Kategorie
Activation	Aktivierungsfunktionswerte berechnen und in den übergebenen Vektor oder Matrix schreiben.	Maschinelles Lernen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
ArgMax	Rückgabe des Index des größten Wertes.	Statistik
ArgMin	Rückgabe des Index des kleinsten Wertes.	Statistik
ArgSort	Rückgabe des sortierten Indexes	Manipulation
Assign	Kopiert eine Matrix, einen Vektor oder ein Array mit automatischer Typ-Umwandlung.	Initialisierung
Average	Berechnet den gewichteten Durchschnitt von Matrix oder Vektorwerten.	Statistik
Cholesky	Berechnen der Cholesky-Zerlegung.	Transformationen
Clip	Begrenzt die Elemente einer Matrix oder eines Vektors auf einen bestimmten Bereich gültiger Werte.	Manipulation
Col	Gibt einen Spaltenvektor zurück. Schreibt einen Vektor in die angegebene Spalte.	Manipulation
Cols	Rückgabe der Anzahl der Spalten einer Matrix.	Eigenschaften
Compare	Vergleich der Elemente von zwei Matrizen oder Vektoren mit der angegebenen Präzision.	Manipulation
CompareByDigits	Vergleicht die Elemente zweier Matrizen oder Vektoren mit einer signifikanten Zahlengenauigkeit.	Manipulation
Cond	Berechnen der Konditionszahl einer Matrix.	Eigenschaften
Convolve	Liefert die diskrete, lineare Faltung (Convolution) von zwei Vektoren.	Produkte
Copy	Gibt eine Kopie der gegebenen Matrix oder des gegebenen Vektors zurück.	Manipulation
Concat	Verketten von 2 Submatrizen zu einer Matrix. Verketten von 2 Vektoren zu einem Vektor.	Manipulation

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
CopyIndicatorBuffer	Abrufen der Daten des spezifizierten Indikator-Puffers im angegebenen Umfang Menge in einen Vektor	Initialisierung
CopyRates	Gibt die historischen Zeitreihendaten der Struktur MqlRates der angegebenen Symbolperiodenlänge in der angegebenen Menge in eine Matrix oder einen Vektor zurück.	Initialisierung
CopyTicks	Weist die Ticks der Struktur MqlTick einer Matrix oder einem Vektor zu.	Initialisierung
CopyTicksRange	Weist die Ticks des angegebenen Zeitintervalls aus der Struktur MqlTick einer Matrix oder einem Vektor zu.	Initialisierung
CorrCoef	Berechnet den Pearson-Korrelationskoeffizienten (linearer Korrelationskoeffizient).	Produkte
Correlate	Berechnung der Kreuzkorrelation von zwei Vektoren.	Produkte
Cov	Berechnet die Kovarianzmatrix.	Produkte
CumProd	Rückgabe des kumulativen Produkts von Matrix- oder Vektorelementen, einschließlich der Elemente entlang der angegebenen Achse.	Statistik
CumSum	Rückgabe der kumulativen Summe der Matrix- oder Vektorelemente, einschließlich derjenigen entlang der angegebenen Achse.	Statistik
Derivative	Berechnet die Ableitungswerte der Aktivierungsfunktion, die dem übergebenen Vektor oder Matrix zugewiesen werden.	Maschinelles Lernen
Det	Berechnen der Determinante einer quadratischen invertierbaren Matrix.	Eigenschaften

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
Diag	Eine Diagonale extrahieren oder eine Diagonalmatrix konstruieren.	Manipulation
Dot	Punktprodukt von zwei Vektoren.	Produkte
Eig	Berechnet die Eigenwerte und rechten Eigenvektoren einer quadratischen Matrix.	Transformationen
EigVals	Berechnet die Eigenwerte einer allgemeinen Matrix.	Transformationen
Eye	Rückgabe einer Matrix mit Einsen auf der Diagonale und Nullen an anderer Stelle.	Initialisierung
Fill	Füllen des angegebenen Wertes einer vorhandenen Matrix oder Vektors.	Initialisierung
Flat	Zugriff auf ein Matrixelement über einen Index statt über zwei.	Manipulation
Full	Erstellung und Rückgabe einer neuen Matrix, die mit dem angegebenen Wert gefüllt wurde.	Initialisierung
GeMM	Die Methode GeMM (General Matrix Multiply) implementiert die allgemeine Multiplikation von zwei Matrizen.	Produkte
HasNan	Rückgabe einer Zahl mit dem Wert NaN in einer Matrix/einem Vektor	Manipulation
Hsplit	Horizontales Teilen einer Matrix in mehrere Submatrizen. Dasselbe wie Split mit axis=0.	Manipulation
Identity	Erzeugt eine Identitätsmatrix mit der angegebenen Größe.	Initialisierung
Init	Initialisierung einer Matrix oder eines Vektors.	Initialisierung
Inner	Inneres Produkt von zwei Matrizen.	Produkte

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
Inv	Berechnung der multiplikativen Inversen einer quadratischen invertierbaren Matrix nach der Jordan-Gauss-Methode.	Lösungen
Kron	Rückgabe des Kronecker-Produkts von zwei Matrizen, Matrix und Vektor, Vektor und Matrix oder zwei Vektoren.	Produkte
LinearRegression	Berechnen eines Vektors/einer Matrix mit berechneten linearen Regressionswerten	Statistik
Loss	Berechnet die Verlustfunktionswerte und schreibt sie in den übergebenen Vektor oder Matrix.	Maschinelles Lernen
LstSq	Rückgabe der Lösung der kleinsten Quadrate von linearen algebraischen Gleichungen (für nicht quadratische oder entartete Matrizen).	Lösungen
LU	Implementierung einer LU-Zerlegung einer Matrix: das Produkt aus einer unteren und einer oberen Dreiecksmatrix.	Transformationen
LUP	Implementiert eine LUP-Faktorisierung mit partieller Permutation, d.h. eine LU-Zerlegung nur mit Zeilenpermutationen: PA=LU.	Transformationen
MatMul	Matrixprodukt von zwei Matrizen.	Produkte
Max	Rückgabe des größten Wertes einer Matrix oder Vektors.	Statistik
Mean	Berechnung des arithmetischen Mittels der Elementwerte.	Statistik
Median	Berechnung des Medians der Matrix- oder Vektorelemente.	Statistik
Min	Rückgabe des kleinsten Wertes einer Matrix oder Vektors.	Statistik
Norm	Rückgabe der Matrix- oder Vektornorm.	Eigenschaften

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
Ones	Erzeugen und Zurückgeben einer neuen Matrix aus Einsen.	Initialisierung
Outer	Berechnet das äußere Produkt von zwei Matrizen oder zwei Vektoren.	Produkte
Percentile	Rückgabe des angegebenen Perzentils der Werte von Matrix- bzw. Vektorelementen oder den Elementen entlang der angegebenen Achse.	Statistik
PInv	Berechnung der Pseudoinverse einer Matrix nach der Moore-Penrose-Methode.	Lösungen
Power	Erhöhen einer quadratischen Matrix auf eine ganzzahlige Potenz.	Produkte
Prod	Rückgabe des Produkts von Matrix- oder Vektorelementen, das auch für die angegebene Achse ausgeführt werden kann.	Statistik
Ptp	Gibt den Wertebereich einer Matrix bzw. Vektors oder der angegebenen Matrixachse zurück.	Statistik
QR	Berechnen der qr-Faktorisierung einer Matrix.	Transformationen
Quantile	Rückgabe des angegebenen Quantils der Werte von Matrix- bzw. Vektorelementen oder Elementen entlang der angegebenen Achse.	Statistik
Random	Statische Funktion. Sie erstellt und gibt eine neue Matrix oder einen neuen Vektor mit Zufallswerten zurück. Zufallswerte werden gleichmäßig innerhalb des angegebenen Bereichs generiert.	Initialisierung
Rank	Liefert den Rang der Matrix unter Verwendung der Gaußschen Methode.	Eigenschaften

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
RegressionMetric	Berechnet die Regressionsmetrik als Abweichung von der Regressionslinie, die auf dem angegebenen Datenarray konstruiert wurde.	Statistik
Reshape	Ändern der Form einer Matrix, ohne ihre Daten zu ändern.	Manipulation
Resize	Rückgabe einer neuen Matrix mit geänderter Form und Größe.	Manipulation
Row	Gibt einen Zeilenvektor zurück. Schreiben des Vektors in die angegebene Zeile.	Manipulation
Rows	Rückgabe der Anzahl der Zeilen einer Matrix.	Eigenschaften
Set	Setzt den Wert eines Vektorelements am angegebenen Index.	Manipulation
Size	Rückgabe der Größe des Vektors.	Eigenschaften
SLogDet	Berechnet das Vorzeichen und den Logarithmus der Determinante einer Matrix.	Eigenschaften
Solve	Lösen einer linearen Matrixgleichung oder eines Systems linearer algebraischer Gleichungen.	Lösungen
Sort	Nach einem Platz sortieren.	Manipulation
Spectrum	Berechnung des Spektrums einer Matrix als die Menge ihrer Eigenwerte aus dem Produkt AT*A.	Eigenschaften
Split	Aufteilen einer Matrix in mehrere Submatrizen.	Manipulation
Std	Gibt die Standardabweichung der Werte von Matrix- bzw. Vektorelementen oder Elementen entlang der angegebenen Achse zurück.	Statistik
Sum	Rückgabe der Summe der Matrix- oder Vektorelemente,	Statistik

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Kategorie
	die auch für die angegebene Achse (Achsen) ausgeführt werden kann.	
SVD	Zerlegung in Singulärwerte.	Transformationen
SwapCols	Vertauschen der Spalten in einer Matrix.	Manipulation
SwapRows	Vertauschen der Zeilen in einer Matrix.	Manipulation
Trace	Rückgabe der Summe entlang der Diagonalen der Matrix.	Eigenschaften
Transpose	Transponieren (Vertauschen der Achsen) und Rückgabe der geänderten Matrix.	Manipulation
Tri	Konstruiert eine Matrix mit Einsen auf der angegebenen Diagonalen und darunter und Nullen anderswo.	Initialisierung
TriL	Gibt die Kopie einer Matrix zurück, bei der die Elemente oberhalb der k-ten Diagonale auf Null gesetzt sind. Untere Dreiecksmatrix.	Manipulation
TriU	Gibt die Kopie einer Matrix zurück, bei der die Elemente unterhalb der k-ten Diagonale auf Null gesetzt sind. Obere Dreiecksmatrix.	Manipulation
Var	Berechnung der Varianz der Werte von Matrix- oder Vektorelementen.	Statistik
Vsplit	Teilt eine Matrix vertikal in mehrere Submatrizen auf. Entspricht Split mit Achse=1	Manipulation
Zeros	Erzeugt eine neue, mit Nullen gefüllte Matrix und gibt sie zurück.	Initialisierung

© 2000-2025, MetaQuotes Ltd.

Typen von Matrix und Vektor

Matrix und Vektor sind spezielle Datentypen in MQL5, die lineare Algebra-Operationen ermöglichen. Es gibt die folgenden Datentypen:

- matrix — eine Matrix mit Elementen des Typs double.
- matrixf — eine Matrix mit Elementen des Typs float.
- matrixc — eine Matrix mit Elementen des Typs complex.
- vector — ein Vektor mit Elementen des Typs double.
- vectorf — ein Vektor mit Elementen des Typs float.
- vectorc — ein Vektor mit Elementen des Typs complex.

Template-Funktionen unterstützen Notationen wie matrix<double>, matrix<float>, vector<double>, vector<float> anstelle der entsprechenden Typen.

Methoden zur Initialisierung von Matrizen und Vektoren

Funktion	Aktion
Eye	Rückgabe einer Matrix mit Einsen auf der Diagonale und Nullen an anderer Stelle.
Identity	Erzeugt eine Identitätsmatrix mit der angegebenen Größe.
Ones	Erzeugen und Zurückgeben einer neuen Matrix aus Einsen.
Zeros	Erzeugt eine neue, mit Nullen gefüllte Matrix und gibt sie zurück.
Full	Erstellung und Rückgabe einer neuen Matrix, die mit dem angegebenen Wert gefüllt wurde.
Tri	Konstruktion einer Matrix mit Einsen auf und unterhalb der gegebenen Diagonalen und Nullen an anderen Stellen
Init	Initialisieren einer Matrix.
Fill	Füllen des angegebenen Wertes einer vorhandenen Matrix oder Vektors.

© 2000-2025, MetaQuotes Ltd.

Enumeration der Matrix- und Vektormethoden

Dieser Abschnitt beschreibt die Enumeration, die in verschiedenen Matrix- und Vektor-Methoden verwendet werden.

ENUM_AVERAGE_MODE

Enumeration der Glättungstypen

ID	Beschreibung
AVERAGE_NONE	Keine Glättung. Die Ergebnisse werden für jedes Label separat angegeben.
AVERAGE_BINARY	Ergebnis für Label 1 bei binärer Klassifizierung.
AVERAGE_MICRO	Ergebnismatrix der Durchschnittsfehler (Konfusionsmatrix).
AVERAGE_MACRO	Durchschnittliches Ergebnis aus den Ergebnissen der Fehlermatrizen der einzelnen Labels.
AVERAGE_WEIGHTED	Gewichtetes Durchschnittsergebnis.

ENUM_VECTOR_NORM

Enumeration für die Vektor-Methode vector::Norm.

ID	Beschreibung
VECTOR_NORM_INF	Die unendliche Norm
VECTOR_NORM_MINUS_INF	Negative unendliche Norm
VECTOR_NORM_P	P-Norm

ENUM_MATRIX_NORM

Enumeration der Matrizennormen für Matrix::Norm und für den Erhalt der Matrix::Cond Matrixbedingungsnummer.

ID	Beschreibung
MATRIX_NORM_FROBENIUS	Frobenius-Norm
MATRIX_NORM_SPECTRAL	Spektrale Norm
MATRIX_NORM_NUCLEAR	Nukleare Norm
MATRIX_NORM_INF	Die unendliche Norm

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
MATRIX_NORM_P1	P1-Norm
MATRIX_NORM_P2	P2-Norm
MATRIX_NORM_MINUS_INF	Negative unendliche Norm
MATRIX_NORM_MINUS_P1	Negative P1-Norm
MATRIX_NORM_MINUS_P2	Negative P2-Norm

ENUM_VECTOR_CONVOLVE

Enumeration für Faltungsvektoren vektor::Convolve und Kreuzkorrelationen vektor::Correlate.

ID	Beschreibung
VECTOR_CONVOLVE_FULL	Vollständige Faltung
VECTOR_CONVOLVE_SAME	Faltung mit Typ same
VECTOR_CONVOLVE_VALID	Faltung mit Typ valid

ENUM_REGRESSION_METRIC

Enumeration von Regressionsmetriken für vektor::RegressionMetric.

ID	Beschreibung
REGRESSION_MAE	Mittlerer absoluter Fehler
REGRESSION_MSE	Mittlerer quadratischer Fehler
REGRESSION_RMSE	Quadratwurzel des mittleren quadratischen Fehlers (MSE)
REGRESSION_R2	R-Quadrat
REGRESSION_MAPE	Mittlerer absoluter Fehler in Prozent
REGRESSION_MSPE	Prozentualer mittlerer quadratischer Fehler
REGRESSION_RMSLE	Logarithmierter prozentualer mittlerer quadratischer Fehler
REGRESSION_SMAPE	Symmetrischer mittlerer absoluter prozentualer Fehler
REGRESSION_MAXE	Maximaler absoluter Fehler
REGRESSION_MEDE	Median des absoluten Fehlers
REGRESSION_MPD	Mittlere Poisson-Abweichung
REGRESSION_MGD	Mittlere Gamma-Abweichung

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
REGRESSION_EXPV	Erklärbare Varianz

ENUM_CLASSIFICATION_METRIC

Enumeration der Metriken für Klassifizierungsprobleme.

ID	Beschreibung
CLASSIFICATION_ACCURACY	Modellqualität in Form der Vorhersagegenauigkeit für alle Klassen.
CLASSIFICATION_AVERAGE_PRECISION	Durchschnittliche Modellgenauigkeit.
CLASSIFICATION_BALANCED_ACCURACY	Ausgewogene Vorhersagegenauigkeit.
CLASSIFICATION_F1	F1-Score. Harmonisches Mittel zwischen der Modellpräzision und dem Recall.
CLASSIFICATION_JACCARD	Jaccard-Punktzahl
CLASSIFICATION_PRECISION	Genauigkeit des Modells bei der Vorhersage wahrer positiver Ergebnisse für die Zielklasse.
CLASSIFICATION_RECALL	Vollständigkeit des Modells.
CLASSIFICATION_ROC_AUC	Fläche unter der Fehlerkurve.
CLASSIFICATION_TOP_K_ACCURACY	Häufigkeit, mit der das richtige Label an der Spitze von k vorhergesagten Labels erscheint.

ENUM_LOSS_FUNCTION

Enumeration für Verlustfunktionsberechnungen vector::Loss.

ID	Beschreibung
LOSS_MSE	Mittlerer quadratischer Fehler
LOSS_MAE	Mittlerer absoluter Fehler
LOSS_CCE	Kategoriale Kreuzentropie
LOSS_BCE	Binäre Kreuzentropie
LOSS_MAPE	Mittlerer absoluter Fehler in Prozent
LOSS_MSLE	Mittlerer logarithmischer Fehler
LOSS_KLD	Kullback-Leibler-Divergenz
LOSS_COSINE	Cosinus-Ähnlichkeit/Nähe

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
LOSS_POISSON	Poisson-Verlustfunktion
LOSS_HINGE	Würfelförmig-lineare Verlustfunktion (Hinge loss)
LOSS_SQ_HINGE	Quadratische stückweise lineare Verlustfunktion nach Hinge
LOSS_CAT_HINGE	Kategoriale stückweise lineare Verlustfunktion nach Hinge
LOSS_LOG_COSH	Der Logarithmus des hyperbolischen Kosinus
LOSS_HUBER	Huber-Verlustfunktion

ENUM_ACTIVATION_FUNCTION

Enumeration für den Vektor der Aktivierungsfunktion vector::Activation und für den Vektor der Ableitung der Aktivierungsfunktion vector::Derivative.

ID	Beschreibung
AF_ELU	Exponential-Lineareinheit
AF_EXP	Exponentiell
AF_GELU	Gaußscher Fehler lineare Einheit
AF_HARD_SIGMOID	Hartes Sigmoid
AF_LINEAR	Linear
AF_LRELU	Leaky gleichgerichtete Lineareinheit
AF_RELU	Abgeschnittene lineare ReLU
AF_SELU	Skalierte lineare Exponentialfunktion (Skalierte ELU)
AF_SIGMOID	Sigmoid
AF_SOFTMAX	Softmax
AF_SOFTPLUS	Softplus
AF_SOFTSIGN	Softsign
AF_SWISH	Swish-Funktion
AF_TANH	Hyperbolischer Tangens
AF_TRELU	Schwellenwertgesteuerte gleichgerichtete Lineareinheit

ENUM_SORT_MODE

Enumeration der Sortierarten für die Funktion Sort.

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
SORT_ASCENDING	Aufsteigend sortieren
SORT_DESCENDING	Absteigend sortieren

ENUM_MATRIX_AXIS

Enumeration zur Angabe der Achse in allen statistischen Funktionen für Matrizen.

ID	Beschreibung
AXIS_NONE	Die Achse ist nicht angegeben. Die Berechnung erfolgt über alle Matrixelemente, als ob es sich um einen Vektor handeln würde (siehe die Methode Flat).
AXIS_HORZ	Horizontale Achse
AXIS_VERT	Vertikale Achse

© 2000-2025, MetaQuotes Ltd.

Initialisierung

Es gibt mehrere Möglichkeiten, Matrizen und Vektoren zu deklarieren und zu initialisieren.

Funktion	Aktion
Assign	Kopiert eine Matrix, einen Vektor oder ein Array mit automatischer Typ-Umwandlung.
CopyIndicatorBuffer	Abrufen der Daten des spezifizierten Indikator-Puffers im angegebenen Umfang Menge in einen Vektor
CopyRates	Gibt die historischen Zeitreihendaten der Struktur MqlRates der angegebenen Symbolperiodenlänge in der angegebenen Menge in eine Matrix oder einen Vektor zurück.
CopyTicks	Weist die Ticks der Struktur MqlTick einer Matrix oder einem Vektor zu.
CopyTicksRange	Weist die Ticks des angegebenen Zeitintervalls aus der Struktur MqlTick einer Matrix oder einem Vektor zu.
Eye	Rückgabe einer Matrix mit Einsen auf der Diagonale und Nullen an anderer Stelle.
Identity	Erzeugt eine Identitätsmatrix mit der angegebenen Größe.
Ones	Erzeugen und Zurückgeben einer neuen Matrix aus Einsen.
Zeros	Erzeugt eine neue, mit Nullen gefüllte Matrix und gibt sie zurück.
Full	Erstellung und Rückgabe einer neuen Matrix, die mit dem angegebenen Wert gefüllt wurde.
Tri	Konstruktion einer Matrix mit Einsen auf und unterhalb der gegebenen Diagonalen und Nullen an anderen Stellen
Init	Initialisieren einer Matrix.
Fill	Füllen des angegebenen Wertes einer vorhandenen Matrix oder Vektors.
Random	Statische Funktion. Sie erstellt und gibt eine neue Matrix oder einen neuen Vektor mit Zufallswerten zurück. Zufallswerte werden gleichmäßig innerhalb des angegebenen Bereichs generiert.

Deklaration ohne Angabe der Größe (keine Speicherzuweisung für die Daten):

```
matrix          matrix_a;   // Matrix vom Typ double
matrix<double>  matrix_a1;  // Alternative, eine Matrix vom Typ double zu deklarieren
matrixf         matrix_a2;  // float-Matrix
matrix<float>   matrix_a3;  // float-Matrix
vector          vector_a;   // double-Vektor
vector<double>  vector_a1;
vectorf         vector_a2;  // float-Vektor
vector<float>   vector_a3;
```

© 2000-2025, MetaQuotes Ltd.

Deklaration mit der angegebenen Größe (mit Speicherzuweisung für die Daten, aber ohne jegliche Initialisierung):

```
  matrix         matrix_a(128,128);           // die Parameter können entweder Konstar
  matrix<double> matrix_a1(InpRows,InpCols);  // oder Variablen
  matrixf        matrix_a2(1,128);            // analog bei einem horizontalen Vektor
  matrix<float>  matrix_a3(InpRows,1);        // analog bei einem vertikalen Vektor
  vector         vector_a(256);
  vector<double> vector_a1(InpSize);
  vectorf        vector_a2(SomeFunc());       // Die Funktion SomeFunc gibt eine Zahl
  vector<float>  vector_a3(InpSize+16);       // Ausdruck kann als Parameter verwendet
```

Deklaration mit Initialisierung (Matrix- und Vektorgrößen werden durch die Initialisierungssequenz bestimmt):

```
  matrix         matrix_a={{0.1,0.2,0.3},{0.4,0.5,0.6}};
  matrix<double> matrix_a1=matrix_a;                      // es müssen Matrizen dessel
  matrixf        matrix_a2={{1,0,0},{0,1,0},{0,0,1}};
  matrix<float>  matrix_a3={{1,2},{3,4}};
  vector         vector_a={-5,-4,-3,-2,-1,0,1,2,3,4,5};
  vector<double> vector_a1={1,5,2.4,3.3};
  vectorf        vector_a2={0,1,2,3};
  vector<float>  vector_a3=vector_a2;                     // es müssen Vektoren dessel
```

Deklaration mit Initialisierung:

```
template<typename T>
void MatrixArange(matrix<T> &mat,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<mat.Rows(); i++)
     {
      for(ulong j=0; j<mat.Cols(); j++,value+=step)
         mat[i][j]=value;
     }
  }
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
...

  matrix  matrix_a(size_m,size_k,MatrixArange,-M_PI,0.1); // zunächst wird eine nicht
  matrixf matrix_a1(10,20,MatrixArange);                  // nach der Erstellung der M
```

© 2000-2025, MetaQuotes Ltd.

```
vector  vector_a(size,VectorArange,-10.0);              // nach der Erstellung der M
vectorf vector_a1(128,VectorArange);
```

Bitte beachten Sie, dass die Dimensionen der Matrix oder des Vektors geändert werden können, da der Speicher für die Daten immer dynamisch ist.

Statische Methoden

Statische Methoden zur Erzeugung von Matrizen und Vektoren der angegebenen Größe, die auf eine bestimmte Weise initialisiert werden:

```
matrix          matrix_a =matrix::Eye(4,5,1);
matrix<double> matrix_a1=matrix::Full(3,4,M_PI);
matrixf         matrix_a2=matrixf::Identity(5,5);
matrixf<float> matrix_a3=matrixf::Ones(5,5);
matrix          matrix_a4=matrix::Tri(4,5,-1);
vector          vector_a =vector::Ones(256);
vectorf         vector_a1=vector<float>::Zeros(16);
vector<float>   vector_a2=vectorf::Full(128,float_value);
```

Methoden zur Initialisierung bereits erstellter Matrizen und Vektoren:

```
matrix  matrix_a;
matrix_a.Init(size_m,size_k,MatrixArange,-M_PI,0.1);
matrixf matrix_a1(3,4);
matrix_a1.Init(10,20,MatrixArange);
vector  vector_a;
vector_a.Init(128,VectorArange);
vectorf vector_a1(10);
vector_a1.Init(vector_size,VectorArange,start_value,step);

matrix_a.Fill(double_value);
vector_a1.Fill(FLT_MIN);
matrix_a1.Identity();
```

© 2000-2025, MetaQuotes Ltd.

Assign

Kopiert eine Matrix, einen Vektor oder ein Array mit automatischer Typ-Umwandlung.

```
bool matrix::Assign(
  const matrix<T>  &mat     // kopierte Matrix
   );
bool matrix::Assign(
  const void       &array[] // kopiertes Array
   );
bool vector::Assign(
  const vector<T>  &vec     // kopierter Vektor
   );
bool vector::Assign(
  const void       &array[] // kopiertes Array
   );
```

Parameter

M, V oder Array

[in] Die Matrix, der Vektor oder das Array, aus dem die Werte kopiert werden.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls — false.

Hinweis

Im Gegensatz zu Copy erlaubt die Methode Assign auch das Kopieren von Arrays. In diesem Fall findet eine automatische Typ-Umwandlung statt, wobei die resultierende Matrix oder der Vektor an die Größe des kopierten Arrays angepasst wird.

Beispiel:

```
//--- Kopieren der Matrizen
  matrix a= {{2, 2}, {3, 3}, {4, 4}};
  matrix b=a+2;
  matrix c;
  Print("matrix a \n", a);
  Print("matrix b \n", b);
  c.Assign(b);
  Print("matrix c \n", a);

//--- Kopieren des Arrays in die Matrix
  matrix double_matrix=matrix::Full(2,10,3.14);
  Print("double_matrix before Assign() \n", double_matrix);
  int int_arr[5][5]= {{1, 2}, {3, 4}, {5, 6}};
  Print("int_arr: ");
  ArrayPrint(int_arr);
```

© 2000-2025, MetaQuotes Ltd.

```
double_matrix.Assign(int_arr);
Print("double_matrix after Assign(int_arr) \n", double_matrix);
/*
 matrix a
 [[2,2]
  [3,3]
  [4,4]]
 matrix b
 [[4,4]
  [5,5]
  [6,6]]
 matrix c
 [[2,2]
  [3,3]
  [4,4]]

 double_matrix before Assign()
 [[3.14,3.14,3.14,3.14,3.14,3.14,3.14,3.14,3.14,3.14]
  [3.14,3.14,3.14,3.14,3.14,3.14,3.14,3.14,3.14,3.14]]

 int_arr:
     [,0][,1][,2][,3][,4]
 [0,]   1   2   0   0   0
 [1,]   3   4   0   0   0
 [2,]   5   6   0   0   0
 [3,]   0   0   0   0   0
 [4,]   0   0   0   0   0

 double_matrix after Assign(int_arr)
 [[1,2,0,0,0]
  [3,4,0,0,0]
  [5,6,0,0,0]
  [0,0,0,0,0]
  [0,0,0,0,0]]

*/
```

Siehe auch

Copy

© 2000-2025, MetaQuotes Ltd.

CopyIndicatorBuffer

Abrufen der Daten des spezifizierten Indikator-Puffers im angegebenen Umfang Menge in einen Vektor.

Die Daten werden in den Vektor kopiert, wobei das älteste Element an den Anfang des für den Vektor zugewiesenen physischen Speichers gesetzt wird. Es gibt drei Funktionsmöglichkeiten.

Zugriff über die Anfangsposition und die Anzahl der gewünschten Elemente

```
bool  vector::CopyIndicatorBuffer(
   long      indicator_handle,     // Indikator-Handle
   ulong     buffer_index,         // Nummer des Indikatorpuffers
   ulong     start_pos,            // Startposition der Kopie
   ulong     count                 // Anzahl der zu kopierenden Elemente
   );
```

Zugriff über das Startdatum und die Anzahl der benötigten Elemente

```
bool  vector::CopyIndicatorBuffer(
   long      indicator_handle,     // Indikator-Handle
   ulong     buffer_index,         // Nummer des Indikatorpuffers
   datetime  start_time,           // Zeitpunkt, ab dem kopiert werden soll
   ulong     count                 // Anzahl der zu kopierenden Elemente
   );
```

Zugriff mit Anfangs- und Endzeitpunkt des gewünschten Zeitraums

```
bool  vector::CopyIndicatorBuffer(
   long      indicator_handle,     // Indikator-Handle
   ulong     buffer_index,         // Nummer des Indikatorpuffers
   datetime  start_time,           // Zeitpunkt, ab dem kopiert werden soll
   datetime  stop_time             // Zeitpunkt, bis zu dem soll kopiert werden
   );
```

Parameter

indicator_handle

[in] Das von der entsprechenden Indikatorfunktion erhaltene Indikator-Handle.

buffer_index

[in] Die Nummer des Indikatorpuffers.

start_pos

[in] Die Indexnummer des ersten zu kopierenden Elements.

count

[in] Die Anzahl der kopierten Elemente.

start_time

[in] Der Zeitpunkt des Balkens des ersten Elements.

stop_time

© 2000-2025, MetaQuotes Ltd.

[in] Der Zeitpunkt des letzten Elements.

Rückgabewert

Die Funktion liefert bei Erfolg 'true' sonst, wenn ein Fehler aufgetreten ist, 'false'.

Hinweis

Die Elemente der kopierten Daten (der Indikatorpuffer mit dem Index buffer_index) werden von der Gegenwart in die Vergangenheit heruntergezählt, sodass die Startposition 0 der aktuellen Balken (der Indikatorwert für den aktuellen Balken) bedeutet.

Wenn Sie eine unbekannte Datenmenge kopieren, sollten Sie einen Vektor deklarieren, ohne eine Größe anzugeben (ohne den Daten Speicher vorab zu festzulegen), da die Funktion CopyBuffer() versucht, die Größe des Empfangsvektors an die Menge der kopierenden Daten anzupassen.

Wenn ein teilweises Kopieren der Indikatorwerte erforderlich ist, sollten Sie einen Zwischenvektor verwenden, in den die gewünschte Menge kopiert wird. Aus diesem Zwischenvektor können Sie die erforderliche Anzahl von Werten einzeln an die erforderlichen Stellen des Empfangsvektors kopieren.

Wenn Sie eine vorher festgelegte Datenmenge kopieren, ist es empfehlenswert, einen Vektor vorab zu deklarieren und seine Größe anzugeben, um eine unnötige Neuzuweisung von Speicher zu vermeiden.

Wenn Daten von einem Indikator angefordert werden, gibt die Funktion sofort false zurück, wenn die angeforderten Zeitreihen noch nicht erstellt wurden oder vom Server heruntergeladen werden müssen, während sie mit dem Laden/Konstruieren beginnt.

Wenn Daten von einem EA oder einem Skript angefordert werden, wird das Herunterladen vom Server eingeleitet, wenn das Terminal die entsprechenden Daten nicht lokal hat, oder die Konstruktion der erforderlichen Zeitreihen beginnt, wenn die Daten aus der lokalen Historie konstruiert werden können, aber die erforderlichen Zeitrahmen noch nicht bereit sind. Die Funktion gibt die Menge zurück, die zum Zeitpunkt des Ablaufs der Zeitüberschreitung zur Verfügung steht.

Siehe auch

CopyBuffer

© 2000-2025, MetaQuotes Ltd.

CopyRates

Gibt die historischen Zeitreihendaten der Struktur MqlRates der angegebenen Symbolperiodenlänge in der angegebenen Menge in eine Matrix oder einen Vektor zurück. Die Elemente werden von der Gegenwart in die Vergangenheit gereiht und das heißt, dass die Startposition 0 gleich der aktuellen Bar ist.

Die Daten werden so kopiert, dass das älteste Element an den Anfang der Matrix oder des Vektors gesetzt wird. Es gibt drei Funktionsmöglichkeiten.

Zugriff über die Anfangsposition und die Anzahl der gewünschten Elemente

```
bool  matrix::CopyRates(
   string           symbol,       // Symbolname
   ENUM_TIMEFRAMES  period,       // Zeitrahmen
   ulong            rates_mask,   // Flag-Kombination zur Bestimmung der gewünschten Z
   ulong            start,        // Index des Anfangsbalkens ab dem kopiert wird
   ulong            count         // wie viele kopieren
   );
```

Zugriff mit Anfangsdatum und der Anzahl der benötigten Elemente

```
bool  matrix::CopyRates(
   string           symbol,       // Symbolname
   ENUM_TIMEFRAMES  period,       // Zeitrahmen
   ulong            rates_mask,   // Flag-Kombination zur Bestimmung der gewünschten Z
   datetime         from,         // Anfangszeitpunkt
   ulong            count         // wie viele kopieren
   );
```

Zugriff mit Anfangs- und Endzeitpunkt des gewünschten Zeitraums

```
bool  matrix::CopyRates(
   string           symbol,       // Symbolname
   ENUM_TIMEFRAMES  period,       // Zeitrahmen
   ulong            rates_mask,   // Flag-Kombination zur Bestimmung der gewünschten Z
   datetime         from,         // Anfangszeitpunkt
   datetime         to            // Endzeitpunkt
   );
```

Vektor-Methoden

Zugriff über die Anfangsposition und die Anzahl der gewünschten Elemente

```
bool  vector::CopyRates(
   string           symbol,       // Symbolname
   ENUM_TIMEFRAMES  period,       // Zeitrahmen
   ulong            rates_mask,   // Flag-Kombination zur Bestimmung der gewünschten Z
   ulong            start,        // Index des Anfangsbalkens ab dem kopiert wird
   ulong            count         // wie viele kopieren
   );
```

© 2000-2025, MetaQuotes Ltd.

Zugriff mit Anfangsdatum und der Anzahl der benötigten Elemente

```
bool  vector::CopyRates(
   string           symbol,       // Symbolname
   ENUM_TIMEFRAMES  period,       // Zeitrahmen
   ulong            rates_mask,   // Flag-Kombination zur Bestimmung der gewünschten Z
   datetime         from,         // Anfangszeitpunkt
   ulong            count         // wie viele kopieren
   );
```

Zugriff mit Anfangs- und Endzeitpunkt des gewünschten Zeitraums

```
bool  vector::CopyRates(
   string           symbol,       // Symbolname
   ENUM_TIMEFRAMES  period,       // Zeitrahmen
   ulong            rates_mask,   // Flag-Kombination zur Bestimmung der gewünschten Z
   datetime         from,         // Anfangszeitpunkt
   datetime         to            // Endzeitpunkt
   );
```

Parameter

symbol

[in] Symbolname

period

[in] Zeitrahmen.

rates_mask

[in] Kombination der Flags aus der Enumeration ENUM_COPY_RATES, die den Typ der angeforderten Zeitreihe angeben. Beim Kopieren in einen Vektor kann nur ein Wert aus der Enumeration ENUM_COPY_RATES angegeben werden, andernfalls tritt ein Fehler auf.<Segment 1412>

Rstart

[in] Index des ersten zu kopierenden Elements.

count

[in] Anzahl der zu kopierenden Elemente.

from

[in] Zeit des Balkens des ersten Elements.

ändert auf

[in] Zeit des Balkens des letzten Elements.

Rückgabewert

Rückgabe von true bei Erfolg, sonst false im Falle eines Fehlers.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Wenn das Intervall der angeforderten Daten komplett außerhalb der auf dem Server verfügbaren Daten liegt, dann liefert die Funktion false. Wenn die Daten außerhalb von TERMINAL_MAXBARS (maximale Anzahl von Balken im Chart) angefordert werden, gibt die Funktion ebenfalls falsezurück.

Wenn Daten von einem EA oder einem Skript angefordert werden, wird ein Download vom Server initiiert, wenn das Terminal die entsprechenden Daten nicht lokal hat, oder die Konstruktion der notwendigen Zeitreihen beginnt, wenn die Daten aus der lokalen Historie konstruiert werden können, aber noch nicht bereit sind. Die Funktion gibt die Menge zurück, die bis zum Ablauf der Zeitüberschreitung zur Verfügung stehen wird, der Download der Historie wird jedoch fortgesetzt, und die Funktion gibt bei der nächsten ähnlichen Anforderung weitere Daten zurück.

Bei der Abfrage von Daten nach dem Startdatum und der Anzahl der erforderlichen Elemente werden nur Daten zurückgegeben, deren Datum kleiner (vor) oder gleich dem angegebenen Datum ist. Das Intervall wird auf eine Sekunde genau festgelegt und berücksichtigt. Mit anderen Worten: Das Eröffnungsdatum eines jeden Balkens, für den der Wert zurückgegeben wird (Volumen, Spread, Open, High, Low, Close oder Time), ist immer gleich oder kleiner als das angegebene Datum.

Wenn Daten in einem bestimmten Datumsbereich angefordert werden, werden nur Daten zurückgegeben, die in das angeforderte Intervall fallen. Das Intervall wird auf eine Sekunde genau festgelegt und berücksichtigt. Mit anderen Worten, die Eröffnungszeit eines jeden Balkens, für den der Wert zurückgegeben wird (Volumen, Spread, Indikatorpufferwert, Open, High, Low, Close oder Time), liegt immer im angeforderten Intervall.

Wenn zum Beispiel der aktuelle Wochentag Samstag ist, gibt die Funktion 0 zurück, wenn versucht wird, die Daten des wöchentlichen Zeitrahmens zu kopieren und *start_time=Last_Tuesday* und *stop_time=Last_Friday* einstellt wurde, weil die Eröffnungszeit auf dem wöchentlichen Zeitrahmen immer auf den Sonntag fällt, aber kein einziger Balken einer Woche in den angegebenen Bereich fällt.

Wenn Sie einen Wert für den aktuellen, noch unvollständigen Balken benötigen, können Sie die erste Aufrufform mit der Angabe von *start_pos*=0 und *count*=1 verwenden.

ENUM_COPY_RATES

Die Enumeration ENUM_COPY_RATES enthält die Flags zur Angabe des Datentyps, der an die Matrix oder das Array übergeben werden soll. Die Flag-Kombination ermöglicht es, mehrere Zeitreihen aus der Historie in einer Anfrage zu erhalten. Die Reihenfolge der Zeilen in der Matrix entspricht der Reihenfolge der Werte in der Enumeration ENUM_COPY_RATES. Mit anderen Worten: Die Zeile mit den hohen Daten wird in der Matrix immer höher stehen als die Zeile mit den niedrigen Daten.

ID	Wert	Beschreibung
COPY_RATES_OPEN	1	Zeitreihe der Eröffnungspreise
COPY_RATES_HIGH	2	Zeitreihe der Hochs
COPY_RATES_LOW	4	Zeitreihe der Tiefs
COPY_RATES_CLOSE	8	Zeitreihe der Schlusskurse
COPY_RATES_TIME	16	Zeitreihe (Bar open time)

© 2000-2025, MetaQuotes Ltd.

ID	Wert	Beschreibung
		Die Ermittlung der Zeit eines Vektors vom Typ float und der Matrix (vectord und matrixf) führt zu Verlusten von ~100 Sekunden, da bei float die Genauigkeit stark eingeschränkt ist und ganze Zahlen größer als 1<<24 in float nicht genau dargestellt werden können.
COPY_RATES_VOLUME_TICK	32	Tick Volumina
COPY_RATES_VOLUME_REAL	64	Handelsvolumina
COPY_RATES_SPREAD	128	Spreads
Kombination		
COPY_RATES_OHLC	15	Zeitreihen von Open, High, Low, Close
COPY_RATES_OHLCT	31	Zeitreihen von Open, High, Low, Close, Time
Anordnung der Daten		
COPY_RATES_VERTICAL	32768	Die Reihen werden entlang der vertikalen Achse in die Matrix kopiert. Die erhaltenen Reihenwerte werden in der Matrix vertikal angeordnet, d. h. die ältesten Daten stehen in der ersten Zeile, während die jüngsten Daten in der letzten Matrixzeile stehen. Beim standardmäßigen Kopieren werden die Datenreihen in eine Matrix entlang der horizontalen Achse eingefügt. Dies Flag ist nur beim Kopieren in eine Matrix anwendbar.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- Abruf der Kurse in die Matrix
  matrix matrix_rates;
  if(matrix_rates.CopyRates(Symbol(), PERIOD_CURRENT, COPY_RATES_OHLCT, 1, 10))
    Print("matrix rates: \n", matrix_rates);
  else
    Print("matrix_rates.CopyRates failed. Fehlernummer ", GetLastError());
//--- check
  MqlRates mql_rates[];
```

© 2000-2025, MetaQuotes Ltd.

```
  if(CopyRates(Symbol(), PERIOD_CURRENT, 1, 10, mql_rates)>0)
   {
    Print("mql_rates array:");
    ArrayPrint(mql_rates);
   }
  else
    Print("CopyRates(Symbol(), PERIOD_CURRENT,1, 10, mql_rates). Fehlernummer ", GetLa
//--- Abruf der Kurse in den Vektor = ungültiger Aufruf
  vector vector_rates;
  if(vector_rates.CopyRates(Symbol(), PERIOD_CURRENT, COPY_RATES_OHLC, 1, 15))
    Print("vector_rates COPY_RATES_OHLC: \n", vector_rates);
  else
    Print("vector_rates.CopyRates COPY_RATES_OHLC failed. Fehlernummer ", GetLastError
//--- Abruf der Schlusskurse in den Vektor
  if(vector_rates.CopyRates(Symbol(), PERIOD_CURRENT, COPY_RATES_CLOSE, 1, 15))
    Print("vector_rates COPY_RATES_CLOSE: \n", vector_rates);
  else
    Print("vector_rates.CopyRates failed. Fehlernummer ", GetLastError());
 };
/*
   matrix rates:
   [[0.99686,0.99638,0.99588,0.99441,0.99464,0.99594,0.99698,0.99758,0.99581,0.995280
    [0.99708,0.99643,0.99591,0.9955000000000001,0.99652,0.99795,0.99865,0.99764,0.996
    [0.9961100000000001,0.99491,0.99426,0.99441,0.99448,0.99494,0.9964499999999999,0.9
    [0.99641,0.99588,0.99441,0.99464,0.99594,0.99697,0.99758,0.99581,0.995280000000000
    [1662436800,1662440400,1662444000,1662447600,1662451200,1662454800,1662458400,1662
   mql_rates array:
                    [time]   [open]   [high]    [low] [close] [tick_volume] [spread] [rea
   [0] 2022.09.06 04:00:00 0.99686 0.99708 0.99611 0.99641           4463        0
   [1] 2022.09.06 05:00:00 0.99638 0.99643 0.99491 0.99588           4519        0
   [2] 2022.09.06 06:00:00 0.99588 0.99591 0.99426 0.99441           3060        0
   [3] 2022.09.06 07:00:00 0.99441 0.99550 0.99441 0.99464           3867        0
   [4] 2022.09.06 08:00:00 0.99464 0.99652 0.99448 0.99594           5280        0
   [5] 2022.09.06 09:00:00 0.99594 0.99795 0.99494 0.99697           7227        0
   [6] 2022.09.06 10:00:00 0.99698 0.99865 0.99645 0.99758          10130        0
   [7] 2022.09.06 11:00:00 0.99758 0.99764 0.99472 0.99581           7012        0
   [8] 2022.09.06 12:00:00 0.99581 0.99604 0.99360 0.99528           6166        0
   [9] 2022.09.06 13:00:00 0.99528 0.99570 0.99220 0.99259           6950        0
   vector_rates.CopyRates COPY_RATES_OHLC failed. Error 4003
   vector_rates COPY_RATES_CLOSE:
   [0.9931,0.99293,0.99417,0.99504,0.9968399999999999,0.99641,0.99588,0.99441,0.99464,
*/
```

Siehe auch

Zugang zu Zeitreihen und Daten der Indikatoren, CopyRates

© 2000-2025, MetaQuotes Ltd.

CopyTicks

Abruf der Ticks aus der Struktur MqlTick Struktur in eine Matrix oder einen Vektor. Die Elemente werden von der Vergangenheit bis zur Gegenwart gezählt, was bedeutet, dass der Tick mit dem Index 0 der älteste ist. Um einen Tick zu analysieren, prüft man das Feld *Flags*, das anzeigt, was genau sich in dem Tick geändert hat.

```
bool  matrix::CopyTicks(
   string           symbol,              // Symbolname
   ulong            flags,               // Flag, das den Typ der verlangten Ticks s
   ulong            from_msc,            // Zeitpunkt, ab dem die Ticks abgerufen we
   ulong            count                // Anzahl der Ticks, die abgerufen werden
   );
```

Vektor-Methode

```
bool  vector::CopyTicks(
   string           symbol,              // Symbolname
   ulong            flags,               // Flag, das den Typ der verlangten Ticks s
   ulong            from_msc,            // Zeitpunkt, ab dem die Ticks abgerufen we
   ulong            count                // Anzahl der Ticks, die abgerufen werden
   );
```

Parameter

symbol

[in] Symbolname

flags

[in] Eine Kombination von Flags aus der Enumeration ENUM_COPY_TICKS, die den Inhalt der angeforderten Daten angibt. Beim Kopieren in einen Vektor kann nur ein Wert aus der Enumeration ENUM_COPY_TICKS angegeben werden, sonst tritt ein Fehler auf.

from_msc

[in] Zeitpunkt, ab dem Ticks angefordert werden. Die Zeit wird in Millisekunden seit dem 01/01/1970 angegeben. Ist from_msc=0, wird die letzte Anzahl von Ticks zurückgegeben, die 'count' entspricht.

count

[in] Die Anzahl der angeforderten Ticks. Wenn die Parameter 'from_msc' und 'count' nicht angegeben werden, werden alle verfügbaren Ticks, jedoch nicht mehr als 2000, abgerufen.

Rückgabewert

Gibt bei Erfolg true zurück oder false, wenn ein Fehler aufgetreten ist.

Hinweis

Der erste Aufruf von CopyTicks() leitet die Synchronisation der auf der Festplatte gespeicherten Tick-Datenbank des betreffenden Symbols ein. Wenn die lokale Datenbasis nicht alle angeforderten Ticks enthält, werden die fehlenden Ticks automatisch vom Handelsserver heruntergeladen. Die in CopyTicks() angegebenen Ticks von ***from_msc*** bis zum aktuellen Zeitpunkt werden synchronisiert.

© 2000-2025, MetaQuotes Ltd.

Danach werden alle Ticks, die für dieses Symbol eintreffen, der Tick-Datenbasis hinzugefügt, sodass sie im synchronisierten Zustand bleibt.

Wenn die Parameter *from_msc* und *count* nicht angegeben werden, werden alle verfügbaren Ticks, aber nicht mehr als 2000, in die Matrix bzw. den Vektor geschrieben.

In Indikatoren liefert die Methode CopyTicks() sofort das Ergebnis: Wenn sie von einem Indikator aufgerufen wird, liefert CopyTick() sofort alle verfügbaren Ticks eines Symbols und startet eine Synchronisation der Tick-Datenbank, wenn die verfügbaren Daten nicht ausreichen. Alle Indikatoren für dasselbe Symbol arbeiten in einem gemeinsamen Thread, sodass der Indikator nicht auf das Ende der Synchronisierung warten kann. Nach der Synchronisierung gibt CopyTicks() beim nächsten Aufruf alle angeforderten Ticks zurück. In Indikatoren wird die Funktion OnCalculate() nach dem Eintreffen jedes Ticks aufgerufen.

In Expert Advisors und Skripten kann CopyTicks() bis zu 45 Sekunden auf das Ergebnis warten: Im Gegensatz zu Indikatoren arbeitet jeder Expert Advisor oder jedes Skript in einem eigenen Thread und kann daher bis zu 45 Sekunden auf den Abschluss der Synchronisation warten. Wenn die erforderliche Anzahl von Ticks in dieser Zeit nicht synchronisiert werden kann, gibt CopyTicks() die verfügbaren Ticks bis zum Timeout zurück und setzt dann die Synchronisierung fort. OnTick() in Expert Advisors reagiert nicht auf jeden Tick, sondern benachrichtigt den Expert Advisor nur über Veränderungen auf dem Markt. Dabei kann es sich um eine Reihe von Änderungen handeln: das Terminal kann gleichzeitig mehrere Ticks empfangen, während OnTick() nur einmal aufgerufen wird, um den Expert Advisor über den neuesten Marktzustand zu informieren.

Rate der Datenrückgabe: Das Terminal speichert 4096 letzte Ticks für jedes Instrument im Fast-Access-Cache (65536 Ticks für Symbole mit laufender Markttiefe). Anfragen, die diese Daten betreffen, werden am schnellsten ausgeführt. Wenn die angeforderten Ticks für die aktuelle Handelssitzung außerhalb des Cache liegen, ruft CopyTicks() die im Terminalspeicher gespeicherten Ticks auf. Diese Anfragen benötigen mehr Zeit zur Ausführung. Am langsamsten sind die Anfragen, die Ticks für andere Tage anfordern, da die Daten in diesem Fall vom Laufwerk gelesen werden.

ENUM_COPY_TICKS

Die Enumeration ENUM_COPY_TICKS enthält die Flags zur Angabe des Datentyps, der an die Matrix oder das Array übergeben werden soll. Die Flag-Kombination ermöglicht es, mehrere Zeitreihen aus der Historie in einer Anfrage zu erhalten. Die Reihenfolge der Zeilen in der Matrix entspricht der Reihenfolge der Werte in der Enumeration ENUM_COPY_TICKS. Mit anderen Worten: Die Zeile mit den hohen Daten wird in der Matrix immer höher stehen als die Zeile mit den niedrigen Daten.

ID	Wert	Beschreibung
COPY_TICKS_INFO	1	Ticks mit Bid- und/oder Ask-Preisänderungen
COPY_TICKS_TRADE	2	Ticks mit letzter Preisänderung und/oder Volumensänderung
COPY_TICKS_ALL	3	Alle Ticks
COPY_TICKS_TIME_MS	1<<8	Tickzeit in Millisekunden
COPY_TICKS_BID	1<<9	Geldkurs (Bid)
COPY_TICKS_ASK	1<<10	Briefkurs (Ask)
COPY_TICKS_LAST	1<<11	Last-Preis (Preis des letzten Deals)

© 2000-2025, MetaQuotes Ltd.

ID	Wert	Beschreibung
COPY_TICKS_VOLUME	1<<12	Volumen des Last-Preises
COPY_TICKS_FLAGS	1<<13	Tick-Flags
Anordnung der Daten		
COPY_TICKS_VERTICAL	32768	Ticks werden entlang der vertikalen Achse in die Matrix kopiert. Die erhaltenen Ticks werden in der Matrix vertikal angeordnet, d. h. die ältesten Ticks befinden sich in der ersten Zeile, während die jüngsten Ticks in der letzten Matrixzeile stehen. Beim standardmäßigen Kopieren werden die Ticks in eine Matrix entlang der horizontalen Achse eingefügt. Dies Flag ist nur beim Kopieren in eine Matrix anwendbar.

Analysieren Sie die Tick-Flags, um herauszufinden, welche Daten sich geändert haben:

- TICK_FLAG_BID — der Tick hat den Geldkurs (Bid) geändert.
- TICK_FLAG_ASK — das Häkchen hat den Briefkurs (Ask) verändert.
- TICK_FLAG_LAST — der Tick hat den Preis des letzten Deals geändert.
- TICK_FLAG_VOLUME — der Tick hat das Volumen verändert.
- TICK_FLAG_BUY — der Tick ist das Ergebnis eines Kaufes.
- TICK_FLAG_SELL — der Tick ist das Ergebnis eines Verkaufs.

Siehe auch

Zugang zu Zeitreihen und Daten der Indikatoren, CopyRates

© 2000-2025, MetaQuotes Ltd.

CopyTicksRange

Weist die Ticks aus der Struktur MqlTick einer Matrix oder einem Vektor innerhalb der angegebenen Zeitspanne zu. Die Elemente werden von der Vergangenheit bis zur Gegenwart gezählt, d.h. der Tick mit Index 0 ist der älteste. Um einen Tick zu analysieren, prüft man das Feld *Flags*, das anzeigt, was genau sich in dem Tick geändert hat.

```
bool  matrix::CopyTicksRange(
   string           symbol,              // Symbolname
   ulong            flags,               // Flag, das den Typ der verlangten Ticks s
   ulong            from_msc,            // Zeitpunkt, ab dem die Ticks abgerufen we
   ulong            to_msc               // Zeitpunkt, bis zu der Ticks angefordert
   );
```

Vektor-Methode

```
bool  vector::CopyTicksRange(
   string           symbol,              // Symbolname
   ulong            flags,               // Flag, das den Typ der verlangten Ticks s
   ulong            from_msc,            // Zeitpunkt, ab dem die Ticks abgerufen we
   ulong            to_msc               // Zeitpunkt, bis zu der Ticks angefordert
   );
```

Parameter

symbol

[in] Symbolname

flags

[in] Eine Kombination von Flags aus der Enumeration ENUM_COPY_TICKS, die den Inhalt der angeforderten Daten angibt. Beim Kopieren in einen Vektor kann nur ein Wert aus der Enumeration ENUM_COPY_TICKS angegeben werden, sonst tritt ein Fehler auf.

from_msc

[in] Zeitpunkt, ab dem Ticks angefordert werden. Die Zeit wird in Millisekunden seit dem 01/01/1970 angegeben. Wird der Parameter 'from_msc' nicht angegeben, werden die Ticks vom Beginn der Historie an zurückgegeben. Zurückgegeben werden die Ticks mit einer Zeit >= from_msc.

to_msc

[in] Zeit, bis zu der Ticks angefordert werden. Die Zeit wird in Millisekunden seit dem 01/01/1970 angegeben. Zurückgegeben werden die Ticks mit einer Zeit <= to_msc.<= to_msc Wenn der Parameter to_msc nicht angegeben wird, werden alle Ticks bis zum Ende der Historie zurückgegeben.

Rückgabewert

Gibt bei Erfolg true zurück oder false, wenn ein Fehler aufgetreten ist. GetLastError() kann die folgenden Fehler zurückgeben:

- ERR_HISTORY_TIMEOUT — Timeout für die Tick-Synchronisation ist abgelaufen, die Funktion hat alles zurückgegeben, was sie hatte.

© 2000-2025, MetaQuotes Ltd.

- ERR_HISTORY_SMALL_BUFFER — der statische Puffer ist zu klein. Nur die Menge, die das Array speichern kann, wurde zurückgegeben.
- ERR_NOT_ENOUGH_MEMORY — nicht genug Speicher, um historische Daten aus dem angegebenen Bereich in ein dynamisches Tick-Array zu empfangen. Es konnte nicht genügend Speicher für das Tick-Array zugewiesen werden.

Analysieren Sie die Tick-Flags, um herauszufinden, welche Daten sich geändert haben:

- TICK_FLAG_BID — der Tick hat den Geldkurs (Bid) geändert.
- TICK_FLAG_ASK — das Häkchen hat den Briefkurs (Ask) verändert.
- TICK_FLAG_LAST — der Tick hat den Preis des letzten Deals geändert.
- TICK_FLAG_VOLUME — der Tick hat das Volumen verändert.
- TICK_FLAG_BUY — der Tick ist das Ergebnis eines Kaufes.
- TICK_FLAG_SELL — der Tick ist das Ergebnis eines Verkaufs.

Hinweis

Die Methode CopyTicksRange() wird verwendet, um Ticks aus genau dem angegebenen Bereich abzufragen. Zum Beispiel Ticks für einen bestimmten Tag in der Geschichte. CopyTicks() Erlaubt auch die Angabe nur des Startdatums, z.B. um alle Ticks vom Monatsanfang bis jetzt zu erhalten.

Siehe auch

Zugang zu Zeitreihen und Daten der Indikatoren, CopyRates

© 2000-2025, MetaQuotes Ltd.

Eye

Eine statische Funktion. Konstruiert eine Matrix einer bestimmten Größe mit Einsen auf der Hauptdiagonalen und Nullen an anderen Stellen. Gibt eine Matrix mit Einsen auf der Diagonale und Nullen an anderen Stellen zurück.

```
static matrix matrix::Eye(
  const ulong  rows,         // Zeilenanzahl
  const ulong  cols,         // Spaltenanzahl
  const int    ndiag=0       // Index der Diagonalen
   );
```

Parameter

rows

[in] Anzahl der Zeilen in der Ausgabe.

cols

[in] Anzahl der Spalten in der Ausgabe.

ndiag=0[in] Index der Diagonale: 0 (Standardwert) bezieht sich auf die Hauptdiagonale, ein positiver Wert bezieht sich auf eine obere Diagonale und ein negativer Wert auf eine untere Diagonale.RückgabewertEine Matrix, bei der alle Elemente gleich Null sind, mit Ausnahme der k-ten Diagonale, deren Werte gleich Eins sind.

[in] Index der Diagonale: 0 (Standardwert) bezieht sich auf die Hauptdiagonale, ein positiver Wert bezieht sich auf eine obere Diagonale und ein negativer Wert auf eine untere Diagonale.

Rückgabewert

Eine Matrix, in der alle Elemente gleich Null sind, mit Ausnahme der k-ten Diagonale, deren Werte gleich Eins sind.

MQL5 Beispiel:

```
  matrix eye=matrix::Eye(3, 3);
  Print("eye = \n", eye);

  eye=matrix::Eye(4, 4,1);
  Print("eye = \n", eye);
  /*
   eye =
   [[1,0,0]
    [0,1,0]
    [0,0,1]]
   eye =
   [[0,1,0,0]
    [0,0,1,0]
    [0,0,0,1]
    [0,0,0,0]]
```

© 2000-2025, MetaQuotes Ltd.

```
   */
```

Python Beispiel:

```
 np.eye(3, dtype=int)
array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])

 np.eye(4, k=1)
array([[0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.],
       [0., 0., 0., 0.]])
```

© 2000-2025, MetaQuotes Ltd.

Identity

Es ist eine statische Funktion, die eine Identitätsmatrix mit der angegebenen Größe (nicht notwendigerweise quadratisch) erzeugt. Eine Identitätsmatrix enthält Einsen auf der Hauptdiagonale und Nullen an den anderen Stellen. Die Hauptdiagonale besteht aus den Matrixelementen mit gleichen Zeilen- und Spaltenindizes, z. B. [0,0],[1,1],[2,2] usw. Es wird eine neue Identitätsmatrix erzeugt.

Es gibt auch die Methode Identity, die eine bereits vorhandene Matrix in eine Identitätsmatrix umwandelt.

```
static matrix matrix::Identity(
  const ulong  rows,         // Zeilenanzahl
  const ulong  cols,         // Spaltenanzahl
   );

void matrix::Identity();
```

Parameter

rows

[in] Zeilenanzahl (und Spalten) in einer n x n Matrix.

Rückgabewert

Rückgabe der Identitätsmatrix. Die Identitätsmatrix ist eine quadratische Matrix mit Einsen auf der Hauptdiagonalen.

MQL5 Beispiel:

```
  matrix identity=matrix::Identity(3,3);
  Print("identity = \n", identity);
/*
   identity =
   [[1,0,0]
    [0,1,0]
    [0,0,1]]
*/
  matrix identity2(3,5);
  identity2.Identity();
  Print("identity2 = \n", identity2);
/*
   identity2 =
   [[1,0,0,0,0]
    [0,1,0,0,0]
    [0,0,1,0,0]]
*/
```

Python Beispiel:

```
np.identity(3)
```

© 2000-2025, MetaQuotes Ltd.

```
array([[1.,  0.,  0.],
       [0.,  1.,  0.],
       [0.,  0.,  1.]])
```

© 2000-2025, MetaQuotes Ltd.

Ones

Die statische Funktion erstellt und retourniert eine neue Matrix, die mit Einsen gefüllt wurde.

```
static matrix matrix::Ones(
  const ulong  rows,      // Zeilenanzahl
  const ulong  cols       // Spaltenanzahl
   );

static vector vector::Ones(
  const ulong  size,      // Vektorgröße
   );
```

Parameter

rows

[in] Zeilenanzahl.

cols

[in] Spaltenanzahl.

Rückgabewert

Eine neue Matrix mit gegebenen Zeilen und Spalten, gefüllt mit Einsen.

MQL5 Beispiel:

```
  matrix ones=matrix::Ones(4, 4);
  Print("ones = \n", ones);
/*
ones =
   [[1,1,1,1]
    [1,1,1,1]
    [1,1,1,1]
    [1,1,1,1]]
*/
```

Python Beispiel:

```
np.ones((4, 1))
array([[1.],
       [1.]])
```

© 2000-2025, MetaQuotes Ltd.

Zeros

Die statische Funktion erstellt und retourniert eine neue Matrix, die mit Nullen gefüllt wurde.

```
static matrix matrix::Zeros(
  const ulong  rows,      // Zeilenanzahl
  const ulong  cols       // Spaltenanzahl
   );

static vector vector::Zeros(
  const ulong  size,      // Vektorgröße
   );
```

Parameter

rows

[in] Zeilenanzahl.

cols

[in] Spaltenanzahl.

Rückgabewert

Eine neue Matrix mit angegebenen Zeilen- und Spaltenanzahl, die mit Nullen gefüllt ist.

MQL5 Beispiel:

```
  matrix zeros=matrix::Zeros(3, 4);
  Print("zeros = \n", zeros);
/*
zeros =
   [[0,0,0,0]
    [0,0,0,0]
    [0,0,0,0]]
*/
```

Python Beispiel:

```
np.zeros((2, 1))
array([[ 0.],
       [ 0.]])
```

© 2000-2025, MetaQuotes Ltd.

Full

Die statische Funktion erstellt und retourniert eine neue Matrix, die mit dem angegebenen Wert gefüllt wurde.

```
static matrix matrix::Full(
  const ulong   rows,       // Zeilenanzahl
  const ulong   cols,       // Spaltenanzahl
  const double  value       // Wert für das Ausfüllen
   );

static vector vector::Full(
  const ulong   size,       // Vektorgröße
  const double  value       // Wert für das Ausfüllen
   );
```

Parameter

rows

[in] Zeilenanzahl.

cols

[in] Spaltenanzahl.

value

[in] Wert, der allen Matrixelementen zugewiesen werden soll.

Rückgabewert

Gibt eine neue Matrix mit den angegebenen Zeilen und Spalten zurück, die mit dem angegebenen Wert gefüllt ist.

MQL5 Beispiel:

```
  matrix full=matrix::Full(3,4,10);
  Print("full = \n", full);
/*
full =
   [[10,10,10,10]
    [10,10,10,10]
    [10,10,10,10]]
*/
```

Beispiel

```
np.full((2, 2), 10)
array([[10, 10],
       [10, 10]])
```

© 2000-2025, MetaQuotes Ltd.

Tri

Dies ist eine statische Funktion, die eine Matrix mit Einsen auf und unterhalb der gegebenen Diagonalen und Nullen an anderen Stellen erstellt.

```
static matrix matrix::Tri(
  const ulong rows,         // Zeilenanzahl
  const ulong cols,         // Spaltenanzahl
  const int   ndiag=0       // Zahl der Diagonale
   );
```

Parameter

rows

[in] Zeilenanzahl im Array.

cols

[in] Spaltenanzahl im Array.

ndiag=0[in] Index der Diagonale: 0 (Standardwert) bezieht sich auf die Hauptdiagonale, ein positiver Wert bezieht sich auf eine obere Diagonale und ein negativer Wert auf eine untere Diagonale.RückgabewertEine Matrix, bei der alle Elemente gleich Null sind, mit Ausnahme der k-ten Diagonale, deren Werte gleich Eins sind.

[in] Die Sub-Diagonale, auf der und unter der das Array gefüllt wird. k = 0 ist die Hauptdiagonale, während k < 0 darunter liegt und k > 0 darüber. Standardwert ist 0.

Rückgabewert

Array, dessen unteres Dreieck mit Einsen gefüllt ist, während an anderer Stelle Null steht.

MQL5 Beispiel:

```
   matrix matrix_a=matrix::Tri(3,4,1);
   Print("Tri(3,4,1)\n",matrix_a);
   matrix_a=matrix::Tri(4,3,-1);
   Print("Tri(4,3,-1)\n",matrix_a);

/*
   Tri(3,4,1)
   [[1,1,0,0]
    [1,1,1,0]
    [1,1,1,1]]
   Tri(4,3,-1)
   [[0,0,0]
    [1,0,0]
    [1,1,0]
    [1,1,1]]
*/
```

Beispiel

```
np.tri(3, 5, 2, dtype=int)
```

© 2000-2025, MetaQuotes Ltd.

```
array([[1, 1, 1, 0, 0],
       [1, 1, 1, 1, 0],
       [1, 1, 1, 1, 1]])
```

© 2000-2025, MetaQuotes Ltd.

Init

Initialisieren einer Matrix.

```
void matrix::Init(
  const ulong  rows,            // Zeilenanzahl
  const ulong  cols,            // Spaltenanzahl
  func_name    init_func=NULL,  // init-Funktion in einem Bereich oder einer statische
   ...         parameters
   );

void vector::Init(
  const ulong  size,            // Vektorgröße
  func_name    init_func=NULL,  // init-Funktion in einem Bereich oder einer statische
   ...         parameters
   );
```

Parameter

rows

[in] Zeilenanzahl.

cols

[in] Spaltenanzahl.

func_name

[in] Initialisierungsfunktion.

...

[in] Parameter der Initialisierungsfunktion.

Rückgabewert

Kein Rückgabewert.

Beispiel

```
template<typename T>
void MatrixArange(matrix<T> &mat,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<mat.Rows(); i++)
     {
      for(ulong j=0; j<mat.Cols(); j++,value+=step)
         mat[i][j]=value;
     }
  }
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
  int size_m=3, size_k=4;
  matrix  m(size_m,size_k,MatrixArange,-2.,0.1); // zunächst wird eine uninitialisiert
  Print("matrix m \n",m);                        // dann wird die Funktion MatrixArang
  matrixf m_float(5,5,MatrixArange,-2.f,0.1f);   // nachdem eine Matrix vom Typ float
  Print("matrix m_float \n",m_float);
  vector  v(size_k,VectorArange,-10.0);          // nachdem ein Vektor erstellt wurde,
  Print("vector v \n",v);
  /*
   matrix m
   [[-2,-1.9,-1.8,-1.7]
    [-1.6,-1.5,-1.399999999999999,-1.299999999999999]
    [-1.199999999999999,-1.099999999999999,-0.9999999999999992,-0.8999999999999992]]
   matrix m_float
   [[-2,-1.9,-1.8,-1.6999999,-1.5999999]
    [-1.4999999,-1.3999999,-1.2999998,-1.1999998,-1.0999998]
    [-0.99999976,-0.89999974,-0.79999971,-0.69999969,-0.59999967]
    [-0.49999967,-0.39999968,-0.29999968,-0.19999969,-0.099999689]
    [3.1292439e-07,0.10000031,0.20000032,0.30000031,0.4000003]]
   vector v
   [-10,-9,-8,-7]
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

Fill

Füllen des angegebenen Wertes einer vorhandenen Matrix oder Vektors.

```
void matrix::Fill(
  const double  value      // Wert für das Ausfüllen
   );

void vector::Fill(
  const double  value      // Wert für das Ausfüllen
   );
```

Parameter

value

[in] Wert, der allen Matrixelementen zugewiesen werden soll.

Rückgabewert

Kein Rückgabewert. Die Matrix wird an der Stelle mit dem angegebenen Wert gefüllt.

Beispiel

```
   matrix matrix_a(2,2);
   matrix_a.Fill(10);
   Print("matrix_a\n",matrix_a);

/*
  matrix_a
  [[10,10]
   [10,10]]
*/
```

© 2000-2025, MetaQuotes Ltd.

Random

Statische Funktion. Sie erstellt und gibt eine neue Matrix oder einen neuen Vektor mit Zufallswerten zurück. Zufallswerte werden gleichmäßig innerhalb des angegebenen Bereichs generiert.

```
static vector vector::Random(
  const ulong   size,       // Vektorlänge
  const double  min=0.0,    // Minimalwert
  const double  max=1.0     // Maximalwert
   );

static matrix matrix::Random(
  const ulong   rows,       // Anzahl der Zeilen
  const ulong   cols        // Anzahl der Spalten
  const float   min=0.0,    // Minimalwert
  const float   max=1.0     // Maximalwert
   );
```

Die Funktion weist einer existierenden Matrix oder Vektor Zufallswerte zu. Die Zufallswerte werden gleichmäßig innerhalb des angegebenen Bereichs generiert.

```
void  vector::Random(
  const double  min=0.0,    // Minimalwert
  const double  max=1.0     // Maximalwert
   );

void  matrix::Random(
  const float  min=0.0,     // Minimalwert
  const float  max=1.0      // Maximalwert
   );
```

Parameters

rows

[in] Anzahl der Zeilen.

cols

[in] Anzahl der Spalten.

size

[in] Vektorlänge.

min=0.0

[in] Der Minimalwert der generierten Zufallszahlen in der Stichprobe.

max=1.0

[in] Der Mindestwert der generierten Zufallszahlen in der Stichprobe.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Es wird eine neue Matrix mit den gegebenen Zeilen und Spalten zurückgegeben, die mit Zufallswerten gefüllt ist.

Beispiel

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector v1= vector::Random(6,0,4);
   Print("vector v1 \n", v1);
   /*
   vector v1
   [3.340834286841758,0.9073984578501895,3.15598511117417,0.7106475841045956,2.294010(
   */

   vector v2(4);
   v2.Random(-3,3);
   Print("vector v2 \n", v2);
   /*
   vector v2
   [-2.642032080246609,1.008521607147651,1.340623848219547,-2.789016363161853]
   */

   matrix m1 = matrix::Random(5, 5, -1, 3);
   Print("matrix m1 \n", m1);
   /*
   matrix m1
   [[1.468433369525809,1.199383023707052,0.6154667729547869,2.830012157519816,-0.06995
    [2.638591175814763,2.966658948912904,2.001837257994767,-0.08570802469870731,0.826(
    [0.982665140643405,0.5825661973987162,-0.6615881071569371,0.4069533109215444,-0.16
    [-0.5791836119208447,-0.9332020478167575,2.801898707589577,-0.09887318478405316,-(
    [-0.1069569017521471,-0.04576879345014895,0.6320100840255642,-0.3805729270401813,(
   */

   matrix m2(3, 3);
   m2.Random(-2, 2);
   Print("matrix m2 \n", m2);
   /*
   matrix m2
   [[-1.928191291151613,-1.222873445419332,-1.965333905062949]
    [-1.077723600686929,1.053317863273741,-0.1786824246353196]
    [-1.965241401448204,1.620956906029246,-1.696386492740453]]
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

Manipulation von Matrizen und Vektoren

Dies sind Methoden für grundlegende Matrixoperationen: Füllen, Kopieren, einen Teil einer Matrix erhalten, Transponieren, Aufteilen und Sortieren.

Außerdem gibt es mehrere Methoden für Operationen mit Matrixzeilen und -spalten.

Funktion	Aktion
HasNan	Rückgabe einer Zahl mit dem Wert NaN in einer Matrix/einem Vektor.
Transpose	Umkehrung oder Permutation der Achsen einer Matrix; gibt die geänderte Matrix zurück.
TriL	Gibt die Kopie einer Matrix zurück, bei der die Elemente oberhalb der k-ten Diagonale auf Null gesetzt sind. Untere Dreiecksmatrix.
TriU	Gibt die Kopie einer Matrix zurück, bei der die Elemente unterhalb der k-ten Diagonale auf Null gesetzt sind. Obere Dreiecksmatrix.
Diag	Eine Diagonale extrahieren oder eine Diagonalmatrix konstruieren.
Row	Gibt einen Zeilenvektor zurück. Schreibt einen Vektor in die angegebene Zeile.
Col	Gibt einen Spaltenvektor zurück. Schreibt einen Vektor in die angegebene Spalte.
Copy	Gibt eine Kopie der gegebenen Matrix oder des gegebenen Vektors zurück.
Concat	Verketten von 2 Submatrizen zu einer Matrix. Verketten von 2 Vektoren zu einem Vektor.
Compare	Vergleich der Elemente von zwei Matrizen oder Vektoren mit der angegebenen Präzision.
CompareByDigits	Vergleicht die Elemente zweier Matrizen oder Vektoren mit der Genauigkeit von signifikanten Dezimalstellen.
Flat	Ermöglicht die Adressierung eines Matrixelements über einen Index anstelle von zwei.
Clip	Limit der Elemente einer Matrix/eines Vektors in einen angegebenen Bereich gültiger Werte.
Reshape	Ändern der Form einer Matrix, ohne ihre Daten zu ändern.
Resize	Rückgabe einer neuen Matrix mit geänderter Form und Größe.
SwapRows	Vertauschen der Zeilen in einer Matrix.
SwapCols	Vertauschen der Spalten in einer Matrix.
Split	Aufteilen einer Matrix in mehrere Submatrizen.

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
Hsplit	Teilt eine Matrix vertikal in mehrere Submatrizen auf. Das Gleiche wie Split mit axis=0.
Vsplit	Teilt eine Matrix vertikal in mehrere Submatrizen auf. Das Gleiche wie Split mit axis=1.
ArgSort	Indirektes Sortieren einer Matrix oder Vektors.
Sort	Sortieren einer Matrix oder Vektors.

© 2000-2025, MetaQuotes Ltd.

HasNan

Rückgabe einer Zahl mit dem Wert NaN in einer Matrix/einem Vektor.

```
ulong vector::HasNan();

ulong matrix::HasNan();
```

Rückgabewert

Die Anzahl der Matrix-/Vektor-Elemente, die einen Wert NaN enthalten.

Hinweis

Beim Vergleich des entsprechenden Paares von Elementen mit NaN-Werten betrachten die Methoden Compare und CompareByDigits diese Elemente als gleich, während bei einem üblichen Vergleich von Fließkommazahlen NaN != NaN.

Beispiel:

```
void OnStart(void)
  {
   double x=sqrt(-1);

   Print("single: ",x==x);

   vector<double> v1={x};
   vector<double> v2={x};

   Print("vector: ", v1.Compare(v2,0)==0);
  }

/* Ergebnis:

 single: false
 vector: true
*/
```

Siehe auch

MathClassify, Compare, CompareByDigits

© 2000-2025, MetaQuotes Ltd.

Transpose

Matrix-Transposition. Umkehrung oder Vertauschung der Achsen einer Matrix; gibt die geänderte Matrix zurück.

```
matrix matrix::Transpose()
```

Rückgabewert

Transponierte Matrix.

Ein einfacher Algorithmus zur Transposition von Matrizen in MQL5:

```
matrix MatrixTranspose(const matrix& matrix_a)
  {
   matrix matrix_c(matrix_a.Cols(),matrix_a.Rows());

   for(ulong i=0; i<matrix_c.Rows(); i++)
      for(ulong j=0; j<matrix_c.Cols(); j++)
         matrix_c[i][j]=matrix_a[j][i];

   return(matrix_c);
  }
```

MQL5 Beispiel:

```
  matrix a= {{0, 1, 2}, {3, 4, 5}};
  Print("matrix a \n", a);
  Print("a.Transpose() \n", a.Transpose());

  /*
  matrix a
  [[0,1,2]
   [3,4,5]]
  a.Transpose()
  [[0,3]
   [1,4]
   [2,5]]
  */
```

Python Beispiel:

```
import numpy as np

a = np.arange(6).reshape((2,3))
print("a \n",a)
print("np.transpose(a) \n",np.transpose(a))
```

© 2000-2025, MetaQuotes Ltd.

```
a
 [[0 1 2]
 [3 4 5]]
np.transpose(a)
 [[0 3]
 [1 4]
 [2 5]]
```

© 2000-2025, MetaQuotes Ltd.

TransposeConjugate

Transposing a complex matrix with conjugation. Reverse or permute the axes of a matrix by changing a sign of an imaginary part of a complex number, return the modified matrix.

```
matrix matrix::TransposeConjugate()
```

Return Value

Transposed complex conjugate matrix.

Simple algorithm of transposing a complex matrix with conjugation - method explanation

```
//--- Complex matrix transpose function with conjugation
matrix MatrixTranspose(const matrix& matrix_a) {
    //--- create a new matrix_c with dimensions inverse to matrix_a
    matrix matrix_c(matrix_a.Cols(), matrix_a.Rows());

    //--- go through all the rows of the new matrix
    for (ulong i = 0; i < matrix_c.Rows(); i++) {
        //--- go through all the columns of the new matrix
        for (ulong j = 0; j < matrix_c.Cols(); j++) {
            //--- transfer the real part of the element by transposing the index
            matrix_c[i][j].real = matrix_a[j][i].real;

            //--- transfer the imaginary part of the element by changing the sign (cor
            matrix_c[i][j].imag = -matrix_a[j][i].imag;
        }
    }

    //--- return the transposed matrix with conjugation
    return (matrix_c);
}
```

© 2000-2025, MetaQuotes Ltd.

TriL

Gibt die Kopie einer Matrix zurück, bei der die Elemente oberhalb der k-ten Diagonale auf Null gesetzt sind. Untere Dreiecksmatrix

```
matrix matrix::Tril(
  const int     ndiag=0      // Index der Diagonalen
   );
```

Parameter

ndiag=0[in] Index der Diagonale: 0 (Standardwert) bezieht sich auf die Hauptdiagonale, ein positiver Wert bezieht sich auf eine obere Diagonale und ein negativer Wert auf eine untere Diagonale.RückgabewertEine Matrix, bei der alle Elemente gleich Null sind, mit Ausnahme der k-ten Diagonale, deren Werte gleich Eins sind.

[in] Diagonale, über der die Elemente auf Null gesetzt werden. ndiag = 0 (Standardeinstellung) ist die Hauptdiagonale, ndiag < 0 ist darunter und ndiag > 0 ist darüber.

Rückgabewert

Array, dessen unteres Dreieck mit Einsen gefüllt ist, während an anderer Stelle Null steht.

MQL5 Beispiel:

```
   matrix a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
   matrix b=a.TriL(-1);
   Print("matrix b \n",b);

  /*
  matrix_c
  [[0,0,0]
  [4,0,0]
  [7,8,0]
  [10,11,12]]
  */
```

Python Beispiel:

```
import numpy as np

a=np.tril([[1,2,3],[4,5,6],[7,8,9],[10,11,12]], -1)

[[ 0  0  0]
 [ 4  0  0]
 [ 7  8  0]
 [10 11 12]]
```

© 2000-2025, MetaQuotes Ltd.

TriU

Gibt die Kopie einer Matrix zurück, bei der die Elemente unterhalb der k-ten Diagonale auf Null gesetzt sind. Obere Dreiecksmatrix

```
matrix matrix::Triu(
  const int     ndiag=0      // Index der Diagonalen
   );
```

Parameter

ndiag=0[in] Index der Diagonale: 0 (Standardwert) bezieht sich auf die Hauptdiagonale, ein positiver Wert bezieht sich auf eine obere Diagonale und ein negativer Wert auf eine untere Diagonale.RückgabewertEine Matrix, bei der alle Elemente gleich Null sind, mit Ausnahme der k-ten Diagonale, deren Werte gleich Eins sind.

[in] Diagonale, unter der die Elemente auf Null gesetzt werden. ndiag = 0 (Standardeinstellung) ist die Hauptdiagonale, ndiag < 0 ist darunter und ndiag > 0 ist darüber.

MQL5 Beispiel:

```
   matrix a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
   matrix b=a.TriU(-1);
   Print("matrix b \n",b);

  /*
  matrix b
  [[1,2,3]
   [4,5,6]
   [0,8,9]
   [0,0,12]]
  */
```

Python Beispiel:

```
import numpy as np

a=np.triu([[1,2,3],[4,5,6],[7,8,9],[10,11,12]], -1)
print(a)

[[ 1  2  3]
 [ 4  5  6]
 [ 0  8  9]
 [ 0  0 12]]
```

© 2000-2025, MetaQuotes Ltd.

Diag

Eine Diagonale extrahieren oder eine Diagonalmatrix konstruieren.

```
vector matrix::Diag(
  const int      ndiag=0      // Zahl der Diagonale
   );

void matrix::Diag(
  const vector   v,           // Diagonalvektor
  const int      ndiag=0      // Zahl der Diagonale
   );
```

Parameter

v

[in] Ein Vektor, dessen Elemente in der entsprechenden Diagonale enthalten sind (ndiag=0 ist die Hauptdiagonale).

ndiag=0[in] Index der Diagonale: 0 (Standardwert) bezieht sich auf die Hauptdiagonale, ein positiver Wert bezieht sich auf eine obere Diagonale und ein negativer Wert auf eine untere Diagonale.RückgabewertEine Matrix, bei der alle Elemente gleich Null sind, mit Ausnahme der k-ten Diagonale, deren Werte gleich Eins sind.

[in] Die betreffende Diagonale. Standardwert ist 0. Verwenden Sie ndiag>0 für Diagonalen oberhalb der Hauptdiagonale und ndiag<0 für Diagonalen unterhalb der Hauptdiagonale.

Hinweis

Auch für nicht allozierte Matrizen (die keine Dimensionen haben) kann eine Diagonale gesetzt werden. In diesem Fall wird eine Nullmatrix der Größe erstellt, deren Größe der Größe des Diagonalvektors entspricht, woraufhin die Vektorwerte in die entsprechende Diagonale eingefügt werden. Wenn die Diagonale auf eine bereits vorhandene Matrix gesetzt wird, ändern sich die Dimensionen der Matrix nicht und die Werte der Matrixelemente außerhalb des Diagonalvektors ändern sich nicht.

Beispiel

```
   vector v1={1,2,3};
   matrix m1;
   m1.Diag(v1);
   Print("m1\n",m1);
   matrix m2;
   m2.Diag(v1,-1);
   Print("m2\n",m2);
   matrix m3;
   m3.Diag(v1,1);
   Print("m3\n",m3);
   matrix m4=matrix::Full(4,5,9);
```

© 2000-2025, MetaQuotes Ltd.

```
 m4.Diag(v1,1);
 Print("m4\n",m4);

 Print("diag -1 - ",m4.Diag(-1));
 Print("diag 0 - ",m4.Diag());
 Print("diag 1 - ",m4.Diag(1));

/*

m1
[[1,0,0]
[0,2,0]
[0,0,3]]
m2
[[0,0,0]
[1,0,0]
[0,2,0]
[0,0,3]]
m3
[[0,1,0,0]
[0,0,2,0]
[0,0,0,3]]
m4
[[9,1,9,9,9]
[9,9,2,9,9]
[9,9,9,3,9]
[9,9,9,9,9]]
diag -1 - [9,9,9]
diag 0 - [9,9,9,9]
diag 1 - [1,2,3,9]
*/
```

© 2000-2025, MetaQuotes Ltd.

Row

Gibt einen Zeilenvektor zurück. Schreibt einen Vektor in die angegebene Zeile.

```
vector matrix::Row(
  const ulong   nrow      // Zeilennummer
   );

void matrix::Row(
  const vector  v,        // Zeile des Vektors
  const ulong   nrow      // Zeilennummer
   );
```

Parameter

nrow

[in] Zeilennummer.

Rückgabewert

Vektor.

Hinweis

Auch für nicht allozierte Matrizen (die keine Dimensionen haben) kann eine Zeile festgelegt werden. In diesem Fall wird eine Nullmatrix mit der Größe des Vektors Größe x Zeilennummer+1 erstellt, woraufhin die Werte der Vektorelemente in die entsprechende Zeile eingefügt werden. Wird die Zeile auf eine bereits existierende Matrix gesetzt, ändern sich die Matrixdimensionen nicht und die Werte der Matrixelemente außerhalb des Zeilenvektors ändern sich nicht.

Beispiel

```
   vector v1={1,2,3};
   matrix m1;
   m1.Row(v1,1);
   Print("m1\n",m1);
   matrix m2=matrix::Full(4,5,7);
   m2.Row(v1,2);
   Print("m2\n",m2);

   Print("row 1 - ",m2.Row(1));
   Print("row 2 - ",m2.Row(2));

  /*
  m1
  [[0,0,0]
  [1,2,3]]
  m2
```

© 2000-2025, MetaQuotes Ltd.

```
   [[7,7,7,7,7]
   [7,7,7,7,7]
   [1,2,3,7,7]
   [7,7,7,7,7]]
   row 1 - [7,7,7,7,7]
   row 2 - [1,2,3,7,7]
   */
```

© 2000-2025, MetaQuotes Ltd.

Col

Gibt einen Spaltenvektor zurück. Schreibt einen Vektor in die angegebene Spalte.

```
vector matrix::Col(
  const ulong   ncol      // Spaltennummer
   );

void matrix::Col(
  const vector  v,        // Spaltenvektor
  const ulong   ncol      // Spaltennummer
   );
```

Parameter

ncol

[in] Spaltennummer.

Rückgabewert

Vektor.

Hinweis

Auch für nicht allozierte Matrizen (die keine Dimensionen haben) kann eine Spalte festgelegt werden. In diesem Fall wird eine Nullmatrix mit der Größe des Vektors Größe x Spaltennummer+1 erstellt, woraufhin die Werte der Vektorelemente in die entsprechende Spalte eingefügt werden. Wird die Spalte auf eine bereits existierende Matrix gesetzt, ändern sich die Matrixdimensionen nicht und die Werte der Matrixelemente außerhalb des Spaltenvektors ändern sich nicht.

Beispiel

```
   vector v1={1,2,3};
   matrix m1;
   m1.Col(v1,1);
   Print("m1\n",m1);
   matrix m2=matrix::Full(4,5,8);
   m2.Col(v1,2);
   Print("m2\n",m2);

   Print("col 1 - ",m2.Col(1));
   Print("col 2 - ",m2.Col(2));

  /*
  m1
  [[0,1]
  [0,2]
  [0,3]]
  m2
```

© 2000-2025, MetaQuotes Ltd.

```
   [[8,8,1,8,8]
   [8,8,2,8,8]
   [8,8,3,8,8]
   [8,8,8,8,8]]
   col 1 - [8,8,8,8]
   col 2 - [1,2,3,8]
   */
```

© 2000-2025, MetaQuotes Ltd.

Copy

Erstellt eine Kopie der angegebenen Matrix oder Vektor

```
bool matrix::Copy(
  const matrix&  a     // zu kopierende Matrix
   );
bool vector::Copy(
  const vector&  v     // zu kopierender Vektor
   );
```

Parameter

v

[in] zu kopierende Matrix oder Vektor.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

MQL5 Beispiel:

```
  matrix a=matrix::Eye(3, 4);
  matrix b;
  b.Copy(a);
  matrix c=a;
  Print("matrix b \n", b);
  Print("matrix_c \n", c);

  /*
  /*
  matrix b
  [[1,0,0,0]
  [0,1,0,0]
  [0,0,1,0]]
  matrix_c
  [[1,0,0,0]
  [0,1,0,0]
  [0,0,1,0]]
  */
  */
```

Python Beispiel:

```
import numpy as np

a = np.eye(3,4)
print('a \n',a)
b = a
```

© 2000-2025, MetaQuotes Ltd.

```
print('b \n',b)
c = np.copy(a)
print('c \n',c)

a
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]]
b
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]]
c
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]]
```

© 2000-2025, MetaQuotes Ltd.

Concat

Verketten von 2 Submatrizen zu einer Matrix. Verketten von 2 Vektoren zu einem Vektor.

```
vector vector::Concat(
  const vector&  second_part      // zweiter zu verkettender Vektor
   );

vector matrix::Concat(
  const matrix&  second_part      // zweite zu verkettende Matrix
   );

matrix matrix::Split(
  const matrix&  second_part,     // zweite zu verkettende Matrix
  const int      axis             // Achse
   );
```

Parameters

second_part

[in] Zweiter zu verkettender Vektor oder Matrix. Wenn Matrizen entlang einer der Achsen verkettet werden, müssen die Größen der Matrizen entsprechend der Achse konsistent sein.

axis

[in] Achse. 0 - horizontale Achse, 1 - vertikale Achse.

Rückgabewerte

Es wird ein Vektor zurückgegeben, wenn Vektoren oder Matrizen ohne Achsenparameter verkettet wurden, oder eine Matrix, die entlang einer horizontalen oder vertikalen Achse verkettet wurde.

Example

```
   vector vector_a={1,2,3,4};
   vector vector_b={5,6,7};
   vector vector_c=vector_a.Concat(vector_b);
   Print("vector_c=",vector_c);

   matrix matrix_a={{1,2,3},{4,5,6}};
   matrix matrix_b={{7,8,9},{10,11,12}};
   vector_c=matrix_a.Concat(matrix_b);
   Print("vector_c=",vector_c);

   matrix matrix_c0=matrix_a.Concat(matrix_b,0);
   Print("matrix_c0=\n",matrix_c0);

   matrix matrix_c1=matrix_a.Concat(matrix_b,1);
   Print("matrix_c1=\n",matrix_c1);

  /*
```

© 2000-2025, MetaQuotes Ltd.

```
vector_c=[1,2,3,4,5,6,7]
vector_c=[1,2,3,4,5,6,7,8,9,10,11,12]
matrix_c0=
[[1,2,3]
 [4,5,6]
 [7,8,9]
 [10,11,12]]
matrix_c1=
[[1,2,3,7,8,9]
 [4,5,6,10,11,12]]
*/
```

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleich der Elemente von zwei Matrizen oder Vektoren mit der angegebenen Präzision.

```
ulong vector::Compare(
  const vector& vec,          // zu vergleichender Vektor
  const double  epsilon       // Präzision
   );

ulong matrix::Compare(
  const matrix& mat,          // zu vergleichende Matrix
  const double  epsilon       // Präzision
   );
```

Parameter

vector_b

[in] zu vergleichende Matrix.

epsilon

[in] Präzision.

Rückgabewert

Die Anzahl der nicht übereinstimmenden Elemente der zu vergleichenden Matrizen oder Vektoren: 0, wenn die Matrizen gleich sind, sonst größer als 0.

Hinweis

Die Vergleichsoperatoren == oder != führen einen exakten, elementweisen Vergleich durch. Es ist bekannt, dass der exakte Vergleich von reellen Zahlen nur von begrenztem Nutzen ist, daher wurde die Epsilon-Vergleichsmethode hinzugefügt. Es kann vorkommen, dass eine Matrix Elemente in einem Bereich enthalten kann, z. B. von 1e-20 to 1e+20. Solche Matrizen können mit dem elementweisen Vergleich bis zu signifikanten Zahlen verarbeitet werden.

Bei komplexen Matrizen/Vektoren beinhaltet der Vergleich die Schätzung des Abstands zwischen komplexen Zahlen. Der Abstand wird berechnet durch sqrt(pow(r1-r2, 2) + pow(i1-i2, 2) und ist eine reelle Zahl, die direkt mit epsilon verglichen werden kann.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4}};
   matrix matrix_i=matrix::Identity(3,3);
   matrix matrix_c=matrix_a.Inv();
   matrix matrix_check=matrix_a.MatMul(matrix_c);
   Print("matrix_check\n",matrix_check);

   ulong errors=matrix_check.Compare(matrix::Identity(3,3),1e-15);
   Print("errors=",errors);
```

© 2000-2025, MetaQuotes Ltd.

```
/*
matrix_check
[[1,0,0]
[4.440892098500626e-16,1,8.881784197001252e-16]
[4.440892098500626e-16,2.220446049250313e-16,0.9999999999999996]]
errors=0

*/
```

© 2000-2025, MetaQuotes Ltd.

CompareByDigits

Vergleicht die Elemente zweier Matrizen oder Vektoren mit der Präzision der signifikanten Dezimalstellen.

```
ulong vector::CompareByDigits(
  const vector& vec,          // zu vergleichender Vektor
  const int     digits        // Anzahl der signifikanten Dezimalstellen
   );

ulong matrix::CompareByDigits(
  const matrix& mat,          // zu vergleichende Matrix
  const int     digits        // Anzahl der signifikanten Dezimalstellen
   );
```

Parameter

vector_b

[in] zu vergleichende Matrix.

digits

[in] Anzahl der signifikanten Dezimalstellen für den Vergleich.

epsilon

[in] Präzision des Vergleichs. Wenn sich zwei Werte im Absolutwert um weniger als die angegebene Präzision unterscheiden, werden sie als gleich angesehen.

Rückgabewert

Die Anzahl der nicht übereinstimmenden Elemente der zu vergleichenden Matrizen oder Vektoren: 0, wenn die Matrizen gleich sind, sonst größer als 0.

Hinweis

Die Vergleichsoperatoren == oder != führen einen exakten, elementweisen Vergleich durch. Es ist bekannt, dass der exakte Vergleich von reellen Zahlen nur von begrenztem Nutzen ist, daher wurde die Epsilon-Vergleichsmethode hinzugefügt. Es kann vorkommen, dass eine Matrix Elemente in einem Bereich enthalten kann, z. B. von 1e-20 to 1e+20. Solche Matrizen können elementweise mit der Genauigkeit von signifikanten Dezimalstellen vergleichen werden.

Beispiel

```
   int    size_m=128;
   int    size_k=256;
   matrix matrix_a(size_m,size_k);
//--- füllen der Matrix
   double  value=0.0;
   for(int i=0; i<size_m; i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      for(int j=0; j<size_k; j++)
        {
         if(i==j)
            matrix_a[i][j]=1.0+i;
         else
           {
            value+=1.0;
            matrix_a[i][j]=value/1e+20;
           }
        }
     }
//--- Erstellen einer weiteren Matrix
   matrix matrix_c = matrix_a * -1;

   ulong errors_epsilon=matrix_a.Compare(matrix_c,1e-15);
   ulong errors_digits=matrix_a.CompareByDigits(matrix_c,15);

   printf("Compare matrix %d x %d  errors_epsilon=%I64u  errors_digits=%I64u",size_m,s


  /*
  Compare matrix 128 x 256  errors_epsilon=128  errors_digits=32768
  */
```

© 2000-2025, MetaQuotes Ltd.

CompareEqual

Perform an absolute comparison of two matrices by unfolding successive rows into one-dimensional vectors.

```
ulong vector::Compare(
  const vector& vec    // vector to compare
   );

ulong matrix::CompareEqual(
  const matrix& mat     // matrix to compare
   );
```

Parameters

vec

[in] Vector to compare.

mat

[in] Matrix to compare.

Method description

Let us have two matrices: matrix A the method is called for and matrix B, which is passed as a method parameter. The comparison is performed as follows:

1. Matrices are expanded into one-dimensional vectors by successive concatenation of rows.
2. Vectors are compared element by element until the first mismatched element.
3. Depending on the comparison results, one of the values described below is returned.

Return Value

-1 — if the matrix A element is less than the corresponding matrix B element.

0 — if all elements of A and B matrices are identical.

1 — if the matrix A element exceeds the corresponding matrix B element.

Note

NaN value elements are taken into account when comparing.

NaN value elements are considered equal if they are present in both matrices at corresponding positions.

The NaN sign is not taken into account in the comparison.

An element with a NaN value is considered to be less than any other numeric value.

Example

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   matrix matrix_a= {{10, 3, 2}, {1, 8, 12}, {6, 5, 4}};
   matrix matrix_i=matrix::Identity(3, 3);
   matrix matrix_c=matrix_a.Inv();
   matrix matrix_check=matrix_a.MatMul(matrix_c);
   Print("matrix_check\n", matrix_check);

   ulong errors=matrix_check.Compare(matrix::Identity(3, 3), 1e-15);
   Print("errors=", errors);
   /*
   matrix_check
   [[1,0,0]
   [4.440892098500626e-16,1,8.881784197001252e-16]
   [4.440892098500626e-16,2.220446049250313e-16,0.9999999999999996]]
   errors=0
   */

//---  absolute comparison of matrices
   matrix<double> A = matrix_a;  // Matrix A initialization
   matrix<double> B = matrix_c;  // Matrix B initialization
   int result = A.CompareEqual(B);
   switch(result)
     {
      case -1:
         Print("Matrix A is smaller than matrix B");
         break;
      case 0:
         Print("Matrices A and B are identical");
         break;
      case 1:
         Print("Matrix A is greater than matrix B");
         break;
      case -2:
         Print("Error! Matrix A is not initialized");
         break;
      case 2:
         Print("Error! Matrix B is not initialized");
         break;
      case -3:
         Print("Error! The size of matrix A is less than the size of matrix B");
         break;
      case 3:
         Print("Error! The size of matrix A is greater than the size of matrix B");
         break;
      default:
         Print("Error! Unknown error");
```

© 2000-2025, MetaQuotes Ltd.

```
            break;
        }
    }
```

© 2000-2025, MetaQuotes Ltd.

Flat

Ermöglicht die Adressierung eines Matrixelements über einen Index anstelle von zwei.

```
bool matrix::Flat(
  const ulong   index,      //
  const double  value       // zu setzender Wert
   );

double matrix::Flat(
  const ulong   index,      //
   );
```

Parameter

index

[in] Index für Flat

value

[in] Wert, der der Stelle des Index zugewiesen werden soll.

Rückgabewert

Wert nach gegebenem Index.

Hinweis

Für die Matrix mat(3,3) kann der Zugriff wie folgt geschrieben werden:

- Lesen: 'x=mat.Flat(4)', was gleichbedeutend ist mit 'x=mat[1][1]'
- Schreiben: 'mat.Flat(5, 42)', was 'mat[1][2]=42' entspricht

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);
   ulong arg_max=matrix_a.ArgMax();
   Print("max_value=",matrix_a.Flat(arg_max));
   matrix_a.Flat(arg_max,0);
   arg_max=matrix_a.ArgMax();
   Print("max_value=",matrix_a.Flat(arg_max));


  /*
  matrix_a
  [[10,3,2]
   [1,8,12]
   [6,5,4]
   [7,11,9]]
```

© 2000-2025, MetaQuotes Ltd.

```
   max_value=12.0
   max_value=11.0
   */
```

© 2000-2025, MetaQuotes Ltd.

Clip

Limitiert die Elemente einer Matrix/eines Vektors auf einen angegebenen Bereich gültiger Werte.

```
bool matrix::Clip(
  const double  min_value,     // Minimalwert
  const double  max_value      // Maximalwert
   );
bool vector::Clip(
  const double  min_value,     // Minimalwert
  const double  max_value      // Maximalwert
   );
```

Parameter

min_value

[in] Minimalwert.

max_value

[in] Maximalwert.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Die Matrix (oder der Vektor) wird am Speicherort verarbeitet. Es werden keine Kopien erstellt. Es werden keine Kopien erstellt.

Beispiel

```
   matrix matrix_a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
   bool res=matrix_a.Clip(4,8);
   Print("matrix_a\n",matrix_a);

  /*
  matrix_a
  [[4,4,4]
   [4,5,6]
   [7,8,8]
   [8,8,8]]
  */
```

© 2000-2025, MetaQuotes Ltd.

Reshape

Ändern der Form einer Matrix, ohne ihre Daten zu ändern.

```
void  Reshape(
  const ulong  rows,      // neue Zeilenanzahl.
  const ulong  cols       // neue Spaltenanzahl.
   );
```

Parameter

rows

[in] Neue Zeilenanzahl.

cols

[in] Neue Spaltenanzahl.

Hinweis

Die Matrix (oder der Vektor) wird am Speicherort verarbeitet. Es werden keine Kopien erstellt. Es werden keine Kopien erstellt. Es kann eine beliebige Größe angegeben werden, d. h. rows_new*cols_new!=rows_old*cols_old. Wenn der Matrixpuffer vergrößert wird, sind die zusätzlichen Werte undefiniert.

Beispiel

```
   matrix matrix_a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};

   Print("matrix_a\n",matrix_a);
   matrix_a.Reshape(2,6);
   Print("Reshape(2,6)\n",matrix_a);
   matrix_a.Reshape(3,5);
   Print("Reshape(3,5)\n",matrix_a);
   matrix_a.Reshape(2,4);
   Print("Reshape(2,4)\n",matrix_a);

  /*
  matrix_a
  [[1,2,3]
   [4,5,6]
   [7,8,9]
   [10,11,12]]
  Reshape(2,6)
  [[1,2,3,4,5,6]
   [7,8,9,10,11,12]]
  Reshape(3,5)
  [[1,2,3,4,5]
   [6,7,8,9,10]
   [11,12,0,3,0]]
  Reshape(2,4)
```

© 2000-2025, MetaQuotes Ltd.

```
   [[1,2,3,4]
    [5,6,7,8]]
   */
```

© 2000-2025, MetaQuotes Ltd.

Resize

Rückgabe einer neuen Matrix mit geänderter Form und Größe.

```
bool matrix::Resize(
  const ulong  rows,     //
  const ulong  cols,     // neue Spaltenanzahl.
  const ulong  reserve=0 // reservierte Anzahl von Elementen.
   );

bool vector::Resize(
  const ulong  size,     // neue Größe.
  const ulong  reserve=0 // reservierte Anzahl von Elementen.
   );
```

Parameter

rows

[in] Neue Zeilenanzahl.

cols

[in] Neue Spaltenanzahl.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Die Matrix (oder der Vektor) wird am Speicherort verarbeitet. Es werden keine Kopien erstellt. Es werden keine Kopien erstellt. Es kann eine beliebige Größe angegeben werden, d. h. rows_new*cols_new!=rows_old*cols_old. Im Gegensatz zu Reshape wird die Matrix zeilenweise verarbeitet. Wenn die Anzahl der Spalten erhöht wird, sind die Werte der zusätzlichen Spalten undefiniert. Wenn die Anzahl der Zeilen erhöht wird, sind die Werte der Elemente in den neuen Zeilen undefiniert. Wenn die Anzahl der Spalten verringert wird, wird jede Zeile der Matrix abgeschnitten.

Beispiel

```
matrix matrix_a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
Print("matrix_a\n",matrix_a);
matrix_a.Resize(2,6);
Print("Ressize(2,6)\n",matrix_a);
matrix_a.Resize(3,5);
Print("Resize(3,5)\n",matrix_a);
matrix_a.Resize(2,4);
Print("Resize(2,4)\n",matrix_a);

/*
```

© 2000-2025, MetaQuotes Ltd.

```
   matrix_a
   [[1,2,3]
    [4,5,6]
    [7,8,9]
    [10,11,12]]
   Ressize(2,6)
   [[1,2,3,4,5,6]
    [4,5,6,10,11,12]]
   Resize(3,5)
   [[1,2,3,4,5]
    [4,5,6,10,11]
    [11,12,3,8,8]]
   Resize(2,4)
   [[1,2,3,4]
    [4,5,6,10]]
   */
```

© 2000-2025, MetaQuotes Ltd.

Set

Setzt den Wert eines Vektorelements am angegebenen Index.

```
bool vector::Set(
  ulong   index,      // Element-Index
  double  value       // Wert
   );
```

Parameter

index

[in] Index des Elements, dem der Wert zuzuweisen ist.

value

[in] Wert.

Rückgabewert

Liefert bei Erfolg true, ansonsten false.

Hinweis

Die Set-Methode tut dasselbe wie die Zuweisung eines Wertes mittels der eckigen Klammern, nämlich: *vector[index]=value*. Die Methode wurde hinzugefügt, um die Übertragung eines Codes aus Sprachen zu vereinfachen, in denen diese Art der Notation verwendet wird. Das folgende Beispiel zeigt beide Optionen zum Füllen des Vektors mit Werten am angegebenen Index.

Beispiel:

```
void OnStart()
  {
//---
   vector v1(10, VectorAssignValues);
   Print("v1 = ", v1);

   vector v2(10, VectorSetValues);
   Print("v2 = ", v2);
  }
 /* Ergebnis
  v1 = [1,2,4,8,16,32,64,128,256,512]
  v2 = [1,2,4,8,16,32,64,128,256,512]
  */
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit Potenzen einer Zahl durch die Zuweisung |
//+------------------------------------------------------------------+
void VectorAssignValues(vector& v, double initial=1)
  {
   double value=initial;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(ulong k=0; k<v.Size(); k++)
     {
      v[k]=value;
      value*=2;
     }
  }
//+------------------------------------------------------------------------+
//| Füllen eines Vektors mit Potenzen einer Zahl mit der Methode Set       |
//+------------------------------------------------------------------------+
void VectorSetValues(vector& v, double initial=1)
  {
   double value=initial;
   for(ulong k=0; k<v.Size(); k++)
     {
      v.Set(k, value);
      value*=2;
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SwapRows

Vertauschen der Zeilen in einer Matrix.

```
bool matrix::SwapRows(
  const ulong  row1,     // Index der ersten Zeile
  const ulong  row2      // Index der zweiten Zeile
   );
```

Parameter

row1

[in] Index der ersten Zeile.

row2

[in] Index der zweiten Zeile.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Beispiel

```
   matrix matrix_a={{1,2,3,4},
                    {5,6,7,8},
                    {9,10,11,12},
                    {13,14,15,16}};
   matrix matrix_i=matrix::Identity(4,4);
   matrix matrix_a1=matrix_a;
   matrix_a1.SwapRows(0,3);
   Print("matrix_a1\n",matrix_a1);

   matrix matrix_p=matrix_i;
   matrix_p.SwapRows(0,3);
   matrix matrix_c1=matrix_p.MatMul(matrix_a);
   Print("matrix_c1\n",matrix_c1);

  /*
  matrix_a1
  [[13,14,15,16]
   [5,6,7,8]
   [9,10,11,12]
   [1,2,3,4]]
  matrix_c1
  [[13,14,15,16]
   [5,6,7,8]
   [9,10,11,12]
   [1,2,3,4]]
  */
```

© 2000-2025, MetaQuotes Ltd.

SwapCols

Vertauschen der Spalten in einer Matrix.

```
bool matrix::SwapCols(
  const ulong  row1,     // Index der ersten Spalte
  const ulong  row2      // Index der zweiten Spalte
   );
```

Parameter

col1

[in] Index der ersten Spalte.

col2

[in] Index der zweiten Spalte.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Beispiel

```
   matrix matrix_a={{1,2,3,4},
                    {5,6,7,8},
                    {9,10,11,12},
                    {13,14,15,16}};
   matrix matrix_i=matrix::Identity(4,4);
   matrix matrix_a1=matrix_a;
   matrix_a1.SwapCols(0,3);
   Print("matrix_a1\n",matrix_a1);

   matrix matrix_p=matrix_i;
   matrix_p.SwapCols(0,3);
   matrix matrix_c1=matrix_a.MatMul(matrix_p);
   Print("matrix_c1\n",matrix_c1);

  /*
  matrix_a1
  [[4,2,3,1]
   [8,6,7,5]
   [12,10,11,9]
   [16,14,15,13]]
  matrix_c1
  [[4,2,3,1]
   [8,6,7,5]
   [12,10,11,9]
   [16,14,15,13]]
  */
```

© 2000-2025, MetaQuotes Ltd.

Split

Aufteilen einer Matrix in mehrere Submatrizen.

```
bool matrix::Split(
  const ulong  parts,      // Anzahl der Submatrizen
  const int    axis,       // Achse
  matrix&      splitted[]  // Array der resultierenden Submatrizen
   );

void matrix::Split(
  const ulong& parts[],    // Größen der Submatrizen
  const int    axis,       // Achse
  matrix&      splitted[]  // Array der resultierenden Submatrizen
   );
```

Parameter

parts

[in] Die Anzahl der Submatrizen, in die die Matrix aufgeteilt werden soll.

axis

[in] Achse. 0 - horizontale Achse, 1 - vertikale Achse.

splitted

[out] Array der resultierenden Submatrizen.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Wird die Anzahl der Submatrizen angegeben, so erhält man gleich große Teilmatrizen. Das bedeutet, dass die Größe der Matrix (0 - die Anzahl der Zeilen, 1 - die Anzahl der Spalten) ohne Rest durch "parts" teilbar sein muss. Submatrizen unterschiedlicher Größe können mit Hilfe eines Arrays von Submatrixgrößen erhalten werden. Die Elemente des Arrays mit den Größen werden so lange verwendet, bis die gesamte Matrix aufgeteilt ist. Wenn das Array der Größen zu Ende ist und die Matrix noch nicht vollständig aufgeteilt wurde, ist der ungeteilte Rest die letzte Submatrix.

Beispiel

```
   matrix matrix_a={{ 1, 2, 3, 4, 5, 6},
                    { 7, 8, 9,10,11,12},
                    {13,14,15,16,17,18},
                    {19,20,21,22,23,24},
                    {25,26,27,28,29,30}};
   matrix splitted[];
   ulong  parts[]={2,2};
```

© 2000-2025, MetaQuotes Ltd.

```
 bool res=matrix_a.Split(2,0,splitted);
 Print(res,"  ",GetLastError());
 ResetLastError();
 for(uint i=0; i<splitted.Size(); i++)
    Print("splitted ",i,"\n",splitted[i]);

 res=matrix_a.Split(2,1,splitted);
 Print(res,"  ",GetLastError());
 for(uint i=0; i<splitted.Size(); i++)
    Print("splitted ",i,"\n",splitted[i]);

 res=matrix_a.Split(parts,0,splitted);
 Print(res,"  ",GetLastError());
 for(uint i=0; i<splitted.Size(); i++)
    Print("splitted ",i,"\n",splitted[i]);

/*
false  4003
true  0
splitted 0
[[1,2,3]
 [7,8,9]
 [13,14,15]
 [19,20,21]
 [25,26,27]]
splitted 1
[[4,5,6]
 [10,11,12]
 [16,17,18]
 [22,23,24]
 [28,29,30]]
true  0
splitted 0
[[1,2,3,4,5,6]
 [7,8,9,10,11,12]]
splitted 1
[[13,14,15,16,17,18]
 [19,20,21,22,23,24]]
splitted 2
[[25,26,27,28,29,30]]
*/
```

© 2000-2025, MetaQuotes Ltd.

Hsplit

Horizontales Teilen einer Matrix in mehrere Submatrizen. Dasselbe wie Split mit axis=0.

```
bool matrix::Hsplit(
  const ulong  parts,      // Anzahl der Submatrizen
  matrix&      splitted[]  // Array der resultierenden Submatrizen
   );

void matrix::Hsplit(
  const ulong& parts[],    // Größen der Submatrizen
  matrix&      splitted[]  // Array der resultierenden Submatrizen
   );
```

Parameter

parts

[in] Die Anzahl der Submatrizen, in die die Matrix aufgeteilt werden soll.

splitted

[out] Array der resultierenden Submatrizen.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Wird die Anzahl der Submatrizen angegeben, so erhält man gleich große Teilmatrizen. Das bedeutet, dass die Anzahl der Zeilen durch "parts" ohne Rest teilbar sein muss. Submatrizen unterschiedlicher Größe können mit Hilfe eines Arrays von Submatrixgrößen erhalten werden. Die Elemente des Arrays mit den Größen werden so lange verwendet, bis die gesamte Matrix aufgeteilt ist. Wenn das Array der Größen zu Ende ist und die Matrix noch nicht vollständig aufgeteilt wurde, ist der ungeteilte Rest die letzte Submatrix.

Beispiel

```
   matrix matrix_a={{ 1, 2, 3, 4, 5, 6},
                    { 7, 8, 9,10,11,12},
                    {13,14,15,16,17,18},
                    {19,20,21,22,23,24},
                    {25,26,27,28,29,30}};
   matrix splitted[];
   ulong  parts[]={2,4};

   bool res=matrix_a.Hsplit(2,splitted);
   Print(res,"  ",GetLastError());
   ResetLastError();
   for(uint i=0; i<splitted.Size(); i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("splitted ",i,"\n",splitted[i]);

   res=matrix_a.Hsplit(5,splitted);
   Print(res,"  ",GetLastError());
   for(uint i=0; i<splitted.Size(); i++)
      Print("splitted ",i,"\n",splitted[i]);

   res=matrix_a.Hsplit(parts,splitted);
   Print(res,"  ",GetLastError());
   for(uint i=0; i<splitted.Size(); i++)
      Print("splitted ",i,"\n",splitted[i]);


  /*
  false  4003
  true  0
  splitted 0
  [[1,2,3,4,5,6]]
  splitted 1
  [[7,8,9,10,11,12]]
  splitted 2
  [[13,14,15,16,17,18]]
  splitted 3
  [[19,20,21,22,23,24]]
  splitted 4
  [[25,26,27,28,29,30]]
  true  0
  splitted 0
  [[1,2,3,4,5,6]
  [7,8,9,10,11,12]]
  splitted 1
  [[13,14,15,16,17,18]
  [19,20,21,22,23,24]
  [25,26,27,28,29,30]]
  */
```

© 2000-2025, MetaQuotes Ltd.

Vsplit

Teilt eine Matrix vertikal in mehrere Submatrizen auf. Entspricht Split mit Achse=1

```
bool matrix::Vsplit(
  const ulong  parts,      // Anzahl der Submatrizen
  matrix&      splitted[]  // Array der resultierenden Submatrizen
   );

void matrix::Vsplit(
  const ulong& parts[],    // Größen der Submatrizen
  matrix&      splitted[]  // Array der resultierenden Submatrizen
   );
```

Parameter

parts

[in] Die Anzahl der Submatrizen, in die die Matrix aufgeteilt werden soll.

splitted

[out] Array der resultierenden Submatrizen.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Wird die Anzahl der Submatrizen angegeben, so erhält man gleich große Teilmatrizen. Das bedeutet, dass die Anzahl der Spalten ohne Rest durch "parts" teilbar sein muss. Submatrizen unterschiedlicher Größe können mit Hilfe eines Arrays von Submatrixgrößen erhalten werden. Die Elemente des Arrays mit den Größen werden so lange verwendet, bis die gesamte Matrix aufgeteilt ist. Wenn das Array der Größen zu Ende ist und die Matrix noch nicht vollständig aufgeteilt wurde, ist der ungeteilte Rest die letzte Submatrix.

Beispiel

```
   matrix matrix_a={{ 1, 2, 3, 4, 5, 6},
                    { 7, 8, 9,10,11,12},
                    {13,14,15,16,17,18}};
   matrix splitted[];
   ulong  parts[]={2,3};

   matrix_a.Vsplit(2,splitted);
   for(uint i=0; i<splitted.Size(); i++)
      Print("splitted ",i,"\n",splitted[i]);

   matrix_a.Vsplit(3,splitted);
   for(uint i=0; i<splitted.Size(); i++)
      Print("splitted ",i,"\n",splitted[i]);
```

© 2000-2025, MetaQuotes Ltd.

```
 matrix_a.Vsplit(parts,splitted);
 for(uint i=0; i<splitted.Size(); i++)
    Print("splitted ",i,"\n",splitted[i]);


/*
   splitted 0
   [[1,2,3]
    [7,8,9]
    [13,14,15]]
   splitted 1
   [[4,5,6]
    [10,11,12]
    [16,17,18]]

   splitted 0
   [[1,2]
    [7,8]
    [13,14]]
   splitted 1
   [[3,4]
    [9,10]
    [15,16]]
   splitted 2
   [[5,6]
    [11,12]
    [17,18]]

   splitted 0
   [[1,2]
    [7,8]
    [13,14]]
   splitted 1
   [[3,4,5]
    [9,10,11]
    [15,16,17]]
   splitted 2
   [[6]
    [12]
    [18]]

*/
```

© 2000-2025, MetaQuotes Ltd.

ArgSort

Indirektes Sortieren einer Matrix oder Vektors.

```
vector vector::Sort(
  func_name  compare_func=NULL,  // Vergleichsfunktion
  T          context             // Parameter für die Nutzerfunktion zum Sortieren
   );

matrix matrix::Sort(
  func_name  compare_func=NULL   // Vergleichsfunktion
  T          context             // Parameter für die Nutzerfunktion zum Sortieren
   );

matrix matrix::Sort(
  const int  axis,               // Achse für die Sortierung
  func_name  compare_func=NULL   // Vergleichsfunktion
  T          context             // Parameter für die Nutzerfunktion zum Sortieren
   );
```

Parameter

axis

[in] Die Achse, nach der sortiert werden soll: 0 ist horizontal, 1 ist vertikal.

func_name

[in] Vergleichsfunktion. Sie können einen der Werte der Enumeration ENUM_SORT_MODE oder Ihre eigene Vergleichsfunktion angeben. Wenn keine Funktion angegeben wird, wird aufsteigend sortiert.

Eine eigene Vergleichsfunktion kann von zwei Typen sein:

- int comparator(T x1,T x2)
- int comparator(T x1,T x2,TContext context)

Hier ist T der Typ der Matrix oder des Vektors, und TContex ist der Typ der "Kontext"-Variablen, die als zusätzlicher Parameter an die Sortiermethode übergeben wird.

context

[in] Zusätzlicher optionaler Parameter, der an eine nutzerdefinierte Sortierfunktion übergeben werden kann.

Rückgabewert

Vektor oder Matrix mit den Indizes der sortierten Elemente. Das Ergebnis [4,2,0,1,3] bedeutet zum Beispiel, dass an der Position Null ein Element mit dem Index 4 stehen sollte, an der Position Eins ein Element mit dem Index 2 usw.

© 2000-2025, MetaQuotes Ltd.

Sort

Sortieren einer Matrix oder Vektors.

```
void vector::Sort(
  func_name  compare_func=NULL,  // Vergleichsfunktion
  T          context             // Parameter für die Nutzerfunktion zum Sortieren
   );

void matrix::Sort(
  func_name  compare_func=NULL   // Vergleichsfunktion
  T          context             // Parameter für die Nutzerfunktion zum Sortieren
   );

void matrix::Sort(
  const int  axis,               // Achse für die Sortierung
  func_name  compare_func=NULL   // Vergleichsfunktion
  T          context             // Parameter für die Nutzerfunktion zum Sortieren
   );
```

Parameter

axis

[in] Die Achse, nach der sortiert werden soll: 0 ist horizontal, 1 ist vertikal.

func_name

[in] Vergleichsfunktion. Sie können einen der Werte der Enumeration ENUM_SORT_MODE oder Ihre eigene Vergleichsfunktion angeben. Wenn keine Funktion angegeben wird, wird aufsteigend sortiert.

Eine eigene Vergleichsfunktion kann von zwei Typen sein:

- int comparator(T x1,T x2)
- int comparator(T x1,T x2,TContext context)

Hier ist T der Typ der Matrix oder des Vektors, und TContex ist der Typ der "Kontext"-Variablen, die als zusätzlicher Parameter an die Sortiermethode übergeben wird.

context

[in] Zusätzlicher optionaler Parameter, der an eine nutzerdefinierte Sortierfunktion übergeben werden kann.

Rückgabewert

Keiner. Die Sortierung erfolgt innerhalb, d. h. sie wird auf die Daten der Matrix/des Vektors angewendet, für die die Sortiermethode aufgerufen wird.

Beispiel

```
//+------------------------------------------------------------------+
//| Sortierfunktion                                                  |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int MyDoubleComparator(double x1,double x2,int sort_mode=0)
  {
   int res=x1<x2 ? -1 : (x1>x2 ? 1 : 0);
   return(sort_mode==0 ? res : -res);
  }
//+------------------------------------------------------------------+
//| Script start Funktion                                            |
//+------------------------------------------------------------------+
void OnStart()
  {
   //--- Ausfüllen des Vektors
   vector v(100);
   //--- aufsteigend sortieren
   v.Sort(MyDoubleComparator);   // ein zusätzlicher Parameter mit dem Standardwert "(
   Print(v);
   // absteigend sortieren
   v.Sort(MyDoubleComparator,1); // hier wird der zusätzliche Parameter '1' ausdrückli
   Print(v);
  }
```

© 2000-2025, MetaQuotes Ltd.

Mathematische Operationen mit Matritzen und Vektoren

Mathematische Operationen, einschließlich Addition, Subtraktion, Multiplikation und Division, können mit Matrizen und Vektoren elementweise durchgeführt werden.

Mathematische Funktionen wurden ursprünglich entwickelt, um relevante Operationen mit skalaren Werten durchzuführen. Die meisten der Funktionen können auf Matrizen und Vektoren angewendet werden. Dazu gehören MathAbs, MathArccos, MathArcsin, MathArctan, MathCeil, MathCos, MathExp, MathFloor, MathLog, MathLog10, MathMod, MathPow, MathRound, MathSin, MathSqrt, MathTan, MathExpm1, MathLog1p, MathArccosh, MathArcsinh, MathArctanh, MathCosh, MathSinh, und MathTanh. Diese Operationen implizieren eine elementweise Verarbeitung von Matrizen und Vektoren. Beispiel

```
//---
  matrix a= {{1, 4}, {9, 16}};
  Print("matrix a=\n",a);
  a=MathSqrt(a);
  Print("MatrSqrt(a)=\n",a);
  /*
   matrix a=
   [[1,4]
    [9,16]]
   MatrSqrt(a)=
   [[1,2]
    [3,4]]
  */
```

Für MathMod und MathPow kann das zweite Element entweder ein Skalar oder eine Matrix/ein Vektor der entsprechenden Größe sein.

Das folgende Beispiel zeigt, wie man die Standardabweichung durch Anwendung von mathematischen Funktionen auf einen Vektor berechnet.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- Verwendung der Initialisierungsfunktion zum Füllen des Vektors
  vector r(10, ArrayRandom); // Array mit Zufallszahlen von 0 bis 1
//--- berechnen des Mittelwerts
  double avr=r.Mean();       // Mittelwert des Arrays
  vector d=r-avr;            // Berechnen eines Arrays der Abweichungen vom Mittelwert
  Print("avr(r)=", avr);
  Print("r=", r);
  Print("d=", d);
  vector s2=MathPow(d, 2);   // Array der quadrierten Abweichungen
  double sum=s2.Sum();       // Summe der quadrierten Abweichungen
//--- Berechnen der Standardabweichung mit zwei verschiedenen Methoden
  double std=MathSqrt(sum/r.Size());
  Print(" std(r)=", std);
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("r.Std()=", r.Std());
 }
/*
   avr(r)=0.5300302133243813
   r=[0.8346201971495713,0.8031556138798182,0.6696676534318063,0.05386516922513505,0.54
   d=[0.30458998382519,0.2731254005554369,0.1396374401074251,-0.4761650440992462,0.0190
    std(r)=0.2838269732183663
   r.Std()=0.2838269732183663
*/
//+------------------------------------------------------------------+
//| Füllen des Vektors mit Zufallszahlen                             |
//+------------------------------------------------------------------+
void ArrayRandom(vector& v)
 {
  for(ulong i=0; i<v.Size(); i++)
    v[i]=double(MathRand())/32767.;
 }
```

© 2000-2025, MetaQuotes Ltd.

Mathematische Operationen

Mathematische Operationen, einschließlich Addition, Subtraktion, Multiplikation und Division, können mit Matrizen und Vektoren elementweise durchgeführt werden.

Beide Matrizen oder Vektoren müssen vom gleichen Typ sein und die gleichen Dimensionen haben. Jedes Element der Matrix wirkt auf das entsprechende Element der zweiten Matrix.

Sie können auch einen Skalar des entsprechenden Typs (double, float oder complex) als zweiten Term (Multiplikator, Subtrahend oder Divisor) verwenden. In diesem Fall wirkt jedes Element der Matrix oder des Vektors auf den angegebenen Skalar.

```
   matrix matrix_a={{0.1,0.2,0.3},{0.4,0.5,0.6}};
   matrix matrix_b={{1,2,3},{4,5,6}};

   matrix matrix_c1=matrix_a+matrix_b;
   matrix matrix_c2=matrix_b-matrix_a;
   matrix matrix_c3=matrix_a*matrix_b;    // Hadamard-Produkt, es darf nicht mit einem M
   matrix matrix_c4=matrix_b/matrix_a;

   matrix_c1=matrix_a+1;
   matrix_c2=matrix_b-double_value;
   matrix_c3=matrix_a*M_PI;
   matrix_c4=matrix_b/0.1;

//--- Operationen im Innern sind möglich
   matrix_a+=matrix_b;
   matrix_a/=2;
```

Die gleichen Operationen sind für Vektoren verfügbar.

© 2000-2025, MetaQuotes Ltd.

Mathematische Funktionen

Die folgenden mathematischen Funktionen können auf Matrizen und Vektoren angewendet werden: MathAbs, MathArccos, MathArcsin, MathArctan, MathCeil, MathCos, MathExp, MathFloor, MathLog, MathLog10, MathMod, MathPow, MathRound, MathSin, MathSqrt, MathTan, MathExpm1, MathLog1p, MathArccosh, MathArcsinh, MathArctanh, MathCosh, MathSinh, MathTanh. Diese Operationen implizieren eine elementweise Verarbeitung von Matrizen und Vektoren.

Bei MathMod und MathPow kann das zweite Element entweder ein Skalar oder eine Matrix/ein Vektor der entsprechenden Größe sein.

```
   matrix<T> mat1(128,128);
   matrix<T> mat3(mat1.Rows(),mat1.Cols());
   ulong     n,size=mat1.Rows()*mat1.Cols();
...
   mat2=MathPow(mat1,(T)1.9);
   for(n=0; n<size; n++)
     {
      T res=MathPow(mat1.Flat(n),(T)1.9);
      if(res!=mat2.Flat(n))
         errors++;
     }
   mat2=MathPow(mat1,mat3);
   for(n=0; n<size; n++)
     {
      T res=MathPow(mat1.Flat(n),mat3.Flat(n));
      if(res!=mat2.Flat(n))
         errors++;
     }
...
   vector<T> vec1(16384);
   vector<T> vec3(vec1.Size());
   ulong     n,size=vec1.Size();
...
   vec2=MathPow(vec1,(T)1.9);
   for(n=0; n<size; n++)
     {
      T res=MathPow(vec1[n],(T)1.9);
      if(res!=vec2[n])
         errors++;
     }
   vec2=MathPow(vec1,vec3);
   for(n=0; n<size; n++)
     {
      T res=MathPow(vec1[n],vec3[n]);
      if(res!=vec2[n])
         errors++;
     }
```

© 2000-2025, MetaQuotes Ltd.

Matrix- und Vektor-Produkte

Matrix- und Vektorproduktberechnungen umfassen:

- Matrix-Multiplikation
- Vektor-Multiplikation
- Berechnung der Kovarianzmatrix
- Berechnung der Kreuzkorrelation von zwei Vektoren
- Berechnung der Faltung von zwei Vektoren
- Berechnung des Korrelationskoeffizienten

Funktion	Aktion
MatMul	Matrixprodukt von zwei Matrizen.
GeMM	Generelle Matrix Multiplikation (GeMM)
Power	Quadratische Matrix mit einer ganzen Hochzahl potenzieren
Dot	Punktprodukt von zwei Vektoren.
Kron	Rückgabe des Kronecker-Produkts von zwei Matrizen, Matrix und Vektor, Vektor und Matrix oder zwei Vektoren.
Inner	Inneres Produkt von zwei Matrizen.
Outer	Berechnet das äußere Produkt von zwei Matrizen oder zwei Vektoren.
CorrCoef	Berechnet den Pearson-Korrelationskoeffizienten (linearer Korrelationskoeffizient).
Cov	Berechnet die Kovarianzmatrix.
Correlate	Berechnung der Kreuzkorrelation von zwei Vektoren.
Convolve	Liefert die diskrete, lineare Faltung (Convolution) von zwei Vektoren.

© 2000-2025, MetaQuotes Ltd.

MatMul

Die Methode MatMul, die die Multiplikation von Matrizen und Vektoren ermöglicht, hat mehrere Überladungen.

Multiplikation einer Matrix mit einer Matrix: `Matrix[M][K] * Matrix[K][N] = Matrix[M][N]`

```
matrix  matrix::MatMul(
  const matrix&  b      // zweite Matrix
   );
```

Multiplikation eines Vektors mit einem Vektor: `horizontaler Vektor[K] * Matrix[K][N] = horizontaler Vektor[N]`

```
vector  vector::MatMul(
  const matrix&  b      // Matrix
   );
```

Multiplikation einer Matrix mit einem Vektor: `Matrix[M][K] * vertikaler Vektor[K] = vertikaler Vektor[M]`

```
vector  matrix::MatMul(
  const vector&  b      // Vektor
   );
```

Skalare Vektormultiplikation: `horizontaler Vektor * vertikaler Vektor = Dot- Wert`

```
scalar  vector::MatMul(
  const vector&  b      // zweiter Vektor
   );
```

Parameter

b

[in] Matrix oder Vektor.

Rückgabewert

Matrix, Vektor oder Skalar, je nach der verwendeten Methode.

Hinweis

Die Matrizen sollten für die Multiplikation kompatibel sein, d. h. die Anzahl der Spalten in der ersten Matrix sollte gleich der Anzahl der Zeilen in der zweiten Matrix sein. Die Matrixmultiplikation ist nicht kommutativ: Das Ergebnis der Multiplikation der ersten Matrix mit der zweiten ist im allgemeinen Fall nicht gleich dem Ergebnis der Multiplikation der zweiten Matrix mit der ersten.

Das Matrixprodukt besteht aus allen möglichen Kombinationen von Skalarprodukten der Zeilenvektoren der ersten Matrix und der Spaltenvektoren der zweiten Matrix.

Bei der Skalarmultiplikation müssen die Vektoren die gleiche Länge haben.

Bei der Multiplikation eines Vektors mit einer Matrix muss die Länge des Vektors genau der Anzahl der Spalten der Matrix entsprechen.

© 2000-2025, MetaQuotes Ltd.

Naiver Algorithmus zur Matrixmultiplikation in MQL5:

```
matrix MatrixProduct(const matrix& matrix_a, const matrix& matrix_b)
  {
   matrix matrix_c;

   if(matrix_a.Cols()!=matrix_b.Rows())
      return(matrix_c);

   ulong M=matrix_a.Rows();
   ulong K=matrix_a.Cols();
   ulong N=matrix_b.Cols();
   matrix_c=matrix::Zeros(M,N);

   for(ulong m=0; m<M; m++)
      for(ulong k=0; k<K; k++)
         for(ulong n=0; n<N; n++)
            matrix_c[m][n]+=matrix_a[m][k]*matrix_b[k][n];

   return(matrix_c);
  }
```

Beispiel einer Matrix-Multiplikation

```
   matrix a={{1, 0, 0},
             {0, 1, 0}};
   matrix b={{4, 1},
             {2, 2},
             {1, 3}};
   matrix c1=a.MatMul(b);
   matrix c2=b.MatMul(a);
   Print("c1 = \n", c1);
   Print("c2 = \n", c2);
/*
   c1 =
   [[4,1]
    [2,2]]
   c2 =
   [[4,1,0]
    [2,2,0]
    [1,3,0]]
*/
```

Ein Beispiel für die Multiplikation eines horizontalen Vektors mit einer Matrix

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- erstellen einer 3x5 Matrix
   matrix m35;
   m35.Init(3, 5, Arange);
//---
   vector v3 = {1, 2, 3};
   Print("Product of horizontal vector v and matrix m[3,5]");
   Print("On the left, vector v3 = ", v3);
   Print("On the right, matrix m35 = \n", m35);
   Print("v3.MatMul(m35) = horizontal vector v[5] \n", v3.MatMul(m35));

  /* Ergebnis
    Product of horizontal vector v3 and matrix m[3,5]
    On the left, vector v3 = [1,2,3]
    On the right, matrix m35 =
    [[0,1,2,3,4]
     [5,6,7,8,9]
     [10,11,12,13,14]]
    v3.MatMul(m35) = horizontal vector v[5]
    [40,46,52,58,64]
   */
  }
//+------------------------------------------------------------------+
//|  Füllen der Matrix mit ansteigenden Werten                       |
//+------------------------------------------------------------------+
void Arange(matrix & m, double start = 0, double step = 1)
  {
//---
   ulong cols = m.Cols();
   ulong rows = m.Rows();
   double value = start;
   for(ulong r = 0; r < rows; r++)
     {
      for(ulong c = 0; c < cols; c++)
        {
         m[r][c] = value;
         value += step;
        }
     }
//---
  }
```

An example of how to multiply a matrix by a vertical vector

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- erstellen einer 3x5 Matrix
   matrix m35;
   m35.Init(3, 5, Arange);
//---
   Print("Product of matrix m[3,5] and vertical vector v[5]");
   vector v5 = {1,2,3,4,5};
   Print("On the left, m35 = \n",m35);
   Print("On the right v5 = ",v5);
   Print("m35.MatMul(v5) = vertical vector v[3] \n",m35.MatMul(v5));

  /* Ergebnis
   Product of matrix m[3,5] and vertical vector v[5]
   On the left, m35 =
   [[0,1,2,3,4]
    [5,6,7,8,9]
    [10,11,12,13,14]]
   On the right, v5 = [1,2,3,4,5]
   m35.MatMul(v5) = vertical vector v[3]
   [40,115,190]
   */
  }
//+------------------------------------------------------------------+
//|  Füllen der Matrix mit ansteigenden Werten                       |
//+------------------------------------------------------------------+
void Arange(matrix & m, double start = 0, double step = 1)
  {
//---
   ulong cols = m.Cols();
   ulong rows = m.Rows();
   double value = start;
   for(ulong r = 0; r < rows; r++)
     {
      for(ulong c = 0; c < cols; c++)
        {
         m[r][c] = value;
         value += step;
        }
     }
//---
  }
```

Ein Beispiel für das Skalarprodukt (dot) von Vektoren

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- Skalarprodukt eines horizontalen Vektors und eines vertikalen Vektors
   vector a= {1, 2, 3};  // horizontal vector
   vector b= {4, 5, 6};  // vertical vector
   Print("a = ", a);
   Print("b = ", b);
   Print("1) a.MatMul(b) = ", a.MatMul(b));
   //--- man sieht, dass die Methode Dot das gleiche Ergebnis liefert
   Print("2) a.Dot(b) = ", a.Dot(b));

  /* Ergebnis
   a = [1,2,3]
   b = [4,5,6]
   1) a.MatMul(b) = 32.0
   2) a.Dot(b) = 32.0
    */
  }
```

Siehe auch

Dot, GeMM

© 2000-2025, MetaQuotes Ltd.

GeMM

Die Methode GeMM (General Matrix Multiply) implementiert die allgemeine Multiplikation von zwei Matrizen. Die Operation ist definiert als C ← α A B + β C, wobei die Matrizen A und B optional transponiert werden können. Bei einer normalen Multiplikation von Matrizen AB (MatMul) wird angenommen, dass der Skalar *Alpha* gleich 1 und *Beta* gleich Null ist.

Der Hauptunterschied zwischen GeMM und MatMul in Bezug auf die Effizienz besteht darin, dass MatMul immer ein neues Matrix-/Vektorobjekt erzeugt, während GeMM mit einem bestehenden Matrixobjekt arbeitet und es nicht neu erzeugt. Wenn Sie also GeMM verwenden und den Speicher für die entsprechende Matrix im Voraus zuweisen, müssen Sie bei der Arbeit mit denselben Matrixgrößen keinen Speicher neu zuweisen. Dies kann ein wichtiger Vorteil von GeMM für Massenberechnungen sein, z. B. bei der Durchführung von Optimierungen in einem Strategietester oder beim Training eines neuronalen Netzes.

Ähnlich wie MatMul hat auch GeMM 4 Überladungen. Die Semantik der vierten Überladung wurde jedoch geändert, um die Multiplikation eines vertikalen Vektors mit einem horizontalen Vektor zu ermöglichen.

Bei einem bestehenden Matrix-/Vektor-Objekt ist es nicht notwendig, den Speicher vorab zuzuweisen. Der Speicher wird beim ersten GeMM-Aufruf zugewiesen und mit Nullen gefüllt.

Multiplikation einer Matrix mit einer Matrix: `Matrix C[M][N] = α * ( Matrix A[M][K] * Matrix B[K][N]) + β * matrix C[M][N]`

```
bool  matrix::GeMM(
  const matrix &A,     // erste Matrix
  const matrix &B,     // zweite Matrix
  double alpha,        // der Multiplikator alpha für das Produkt AB
  double beta,         // der Multiplikator beta für die Matrix C
  uint   flags         // eine Kombination der Werte von ENUM_GEMM (bitwise OR), die be
   );
```

Multiplikation eines Vektors mit einer Matrix: `Vektor C[N] = α * ( Vektor A[K] * Matrix B[K][N]) + β * Vektor C[N]`

```
bool  vector::GeMM(
  const vector &A,     // horizontaler Vektor
  const matrix &B,     // Matrix
  double alpha,        // der Multiplikator alpha für das Produkt AB
  double beta,         // der Multiplikator beta für den Vektor C
  uint   flags         // Wert der Enumeration ENUM_GEMM, der bestimmt, ob die Matrix A
   );
```

Multiplikation einer Matrix mit einem Vektor: `Vektor C[M] = α * ( Matrix A[M][K] * Vektor B[K] * ) + β * Vektor C[M]`

```
bool  vector::GeMM(
  const matrix &A,     // Matrix
  const vector &B,     // vertikaler Vektor
  double alpha,        // der Multiplikator alpha für das Produkt AB
  double beta,         // der Multiplikator beta für den Vektor C
```

© 2000-2025, MetaQuotes Ltd.

```
  uint   flags        // Wert der Enumeration ENUM_GEMM, der bestimmt, ob die Matrix
   );
```

Multiplikation eines Vektors mit einem Vektor: `Matrix C[M][N] = α * ( Vektor A[M] * Vektor B[N] * ) + β * matrix C[M][N]`. Diese Überladung gibt eine Matrix zurück, im Gegensatz zu MatMul, die einen Skalar zurückgibt.

```
bool  matrix::GeMM(
  const vector &A,     // erster Vektor
  const vector &B,     // zweiter Vektor
  double alpha,        // der Multiplikator alpha für das Produkt AB
  double beta,         // beta für die Matrix C
  uint   flags         // Wert der Enumeration ENUM_GEMM, der bestimmt, ob die Matrix
   );
```

Parameter

A

[in] Matrix oder Vektor.

B

[in] Matrix oder Vektor.

alpha

[in] Der Multiplikator alpha für das Produkt AB.

beta

[in] Der Multiplikator beta für die Ergebnismatrix C.

flags

[in] Der Wert der Enumeration ENUM_GEMM, der bestimmt, ob Matrizen A, B und C transponiert sind.

Rückgabewert

Gibt true bei Erfolg zurück oder false andernfalls.

ENUM_GEMM

Die Enumeration er Flags der Methoden von GeMM.

ID	Beschreibung
TRANSP_A	Verwendung der transponierten Matrix A
TRANSP_B	Verwendung der transponierten Matrix B
TRANSP_C	Verwendung der transponierten Matrix

Hinweis

© 2000-2025, MetaQuotes Ltd.

Als Parameter A und B können Matrizen und Vektoren vom Typ float, double und complex verwendet werden. Die Template-Varianten der GeMM-Methode sind wie folgt

```
bool matrix<T>::GeMM(const matrix<T> &A,const matrix<T> &B,T alpha,T beta,ulong flags)
bool matrix<T>::GeMM(const vector<T> &A,const vector<T> &B,T alpha,T beta,ulong flags)

bool vector<T>::GeMM(const vector<T> &A,const matrix<T> &B,T alpha,T beta,ulong flags)
bool vector<T>::GeMM(const matrix<T> &A,const vector<T> &B,T alpha,T beta,ulong flags)
```

Die allgemeine Matrixmultiplikationsfunktion wird im Wesentlichen wie folgt beschrieben:

```
C[m,n] = α *Sum(A[m,k]*B[k,n]) + β*C[m,n]
```

mit den folgenden Größen: Matrix A ist M x K, Matrix B ist K x N und Matrix C ist M x N.

Die Matrizen müssen also für die Multiplikation kompatibel sein, d. h. die Anzahl der Spalten der ersten Matrix muss gleich der Anzahl der Zeilen der zweiten Matrix sein. Die Matrixmultiplikation ist nicht kommutativ: Das Ergebnis der Multiplikation der ersten Matrix mit der zweiten ist im allgemeinen Fall nicht gleich dem Ergebnis der Multiplikation der zweiten Matrix mit der ersten.

Beispiel:

```
void OnStart()
  {
   vector vector_a= {1, 2, 3, 4, 5};
   vector vector_b= {4, 3, 2, 1};
   matrix matrix_c;
//--- GeMM für zwei Vektoren berechnen
   matrix_c.GeMM(vector_a, vector_b, 1, 0);
   Print("matrix_c:\n ", matrix_c, "\n");
   /*
   matrix_c:
    [[4,3,2,1]
    [8,6,4,2]
    [12,9,6,3]
    [16,12,8,4]
    [20,15,10,5]]
   */
//--- Erstellen von Matrizen als Vektoren
   matrix matrix_a(5, 1);
   matrix matrix_b(1, 4);
   matrix_a.Col(vector_a, 0);
   matrix_b.Row(vector_b, 0);
   Print("matrix_a:\n ", matrix_a);
   Print("matrix_b:\n ", matrix_b);
   /*
   matrix_a:
```

© 2000-2025, MetaQuotes Ltd.

```
   [[1]
   [2]
   [3]
   [4]
   [5]]
   matrix_b:
   [[4,3,2,1]]
   */
//-- GeMM für zwei Matrizen berechnen und das gleiche Ergebnis erhalten
   matrix_c.GeMM(matrix_a, matrix_b, 1, 0);
   Print("matrix_c:\n ", matrix_c);
   /*
   matrix_c:
    [[4,3,2,1]
    [8,6,4,2]
    [12,9,6,3]
    [16,12,8,4]
    [20,15,10,5]]
   */
  }
```

Siehe auch

MatMul

© 2000-2025, MetaQuotes Ltd.

Power

Erhöhen einer quadratischen Matrix auf die (ganzzahlige) Potenz.

```
matrix matrix::Power(
  const int  power      // Potenz
   );
```

Parameter

power

[in] Der Exponent kann jede ganze Zahl sein, positiv, negativ oder Null.

Rückgabewert

Matrix.

Hinweis

Die resultierende Matrix hat die gleiche Größe wie die ursprüngliche Matrix. Bei einer Potenz hoch 0 wird die Identitätsmatrix zurückgegeben. Die positive Potenz n bedeutet, dass die ursprüngliche Matrix n-mal mit sich selbst multipliziert wird. Die negative Potenz -n bedeutet, dass die ursprüngliche Matrix zunächst invertiert und dann die invertierte Matrix n-mal mit sich selbst multipliziert wird.

Ein einfacher Algorithmus zum Erhöhen einer Matrix auf eine Potenz in MQL5:

```
bool MatrixPower(matrix& c, const matrix& a, const int power)
  {
//--- die Matrix muss quadratisch sein
   if(a.Rows()!=a.Cols())
      return(false);
//--- die Größe er Ergebnismatrix ist genau gleich
   ulong  rows=a.Rows();
   ulong  cols=a.Cols();
   matrix result(rows,cols);
//--- bei einer Potenz Null wird die Identitätsmatrix zurückgegeben
   if(power==0)
      result.Identity();
   else
     {
      //--- bei einem negativen Wert wird die Matrix invertiert
      if(power<0)
        {
         matrix inverted=a.Inv();
         result=inverted;
         for(int i=-1; i>power; i--)
            result=result.MatMul(inverted);
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      else
        {
         result=a;
         for(int i=1; i<power; i++)
            result=result.MatMul(a);
        }
     }
//---
   c=result;
   return(true);
  }
```

MQL5 Beispiel:

```
  matrix i= {{0, 1}, {-1, 0}};
  Print("i:\n", i);

  Print("i.Power(3):\n", i.Power(3));

  Print("i.Power(0):\n", i.Power(0));

  Print("i.Power(-3):\n", i.Power(-3));

  /*
  i:
  [[0,1]
   [-1,0]]

  i.Power(3):
  [[0,-1]
   [1,0]]

  i.Power(0):
  [[1,0]
   [0,1]]

  i.Power(-3):
  [[0, -1]
   [1,0]]
  */
```

Python Beispiel:

```
import numpy as np
from numpy.linalg import matrix_power

# matrix equiv. of the imaginary unit
```

© 2000-2025, MetaQuotes Ltd.

```
i = np.array([[0, 1], [-1, 0]])
print("i:\n",i)

# should = -i
print("matrix_power(i, 3) :\n",matrix_power(i, 3) )

print("matrix_power(i, 0):\n",matrix_power(i, 0))

# should = 1/(-i) = i, but w/ f.p. elements
print("matrix_power(i, -3):\n",matrix_power(i, -3))

i:
 [[ 0  1]
 [-1  0]]

matrix_power(i, 3) :
 [[ 0 -1]
 [ 1  0]]

matrix_power(i, 0):
 [[1 0]
 [0 1]]

matrix_power(i, -3):
 [[ 0.  1.]
 [-1.  0.]]
```

© 2000-2025, MetaQuotes Ltd.

Dot

Punktprodukt von zwei Vektoren.

```
double vector::Dot(
  const vector&  b      // zweiter Vektor
   );
```

Parameter

b

[in] Vektor.

Rückgabewert

Skalar.

Hinweis

Das Punktprodukt von zwei Matrizen ist das Matrixprodukt matrix::MatMul().

Ein einfacher Algorithmus für das Skalarprodukt von Vektoren in MQL5:

```
double VectorDot(const vector& vector_a, const vector& vector_b)
  {
   double dot=0;

   if(vector_a.Size()==vector_b.Size())
     {
      for(ulong i=0; i<vector_a.Size(); i++)
         dot+=vector_a[i]*vector_b[i];
     }

   return(dot);
  }
```

MQL5 Beispiel:

```
   for(ulong i=0; i<rows; i++)
     {
      vector v1=a.Row(i);
      for(ulong j=0; j<cols; j++)
        {
         vector v2=b.Row(j);
         result[i][j]=v1.Dot(v2);
        }
     }
```

© 2000-2025, MetaQuotes Ltd.

Python Beispiel:

```
import numpy as np

a = [1, 0, 0, 1]
b = [4, 1, 2, 2]
print(np.dot(a, b))

>>> 6
```

© 2000-2025, MetaQuotes Ltd.

Kron

Rückgabe des Kronecker-Produkts von zwei Matrizen, Matrix und Vektor, Vektor und Matrix oder zwei Vektoren.

```
matrix matrix::Kron(
  const matrix&  b      // zweite Matrix
   );

matrix matrix::Kron(
  const vector&  b      // Vektor
   );

matrix vector::Kron(
  const matrix&  b      // Matrix
   );

matrix vector::Kron(
  const vector&  b      // zweiter Vektor
   );
```

Parameter

b

[in] zweite Matrix.

Rückgabewert

Matrix.

Hinweis

Das Kronecker-Produkt wird auch als Blockmatrix-Multiplikation bezeichnet.

Ein einfacher Algorithmus für das Kronecker-Produkt für zwei Matrizen in MQL5:

```
matrix MatrixKronecker(const matrix& matrix_a,const matrix& matrix_b)
  {
   ulong  M=matrix_a.Rows();
   ulong  N=matrix_a.Cols();
   ulong  P=matrix_b.Rows();
   ulong  Q=matrix_b.Cols();
   matrix matrix_c(M*P,N*Q);

   for(ulong m=0; m<M; m++)
      for(ulong n=0; n<N; n++)
         for(ulong p=0; p<P; p++)
```

© 2000-2025, MetaQuotes Ltd.

```
            for(ulong q=0; q<Q; q++)
               matrix_c[m*P+p][n*Q+q]=matrix_a[m][n] * matrix_b[p][q];

   return(matrix_c);
  }
```

MQL5 Beispiel:

```
   matrix a={{1,2,3},{4,5,6}};
   matrix b=matrix::Identity(2,2);
   vector v={1,2};

   Print(a.Kron(b));
   Print(a.Kron(v));

  /*
   [[1,0,2,0,3,0]
    [0,1,0,2,0,3]
    [4,0,5,0,6,0]
    [0,4,0,5,0,6]]

   [[1,2,2,4,3,6]
    [4,8,5,10,6,12]]
  */
```

Python Beispiel:

```
import numpy as np

A = np.arange(1,7).reshape(2,3)
B = np.identity(2)
V = [1,2]
print(np.kron(A, B))
print("")
print(np.kron(A, V))

[[1. 0. 2. 0. 3. 0.]
 [0. 1. 0. 2. 0. 3.]
 [4. 0. 5. 0. 6. 0.]
 [0. 4. 0. 5. 0. 6.]]

[[ 1  2  2  4  3  6]
 [ 4  8  5 10  6 12]]
```

© 2000-2025, MetaQuotes Ltd.

Inner

Inneres Produkt von zwei Matrizen.

```
matrix matrix::Inner(
  const matrix&  b      // zweite Matrix
   );
```

Parameter

b

[in] Matrix.

Rückgabewert

Matrix.

Hinweis

Das innere Produkt für zwei Vektoren ist das Punktprodukt der beiden Vektoren vector::Dot().

Ein einfacher Algorithmus für das innere Produkt von zwei Matrizen in MQL5:

```
bool MatrixInner(matrix& c, const matrix& a, const matrix& b)
  {
//--- die Spaltenzahlen müssen gleich sein
   if(a.Cols()!=b.Cols())
      return(false);
//--- die Größe der sich ergebenden Matrix hängt von der Anzahl der Vektoren in jeder
   ulong  rows=a.Rows();
   ulong  cols=b.Rows();
   matrix result(rows,cols);
//---
   for(ulong i=0; i<rows; i++)
     {
      vector v1=a.Row(i);
      for(ulong j=0; j<cols; j++)
        {
         vector v2=b.Row(j);
         result[i][j]=v1.Dot(v2);
        }
     }
//---
   c=result;
   return(true);
  }
```

MQL5 Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
   matrix a={{0,1,2},{3,4,5}};
   matrix b={{0,1,2},{3,4,5},{6,7,8}};
   matrix c=a.Inner(b);
   Print(c);
   matrix a1={{0,1,2}};
   matrix c1=a1.Inner(b);
   Print(c1);

  /*
  [[5,14,23]
  [14,50,86]]
  [[5,14,23]]
  */
```

Python Beispiel:

```
import numpy as np

A = np.arange(6).reshape(2, 3)
B = np.arange(9).reshape(3, 3)
A1= np.arange(3)
print(np.inner(A, B))
print("");
print(np.inner(A1, B))

import numpy as np

A = np.arange(6).reshape(2, 3)
B = np.arange(9).reshape(3, 3)
A1= np.arange(3)
print(np.inner(A, B))
print("");
print(np.inner(A1, B))
```

© 2000-2025, MetaQuotes Ltd.

Outer

Berechnet das äußere Produkt von zwei Matrizen oder zwei Vektoren.

```
matrix matrix::Outer(
  const matrix&  b      // zweite Matrix
   );

matrix vector::Outer(
  const vector&  b      // zweiter Vektor
   );
```

Parameter

b

[in] Matrix.

Rückgabewert

Matrix.

Hinweis

Das äußere Produkt ist wie das Kronecker-Produkt ebenfalls eine Blockmatrix- (und Vektor-) Multiplikation.

Ein einfacher Algorithmus für das äußere Produkt von zwei Matrizen in MQL5:

```
matrix MatrixOuter(const matrix& matrix_a, const matrix& matrix_b)
  {
//--- Die Größe der resultierenden Matrix hängt von den Matrixgrößen ab
   ulong  rows=matrix_a.Rows()*matrix_a.Cols();
   ulong  cols=matrix_b.Rows()*matrix_b.Cols();
   matrix matrix_c(rows,cols);
   ulong  cols_a=matrix_a.Cols();
   ulong  cols_b=matrix_b.Cols();
//---
   for(ulong i=0; i<rows; i++)
     {
      ulong row_a=i/cols_a;
      ulong col_a=i%cols_a;
      for(ulong j=0; j<cols; j++)
        {
         ulong row_b=j/cols_b;
         ulong col_b=j%cols_b;
         matrix_c[i][j]=matrix_a[row_a][col_a] * matrix_b[row_b][col_b];
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//---
   return(matrix_c);
  }
```

MQL5 Beispiel:

```
   vector vector_a={0,1,2,3,4,5};
   vector vector_b={0,1,2,3,4,5,6};
   Print("vector_a.Outer\n",vector_a.Outer(vector_b));
   Print("vector_a.Kron\n",vector_a.Kron(vector_b));

   matrix matrix_a={{0,1,2},{3,4,5}};
   matrix matrix_b={{0,1,2},{3,4,5},{6,7,8}};
   Print("matrix_a.Outer\n",matrix_a.Outer(matrix_b));
   Print("matrix_a.Kron\n",matrix_a.Kron(matrix_b));

  /*
   vector_a.Outer
   [[0,0,0,0,0,0,0]
    [0,1,2,3,4,5,6]
    [0,2,4,6,8,10,12]
    [0,3,6,9,12,15,18]
    [0,4,8,12,16,20,24]
    [0,5,10,15,20,25,30]]
   vector_a.Kron
   [[0,0,0,0,0,0,0,0,1,2,3,4,5,6,0,2,4,6,8,10,12,0,3,6,9,12,15,18,0,4,8,12,16,20,24,0,
   matrix_a.Outer
   [[0,0,0,0,0,0,0,0,0]
    [0,1,2,3,4,5,6,7,8]
    [0,2,4,6,8,10,12,14,16]
    [0,3,6,9,12,15,18,21,24]
    [0,4,8,12,16,20,24,28,32]
    [0,5,10,15,20,25,30,35,40]]
   matrix_a.Kron
   [[0,0,0,0,1,2,0,2,4]
    [0,0,0,3,4,5,6,8,10]
    [0,0,0,6,7,8,12,14,16]
    [0,3,6,0,4,8,0,5,10]
    [9,12,15,12,16,20,15,20,25]
    [18,21,24,24,28,32,30,35,40]]
   */
```

Python Beispiel:

```
import numpy as np
```

© 2000-2025, MetaQuotes Ltd.

```
A = np.arange(6)
B = np.arange(7)
print("np.outer")
print(np.outer(A, B))
print("np.kron")
print(np.kron(A, B))

A = np.arange(6).reshape(2, 3)
B = np.arange(9).reshape(3, 3)
print("np.outer")
print(np.outer(A, B))
print("np.kron")

np.outer
[[ 0  0  0  0  0  0  0]
 [ 0  1  2  3  4  5  6]
 [ 0  2  4  6  8 10 12]
 [ 0  3  6  9 12 15 18]
 [ 0  4  8 12 16 20 24]
 [ 0  5 10 15 20 25 30]]
np.kron
[ 0  0  0  0  0  0  0  0  1  2  3  4  5  6  0  2  4  6  8 10 12  0  3  6
  9 12 15 18  0  4  8 12 16 20 24  0  5 10 15 20 25 30]
np.outer
[[ 0  0  0  0  0  0  0  0  0]
 [ 0  1  2  3  4  5  6  7  8]
 [ 0  2  4  6  8 10 12 14 16]
 [ 0  3  6  9 12 15 18 21 24]
 [ 0  4  8 12 16 20 24 28 32]
 [ 0  5 10 15 20 25 30 35 40]]
np.kron
[[ 0  0  0  0  1  2  0  2  4]
 [ 0  0  0  3  4  5  6  8 10]
 [ 0  0  0  6  7  8 12 14 16]
 [ 0  3  6  0  4  8  0  5 10]
 [ 9 12 15 12 16 20 15 20 25]
 [18 21 24 24 28 32 30 35 40]]
```

© 2000-2025, MetaQuotes Ltd.

CorrCoef

Berechnung des Pearson-Korrelationskoeffizientens (linearer Korrelationskoeffizient).

```
matrix matrix::CorrCoef(
  const bool    rowvar=true  // Zeilen- oder Spalten-Vektoren von Beobachtungen
);

scalar vector::CorrCoef(
  const vector&  b           // zweiter Vektor
);
```

Rückgabewert

Korrelationskoeffizient des Produkt-Moments nach Pearson.

Hinweis

Der Korrelationskoeffizient liegt im Bereich [-1, 1].

Aufgrund der Fließkommarundung ist das resultierende Array möglicherweise nicht hermitisch, die Diagonalelemente sind möglicherweise nicht 1, und die Elemente erfüllen möglicherweise nicht die Ungleichung abs(a) <= 1. Die Real- und Imaginärteile werden auf das Intervall [-1, 1] abgeschnitten, um diese Situation zu verbessern, was jedoch im komplexen Fall nicht sehr hilfreich ist.

Ein einfacher Algorithmus zur Berechnung des Korrelationskoeffizienten von zwei Vektoren mit MQL5:

```
double VectorCorrelation(const vector& vector_x,const vector& vector_y)
  {
   ulong n=vector_x.Size()<vector_y.Size() ? vector_x.Size() : vector_y.Size();
   if(n<=1)
      return(0);

   ulong  i;
   double xmean=0;
   double ymean=0;
   for(i=0; i<n; i++)
     {
      if(!MathIsValidNumber(vector_x[i]))
         return(0);
      if(!MathIsValidNumber(vector_y[i]))
         return(0);
      xmean+=vector_x[i];
      ymean+=vector_y[i];
     }
   xmean/=(double)n;
```

© 2000-2025, MetaQuotes Ltd.

```
   ymean/=(double)n;

   double s=0;
   double xv=0;
   double yv=0;
   double t1=0;
   double t2=0;
//--- Berechnung
   s=0;
   for(i=0; i<n; i++)
     {
      t1=vector_x[i]-xmean;
      t2=vector_y[i]-ymean;
      xv+=t1*t1;
      yv+=t2*t2;
      s+=t1*t2;
     }
//--- check
   if(xv==0 || yv==0)
      return(0);
//--- Ergebnisrückgabe
   return(s/(MathSqrt(xv)*MathSqrt(yv)));
  }
```

MQL5 Beispiel:

```
   vectorf vector_a={1,2,3,4,5};
   vectorf vector_b={0,1,0.5,2,2.5};
   Print("vectors correlation ",vector_a.CorrCoef(vector_b));
//---
   matrixf matrix_a={{1,2,3,4,5},
                     {0,1,0.5,2,2.5}};
   Print("matrix rows correlation\n",matrix_a.CorrCoef());
   matrixf matrix_a2=matrix_a.Transpose();
   Print("transposed matrix cols correlation\n",matrix_a2.CorrCoef(false));
   matrixf matrix_a3={{1.0f, 2.0f, 3.0f, 4.0f, 5.0f},
                      {0.0f, 1.0f, 0.5f, 2.0f, 2.5f},
                      {0.1f, 1.0f, 2.0f, 1.0f, 0.3f}};
   Print("rows correlation\n",matrix_a3.CorrCoef());
   Print("cols correlation\n",matrix_a3.CorrCoef(false));

  /*
   vectors correlation 0.9149913787841797
   matrix rows correlation
   [[1,0.91499138]
    [0.91499138,1]]
   transposed matrix cols correlation
   [[1,0.91499138]
```

© 2000-2025, MetaQuotes Ltd.

```
    [0.91499138,1]]
   rows correlation
   [[1,0.91499138,0.08474271]
    [0.91499138,1,-0.17123166]
    [0.08474271,-0.17123166,1]]
   cols correlation
   [[1,0.99587059,0.85375023,0.91129309,0.83773589]
    [0.99587059,1,0.80295509,0.94491106,0.88385159]
    [0.85375023,0.80295509,1,0.56362146,0.43088508]
    [0.91129309,0.94491106,0.56362146,1,0.98827404]
    [0.83773589,0.88385159,0.43088508,0.98827404,1]]
   */
```

Python Beispiel:

```
import numpy as np
va=[1,2,3,4,5]
vb=[0,1,0.5,2,2.5]
print("vectors correlation")
print(np.corrcoef(va,vb))

ma=np.zeros((2,5))
ma[0,:]=va
ma[1,:]=vb
print("matrix rows correlation")
print(np.corrcoef(ma))
print("transposed matrix cols correlation")
print(np.corrcoef(np.transpose(ma),rowvar=False))
print("")

ma1=[[1,2,3,4,5],[0,1,0.5,2,2.5],[0.1,1,0.2,1,0.3]]
print("rows correlation\n",np.corrcoef(ma1))
print("cols correlation\n",np.corrcoef(ma1,rowvar=False))

transposed matrix cols correlation
[[1.         0.91499142]
 [0.91499142 1.        ]]

rows correlation
 [[1.         0.91499142 0.1424941 ]
 [0.91499142 1.         0.39657517]
 [0.1424941  0.39657517 1.        ]]
cols correlation
 [[1.         0.99587059 0.98226063 0.91129318 0.83773586]
 [0.99587059 1.         0.99522839 0.94491118 0.88385151]
 [0.98226063 0.99522839 1.         0.97234063 0.92527551]
 [0.91129318 0.94491118 0.97234063 1.         0.98827406]
 [0.83773586 0.88385151 0.92527551 0.98827406 1.        ]]
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Cov

Berechnung der Kovarianzmatrix.

```
matrix matrix::Cov(
  const bool     rowvar=true  // Zeilen- oder Spaltenvektoren der Beobachtungen
);

matrix matrix::Cov(
  const bool     rowvar,      // Zeilen- oder Spaltenvektoren der Beobachtungen
  const int      ddof         // delta degrees of freedom (Anzahl der Freiheitsgrade)
);

matrix vector::Cov(
  const vector& b             // zweiter Vektor
);

matrix vector::Cov(
  const vector& b,            // zweiter Vektor
  const int      ddof         // Anzahl der Freiheitsgrade
);
```

Parameter

rowwar

[in] Wenn rowvar true ist (Standardeinstellung), dann stellt jede Zeile eine Variable dar, mit Beobachtungen in den Spalten. Andernfalls wird die Beziehung umgedreht: Jede Spalte stellt eine Variable dar, während die Zeilen Beobachtungen enthalten.

b

[in] Zweiter Vektor der Beobachtungen.

ddof

[in] "Delta Degrees of Freedom" (Anzahl der Freiheitsgrade): Der für die Berechnung verwendete Divisor ist N - ddof, wobei N die Anzahl der Elemente darstellt. Standardmäßig ist ddof gleich 1.

Hinweis

Berechnung der Kovarianzmatrix.

Ein einfacher Algorithmus zur Berechnung der Kovarianzmatrix von zwei Vektoren mit MQL5:

```
bool VectorCovariation(const vector& vector_a,const vector& vector_b,matrix& matrix_c)
  {
   int i,j;
   int m=2;
   int n=(int)(vector_a.Size()<vector_b.Size()?vector_a.Size():vector_b.Size());
//--- Prüfungen
   if(n<=1)
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
   for(i=0; i<n; i++)
     {
      if(!MathIsValidNumber(vector_a[i]))
         return(false);
      if(!MathIsValidNumber(vector_b[i]))
         return(false);
     }
//---
   matrix matrix_x(2,n);
   matrix_x.Row(vector_a,0);
   matrix_x.Row(vector_b,1);
   vector t=vector::Zeros(m);
//--- Berechnung
   for(i=0; i<m; i++)
      for(j=0; j<n; j++)
         t[i]+=matrix_x[i][j]/double(n);
   for(i=0; i<m; i++)
      for(j=0; j<n; j++)
         matrix_x[i][j]-=t[i];
//--- syrk C=alpha*A^H*A+beta*C (beta=0 and not considered)
   matrix_c=matrix::Zeros(m,m);
   for(i=0; i<m; i++)
     {
      for(j=0; j<n; j++)
        {
         double v=matrix_x[i][j]/(n-1);
         for(int i_=i; i_<m; i_++)
            matrix_c[i][i_]+=v*matrix_x[i_][j];
        }
     }
//--- Herstellen der Symetrie
   for(i=0; i<m-1; i++)
      for(j=i+1; j<m; j++)
         matrix_c[j][i]=matrix_c[i][j];
//---
   return(true);
  }
```

MQL5 example:

```
   matrix matrix_a={{3,-2.1},{1.1,-1},{0.12,4.3}};
   Print("covariation cols\n",matrix_a.Cov(false));
   Print("covariation rows\n",matrix_a.Cov());

   vector vector_a=matrix_a.Col(0);
   vector vector_b=matrix_a.Col(1);
   Print("covariation vectors\n",vector_a.Cov(vector_b));
```

© 2000-2025, MetaQuotes Ltd.

```
/*
 covariation cols
 [[2.144133333333333,-4.286]
  [-4.286,11.71]]
 covariation rows
 [[13.005,5.355,-10.659]
  [5.355,2.205,-4.389]
  [-10.659,-4.389,8.736199999999998]]
 covariation vectors
 [[2.144133333333333,-4.286]
  [-4.286,11.71]]
*/
```

Python example:

```
import numpy as np
matrix_a=np.array([[3,-2.1],[1.1,-1],[0.12,4.3]])
matrix_c=np.cov(matrix_a,rowvar=False)
print("covariation cols\n",matrix_c)
matrix_c2=np.cov(matrix_a)
print("covariation rows\n",matrix_c2)

vector_a=matrix_a[:,0]
vector_b=matrix_a[:,1]
matrix_c3=np.cov(vector_a,vector_b)
print("covariation vectors\n",matrix_c3)

covariation cols
 [[ 2.14413333 -4.286     ]
 [-4.286      11.71      ]]
covariation rows
 [[ 13.005    5.355  -10.659 ]
 [  5.355    2.205   -4.389 ]
 [-10.659   -4.389    8.7362]]
covariation vectors
 [[ 2.14413333 -4.286     ]
 [-4.286      11.71      ]]
```

© 2000-2025, MetaQuotes Ltd.

Correlate

Berechnung der Kreuzkorrelation von zwei Vektoren.

```
vector vector::Correlate(
  const vector&           v,        // Vektor
  ENUM_VECTOR_CONVOLVE    mode      // Modus
   );
```

Parameter

v

[in] Zweiter Vektor.

mode

[in] Der Parameter 'mode' bestimmt den Berechnungsmodus der linearen Faltung. Wert aus der Enumeration ENUM_VECTOR_CONVOLVE.

Rückgabewert

Kreuz-Korrelation von zwei Vektoren.

Hinweis

Der Parameter 'mode' bestimmt den Berechnungsmodus der linearen Faltung.

Ein einfacher Algorithmus zur Berechnung des Korrelationskoeffizienten von zwei Vektoren mit MQL5:

```
vector VectorCrossCorrelationFull(const vector& a,const vector& b)
  {
   int    m=(int)a.Size();
   int    n=(int)b.Size();
   int    size=m+n-1;
   vector c=vector::Zeros(size);

   for(int i=0; i<n; i++)
      for(int i_=i; i_<i+m; i_++)
         c[i_]+=b[n-i-1]*a[i_-i];

   return(c);
  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
vector VectorCrossCorrelationSame(const vector& a,const vector& b)
  {
   int    m=(int)a.Size();
   int    n=(int)b.Size();
   int    size=MathMax(m,n);
```

© 2000-2025, MetaQuotes Ltd.

```
   vector c=vector::Zeros(size);

   for(int i=0; i<n; i++)
     {
      for(int i_=i; i_<i+m; i_++)
        {
         int k=i_-size/2+1;
         if(k>=0 && k<size)
            c[k]+=b[n-i-1]*a[i_-i];
        }
     }

   return(c);
  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
vector VectorCrossCorrelationValid(const vector& a,const vector& b)
  {
   int    m=(int)a.Size();
   int    n=(int)b.Size();
   int    size=MathMax(m,n)-MathMin(m,n)+1;
   vector c=vector::Zeros(size);

   for(int i=0; i<n; i++)
     {
      for(int i_=i; i_<i+m; i_++)
        {
         int k=i_-n+1;
         if(k>=0 && k<size)
            c[k]+=b[n-i-1]*a[i_-i];
        }
     }

   return(c);
  }
```

MQL5 Beispiel:

```
   vector a={1,2,3,4,5};
   vector b={0,1,0.5};

   Print("full\n",a.Correlate(b,VECTOR_CONVOLVE_FULL));
   Print("same\n",a.Correlate(b,VECTOR_CONVOLVE_SAME));
   Print("valid\n",a.Correlate(b,VECTOR_CONVOLVE_VALID));
   Print("full\n",b.Correlate(a,VECTOR_CONVOLVE_FULL));

  /*
```

© 2000-2025, MetaQuotes Ltd.

```
   full
   [0.5,2,3.5,5,6.5,5,0]
   same
   [2,3.5,5,6.5,5]
   valid
   [3.5,5,6.5]
   full
   [0,5,6.5,5,3.5,2,0.5]
  */
```

Python Beispiel:

```
import numpy as np
a=[1,2,3,4,5]
b=[0,1,0.5]

print("full\n",np.correlate(a,b,'full'))
print("same\n",np.correlate(a,b,'same'));
print("valid\n",np.correlate(a,b,'valid'));
print("full\n",np.correlate(b,a,'full'))

full
 [0.5 2.  3.5 5.  6.5 5.  0. ]
same
 [2.  3.5 5.  6.5 5. ]
valid
 [3.5 5.  6.5]
full
 [0.  5.  6.5 5.  3.5 2.  0.5]
```

© 2000-2025, MetaQuotes Ltd.

Convolve

Liefert die diskrete, lineare Faltung (Convolution) von zwei Vektoren.

```
vector vector::Convolve(
  const vector&            v,         // Vektor
  ENUM_VECTOR_CONVOLVE     mode       // Modus
   );
```

Parameter

v

[out] Zweiter Vektor.

mode

[in] Der Parameter 'mode' bestimmt den Berechnungsmodus der linearen Faltung aus ENUM_VECTOR_CONVOLVE.

Rückgabewert

Diskrete, lineare Faltung (convolution) von zwei Vektoren.

Ein einfacher Algorithmus zur Berechnung der Faltung von zwei Vektoren in MQL5:

```
vector VectorConvolutionFull(const vector& a,const vector& b)
  {
   if(a.Size()<b.Size())
      return(VectorConvolutionFull(b,a));

   int    m=(int)a.Size();
   int    n=(int)b.Size();
   int    size=m+n-1;
   vector c=vector::Zeros(size);

   for(int i=0; i<n; i++)
      for(int i_=i; i_<i+m; i_++)
         c[i_]+=b[i]*a[i_-i];

   return(c);
  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
vector VectorConvolutionSame(const vector& a,const vector& b)
  {
   if(a.Size()<b.Size())
      return(VectorConvolutionSame(b,a));

   int    m=(int)a.Size();
   int    n=(int)b.Size();
```

© 2000-2025, MetaQuotes Ltd.

```
   int    size=MathMax(m,n);
   vector c=vector::Zeros(size);

   for(int i=0; i<n; i++)
     {
      for(int i_=i; i_<i+m; i_++)
        {
         int k=i_-size/2+1;
         if(k>=0 && k<size)
            c[k]+=b[i]*a[i_-i];
        }
     }

   return(c);
  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
vector VectorConvolutionValid(const vector& a,const vector& b)
  {
   if(a.Size()<b.Size())
      return(VectorConvolutionValid(b,a));

   int    m=(int)a.Size();
   int    n=(int)b.Size();
   int    size=MathMax(m,n)-MathMin(m,n)+1;
   vector c=vector::Zeros(size);

   for(int i=0; i<n; i++)
     {
      for(int i_=i; i_<i+m; i_++)
        {
         int k=i_-n+1;
         if(k>=0 && k<size)
            c[k]+=b[i]*a[i_-i];
        }
     }

   return(c);
  }
```

MQL5 Beispiel:

```
  vector a= {1, 2, 3, 4, 5};
  vector b= {0, 1, 0.5};

  Print("full\n", a.Convolve(b, VECTOR_CONVOLVE_FULL));
  Print("same\n", a.Convolve(b, VECTOR_CONVOLVE_SAME));
```

© 2000-2025, MetaQuotes Ltd.

```
  Print("valid\n", a.Convolve(b, VECTOR_CONVOLVE_VALID));

  /*
   full
   [0,1,2.5,4,5.5,7,2.5]
   same
   [1,2.5,4,5.5,7]
   valid
   [2.5,4,5.5]
  */
```

Python Beispiel:

```
import numpy as np
a=[1,2,3,4,5]
b=[0,1,0.5]

print("full\n",np.convolve(a,b,'full'))
print("same\n",np.convolve(a,b,'same'));
print("valid\n",np.convolve(a,b,'valid'));


full
 [0.  1.  2.5 4.  5.5 7.  2.5]
same
 [1.  2.5 4.  5.5 7. ]
valid
 [2.5 4.  5.5]
```

© 2000-2025, MetaQuotes Ltd.

Matrix Transformationen

Die Matrixzerlegung kann in den folgenden Fällen verwendet werden:

- als Zwischenschritt beim Lösen linearer Gleichungssysteme
- bei der Matrixinversion
- bei der Berechnung von Determinanten
- bei der Ermittlung von Eigenwerten und Eigenvektoren einer Matrix
- bei der Berechnung von analytischen Funktionen von Matrizen
- bei der Anwendung der Methode der kleinsten Quadrate
- bei der numerischen Lösung von Differentialgleichungen

Je nach Problemstellung werden verschiedene Arten der Matrixzerlegung verwendet.

Funktion	Aktion
Cholesky	Berechnet die Cholesky-Zerlegung
Eig	Berechnet die Eigenwerte und rechten Eigenvektoren einer quadratischen Matrix.
EigVals	Berechnet die Eigenwerte einer allgemeinen Matrix.
LU	LU-Faktorisierung einer Matrix ist das Produkt aus einer unteren Dreiecksmatrix und einer oberen Dreiecksmatrix
LUP	LUP-Faktorisierung mit partieller Pivotisierung, die sich auf die LU-Zerlegung mit reinen Zeilenpermutationen bezieht: PA=LU
QR	Berechnen der qr-Faktorisierung einer Matrix.
SVD	Singulärwert-Zerlegung

© 2000-2025, MetaQuotes Ltd.

Cholesky

Berechnen der Cholesky-Zerlegung.

```
bool matrix::Cholesky(
  matrix&  L      // Matrix
   );
```

Parameter

L

[out] Untere Dreiecksmatrix.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Gibt die Cholesky-Zerlegung, L * L.H, der quadratischen Matrix a zurück, wobei L eine untere Dreiecksmatrix und .H der Operator für die konjugierte Transponierung ist (der die gewöhnliche Transponierung ist, wenn a reellwertig ist). a muss hermitesch sein (symmetrisch, wenn reellwertig) und positiv-definit. Es wird nicht geprüft, ob a hermitesch ist oder nicht. Außerdem werden nur die unteren Dreiecks- und Diagonalelemente von a verwendet. Nur L wird tatsächlich zurückgegeben.

Beispiel

```
matrix matrix_a= {{5.7998084, -2.1825367}, {-2.1825367, 9.85910595}};
matrix matrix_l;
Print("matrix_a\n", matrix_a);

matrix_a.Cholesky(matrix_l);
Print("matrix_l\n", matrix_l);
Print("check\n", matrix_l.MatMul(matrix_l.Transpose()));

/*
matrix_a
[[5.7998084,-2.1825367]
 [-2.1825367,9.85910595]]
matrix_l
[[2.408279136645086,0]
 [-0.9062640068544704,3.006291985133859]]
check
[[5.7998084,-2.1825367]
 [-2.1825367,9.85910595]]
*/
```

© 2000-2025, MetaQuotes Ltd.

Eig

Berechnung der Eigenwerte und der rechten Eigenvektoren einer Quadratmatrix.

```
bool matrix::Eig(
  matrix&  eigen_vectors,         // Matrix der Eigenvektoren
  vector&  eigen_values           // Vektor der Eigenwerte
   );
```

Komplexe Lösung von Eigenwerten und Eigenvektoren

```
bool matrix::Eig(
  matrix<complex>& eigen_vectors, // Matrix der Eigenvektoren
  vector<complex>& eigen_values   // Vektor der Eigenwerte
   );
```

Parameter

eigen_vectors

[out] Matrix der vertikalen Eigenvektoren.

eigen_values

[out] Vektor der Eigenwerte.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Wenn bei der Berechnung der Eigenwerte eine komplexe Lösung auftritt, wird die Berechnung abgebrochen und der Fehlercode auf 4019 (ERR_MATH_OVERFLOW) gesetzt. Verwenden Sie die komplexe Überladung der Eig-Methode, um eine vollständige Lösung im komplexen Raum zu ermitteln

Wenn ein komplexer Eigenwert einen Imaginäranteil gleich Null hat, dann ist das ein reeller Eigenwert. Dies ist im folgenden Beispiel zu sehen.

Beispiel:

```
void OnStart()
  {
   matrix matrix_a =
     {
        {-3.474589, 1.106384, -9.091977,-3.925227 },
        {-5.522139, 2.366887,-15.162351,-6.357512 },
        { 8.394926,-2.960067, 22.292115, 9.524129 },
        { 7.803242,-2.080287, 19.217706, 8.186645 }
     };

   matrix eigen_vectors;
```

© 2000-2025, MetaQuotes Ltd.

```
   vector eigen_values;

   bool res=matrix_a.Eig(eigen_vectors, eigen_values);
   Print("res=",res,"  error=",GetLastError());
   Print("Eigen vectors:\n",eigen_vectors, "\nEigen Values:\n",eigen_values);

//--- Überprüfung der Korrektheit von A * v = lambda * v
   int vectors=0;
   for(ulong n=0; n<eigen_values.Size(); n++)
     {
      vector eigen_vector=eigen_vectors.Col(n);
      vector vector_d1   =eigen_vector*eigen_values[n];
      vector vector_d2   =matrix_a.MatMul(eigen_vector);

      ulong errors=vector_d1.Compare(vector_d2,1e-13);
      if(errors==0)
         vectors++;
     }
   Print("vectors=",vectors);

//--- Komplexe Lösung
   matrix<complex> eigen_vectors_c;
   vector<complex> eigen_values_c;
   ResetLastError();
   res=matrix_a.Eig(eigen_vectors_c,eigen_values_c);
   Print("res=",res,"  error=",GetLastError());
   Print("Eigen vectors:\n",eigen_vectors_c, "\nEigen Values:\n",eigen_values_c);

//--- Überprüfung der Korrektheit von A * v = lambda * v
   matrixc matrix_c;
   matrix_c.Assign(matrix_a);
   vectors=0;
   for(ulong n=0; n<eigen_values_c.Size(); n++)
     {
      vectorc eigen_vector_c=eigen_vectors_c.Col(n);
      vectorc vector_c1     =eigen_vector_c*eigen_values_c[n];
      vectorc vector_c2     =matrix_c.MatMul(eigen_vector_c);

      ulong errors=vector_c1.Compare(vector_c2,1e-13);
      if(errors==0)
         vectors++;
     }
   Print("vectors=",vectors);
  }
/* Ergebnis
   res=true  error=4019
   Eigen vectors:
   [[0.2649667608713664]
    [0.4488818803991876]
```

© 2000-2025, MetaQuotes Ltd.

```
    [-0.6527335897527492]
    [-0.5497604331807768]]
   Eigen Values:
   [28.94158645962942]
   vectors=1
   res=true  error=0
   Eigen vectors:
   [[(0.2649667608713664,0),(0.2227392219204762,0.3745470492013296),(0.403285439089771
    [(0.4488818803991876,0),(-0.2613452530342438,-0.3685707727819327),(-0.396850612637
    [(-0.6527335897527492,0),(-0.3418479634708521,-0.3299830378162041),(-0.355302103111
    [(-0.5497604331807768,0),(0.523227993452828,0.3262508584080381),(0.351283559614366
   Eigen Values:
   [(28.94158645962942,0),(0.01022501104810897,0.02954980822331488),(0.010225011048108
   vectors=3
*/
```

© 2000-2025, MetaQuotes Ltd.

EigVals

Berechnung der Eigenwerte einer allgemeinen Matrix.

```
bool matrix::EigVals(
  vector&  eigen_values      // Vektor der Eigenwerte
   );
```

Parameter

eigen_values

[out] Vektor der rechten Eigenwerte.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Hinweis

Der einzige Unterschied zwischen EigVals und Eig besteht darin, dass EigVals keine Eigenvektoren berechnet, sondern nur die Eigenwerte berechnet.

© 2000-2025, MetaQuotes Ltd.

LU

LU-Faktorisierung einer Matrix ist das Produkt aus einer unteren Dreiecksmatrix und einer oberen Dreiecksmatrix

```
bool matrix::LU(
  matrix&  L,      // untere Dreiecksmatrix
  matrix&  U       // obere Dreiecksmatrix
   );
```

Parameter

L

[out] Untere Dreiecksmatrix.

U

[out] Obere Dreiecksmatrix.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Beispiel

```
   matrix matrix_a={{1,2,3,4},
                    {5,2,6,7},
                    {8,9,3,10},
                    {11,12,14,4}};
   matrix matrix_l,matrix_u;
//--- LU Zerlegung
   matrix_a.LU(matrix_l,matrix_u);
   Print("matrix_l\n",matrix_l);
   Print("matrix_u\n",matrix_u);
//--- Test, ob A = L * U
   Print("check\n",matrix_l.MatMul(matrix_u));


   /*
   matrix_l
   [[1,0,0,0]
    [5,1,0,0]
    [8,0.875,1,0]
    [11,1.25,0.5904761904761905,1]]
   matrix_u
   [[1,2,3,4]
    [0,-8,-9,-13]
    [0,0,-13.125,-10.625]
    [0,0,0,-17.47619047619047]]
   check
   [[1,2,3,4]
```

© 2000-2025, MetaQuotes Ltd.

```
    [5,2,6,7]
    [8,9,3,10]
    [11,12,14,4]]
  */
```

© 2000-2025, MetaQuotes Ltd.

LUP

LUP-Faktorisierung mit partieller Pivotisierung, die sich auf die LU-Zerlegung mit reinen Zeilenpermutationen bezieht: PA=LU

```
bool  LUP(
  matrix&  L,      // untere Dreiecksmatrix
  matrix&  U,      // obere Dreiecksmatrix
  matrix&  P       // Permutations-Matrix
   );
```

Parameter

L

[out] Untere Dreiecksmatrix.

U

[out] Obere Dreiecksmatrix.

P

[out] Permutations-Matrix

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Beispiel

```
   matrix matrix_a={{1,2,3,4},
                    {5,2,6,7},
                    {8,9,3,10},
                    {11,12,14,4}};
   matrix matrix_l,matrix_u,matrix_p;
//--- LUP Zerlegung
   matrix_a.LUP(matrix_l,matrix_u,matrix_p);
   Print("matrix_l\n",matrix_l);
   Print("matrix_u\n",matrix_u);
   Print("matrix_p\n",matrix_p);
//--- Test, ob P * A = L * U
   Print("P * A\n",matrix_p.MatMul(matrix_a));
   Print("L * U\n",matrix_l.MatMul(matrix_u));

   /*
   matrix_l
   [[1,0,0,0]
    [0.4545454545454545,1,0,0]
    [0.7272727272727273,-0.07894736842105282,1,0]
    [0.09090909090909091,-0.2631578947368421,-0.2262773722627738,1]]
   matrix_u
   [[11,12,14,4]
```

© 2000-2025, MetaQuotes Ltd.

```
   [0,-3.454545454545454,-0.3636363636363633,5.181818181818182]
   [0,0,-7.210526315789473,7.500000000000001]
   [0,0,0,6.697080291970803]]
  matrix_p
  [[0,0,0,1]
   [0,1,0,0]
   [0,0,1,0]
   [1,0,0,0]]
  P * A
  [[11,12,14,4]
   [5,2,6,7]
   [8,9,3,10]
   [1,2,3,4]]
  L * U
  [[11,12,14,4]
   [5,2,6,7]
   [8,9,3.000000000000001,10]
   [1,2,3,4]]
  */
```

© 2000-2025, MetaQuotes Ltd.

QR

Berechnen der qr-Faktorisierung einer Matrix.

```
bool  QR(
  matrix&  Q,      // Matrix mit orthonormalen Spalten
  matrix&  R       // obere Dreiecksmatrix
   );
```

Parameter

Q

[out] Eine Matrix mit orthonormalen Spalten. Bei mode = 'complete' ist das Ergebnis eine orthogonale/unitäre Matrix, je nachdem, ob a reel/komplex ist oder nicht. Die Determinante kann in diesem Fall entweder +/- 1 sein. Falls die Anzahl der Dimensionen im Eingabefeld größer als 2 ist, wird ein Stapel von Matrizen mit den oben genannten Eigenschaften zurückgegeben.

R

[out] Obere Dreiecksmatrix.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Beispiel

```
//---  A*x = b
  matrix A = {{0, 1}, {1, 1}, {1, 1}, {2, 1}};
  Print("A \n", A);
  vector b = {1, 2, 2, 3};
  Print("b \n", b);
//--- A = Q*R
  matrix q, r;
  A.QR(q, r);
  Print("q \n", q);
  Print("r \n", r);
  matrix qr=q.MatMul(r);
  Print("qr \n", qr);
  /*
  A
  [[0,1]
  [1,1]
  [1,1]
  [2,1]]
  b
  [1,2,2,3]
  q
  [[0.4082482904638631,-0.8164965809277259,-1.110223024625157e-16,-0.4082482904638631]
  [0.4625425214347352,-0.03745747856526496,0.7041241452319315,0.5374574785652647]
  [-0.5374574785652648,-0.03745747856526496,0.7041241452319316,-0.4625425214347352]
```

© 2000-2025, MetaQuotes Ltd.

```
[-0.5749149571305296,-0.5749149571305299,-0.09175170953613698,0.5749149571305296]]
r
[[-1.224744871391589,-0.2415816237971962]
[-1.22474487139159,-1.466326495188786]
[1.224744871391589,1.316496580927726]
[1.224744871391589,0.2415816237971961]]
qr
[[-1.110223024625157e-16,1]
[1,0.9999999999999999]
[1,1]
[2,1]]
*/
```

© 2000-2025, MetaQuotes Ltd.

SVD

Singulärwert-Zerlegung

```
bool matrix::SVD(
  matrix&  U,                    // Unitäre Matrix
  matrix&  V,                    // Unitäre Matrix
  vector&  singular_values       // Singulärwertvektor
   );
```

Parameter

U

[out] Unitäre Matrix der Ordnung m, bestehend aus linken singulären Vektoren.

V

[out] Einheitsmatrix der Ordnung n, bestehend aus rechten Singulärvektoren.

singular_values

[out] Singulärwerte

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false.

Beispiel

```
  matrix a= {{0, 1, 2, 3, 4, 5, 6, 7, 8}};
  a=a-4;
  Print("matrix a \n", a);
  a.Reshape(3, 3);
  matrix b=a;
  Print("matrix b \n", b);
//--- SVD-Zerlegung durchführen
  matrix U, V;
  vector singular_values;
  b.SVD(U, V, singular_values);
  Print("U \n", U);
  Print("V \n", V);
  Print("singular_values = ", singular_values);

// Prüf-Block
//--- U * Singulär-Diagonale * V = A
  matrix matrix_s;
  matrix_s.Diag(singular_values);
  Print("matrix_s \n", matrix_s);
  matrix matrix_vt=V.Transpose();
  Print("matrix_vt \n", matrix_vt);
  matrix matrix_usvt=(U.MatMul(matrix_s)).MatMul(matrix_vt);
  Print("matrix_usvt \n", matrix_usvt);
```

© 2000-2025, MetaQuotes Ltd.

```
  ulong errors=(int)b.Compare(matrix_usvt, 1e-9);
  double res=(errors==0);
  Print("errors=", errors);

//---- eine weitere Prüfung
  matrix U_Ut=U.MatMul(U.Transpose());
  Print("U_Ut \n", U_Ut);
  Print("Ut_U \n", (U.Transpose()).MatMul(U));

  matrix vt_V=matrix_vt.MatMul(V);
  Print("vt_V \n", vt_V);
  Print("V_vt \n", V.MatMul(matrix_vt));

  /*
  matrix a
  [[-4,-3,-2,-1,0,1,2,3,4]]
  matrix b
  [[-4,-3,-2]
   [-1,0,1]
   [2,3,4]]
  U
  [[-0.7071067811865474,0.5773502691896254,0.408248290463863]
   [-6.827109697437648e-17,0.5773502691896253,-0.8164965809277256]
   [0.7071067811865472,0.5773502691896255,0.4082482904638627]]
  V
  [[0.5773502691896258,-0.7071067811865474,-0.408248290463863]
   [0.5773502691896258,1.779939029415334e-16,0.8164965809277258]
   [0.5773502691896256,0.7071067811865474,-0.408248290463863]]
  singular_values = [7.348469228349533,2.449489742783175,3.277709923350408e-17]

  matrix_s
  [[7.348469228349533,0,0]
   [0,2.449489742783175,0]
   [0,0,3.277709923350408e-17]]
  matrix_vt
  [[0.5773502691896258,0.5773502691896258,0.5773502691896256]
   [-0.7071067811865474,1.779939029415334e-16,0.7071067811865474]
   [-0.408248290463863,0.8164965809277258,-0.408248290463863]]
  matrix_usvt
  [[-3.999999999999997,-2.999999999999999,-2]
   [-0.9999999999999981,-5.977974170712231e-17,0.9999999999999974]
   [2,2.999999999999999,3.999999999999996]]
  errors=0

  U_Ut
  [[0.9999999999999993,-1.665334536937735e-16,-1.665334536937735e-16]
   [-1.665334536937735e-16,0.9999999999999987,-5.551115123125783e-17]
   [-1.665334536937735e-16,-5.551115123125783e-17,0.9999999999999999]]
```

© 2000-2025, MetaQuotes Ltd.

```
  Ut_U
  [[0.9999999999999993,-5.551115123125783e-17,-1.110223024625157e-16]
   [-5.551115123125783e-17,0.9999999999999987,2.498001805406602e-16]
   [-1.110223024625157e-16,2.498001805406602e-16,0.999999999999999]]
  vt_V
  [[1,-5.551115123125783e-17,0]
   [-5.551115123125783e-17,0.9999999999999996,1.110223024625157e-16]
   [0,1.110223024625157e-16,0.9999999999999996]]
  V_vt
  [[0.9999999999999999,1.110223024625157e-16,1.942890293094024e-16]
   [1.110223024625157e-16,0.9999999999999998,1.665334536937735e-16]
   [1.942890293094024e-16,1.665334536937735e-16,0.9999999999999996]
  */
 }
```

© 2000-2025, MetaQuotes Ltd.

Statistische Methoden

Methoden zur Berechnung der deskriptiven Statistik von Matrizen und Vektoren.

Funktion	Aktion
ArgMax	Rückgabe des Index des größten Wertes.
ArgMin	Rückgabe des Index des kleinsten Wertes.
Max	Rückgabe des größten Wertes einer Matrix oder Vektors.
Min	Rückgabe des kleinsten Wertes einer Matrix oder Vektors.
Ptp	Gibt den Wertebereich einer Matrix bzw. Vektors oder der angegebenen Matrixachse zurück.
Sum	Rückgabe der Summe der Matrix-/Vektorelemente, die auch für die angegebene Achse (Achsen) durchgeführt werden kann.
Prod	Rückgabe des Produkts der Matrix-/Vektorelemente, das auch für die angegebene Achse durchgeführt werden kann.
CumSum	Rückgabe der kumulativen Summe der Matrix- oder Vektorelemente, einschließlich derjenigen entlang der angegebenen Achse.
CumProd	Rückgabe des kumulativen Produkts von Matrix- oder Vektorelementen, einschließlich der Elemente entlang der angegebenen Achse.
Percentile	Rückgabe des angegebenen Perzentils der Werte von Matrix- bzw. Vektorelementen oder den Elementen entlang der angegebenen Achse.
Quantile	Rückgabe des angegebenen Quantils der Werte von Matrix- bzw. Vektorelementen oder Elementen entlang der angegebenen Achse.
Median	Berechnung des Medians der Matrix- oder Vektorelemente.
Mean	Berechnung des arithmetischen Mittels der Elementwerte.
Average	Berechnung des gewichteten Mittelwerts von Matrix-/Vektorwerten.
Std	Rückgabe der Standardabweichung der Matrix-/Vektorwerte oder der Werte der Elemente entlang der angegebenen Achse.
Var	Berechnung der Varianz der Werte von Matrix- oder Vektorelementen.
LinearRegression	Berechnen eines Vektors/einer Matrix mit berechneten linearen Regressionswerten

© 2000-2025, MetaQuotes Ltd.

ArgMax

Rückgabe des Index des größten Wertes.

```
ulong vector::ArgMax();

ulong matrix::ArgMax();

vector matrix::ArgMax(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Index des Maximalwertes.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_max=matrix_a.ArgMax(0);
   vector rows_max=matrix_a.ArgMax(1);
   ulong  matrix_max=matrix_a.ArgMax();

   Print("cols_max=",cols_max);
   Print("rows_max=",rows_max);
   Print("max index ",matrix_max,"  max value ",matrix_a.Flat(matrix_max));

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_max=[0,3,1]
   rows_max=[0,2,0,1]
   max index 5  max value 12.0
   */
```

© 2000-2025, MetaQuotes Ltd.

ArgMin

Rückgabe des Index des kleinsten Wertes.

```
ulong vector::ArgMin();

ulong matrix::ArgMin();

vector matrix::ArgMin(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Index des kleinsten Wertes.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_min=matrix_a.ArgMin(0);
   vector rows_min=matrix_a.ArgMin(1);
   ulong  matrix_min=matrix_a.ArgMin();

   Print("cols_min=",cols_min);
   Print("rows_min=",rows_min);
   Print("min index ",matrix_min,"  min value ",matrix_a.Flat(matrix_min));

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_min=[1,0,0]
   rows_min=[2,0,2,0]
   min index 3  min value 1.0
   */
```

© 2000-2025, MetaQuotes Ltd.

Max

Rückgabe des größten Wertes einer Matrix oder Vektors.

```
double vector::Max();

double matrix::Max();

vector matrix::Max(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Maximum einer Matrix oder Vektors.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_max=matrix_a.Max(0);
   vector rows_max=matrix_a.Max(1);
   double matrix_max=matrix_a.Max();

   Print("cols_max=",cols_max);
   Print("rows_max=",rows_max);
   Print("max value ",matrix_max);

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_max=[10,11,12]
   rows_max=[10,12,6,11]
   max value 12.0
   */
```

© 2000-2025, MetaQuotes Ltd.

Min

Rückgabe des kleinsten Wertes einer Matrix oder Vektors.

```
double vector::Min();

double matrix::Min();

vector matrix::Min(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Minimum einer Matrix oder Vektors.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_min=matrix_a.Min(0);
   vector rows_min=matrix_a.Min(1);
   double matrix_min=matrix_a.Min();

   Print("cols_min=",cols_min);
   Print("rows_min=",rows_min);
   Print("min value ",matrix_min);

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_min=[1,3,2]
   rows_min=[2,1,4,7]
   min value 1.0
   */
```

© 2000-2025, MetaQuotes Ltd.

Ptp

Rückgabe des Wertebereichs einer Matrix/eines Vektors oder der angegebenen Matrixachse, entspricht Max() - Min(). Ptp - Peak to peak (Spitzenwert zu Spitzenwert).

```
double vector::Ptp();

double matrix::Ptp();

vector matrix::Ptp(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Vektor mit Wertebereichen (Maximum - Minimum).

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_ptp=matrix_a.Ptp(0);
   vector rows_ptp=matrix_a.Ptp(1);
   double matrix_ptp=matrix_a.Ptp();

   Print("cols_ptp  ",cols_ptp);
   Print("rows_ptp  ",rows_ptp);
   Print("ptp value  ",matrix_ptp);

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_ptp [9,8,10]
   rows_ptp [8,11,2,4]
   ptp value  11.0
   */
```

© 2000-2025, MetaQuotes Ltd.

Sum

Rückgabe der Summe der Matrix-/Vektorelemente, die auch für die angegebene Achse (Achsen) durchgeführt werden kann.

```
double vector::Sum();

double matrix::Sum();

vector matrix::Sum(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Die Summe der Matrix-/Vektorelemente, die auch für die angegebene Achse (Achsen) durchgeführt werden kann.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_sum=matrix_a.Sum(0);
   vector rows_sum=matrix_a.Sum(1);
   double matrix_sum=matrix_a.Sum();

   Print("cols_sum=",cols_sum);
   Print("rows_sum=",rows_sum);
   Print("sum value ",matrix_sum);

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_sum=[24,27,27]
   rows_sum=[15,21,15,27]
   sum value 78.0
   */
```

© 2000-2025, MetaQuotes Ltd.

Prod

Rückgabe des Produkts der Matrix-/Vektorelemente, das auch für die angegebene Achse durchgeführt werden kann.

```
double vector::Prod(
  const double  initial=1      // initialer Multiplikator
   );

double matrix::Prod(
  const double  initial=1      // initialer Multiplikator
   );

vector matrix::Prod(
  const int     axis,          // Achse
  const double  initial=1      // initialer Multiplikator
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

initial=1

[in] Initialer Multiplikator.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vector cols_prod=matrix_a.Prod(0);
   vector rows_prod=matrix_a.Prod(1);
   double matrix_prod=matrix_a.Prod();

   Print("cols_prod=",cols_prod);
   cols_prod=matrix_a.Prod(0,0.1);
   Print("cols_prod=",cols_prod);
   Print("rows_prod=",rows_prod);
   Print("prod value ",matrix_prod);

  /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_prod=[420,1320,864]
   cols_prod=[42,132,86.400000000000001]
```

© 2000-2025, MetaQuotes Ltd.

```
   rows_prod=[60,96,120,693]
   prod value 479001600.0
  */
```

© 2000-2025, MetaQuotes Ltd.

CumSum

Rückgabe der kumulativen Summe der Matrix- oder Vektorelemente, einschließlich derjenigen entlang der angegebenen Achse.

```
vector vector::CumSum();

vector matrix::CumSum();

matrix matrix::CumSum(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Kumulativen Summe der Elemente entlang der angegebenen Achse

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   matrix cols_cumsum=matrix_a.CumSum(0);
   matrix rows_cumsum=matrix_a.CumSum(1);
   vector cumsum_values=matrix_a.CumSum();

   Print("cols_cumsum\n",cols_cumsum);
   Print("rows_cumsum\n",rows_cumsum);
   Print("cumsum values  ",cumsum_values);

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_cumsum
   [[10,3,2]
    [11,11,14]
    [17,16,18]
    [24,27,27]]
   rows_cumsum
   [[10,13,15]
    [1,9,21]
```

© 2000-2025, MetaQuotes Ltd.

```
    [6,11,15]
    [7,18,27]]
   cumsum values  [10,13,15,16,24,36,42,47,51,58,69,78]
   */
```

© 2000-2025, MetaQuotes Ltd.

CumProd

Rückgabe des kumulativen Produkts von Matrix- oder Vektorelementen, einschließlich der Elemente entlang der angegebenen Achse.

```
vector vector::CumProd();

vector matrix::CumProd();

matrix matrix::CumProd(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse für jede Spalte (d. h. über die Zeilen), 1 — vertikale Achse für jede Zeile (d. h. über die Spalten)

Rückgabewert

Kumulatives Produkt der Elemente entlang der angegebenen Achse.

Beispiel

```
   matrix matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   matrix cols_cumprod=matrix_a.CumProd(0);
   matrix rows_cumprod=matrix_a.CumProd(1);
   vector cumprod_values=matrix_a.CumProd();

   Print("cols_cumprod\n",cols_cumprod);
   Print("rows_cumprod\n",rows_cumprod);
   Print("cumprod values  ",cumprod_values);

   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_cumprod
   [[10,3,2]
    [10,24,24]
    [60,120,96]
    [420,1320,864]]
   rows_cumprod
   [[10,30,60]
```

© 2000-2025, MetaQuotes Ltd.

```
    [1,8,96]
    [6,30,120]
    [7,77,693]]
   cumprod values  [10,30,60,60,480,5760,34560,172800,691200,4838400,53222400,47900160
   */
```

© 2000-2025, MetaQuotes Ltd.

Percentile

Rückgabe des angegebenen Perzentils der Werte von Matrix-/Vektorelementen oder Elementen entlang der angegebenen Achse.

```
double vector::Percentile(
  const int  percent      //
   );

double matrix::Percentile(
  const int  percent      //
   );

vector matrix::Percentile(
  const int  percent,     //
  const int  axis         // Achse
   );
```

Parameter

percent

[in] Zu berechnende Perzentile, die zwischen 0 und 100 einschließlich liegen müssen.

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

[in] Zu berechnende Perzentile, die zwischen 0 und 100 einschließlich liegen müssen.

Hinweis

Gültige Werte für den Parameter "percent" liegen im Bereich [0, 100]. Zur Berechnung der Perzentile wird ein linearer Algorithmus verwendet. Die korrekte Berechnung der Perzentile setzt eine sortierte Reihenfolge voraus.

Beispiel

```
   matrixf matrix_a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
   Print("matrix_a\n",matrix_a);

   vectorf cols_percentile=matrix_a.Percentile(50,0);
   vectorf rows_percentile=matrix_a.Percentile(50,1);
   float matrix_percentile=matrix_a.Percentile(50);

   Print("cols_percentile ",cols_percentile);
   Print("rows_percentile ",rows_percentile);
   Print("percentile value  ",matrix_percentile);
```

© 2000-2025, MetaQuotes Ltd.

```
/*
matrix_a
[[1,2,3]
 [4,5,6]
 [7,8,9]
 [10,11,12]]
cols_percentile [5.5,6.5,7.5]
rows_percentile [2,5,8,11]
percentile value  6.5
*/
```

© 2000-2025, MetaQuotes Ltd.

Quantile

Rückgabe des angegebenen Quantils der Werte von Matrix-/Vektorelementen oder Elementen entlang der angegebenen Achse zurück.

```
double vector::Quantile(
  const double quantile      // Quantil
   );

double matrix::Quantile(
  const double quantile      // Quantil
   );

vector matrix::Quantile(
  const double quantile,     // Quantil
  const int    axis          // Achse
   );
```

Parameter

quantil

[in] Zu berechnendes Quantil, das zwischen 0 und 1 einschließlich liegen muss.

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Quantil: Skalar oder Vektor.

Hinweis

Der Parameter "quantile" nimmt Werte im Bereich [0, 1] an. Zur Berechnung der Quantile wird ein linearer Algorithmus verwendet. Für die korrekte Berechnung der Quantile muss die Folge sortiert sein.

Beispiel

```
   matrixf matrix_a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
   Print("matrix_a\n",matrix_a);

   vectorf cols_quantile=matrix_a.Quantile(0.5,0);
   vectorf rows_quantile=matrix_a.Quantile(0.5,1);
   float matrix_quantile=matrix_a.Quantile(0.5);

   Print("cols_quantile ",cols_quantile);
   Print("rows_quantile ",rows_quantile);
   Print("quantile value  ",matrix_quantile);

   /*
```

© 2000-2025, MetaQuotes Ltd.

```
   matrix_a
   [[1,2,3]
    [4,5,6]
    [7,8,9]
    [10,11,12]]
   cols_quantile [5.5,6.5,7.5]
   rows_quantile [2,5,8,11]
   quantile value  6.5
   */
```

© 2000-2025, MetaQuotes Ltd.

Median

Berechnung des Medians der Matrix-/Vektor-Elemente.

```
double vector::Median();

double matrix::Median();

vector matrix::Median(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Median: Skalar oder Vektor.

Hinweis

Der Median ist der mittlere Wert, der die höchste Hälfte der Matrix-/Vektorelemente von der niedrigsten Hälfte der Elemente trennt. Dasselbe wie Quantil(0.5) und Perzentil(50). Die korrekte Berechnung der Perzentile setzt eine sortierte Reihenfolge voraus.

Beispiel

```
   matrixf matrix_a={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
   Print("matrix_a\n",matrix_a);

   vectorf cols_median=matrix_a.Median(0);
   vectorf rows_median=matrix_a.Median(1);
   float matrix_median=matrix_a.Median();

   Print("cols_median ",cols_median);
   Print("rows_median ",rows_median);
   Print("median value  ",matrix_median);


   /*
   matrix_a
   [[1,2,3]
    [4,5,6]
    [7,8,9]
    [10,11,12]]
   cols_median [5.5,6.5,7.5]
```

© 2000-2025, MetaQuotes Ltd.

```
   rows_median [2,5,8,11]
   median value  6.5
   */
```

© 2000-2025, MetaQuotes Ltd.

Mean

Berechnung des arithmetischen Mittels der Elementwerte.

```
double vector::Mean();

double matrix::Mean();

vector matrix::Mean(
  const int  axis      // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Arithmetischer Mittelwert der Elementwerte

Beispiel

```
   matrixf matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   Print("matrix_a\n",matrix_a);

   vectorf cols_mean=matrix_a.Mean(0);
   vectorf rows_mean=matrix_a.Mean(1);
   float matrix_mean=matrix_a.Mean();

   Print("cols_mean ",cols_mean);
   Print("rows_mean ",rows_mean);
   Print("mean value  ",matrix_mean);


   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_mean [6,6.75,6.75]
   rows_mean [5,7,5,9]
   mean value  6.5
   */
```

© 2000-2025, MetaQuotes Ltd.

Average

Berechnung des arithmetischen Mittels von Matrix-/Vektorwerten.

```
double vector::Average(
  const vector& weigts      // Gewichtsvektor
   );

double matrix::Average(
  const matrix& weigts      // Gewichtsmatrix
);

vector matrix::Average(
  const matrix& weigts,     // Gewichtsmatrix
  const int     axis        // Achse
   );
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Arithmetisches Mittel: skalar oder als Vektor

Hinweis

Die Gewichtsmatrix/-vektor ist mit der Hauptmatrix/-vektor assoziiert.

Beispiel

```
   matrixf matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
   matrixf matrix_w=matrixf::Ones(4,3);
   Print("matrix_a\n",matrix_a);

   vectorf cols_average=matrix_a.Average(matrix_w,0);
   vectorf rows_average=matrix_a.Average(matrix_w,1);
   float matrix_average=matrix_a.Average(matrix_w);

   Print("cols_average ",cols_average);
   Print("rows_average ",rows_average);
   Print("average value  ",matrix_average);


   /*
   matrix_a
   [[10,3,2]
```

© 2000-2025, MetaQuotes Ltd.

```
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_average [6,6.75,6.75]
   rows_average [5,7,5,9]
   average value  6.5
   */ value  6.5
```

© 2000-2025, MetaQuotes Ltd.

Std

Gibt die Standardabweichung der Werte von Matrix-/Vektor-Elementen oder von Elementen entlang der angegebenen Achse zurück.

```
double vector::Std();

double vector::Std(
  const int  ddof      // delta degrees of freedom (Anzahl der Freiheitsgrade)
);

double matrix::Std();

vector matrix::Std(
  const int  axis      // Achse
);

vector matrix::Std(
  const int  axis,     // Achse
  const int  ddof      // Anzahl der Freiheitsgrade
);
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

ddof

[in] "Delta Degrees of Freedom" (Anzahl der Freiheitsgrade): Der in der Berechnung verwendete Divisor ist N - ddof, wobei N die Anzahl der Elemente darstellt. Standardmäßig ist ddof gleich Null.

Rückgabewert

Standard_deviation: Skalar oder Vektor.

Hinweis

Die Standardabweichung ist die Quadratwurzel des Durchschnitts der quadrierten Abweichungen vom Mittelwert, d. h. std = sqrt(mean(x)), wobei x = abs(a - a.mean())**2.

Die durchschnittliche quadratische Abweichung wird typischerweise berechnet als x.sum() / (N - ddof), wobei N = len(x).

Der Ausdruck mit ddof=0 wird manchmal als „Standardabweichung der Grundgesamtheit" bezeichnet. Wenn ddof>0 (meist 1), wird die resultierende Größe manchmal als „Stichprobenstandardabweichung" bezeichnet.

Beispiel

```
   matrixf matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("matrix_a\n",matrix_a);

   vectorf cols_std=matrix_a.Std(0);
   vectorf rows_std=matrix_a.Std(1);
   float matrix_std=matrix_a.Std();

   Print("cols_std ",cols_std);
   Print("rows_std ",rows_std);
   Print("std value  ",matrix_std);


   /*
   matrix_a
   [[10,3,2]
    [1,8,12]
    [6,5,4]
    [7,11,9]]
   cols_std [3.2403703,3.0310888,3.9607449]
   rows_std [3.5590262,4.5460606,0.81649661,1.6329932]
   std value  3.452052593231201
   */
```

© 2000-2025, MetaQuotes Ltd.

Var

Berechnung der Varianz der Werte von Matrix-/Vektor-Elementen.

```
double vector::Var();

double vector::Var(
  const int  ddof      // delta degrees of freedom (Anzahl der Freiheitsgrade)
);

double matrix::Var();

vector matrix::Var(
  const int  axis      // Achse
);

vector matrix::Var(
  const int  axis,     // Achse
  const int  ddof      // Anzahl der Freiheitsgrade
);
```

Parameter

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

ddof

[in] "Delta Degrees of Freedom" (Anzahl der Freiheitsgrade): Der in der Berechnung verwendete Divisor ist N - ddof, wobei N die Anzahl der Elemente darstellt. Standardmäßig ist ddof gleich Null.

Rückgabewert

Varianz: Skalar oder Vektor.

Hinweis

Die Varianz ist der Durchschnitt der quadrierten Abweichungen vom Mittelwert, d. h. var = mean(x), wobei x = abs(a - a.mean())**2.

Der Mittelwert wird normalerweise als x.sum() / (N - ddof) berechnet, wobei N = len(x).

Der Ausdruck mit ddof=0 wird manchmal als „Varianz der Population" bezeichnet. Wenn ddof>0 (am häufigsten verwendet 1), wird die resultierende Größe manchmal als „Stichprobenvarianz" bezeichnet.

Beispiel

```
matrixf matrix_a={{10,3,2},{1,8,12},{6,5,4},{7,11,9}};
Print("matrix_a\n",matrix_a);
```

© 2000-2025, MetaQuotes Ltd.

```
vectorf cols_var=matrix_a.Var(0);
vectorf rows_var=matrix_a.Var(1);
float matrix_var=matrix_a.Var();

Print("cols_var ",cols_var);
Print("rows_var ",rows_var);
Print("var value  ",matrix_var);


/*
matrix_a
[[10,3,2]
 [1,8,12]
 [6,5,4]
 [7,11,9]]
cols_var [10.5,9.1875,15.6875]
rows_var [12.666667,20.666666,0.66666669,2.6666667]
var value  11.916666984558105
*/
```

© 2000-2025, MetaQuotes Ltd.

LinearRegression

Berechnen eines Vektors/einer Matrix mit berechneten linearen Regressionswerten.

```
vector vector::LinearRegression();

matrix matrix::LinearRegression(
  ENUM_MATRIX_AXIS  axis=AXIS_NONE      // die Achse, entlang der die Regression berec
   );
```

Parameter

axis

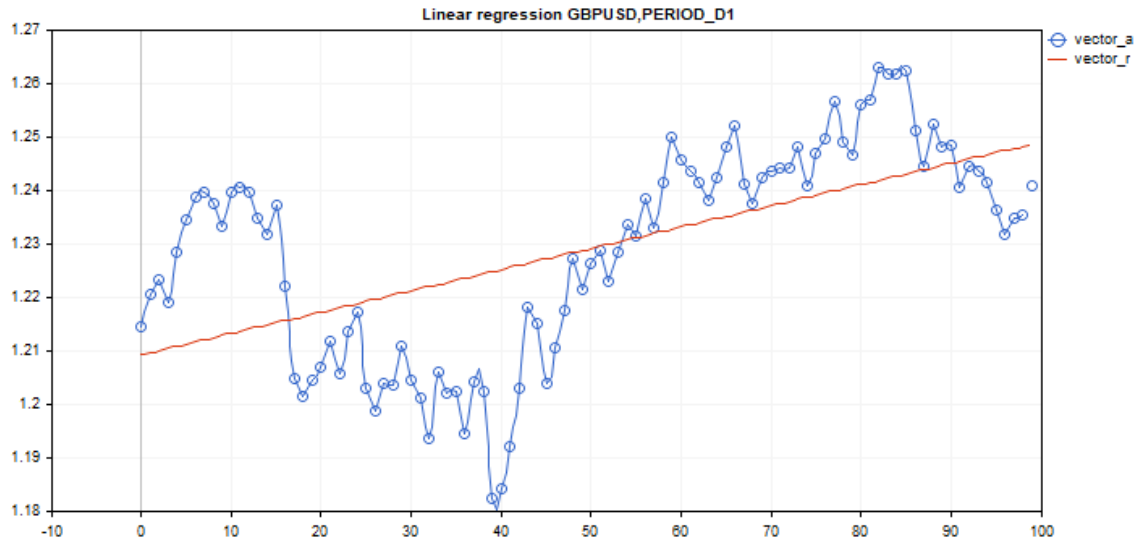
[in] Angabe der Achse, entlang derer die Regression berechnet wird. ENUM_MATRIX_AXIS Enumerationswerte (AXIS_HORZ — horizontale Achse, AXIS_VERT — vertikale Achse).

Rückgabewert

Vektor oder Matrix mit berechneten linearen Regressionswerten.

Hinweis

Die lineare Regression wird mit der Standardregressionsgleichung berechnet: y (x) = a * x + b, wobei a die Steigung der Geraden und b der Achsenabschnitt auf der Y-Achse ist.



Beispiel:

```
#include <Graphics\Graphic.mqh>

#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector vector_a;
   vector_a.CopyRates(_Symbol,_Period,COPY_RATES_CLOSE,1,100);
   vector vector_r=vector_a.LinearRegression();

//--- Darstellung auf dem Chart abschalten
   ChartSetInteger(0,CHART_SHOW,false);

//--- Arrays für das Zeichnen einer Grafik
   double x[];
   double y1[];
   double y2[];
   ArrayResize(x,uint(vector_a.Size()));
   ArrayResize(y1,uint(vector_a.Size()));
   ArrayResize(y2,uint(vector_a.Size()));
   for(ulong i=0; i<vector_a.Size(); i++)
     {
      x[i]=(double)i;
      y1[i]=vector_a[i];
      y2[i]=vector_r[i];
     }

//--- Titel der Grafik
   string title="Linear regression "+_Symbol+","+EnumToString(_Period);

   long   chart=0;
   string name="LinearRegression";

//--- Grafik erstellen
   CGraphic graphic;
   graphic.Create(chart,name,0,0,0,GRAPH_WIDTH,GRAPH_HEIGHT);
   graphic.BackgroundMain(title);
   graphic.BackgroundMainSize(12);

//--- Grafik der Aktivierungsfunktion
   CCurve *curvef=graphic.CurveAdd(x,y1,CURVE_POINTS_AND_LINES);
   curvef.Name("vector_a");
   curvef.LinesWidth(2);
   curvef.LinesSmooth(true);
   curvef.LinesSmoothTension(1);
   curvef.LinesSmoothStep(10);

//--- Ableitungen der Aktivierungsfunktion
   CCurve *curved=graphic.CurveAdd(x,y2,CURVE_LINES);
   curved.Name("vector_r");
   curved.LinesWidth(2);
```

© 2000-2025, MetaQuotes Ltd.

```
   curved.LinesSmooth(true);
   curved.LinesSmoothTension(1);
   curved.LinesSmoothStep(10);
   graphic.CurvePlotAll();
   graphic.Update();

//--- Endlosschleife zur Erkennung von gedrückten Tastaturtasten
   while(!IsStopped())
     {
      //--- zum Beenden des Programms die Escape-Taste drücken
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
         break;
      //--- PdDn drücken, um die Grafik zu speichern
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0)
        {
         string file_names[];
         if(FileSelectDialog("Save Picture",NULL,"All files (*.*)|*.*",FSD_WRITE_FILE,
            continue;
         ChartScreenShot(0,file_names[0],GRAPH_WIDTH,GRAPH_HEIGHT);
        }
      Sleep(10);
     }

//--- aufräumen
   graphic.Destroy();
   ObjectDelete(chart,name);
   ChartSetInteger(0,CHART_SHOW,true);
  }
```

ENUM_MATRIX_AXIS

Enumeration zur Angabe der Achse in allen statistischen Funktionen für Matrizen.

ID	Beschreibung
AXIS_NONE	Die Achse ist nicht angegeben. Die Berechnung erfolgt über alle Matrixelemente, als ob es sich um einen Vektor handeln würde (siehe die Methode Flat).
AXIS_HORZ	Horizontale Achse
AXIS_VERT	Vertikale Achse

© 2000-2025, MetaQuotes Ltd.

Methoden für die Eigenschaften

Diese Methoden ermöglichen die Abfrage von Matrixeigenschaften, wie z.B.:

- Anzahl der Zeilen
- Anzahl der Spalten
- Norm
- Bedingungszahl
- Determinante
- Rang der Matrix
- Spur
- Spektrum

Funktion	Aktion
Rows	Rückgabe der Anzahl der Zeilen einer Matrix.
Cols	Rückgabe der Anzahl der Spalten einer Matrix.
Size	Rückgabe der Größe des Vektors.
Norm	Rückgabe der Matrix- oder Vektornorm.
Cond	Berechnen der Konditionszahl einer Matrix.
Det	Berechnen der Determinante einer quadratischen invertierbaren Matrix.
SLogDet	Berechnet das Vorzeichen und den Logarithmus der Determinante einer Matrix.
Rank	Liefert den Rang der Matrix unter Verwendung der Gaußschen Methode.
Trace	Rückgabe der Summe entlang der Diagonalen der Matrix.
Spectrum	Berechnung des Spektrums einer Matrix als die Menge ihrer Eigenwerte aus dem Produkt AT*A.

© 2000-2025, MetaQuotes Ltd.

Rows

Rückgabe der Anzahl der Zeilen einer Matrix.

```
ulong matrix::Rows()
```

Rückgabewert

Integer.

Beispiel

```
matrix m= {{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}};
m.Reshape(3, 4);
Print("matrix m \n" , m);
Print("m.Rows()=", m.Rows());
Print("m.Cols()=", m.Cols());

/*
 matrix m
 [[1,2,3,4]
  [5,6,7,8]
  [9,10,11,12]]
 m.Rows()=3
 m.Cols()=4
*/
```

© 2000-2025, MetaQuotes Ltd.

Cols

Rückgabe der Anzahl der Spalten einer Matrix.

```
ulong matrix::Cols()
```

Rückgabewert

Integer.

Beispiel

```
matrix m= {{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}};
m.Reshape(3, 4);
Print("matrix m \n" , m);
Print("m.Cols()=", m.Cols());
Print("m.Rows()=", m.Rows());

/*
 matrix m
 [[1,2,3,4]
  [5,6,7,8]
  [9,10,11,12]]
 m.Cols()=4
 m.Rows()=3
*/
```

© 2000-2025, MetaQuotes Ltd.

Size

Rückgabe der Größe des Vektors.

```
ulong vector::Size()
```

Rückgabewert

Integer.

Beispiel

```
matrix m={{1,2,3,4,5,6,7,8,9,10,11,12}};
m.Reshape(3,4);
Print("matrix m\n",m);
vector v=m.Row(1);
Print("v.Size()=",v.Size());
Print("v=",v);


/*
 matrix m
 [[1,2,3,4]
  [5,6,7,8]
  [9,10,11,12]]
 v.Size()=4
 v=[5,6,7,8]
*/
```

© 2000-2025, MetaQuotes Ltd.

Norm

Rückgabe der Matrix- oder Vektornorm.

```
double vector::Norm(
  const ENUM_VECTOR_NORM  norm,     // Vektornorm
  const int               norm_p=2  // Zahl der P-Norm im Falles von VECTOR_NORM_P
   );

double matrix::Norm(
  const ENUM_MATRIX_NORM  norm      // Matrixnorm
   );
```

Parameter

Norm

[in] Art der Norm

Rückgabewert

Matrix- oder Vektornorm

Hinweis

- VECTOR_NORM_INF ist der maximale, absolute Wert unter den Vektorelementen.
- VECTOR_NORM_MINUS_INF ist der minimale, absolute Wert eines Vektors.
- VECTOR_NORM_P ist die P-Norm des Vektors. Wenn norm_p=0 ist, ist dies die Anzahl der Vektorelemente, die nicht Null sind. norm_p=1 ist die Summe der Absolutwerte der Vektorelemente. norm_p=2 ist die Quadratwurzel aus der Summe der Quadrate der Werte der Vektorelemente. P-Norm kann negativ sein.
- MATRIX_NORM_FROBENIUS ist die Quadratwurzel aus der Summe der Quadrate der Werte der Matrixelemente. Die Frobenius-Norm und die Vektor-P2-Norm sind konsistent.
- MATRIX_NORM_SPECTRAL ist der Maximalwert des Matrixspektrums.
- MATRIX_NORM_NUCLEAR ist die Summe der Singulärwerte der Matrix.
- MATRIX_NORM_INF ist die maximale p1-Norm des Vektors unter den vertikalen Vektoren der Matrix. Die Matrix-Inf-Norm und die Vektor-Inf-Norm sind konsistent.
- MATRIX_NORM_MINUS_INF ist die minimale p1-Norm des Vektors unter den vertikalen Vektoren der Matrix.
- MATRIX_NORM_P1 ist die p1-Norm des maximalen Vektors unter den horizontalen Matrixvektoren.
- MATRIX_NORM_MINUS_P1 ist die p1-Norm des minimalen Vektors unter den horizontalen Matrixvektoren.
- MATRIX_NORM_P2 ist der höchste Singulärwert der Matrix.
- MATRIX_NORM_MINUS_P2 ist der niedrigste Singulärwert einer Matrix.

Ein einfacher Algorithmus zur Berechnung der P-Norm eines Vektors in MQL5:

```
double VectorNormP(const vector& v,int norm_value)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   ulong  i;
   double norm=0.0;
//---
   switch(norm_value)
     {
      case 0 :
         for(i=0; i<v.Size(); i++)
            if(v[i]!=0)
               norm+=1.0;
         break;
      case 1 :
         for(i=0; i<v.Size(); i++)
            norm+=MathAbs(v[i]);
         break;
      case 2 :
         for(i=0; i<v.Size(); i++)
            norm+=v[i]*v[i];
         norm=MathSqrt(norm);
         break;
      default :
         for(i=0; i<v.Size(); i++)
            norm+=MathPow(MathAbs(v[i]),norm_value);
         norm=MathPow(norm,1.0/norm_value);
     }
//---
   return(norm);
  }
```

MQL5 Beispiel:

```
  matrix a= {{0, 1, 2, 3, 4, 5, 6, 7, 8}};
  a=a-4;
  Print("matrix a \n", a);
  a.Reshape(3, 3);
  matrix b=a;
  Print("matrix b \n", b);
  Print("b.Norm(MATRIX_NORM_P2)=", b.Norm(MATRIX_NORM_FROBENIUS));
  Print("b.Norm(MATRIX_NORM_FROBENIUS)=", b.Norm(MATRIX_NORM_FROBENIUS));
  Print("b.Norm(MATRIX_NORM_INF)", b.Norm(MATRIX_NORM_INF));
  Print("b.Norm(MATRIX_NORM_MINUS_INF)", b.Norm(MATRIX_NORM_MINUS_INF));
  Print("b.Norm(MATRIX_NORM_P1)=)", b.Norm(MATRIX_NORM_P1));
  Print("b.Norm(MATRIX_NORM_MINUS_P1)=", b.Norm(MATRIX_NORM_MINUS_P1));
  Print("b.Norm(MATRIX_NORM_P2)=", b.Norm(MATRIX_NORM_P2));
  Print("b.Norm(MATRIX_NORM_MINUS_P2)=", b.Norm(MATRIX_NORM_MINUS_P2));

  /*
  matrix a
  [[-4,-3,-2,-1,0,1,2,3,4]]
  matrix b
```

© 2000-2025, MetaQuotes Ltd.

```
  [[-4,-3,-2]
  [-1,0,1]
  [2,3,4]]
  b.Norm(MATRIX_NORM_P2)=7.745966692414834
  b.Norm(MATRIX_NORM_FROBENIUS)=7.745966692414834
  b.Norm(MATRIX_NORM_INF)9.0
  b.Norm(MATRIX_NORM_MINUS_INF)2.0
  b.Norm(MATRIX_NORM_P1)=)7.0
  b.Norm(MATRIX_NORM_MINUS_P1)=6.0
  b.Norm(MATRIX_NORM_P2)=7.348469228349533
  b.Norm(MATRIX_NORM_MINUS_P2)=1.857033188519056e-16
  */
```

Python Beispiel:

```
import numpy as np
from numpy import linalg as LA
a = np.arange(9) - 4
print("a \n",a)
b = a.reshape((3, 3))
print("b \n",b)
print("LA.norm(b)=",LA.norm(b))
print("LA.norm(b, 'fro')=",LA.norm(b, 'fro'))
print("LA.norm(b, np.inf)=",LA.norm(b, np.inf))
print("LA.norm(b, -np.inf)=",LA.norm(b, -np.inf))
print("LA.norm(b, 1)=",LA.norm(b, 1))
print("LA.norm(b, -1)=",LA.norm(b, -1))
print("LA.norm(b, 2)=",LA.norm(b, 2))
print("LA.norm(b, -2)=",LA.norm(b, -2))

a
 [-4 -3 -2 -1  0  1  2  3  4]
b
 [[-4 -3 -2]
 [-1  0  1]
 [ 2  3  4]]
LA.norm(b)= 7.745966692414834
LA.norm(b, 'fro')= 7.745966692414834
LA.norm(b, np.inf)= 9.0
LA.norm(b, -np.inf)= 2.0
LA.norm(b, 1)= 7.0
LA.norm(b, -1)= 6.0
LA.norm(b, 2)= 7.3484692283495345
LA.norm(b, -2)= 1.857033188519056e-16
```

© 2000-2025, MetaQuotes Ltd.

Cond

Berechnen der Konditionszahl einer Matrix.

```
double matrix::Cond(
  const ENUM_MATRIX_NORM  norm      // Matrixnorm
   );
```

Parameter

Norm

[in] Art der Norm aus ENUM_MATRIX_NORM

Rückgabewert

Die Konditionszahl der Matrix Kann unendlich sein.

Hinweis

Die Konditionszahl von x ist definiert als die Norm von x mal der Norm der Inversen von x [1]. Die Norm kann die übliche L2-Norm (Wurzel aus der Summe der Quadrate) oder eine von mehreren anderen Matrixnormen sein.

Die Konditionszahl ist der K-Wert, der dem Produkt aus der Norm der Matrix A und ihrer Inversen entspricht. Matrizen mit einer hohen Konditionszahl werden als schlecht konditioniert bezeichnet. Matrizen mit einer niedrigen Konditionszahl werden als gut konditioniert bezeichnet. Die inverse Matrix wird durch Pseudo-Inversion erhalten, um nicht durch die Kondition der Quadratur und Nicht-Singularität der Matrix eingeschränkt zu sein.

Eine Ausnahme bildet die spektrale Konditionszahl.

Ein einfacher Algorithmus zur Berechnung der spektralen Konditionszahl in MQL5:

```
double MatrixCondSpectral(matrix& a)
  {
   double norm=0.0;
   vector v=a.Spectrum();

   if(v.Size()>0)
     {
      double max_norm=v[0];
      double min_norm=v[0];
      for(ulong i=1; i<v.Size(); i++)
        {
         double real=MathAbs(v[i]);
         if(max_norm<real)
            max_norm=real;
         if(min_norm>real)
            min_norm=real;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
      max_norm=MathSqrt(max_norm);
      min_norm=MathSqrt(min_norm);
      if(min_norm>0.0)
         norm=max_norm/min_norm;
     }

   return(norm);
  }
```

MQL5 Beispiel:

```
matrix a= {{1, 0, -1}, {0, 1, 0}, { 1, 0, 1}};
Print("a.Cond(MATRIX_NORM_P2)=", a.Cond(MATRIX_NORM_P2));
Print("a.Cond(MATRIX_NORM_FROBENIUS)=", a.Cond(MATRIX_NORM_FROBENIUS));
Print("a.Cond(MATRIX_NORM_INF)=", a.Cond(MATRIX_NORM_INF));
Print("a.Cond(MATRIX_NORM_MINUS_INF)=", a.Cond(MATRIX_NORM_MINUS_INF));
Print("a.Cond(MATRIX_NORM_P1)=)", a.Cond(MATRIX_NORM_P1));
Print("a.Cond(MATRIX_NORM_MINUS_P1)=", a.Cond(MATRIX_NORM_MINUS_P1));
Print("a.Cond(MATRIX_NORM_P2)=", a.Cond(MATRIX_NORM_P2));
Print("a.Cond(MATRIX_NORM_MINUS_P2)=", a.Cond(MATRIX_NORM_MINUS_P2));

/*
matrix a
[[1,0,-1]
[0,1,0]
[1,0,1]]
a.Cond(MATRIX_NORM_P2)=1.414213562373095
a.Cond(MATRIX_NORM_FROBENIUS)=3.162277660168379
a.Cond(MATRIX_NORM_INF)=2.0
a.Cond(MATRIX_NORM_MINUS_INF)=0.9999999999999997
a.Cond(MATRIX_NORM_P1)=)2.0
a.Cond(MATRIX_NORM_MINUS_P1)=0.9999999999999998
a.Cond(MATRIX_NORM_P2)=1.414213562373095
a.Cond(MATRIX_NORM_MINUS_P2)=0.7071067811865472
*/
```

Python Beispiel:

```
import numpy as np
from numpy import linalg as LA
a = np.array([[1, 0, -1], [0, 1, 0], [1, 0, 1]])
print("a \n",a)
print("LA.cond(a)=",LA.cond(a))
print("LA.cond(a, 'fro')=",LA.cond(a, 'fro'))
print("LA.cond(a, np.inf)=",LA.cond(a, np.inf))
print("LA.cond(a, -np.inf)=",LA.cond(a, -np.inf))
```

© 2000-2025, MetaQuotes Ltd.

```
print("LA.cond(a, 1)=",LA.cond(a, 1))
print("LA.cond(a, -1)=",LA.cond(a, -1))
print("LA.cond(a, 2)=",LA.cond(a, 2))
print("LA.cond(a, -2)=",LA.cond(a, -2))

a
 [[ 1  0 -1]
 [ 0  1  0]
 [ 1  0  1]]
LA.cond(a)= 1.4142135623730951
LA.cond(a, 'fro')= 3.1622776601683795
LA.cond(a, np.inf)= 2.0
LA.cond(a, -np.inf)= 1.0
LA.cond(a, 1)= 2.0
LA.cond(a, -1)= 1.0
LA.cond(a, 2)= 1.4142135623730951
LA.cond(a, -2)= 0.7071067811865475
```

© 2000-2025, MetaQuotes Ltd.

Det

Berechnen der Determinante einer quadratischen invertierbaren Matrix.

```
double matrix::Det()
```

Rückgabewert

Determinante einer Matrix

Hinweis

Die Determinanten der Matrizen 2. und 3. Ordnung werden nach der Sarrus-Regel berechnet. d2=a11*a22-a12*a21; d3=a11*a22*a33+a12*a23*a31+a13*a21*a32-a13*a22*a31-a11*a23*a32-a12*a21*a33

Die Determinante wird nach der Gaußschen Methode durch Reduktion der Matrix in eine obere Dreiecksform berechnet. Die Determinante einer oberen Dreiecksmatrix ist gleich dem Produkt der Hauptdiagonalelemente.

Wenn mindestens eine Zeile oder Spalte der Matrix Null ist, ist die Determinante Null.

Wenn zwei oder mehr Zeilen oder Spalten der Matrix linear abhängig sind, ist die Determinante gleich Null.

Die Determinante einer Matrix ist gleich dem Produkt ihrer Eigenwerte.

MQL5 Beispiel:

```
matrix m={{1,2},{3,4}};
double det=m.Det();
Print("matrix m\n",m);
Print("det(m)=",det);
/*
matrix m
[[1,2]
 [3,4]]
det(m)=-2.0
*/
```

Python Beispiel:

```
import numpy as np

a = np.array([[1, 2], [3, 4]])
print('a \n',a)
print('nnp.linalg.det(a) \n',np.linalg.det(a))

a
```

© 2000-2025, MetaQuotes Ltd.

```
 [[1 2]
 [3 4]]

np.linalg.det(a)
 -2.0000000000000004
```

© 2000-2025, MetaQuotes Ltd.

SLogDet

Berechnet das Vorzeichen und den Logarithmus der Determinante einer Matrix.

```
double matrix::SLogDet(
  int&  sign      // Vorzeichen
   );
```

Parameter

sign

out] Das Vorzeichen der Determinante. Wenn das Vorzeichen gerade ist, ist die Determinante positiv.

Rückgabewert

Eine Zahl, die das Vorzeichen der Determinante angibt.

Hinweis

Die Determinante wird nach der Gaußschen Methode durch Reduktion der Matrix in eine obere Dreiecksform berechnet. Die Determinante einer oberen Dreiecksmatrix ist gleich dem Produkt der Hauptdiagonalelemente. Der Logarithmus eines Produkts ist gleich der Summe der Logarithmen. Daher kann man im Falle eines Überlaufs bei der Berechnung der Determinante die Methode SLogDet verwenden.

Wenn das Vorzeichen gerade ist, ist die Determinante positiv.

Beispiel

```
a = np.array([[1, 2], [3, 4]])  (sign, logdet) = np.linalg.slogdet(a)  (sign, logdet)
```

© 2000-2025, MetaQuotes Ltd.

Rank

Liefert den Rang der Matrix unter Verwendung der Gaußschen Methode.

```
int  Rank()
```

Rückgabewert

Rang der Matrix.

Hinweis

Der Rang eines Systems von Zeilen (oder Spalten) einer Matrix A mit m Zeilen und n Spalten ist die maximale Anzahl von linear unabhängigen Zeilen (oder Spalten). Mehrere Zeilen (Spalten) heißen linear unabhängig, wenn keine von ihnen linear durch die anderen ausgedrückt werden kann. Der Rang des Zeilensystems ist immer gleich dem Rang des Spaltensystems. Dieser Wert wird als Rang der Matrix bezeichnet.

MQL5 Beispiel:

```
matrix a=matrix::Eye(4, 4);;
Print("matrix a \n", a);
Print("a.Rank()=", a.Rank());

matrix I=matrix::Eye(4, 4);
I[3, 3] = 0.;     // rank deficient matrix
Print("I \n", I);
Print("I.Rank()=", I.Rank());

matrix b=matrix::Ones(1, 4);
Print("b \n", b);
Print("b.Rank()=", b.Rank());;// 1 Dimension - rank 1 außer alle sind 0

matrix  zeros=matrix::Zeros(4, 1);
Print("zeros \n", zeros);
Print("zeros.Rank()=", zeros.Rank());

/*
matrix a
[[1,0,0,0]
[0,1,0,0]
[0,0,1,0]
[0,0,0,1]]
a.Rank()=4

I
[[1,0,0,0]
[0,1,0,0]
```

© 2000-2025, MetaQuotes Ltd.

```
  [0,0,1,0]
  [0,0,0,0]]
  I.Rank()=3

  b
  [[1,1,1,1]]
  b.Rank()=1

  zeros
  [[0]
  [0]
  [0]
  [0]]
  zeros.Rank()=0
  */
```

Python Beispiel:

```
import numpy as np
from numpy.linalg import matrix_rank
a=(np.eye(4)) # Full rank matrix
print("a \n", a)
print("matrix_rank(a)=",matrix_rank(a))
I=np.eye(4)
I[-1,-1] = 0. # rank deficient matrix
print("I \n",I)
print("matrix_rank(I)=",matrix_rank(I))

b=np.ones((4,))
print("b \n",b)
print("matrix_rank(b)=",matrix_rank(b)) # 1 Dimension - Rang 1 außer alle sind 0

zeros=np.zeros((4,))
print("zeroes \n",zeros)
print("matrix_rank(zeros)=",matrix_rank(zeros))

a
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
matrix_rank(a)= 4

I
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 0.]]
```

© 2000-2025, MetaQuotes Ltd.

```
matrix_rank(I)= 3

b
 [1. 1. 1. 1.]
matrix_rank(b)= 1

zeroes
 [0. 0. 0. 0.]
matrix_rank(zeros)= 0
```

© 2000-2025, MetaQuotes Ltd.

Trace

Rückgabe der Summe entlang der Diagonalen der Matrix.

```
double matrix::Trace()
```

Rückgabewert

Die Summe der Diagonalen der Matrix.

Hinweis

Die Spur einer Matrix ist gleich der Summe ihrer Eigenwerte.

MQL5 Beispiel:

```
  matrix a= {{0, 1, 2, 3, 4, 5, 6, 7, 8}};
  a.Reshape(3, 3);
  Print("matrix a \n", a);
  Print("a.Trace() \n", a.Trace());

  /*
  matrix a
  [[0,1,2]
  [3,4,5]
  [6,7,8]]
  a.Trace()
  12.0
  */
```

Python Beispiel:

```
a = np.arange(9).reshape((3,3))
print('a \n',a)
print('np.trace(a) \n',np.trace(a))

a
 [[0 1 2]
 [3 4 5]
 [6 7 8]]

np.trace(a)
 12
```

© 2000-2025, MetaQuotes Ltd.

Spectrum

Berechnung des Spektrums einer Matrix als die Menge ihrer Eigenwerte aus dem Produkt AT*A.

```
vector matrix::Spectrum()
```

Rückgabewert

Spektrums einer Matrix als ein Vektor der Eigenwerte einer Matrix.

Beispiel

```
double MatrixCondSpectral(matrix& a)
  {
   double norm=0.0;
   vector v=a.Spectrum();

   if(v.Size()>0)
     {
      double max_norm=v[0];
      double min_norm=v[0];
      for(ulong i=1; i<v.Size(); i++)
        {
         double real=MathAbs(v[i]);
         if(max_norm<real)
            max_norm=real;
         if(min_norm>real)
            min_norm=real;
        }
      max_norm=MathSqrt(max_norm);
      min_norm=MathSqrt(min_norm);
      if(min_norm>0.0)
         norm=max_norm/min_norm;
     }

   return(norm);
  }
```

© 2000-2025, MetaQuotes Ltd.

Matrixmethoden zur Lösung von linearen Gleichungssystemen

Methoden zum Lösen von linearen Gleichungssystemen und zur Berechnung der inversen Matrix.

Funktion	Aktion
Solve	Lösen einer linearen Matrixgleichung oder eines Systems linearer algebraischer Gleichungen.
LstSq	Rückgabe der Lösung der kleinsten Quadrate von linearen algebraischen Gleichungen (für nicht quadratische oder entartete Matrizen).
Inv	Berechnung der (multiplikativen) Inversen einer quadratischen, nicht entarteten Matrix nach der Jordan-Gauss-Methode.
PInv	Berechnung der Pseudoinverse einer Matrix nach der Moore-Penrose-Methode.

© 2000-2025, MetaQuotes Ltd.

Solve

Lösen einer linearen Matrixgleichung oder eines Systems linearer algebraischer Gleichungen.

```
vector matrix::Solve(
  const vector  b      // Ordinate oder Wert der 'abhängigen Variablen'
   );
```

Parameter

b

[in] Ordinate oder Wert der 'abhängigen Variablen'. (Vektor der freien Terme).

Rückgabewert

Vektor mit der Lösung des Systems a * x = b.

Hinweis

Wenn mindestens eine Matrixzeile oder -spalte Null ist, hat das System keine Lösung.

Wenn zwei oder mehr Matrixzeilen oder -spalten linear abhängig sind, hat das System keine Lösung.

Beispiel

```
//--- SLAE-Lösung
   vector_x=matrix_a.Solve(vector_b);
//--- Prüfung, ob a * x = b
   result_vector=matrix_a.MatMul(vector_x);
   errors=vector_b.Compare(result_vector,1e-12);
```

© 2000-2025, MetaQuotes Ltd.

LstSq

Rückgabe der Lösung der kleinsten Quadrate von linearen algebraischen Gleichungen (für nicht quadratische oder entartete Matrizen).

```
vector matrix::LstSq(
  const vector  b      // Ordinate oder Wert der 'abhängigen Variablen'
   );
```

Parameter

b

[in] Ordinate oder Wert der 'abhängigen Variablen'. (Vektor der freien Terme)

Rückgabewert

Vektor mit der Lösung des Systems a * x = b. Dies gilt nur für Systeme, die eine exakte Lösung haben.

Beispiel

```
   matrix a={{3, 2},
             {4,-5},
             {3, 3}};
   vector b={7,40,3};
//---
   vector x=a.LstSq(b);
//--- check, must be [5, -4]
   Print("x=", x);
//--- Prüfung, muss [7, 40, 3] sein
   vector b1=a.MatMul(x);
   Print("b1=",b1);

/*
  x=[5.000000000000002,-4]
  b1=[7.000000000000005,40.00000000000001,3.000000000000005]
*/
```

© 2000-2025, MetaQuotes Ltd.

Inv

Berechnung der multiplikativen Inversen einer quadratischen invertierbaren Matrix nach der Jordan-Gauss-Methode.

```
matrix matrix::Inv()
```

Rückgabewert

Multiplikative Inverse einer Matrix

Hinweis

Das Produkt aus der ursprünglichen Matrix und der inversen Matrix ist die Identitätsmatrix.

Wenn mindestens eine Zeile oder Spalte der Matrix Null ist, kann die inverse Matrix nicht ermittelt werden.

Wenn zwei oder mehr Matrixzeilen oder -spalten linear abhängig sind, kann die inverse Matrix nicht ermittelt werden.

Beispiel

```
int TestInverse(const int size_m)
  {
   int    i,j,errors=0;
   matrix matrix_a(size_m,size_m);
//--- die quadratische Matrix auffüllen
   MatrixTestFirst(matrix_a);
//--- Zeitmessung in Mikrosekunden
   ulong t1=GetMicrosecondCount();
//--- get the inverse matrix
   matrix inverse=matrix_a.Inv();
//--- Zeitmessung
   ulong t2=GetMicrosecondCount();
//--- Prüfung auf Korrektheit
   matrix identity=matrix_a.MatMul(inverse);
//---
   for(i=0; i<size_m; i++)
     {
      for(j=0; j<size_m; j++)
        {
         double value;
         //--- Einsen müssen auf der Diagonalen liegen
         if(i==j)
            value=1.0;
         else
            value=0.0;
         if(MathClassify(identity[i][j])>FP_ZERO)
```

© 2000-2025, MetaQuotes Ltd.

```
            errors++;
         else
           {
            if(identity[i][j]!=value)
              {
               double diff=MathAbs(identity[i][j]-value);
               //--- zu viele Multiplikationen und Divisionen, reduzieren sie daher di
               if(diff>1e-9)
                  errors++;
              }
           }
        }
     }
//---
   double elapsed_time=double(t2-t1)/1000.0;
   printf("Inversion of matrix %d x %d  %s  errors=%d  time=%.3f ms",size_m,size_m,er
   return(errors);
  }
```

© 2000-2025, MetaQuotes Ltd.

PInv

Berechnung der Pseudoinverse einer Matrix nach der Moore-Penrose-Methode.

```
matrix matrix::PInv()
```

Rückgabewert

Die Pseudoinverse einer Matrix.

Beispiel

```
int TestPseudoInverse(const int size_m, const int size_k)
  {
   matrix matrix_a(size_m,size_k);
   matrix matrix_inverted(size_k,size_m);
   matrix matrix_temp;
   matrix matrix_a2;
//--- füllen der Matrix
   MatrixTestFirst(matrix_a);
//--- invertieren
   matrix_inverted=matrix_a.PInv();
//--- Prüfung auf Korrektheit
   int errors=0;
//--- A * A+ * A = A   (A+ ist eine Pseudoinverse von A)
   matrix_temp=matrix_a.MatMul(matrix_inverted);
   matrix_a2=matrix_temp.MatMul(matrix_a);
   errors=(int)matrix_a.CompareByDigits(matrix_a2,10);

   printf("PseudoInversion %s matrix_size  %d x %d  errors=%d",errors==0?"passed":"fai
//---
   return(errors);
  }
```

© 2000-2025, MetaQuotes Ltd.

Maschinelles Lernen

Diese Methode werden im Bereich des maschinellen Lernens verwendet

Die Aktivierungsfunktion eines neuronalen Netzes bestimmt den Ausgangswert eines Neurons in Abhängigkeit von der gewichteten Summe der Eingänge. Die Wahl der Aktivierungsfunktion hat einen großen Einfluss auf die Leistung des neuronalen Netzes. Verschiedene Modellteile (Schichten) können unterschiedliche Aktivierungsfunktionen verwenden.

Zusätzlich zu allen bekannten Aktivierungsfunktionen bietet MQL5 auch deren Ableitungen. Funktionsableitungen ermöglichen eine effiziente Aktualisierung der Modellparameter auf der Grundlage der beim Lernen erhaltenen Fehler.

Ein neuronales Netz zielt darauf ab, einen Algorithmus zu finden, der den Fehler beim Lernen minimiert, wofür die Verlustfunktion verwendet wird. Der Wert der Verlustfunktion gibt an, um wie viel der vom Modell vorhergesagte Wert vom realen Wert abweicht. Je nach Problemstellung werden unterschiedliche Verlustfunktionen verwendet. So wird beispielsweise der mittlere quadratische Fehler (MSE) für Regressionsprobleme und die binäre Kreuzentropie (BCE) für binäre Klassifikationszwecke verwendet.

Funktion	Aktion
Activation	Aktivierungsfunktionswerte berechnen und in den übergebenen Vektor oder Matrix schreiben
Derivative	Berechnet die Ableitungswerte der Aktivierungsfunktion, die dem übergebenen Vektor oder Matrix zugewiesen werden
Loss	Berechnet die Verlustfunktionswerte und schreibt sie in den übergebenen Vektor oder Matrix
LossGradient	Berechnung eines Vektors oder einer Matrix von Verlustfunktionsgradienten
RegressionMetric	Berechnet die Regressionsmetrik als Abweichung von der Regressionslinie, die auf dem angegebenen Datenarray konstruiert wurde
ConfusionMatrix	Berechnen der Wahrheitsmatrix oder auch Konfusionsmatrix: Richtige und falsche Klassifikationen. Die Methode wird auf den Vektor der vorhergesagten Werte angewendet
ConfusionMatrixMultilabel	Berechnen der Wahrheits- oder Konfusionsmatrix für jedes Label. Die Methode wird auf den Vektor der vorhergesagten Werte angewendet
ClassificationMetric	Berechnung der Klassifizierungsmetrik zur Bewertung der Qualität der vorhergesagten Daten im Vergleich zu den wahren Daten. Die Methode wird auf den Vektor der vorhergesagten Werte angewendet
ClassificationScore	Berechnung der Klassifizierungsmetrik, um die Qualität der vorhergesagten Daten im Vergleich zu den wahren Daten zu bewerten

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
PrecisionRecall	Berechnen der Werte, mit denen eine Precision-Recall-Kurve erstellt wird. Ähnlich wie ClassificationScore wird diese Methode auf den Vektor mit wahren Werte angewendet.
ReceiverOperatingCharacteristic	Berechnet werden die Werte, um die ROC-Kurve (Receiver Operating Characteristic) zu erstellen. Ähnlich wie ClassificationScore wird diese Methode auf den Vektor mit wahren Werte angewendet.

Beispiel

Dieses Beispiel demonstriert das Training eines Modells mit Hilfe von Matrixoperationen. Das Modell wird für die Funktion (a + b + c)^2 / (a^2 + b^2 + c^2) trainiert. Wir geben die Ausgangsdatenmatrix ein, in der a, b und c in verschiedenen Spalten enthalten sind. Das Modell gibt das Funktionsergebnis aus.

```
matrix weights1, weights2, weights3;                // Matrizen der Gewichte
matrix output1, output2, result;                    // Matrizen der Ausgaben der neuror
input int layer1 = 200;                             // die Größe der ersten verdeckten
input int layer2 = 200;                             // die Größe der zweiten verdeckter
input int Epochs = 20000;                           // die Anzahl der Trainingsepochen
input double lr = 3e-6;                             // Lernrate
input ENUM_ACTIVATION_FUNCTION ac_func = AF_SWISH; // Aktivierungsfunktion
//+------------------------------------------------------------------+
//| Script start Funktion                                            |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   int train = 1000;     // Größe des Trainingssatzes
   int test = 10;        // Größe der Teststichprobe
   matrix m_data, m_target;
//--- generieren einer Teststichprobe
   if(!CreateData(m_data, m_target, train))
      return;
//--- Modelltraining
   if(!Train(m_data, m_target, Epochs))
      return;
//--- generieren der Teststichprobe
   if(!CreateData(m_data, m_target, test))
      return;
//--- Test des Modells
   Test(m_data, m_target);
  }
//+------------------------------------------------------------------+
//| Erstellen der Stichprobendaten                                   |
//+------------------------------------------------------------------+
bool CreateData(matrix &data, matrix &target, const int count)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Initialisierung der Ausgangsdaten und der Ergebnismatrix
   if(!data.Init(count, 3) || !target.Init(count, 1))
      return false;
//--- Füllen der Matrix für die Anfangswerte mit Zufallszahlen
   data.Random(-10, 10);
//--- Berechnen der Zielwerte für die Trainingsstichprobe
   vector X1 = MathPow(data.Col(0) + data.Col(1) + data.Col(1), 2);
   vector X2 = MathPow(data.Col(0), 2) + MathPow(data.Col(1), 2) + MathPow(data.Col(2)
   if(!target.Col(X1 / X2, 0))
      return false;
//--- Ergebnisrückgabe
   return true;
  }
//+------------------------------------------------------------------+
//| Trainingsmethode des Models                                      |
//+------------------------------------------------------------------+
bool Train(matrix &data, matrix &target, const int epochs = 10000)
  {
//--- Modell erstellen
   if(!CreateNet())
      return false;
//--- Modelltraining
   for(int ep = 0; ep < epochs; ep++)
     {
      //--- Vorwärtsdurchlauf
      if(!FeedForward(data))
         return false;
      PrintFormat("Epoch %d, loss %.5f", ep, result.Loss(target, LOSS_MSE));
      //--- Backpropagation und Update der Gewichtsmatrix
      if(!Backprop(data, target))
         return false;
     }
//--- Ergebnisrückgabe
   return true;
  }
//+------------------------------------------------------------------+
//| Methode für die Modellerstellung                                 |
//+------------------------------------------------------------------+
bool CreateNet()
  {
//--- Initialisierung der Gewichtsmatrizen
   if(!weights1.Init(4, layer1) || !weights2.Init(layer1 + 1, layer2) || !weights3.Ini
      return false;
//--- Füllen der Gewichtsmatrizen mit Zufallszahlen
   weights1.Random(-0.1, 0.1);
   weights2.Random(-0.1, 0.1);
   weights3.Random(-0.1, 0.1);
//--- Ergebnisrückgabe
```

© 2000-2025, MetaQuotes Ltd.

```
   return true;
  }
//+------------------------------------------------------------------+
//| Feedforward-Methode                                              |
//+------------------------------------------------------------------+
bool FeedForward(matrix &data)
  {
//--- Prüfen der Größe der Ausgangsdaten
   if(data.Cols() != weights1.Rows() - 1)
      return false;
//--- Berechnen der ersten neuronalen Schicht
   matrix temp = data;
   if(!temp.Resize(temp.Rows(), weights1.Rows()) ||
      !temp.Col(vector::Ones(temp.Rows()), weights1.Rows() - 1))
      return false;
   output1 = temp.MatMul(weights1);
//--- Berechnen der Aktivierungsfunktion
   if(!output1.Activation(temp, ac_func))
      return false;
//--- Berechnen der zweiten neuronalen Schicht
   if(!temp.Resize(temp.Rows(), weights2.Rows()) ||
      !temp.Col(vector::Ones(temp.Rows()), weights2.Rows() - 1))
      return false;
   output2 = temp.MatMul(weights2);
//--- Berechnen der Aktivierungsfunktion
   if(!output2.Activation(temp, ac_func))
      return false;
//--- Berechnen der dritten neuronalen Schicht
   if(!temp.Resize(temp.Rows(), weights3.Rows()) ||
      !temp.Col(vector::Ones(temp.Rows()), weights3.Rows() - 1))
      return false;
   result = temp.MatMul(weights3);
//--- Ergebnisrückgabe
   return true;
  }
//+------------------------------------------------------------------+
//| Backpropagation-Methode                                          |
//+------------------------------------------------------------------+
bool Backprop(matrix &data, matrix &target)
  {
//--- Prüfen der Größe der Matrix für die Zielwerte
   if(target.Rows() != result.Rows() ||
      target.Cols() != result.Cols())
      return false;
//--- Berechnen der Ableitung der berechneten Zielwerte
   matrix loss = (target - result) * 2;
//--- Übertragen der Gradienten zur vorherigen Schicht
   matrix gradient = loss.MatMul(weights3.Transpose());
//--- Update der Gewichtsmatrix der letzten Schicht
```

© 2000-2025, MetaQuotes Ltd.

```
   matrix temp;
   if(!output2.Activation(temp, ac_func))
      return false;
   if(!temp.Resize(temp.Rows(), weights3.Rows()) ||
      !temp.Col(vector::Ones(temp.Rows()), weights3.Rows() - 1))
      return false;
   weights3 = weights3 + temp.Transpose().MatMul(loss) * lr;
//--- Anpassung des Fehlergradienten durch die Ableitung der Aktivierungsfunktion
   if(!output2.Derivative(temp, ac_func))
      return false;
   if(!gradient.Resize(gradient.Rows(), gradient.Cols() - 1))
      return false;
   loss = gradient * temp;
//--- Übertragen der Gradienten der tieferen Schicht
   gradient = loss.MatMul(weights2.Transpose());
//--- Update der Gewichtsmatrix der zweiten verdeckten Schicht
   if(!output1.Activation(temp, ac_func))
      return false;
   if(!temp.Resize(temp.Rows(), weights2.Rows()) ||
      !temp.Col(vector::Ones(temp.Rows()), weights2.Rows() - 1))
      return false;
   weights2 = weights2 + temp.Transpose().MatMul(loss) * lr;
//--- Anpassung des Fehlergradienten durch die Ableitung der Aktivierungsfunktion
   if(!output1.Derivative(temp, ac_func))
      return false;
   if(!gradient.Resize(gradient.Rows(), gradient.Cols() - 1))
      return false;
   loss = gradient * temp;
//--- Update der Gewichtsmatrix der ersten verdeckten Schicht
   temp = data;
   if(!temp.Resize(temp.Rows(), weights1.Rows()) ||
      !temp.Col(vector::Ones(temp.Rows()), weights1.Rows() - 1))
      return false;
   weights1 = weights1 + temp.Transpose().MatMul(loss) * lr;
//--- Ergebnisrückgabe
   return true;
  }
//+------------------------------------------------------------------+
//| Testmethode des Models                                           |
//+------------------------------------------------------------------+
bool Test(matrix &data, matrix &target)
  {
//--- Feedforward der Testdaten
   if(!FeedForward(data))
      return false;
//--- Ausdruck der Ergebnisse der Modellberechnung und der tatsächlichen Werte
   PrintFormat("Test loss %.5f", result.Loss(target, LOSS_MSE));
   ulong total = data.Rows();
   for(ulong i = 0; i < total; i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("(%.2f + %.2f + %.2f)^2 / (%.2f^2 + %.2f^2 + %.2f^2) =  Net %.2f, Ta
                  data[i, 0], data[i, 1], data[i, 2], result[i, 0], target[i, 0]);
//--- Ergebnisrückgabe
   return true;
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

Activation

Berechnen der Aktivierungsfunktion und zuweisen der Werte dem übergebenen Vektor bzw. Matrix.

```
bool vector::Activation(
  vector&                    vect_out,      // Vektor für die Werte
  ENUM_ACTIVATION_FUNCTION   activation,    // Aktivierungsfunktion
   ...                                      // weitere Parameter
   );


bool matrix::Activation(
  matrix&                    matrix_out,    // Matrix für die Werte
  ENUM_ACTIVATION_FUNCTION   activation     // Aktivierungsfunktion
   );


bool matrix::Activation(
  matrix&                    matrix_out,    // Matrix für die Werte
  ENUM_ACTIVATION_FUNCTION   activation,    // Aktivierungsfunktion
  ENUM_MATRIX_AXIS           axis,          // Achse
   ...                                      // weitere Parameter
   );
```

Parameter

vect_out/matrix_out

[out] Vektor oder Matrix für die berechneten Werte der Aktivierungsfunktion.

activation

[in] Aktivierungsfunktion aus der Enumeration ENUM_ACTIVATION_FUNCTION.

axis

[in] ENUM_MATRIX_AXIS Wert der Enumeration (AXIS_HORZ — horizontale Achse, AXIS_VERT — vertikale Achse).

...

[in] Weitere Parameter, die für einige Aktivierungsfunktionen erforderlich sind. Wenn keine Parameter angegeben werden, werden die Standardwerte verwendet.

Rückgabewert

Liefert bei Erfolg true, ansonsten false.

Weitere Parameter

Einige Aktivierungsfunktionen akzeptieren zusätzliche Parameter. Wenn keine Parameter angegeben sind, werden die Standardwerte verwendet.

© 2000-2025, MetaQuotes Ltd.

```
AF_ELU  (Exponential Linear Unit)
  double alpha=1.0

Aktivierungsfunktion: if(x>=0) f(x) = x
                      else f(x) = alpha * (exp(x)-1)


AF_LINEAR
  double alpha=1.0
  double beta=0.0

Aktivierungsfunktion: f(x) = alpha*x + beta


AF_LRELU   (Leaky REctified Linear Unit)
  double alpha=0.3

Aktivierungsfunktion: if(x>=0) f(x)=x
                      else f(x) = alpha*x


AF_RELU  (REctified Linear Unit)
  double alpha=0.0
  double max_value=0.0
  double treshold=0.0

Aktivierungsfunktion: if(alpha==0) f(x) = max(x,0)
                      else if(x>max_value) f(x) = x
                      else f(x) = alpha*(x - treshold)


AF_SWISH
  double beta=1.0

Aktivierungsfunktion: f(x) = x / (1+exp(-x*beta))


AF_TRELU   (Thresholded REctified Linear Unit)
  double theta=1.0

Aktivierungsfunktion: if(x>theta) f(x) = x
                      else f(x) = 0


AF_PRELU   (Parametric REctified Linear Unit)
  double alpha[] - learned array of coeefficients
```

© 2000-2025, MetaQuotes Ltd.

```
Aktivierungsfunktion: if(x[i]>=0) f(x)[i] = x[i]
                      else f(x)[i] = alpha[i] * x[i]
```

Hinweis

In künstlichen neuronalen Netzen bestimmt die Aktivierungsfunktion eines Neurons das Ausgangssignal, das durch ein Eingangssignal oder einen Satz von Eingangssignalen definiert ist. Die Wahl der Aktivierungsfunktion hat einen großen Einfluss auf die Leistung des neuronalen Netzes. Verschiedene Modellteile (Schichten) können unterschiedliche Aktivierungsfunktionen verwenden.

Beispiele für die Verwendung zusätzlicher Parameter:

```
vector x={0.1, 0.4, 0.9, 2.0, -5.0, 0.0, -0.1};
vector y;

x.Activation(y,AF_ELU);
Print(y);
x.Activation(y,AF_ELU,2.0);
Print(y);

Print("");
x.Activation(y,AF_LINEAR);
Print(y);
x.Activation(y,AF_LINEAR,2.0);
Print(y);
x.Activation(y,AF_LINEAR,2.0,5.0);
Print(y);

Print("");
x.Activation(y,AF_LRELU);
Print(y);
x.Activation(y,AF_LRELU,1.0);
Print(y);
x.Activation(y,AF_LRELU,0.1);
Print(y);

Print("");
x.Activation(y,AF_RELU);
Print(y);
x.Activation(y,AF_RELU,2.0,0.5);
Print(y);
x.Activation(y,AF_RELU,2.0,0.5,1.0);
Print(y);

Print("");
```

© 2000-2025, MetaQuotes Ltd.

```
   x.Activation(y,AF_SWISH);
   Print(y);
   x.Activation(y,AF_SWISH,2.0);
   Print(y);

   Print("");
   x.Activation(y,AF_TRELU);
   Print(y);
   x.Activation(y,AF_TRELU,0.3);
   Print(y);

   Print("");
   vector a=vector::Full(x.Size(),2.0);
   x.Activation(y,AF_PRELU,a);
   Print(y);

/*  Ergebnisse
   [0.1,0.4,0.9,2,-0.993262053000915,0,-0.095162581964040]
   [0.1,0.4,0.9,2,-1.986524106001829,0,-0.190325163928081]

   [0.1,0.4,0.9,2,-5,0,-0.1]
   [0.2,0.8,1.8,4,-10,0,-0.2]
   [5.2,5.8,6.8,9,-5,5,4.8]

   [0.1,0.4,0.9,2,-1.5,0,-0.03]
   [0.1,0.4,0.9,2,-5,0,-0.1]
   [0.1,0.4,0.9,2,-0.5,0,-0.01]

   [0.1,0.4,0.9,2,0,0,0]
   [0.2,0.8,0.9,2,-10,0,-0.2]
   [-1.8,-1.2,0.9,2,-12,-2,-2.2]

   [0.052497918747894,0.239475064044981,0.6398545523625035,1.761594155955765,-0.033464
   [0.054983399731247,0.275989792451045,0.7723340415895611,1.964027580075817,-0.000226

   [0,0,0,2,0,0,0]
   [0,0.4,0.9,2,0,0,0]

   [0.1,0.4,0.9,2,-10,0,-0.2]
*/
```

© 2000-2025, MetaQuotes Ltd.

Derivative

Berechnet die Ableitungswerte der Aktivierungsfunktion, die dem übergebenen Vektor oder Matrix zugewiesen werden.

```
bool vector::Derivative(
  vector&                   vect_out,      // Vektor für die Werte
  ENUM_ACTIVATION_FUNCTION  activation,    // Aktivierungsfunktion
   ...                                     // weitere Parameter
   );


bool matrix::Derivative(
  matrix&                   matrix_out,    // Matrix für die Werte
  ENUM_ACTIVATION_FUNCTION  activation,    // Aktivierungsfunktion
   );


bool matrix::Derivative(
  matrix&                   matrix_out,    // Matrix für die Werte
  ENUM_ACTIVATION_FUNCTION  activation,    // Aktivierungsfunktion
  ENUM_MATRIX_AXIS          axis,          // Achse
   ...                                     // weitere Parameter
   );
```

Parameter

vect_out/matrix_out

[out] Vektor oder Matrix zum Abrufen der berechneten Werte der Ableitung der Aktivierungsfunktion.

activation

[in] Aktivierungsfunktion aus der Enumeration ENUM_ACTIVATION_FUNCTION.

axis

[in] ENUM_MATRIX_AXIS Wert der Enumeration (AXIS_HORZ — horizontale Achse, AXIS_VERT — vertikale Achse).

...

[in] Die zusätzlichen Parameter sind die gleichen wie die der Aktivierungsfunktionen. Nur einige Aktivierungsfunktionen akzeptieren zusätzliche Parameter. Wenn keine Parameter angegeben werden, werden die Standardwerte verwendet.

Rückgabewert

Liefert bei Erfolg true, ansonsten false.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Funktionsableitungen ermöglichen eine effiziente Aktualisierung von Modellparametern basierend auf dem beim Lernen während der Fehler-Backpropagation erhaltenen Fehler.

© 2000-2025, MetaQuotes Ltd.

Loss

Berechnung des Wertes der Verlustfunktion

```
double vector::Loss(
  const vector&       vect_true,     // Vektor mit tatsächlichen Werten
  ENUM_LOSS_FUNCTION  loss,          // Verlustfunktion
   ...                               // weitere Parameter
   );


double matrix::Loss(
  const matrix&       matrix_true,   // Matrix mit tatsächlichen Werten
  ENUM_LOSS_FUNCTION  loss,          // Verlustfunktion
   );


double matrix::Loss(
  const matrix&       matrix_true,   // Matrix mit tatsächlichen Werten
  ENUM_LOSS_FUNCTION  loss,          // Verlustfunktion
  ENUM_MATRIX_AXIS    axis,          // Achse
   ...                               // weitere Parameter
   );
```

Parameter

vect_true/matrix_true

[in] Vektor oder Matrix mit den tatsächlichen Werten.

loss

[in] Verlustfunktion aus der Enumeration ENUM_LOSS_FUNCTION.

axis

[in] ENUM_MATRIX_AXIS Wert der Enumeration (AXIS_HORZ — horizontale Achse, AXIS_VERT — vertikale Achse).

...

[in] Der zusätzliche Parameter 'delta' kann nur von der Hubert-Verlustfunktion (LOSS_HUBER) verwendet werden.

Rückgabewert

Ein Wert vom Typ double

Wie der Parameter 'delta' in der Hubert-Verlustfunktion (LOSS_HUBER) verwendet wird

```
   double delta = 1.0;
   double error = fabs(y - x);
   if(error<delta)
      loss = 0.5 * error^2;
```

© 2000-2025, MetaQuotes Ltd.

```
   else
      loss = 0.5 * delta^2 + delta * (error - delta);
```

Hinweis

Ein neuronales Netz zielt darauf ab, die Algorithmen zu finden, die den Fehler in der Trainingsstichprobe, für die die Verlustfunktion verwendet wird, minimieren.

Der Wert der Verlustfunktion gibt an, um wie viel der vom Modell vorhergesagte Wert vom realen Wert abweicht.

Je nach Problemstellung werden unterschiedliche Verlustfunktionen verwendet. So wird beispielsweise der mittlere quadratische Fehler (MSE) für Regressionsprobleme und die binäre Kreuzentropie (BCE) für binäre Klassifikationszwecke verwendet.

Beispiel für den Aufruf der Hubert-Verlustfunktion:

```
   vector y_true = {0.0, 1.0, 0.0, 0.0};
   vector y_pred = {0.6, 0.4, 0.4, 0.6};
   double loss=y_pred.Loss(y_true,LOSS_HUBER);
   Print(loss);
   double loss2=y_pred.Loss(y_true,LOSS_HUBER,0.5);
   Print(loss2);

/* Ergebnis
   0.155
   0.15125
*/
```

© 2000-2025, MetaQuotes Ltd.

LossGradient

Berechnung eines Vektors oder einer Matrix der Gradienten einer Verlustfunktion.

```
Vektor vector::LossGradient(
  const vector&       vect_true,     // Vektor mit tatsächlichen Werten
  ENUM_LOSS_FUNCTION  loss,          // Typ der Verlustfunktion
   ...                               // weitere Parameter
   );


matrix matrix::LossGradient(
  const matrix&       matrix_true,   // Matrix mit tatsächlichen Werten
  ENUM_LOSS_FUNCTION  loss,          // Verlustfunktion
   );


matrix matrix::LossGradient(
  const matrix&       matrix_true,   // Matrix mit tatsächlichen Werten
  ENUM_LOSS_FUNCTION  loss,          // Verlustfunktion
  ENUM_MATRIX_AXIS    axis,          // Achse
   ...                               // weitere Parameter
   );
```

Parameter

vect_true/matrix_true

[in] Vektor oder Matrix mit den tatsächlichen Werten.

loss

[in] Verlustfunktion aus der Enumeration ENUM_LOSS_FUNCTION.

axis

[in] ENUM_MATRIX_AXIS Wert der Enumeration (AXIS_HORZ — horizontale Achse, AXIS_VERT — vertikale Achse).

...

[in] Der zusätzliche Parameter 'delta' kann nur von der Hubert-Verlustfunktion (LOSS_HUBER) verwendet werden.

Rückgabewert

Vektor oder Matrix der Werte des Gradienten der Verlustfunktion. Der Gradient ist die partielle Ableitung nach dx (x ist der vorhergesagte Wert) der Verlustfunktion an einem bestimmten Punkt.

Hinweis

Gradienten werden in neuronalen Netzen verwendet, um die Gewichte der Gewichtsmatrix während des Backpropagation-Verfahrens beim Training eines Modells anzupassen.

© 2000-2025, MetaQuotes Ltd.

Ein neuronales Netz zielt darauf ab, die Algorithmen zu finden, die den Fehler in der Trainingsstichprobe, für die die Verlustfunktion verwendet wird, minimieren.

Je nach Problemstellung werden unterschiedliche Verlustfunktionen verwendet. So wird beispielsweise der mittlere quadratische Fehler (MSE) für Regressionsprobleme und die binäre Kreuzentropie (BCE) für binäre Klassifikationszwecke verwendet.

Beispiel für die Berechnung der Gradienten der Verlustfunktionen

```
   matrixf y_true={{ 1, 2, 3, 4 },
                   { 5, 6, 7, 8 },
                   { 9,10,11,12 }};
   matrixf y_pred={{ 1, 2, 3, 4 },
                   {11,10, 9, 8 },
                   { 5, 6, 7,12 }};
   matrixf loss_gradient =y_pred.LossGradient(y_true,LOSS_MAE);
   matrixf loss_gradienth=y_pred.LossGradient(y_true,LOSS_MAE,AXIS_HORZ);
   matrixf loss_gradientv=y_pred.LossGradient(y_true,LOSS_MAE,AXIS_VERT);
   Print("loss gradients\n",loss_gradient);
   Print("loss gradients on horizontal axis\n",loss_gradienth);
   Print("loss gradients on vertical axis\n",loss_gradientv);

/* Ergebnis
   loss gradients
   [[0,0,0,0]
    [0.083333336,0.083333336,0.083333336,0]
    [-0.083333336,-0.083333336,-0.083333336,0]]
   loss gradients on horizontal axis
   [[0,0,0,0]
    [0.33333334,0.33333334,0.33333334,0]
    [-0.33333334,-0.33333334,-0.33333334,0]]
   loss gradients on vertical axis
   [[0,0,0,0]
    [0.25,0.25,0.25,0]
    [-0.25,-0.25,-0.25,0]]
*/
```

© 2000-2025, MetaQuotes Ltd.

RegressionMetric

Berechnung der Regressionsmetrik zur Bewertung der Qualität der vorhergesagten Daten im Vergleich zu den tatsächlichen Daten

```
double vector::RegressionMetric(
   const vector&           vector_true,   // Vektor mit den tatsächlichen Werten
   ENUM_REGRESSION_METRIC  metric         // metrischer Typ
   );

double matrix::RegressionMetric(
   const vector&           matrix_true,   // Matrix mit den tatsächlichen Werten
   ENUM_REGRESSION_METRIC  metric         // metrischer Typ
);

Vektor Matrix::RegressionMetric(
   const vector&           matrix_true,   // Matrix mit den tatsächlichen Werten
   ENUM_REGRESSION_METRIC  metric,        // metrischer Typ
   int                     axis           // Achse
   );
```

Parameter

vector_true/matrix_true

[in] Vektor oder Matrix mit den tatsächlichen Werten.

metric

[in] Metrischer Typ aus der Enumeration ENUM_REGRESSION_METRIC.

axis

[in] Achse. 0 — horizontale Achse, 1 — vertikale Achse.

Rückgabewert

Die berechnete Metrik, die die Qualität der vorhergesagten Daten im Vergleich zu den tatsächlichen Daten bewertet.

Hinweis

- REGRESSION_MAE — mittlerer absoluter Fehler, der die absoluten Differenzen zwischen den vorhergesagten Werten und den entsprechenden tatsächlichen Werten darstellt
- REGRESSION_MSE — mittlerer quadratischer Fehler, der die quadrierten Differenzen zwischen den vorhergesagten Werten und den entsprechenden tatsächlichen Werten darstellt.
- REGRESSION_RMSE — Quadratwurzel des mittleren quadratischen Fehlers (MSE)
- REGRESSION_R2 - 1 — MSE(Regression) / MSE(Mittelwert)
- REGRESSION_MAPE — MAE als Prozentsatz
- REGRESSION_MSPE — MSE in Prozent
- REGRESSION_RMSLE — RMSE berechnet auf einer logarithmischen Skala

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
   vector y_true = {3, -0.5, 2, 7};
   vector y_pred = {2.5, 0.0, 2, 8};
//---
   double mse=y_pred.RegressionMetric(y_true,REGRESSION_MSE);
   Print("mse=",mse);
//---
   double mae=y_pred.RegressionMetric(y_true,REGRESSION_MAE);
   Print("mae=",mae);
//---
   double r2=y_pred.RegressionMetric(y_true,REGRESSION_R2);
   Print("r2=",r2);

  /* Ergebnis
   mae=0.375
   mse=0.5
   r2=0.9486081370449679
   */
```

© 2000-2025, MetaQuotes Ltd.

ConfusionMatrix

Berechnen der Wahrheitsmatrix oder auch Konfusionsmatrix: Richtige und falsche Klassifikationen. Die Methode wird auf den Vektor der vorhergesagten Werte angewendet.

```
Matrix vector::ConfusionMatrix(
   const vector&        vect_true      // Vektor der wahren Werte
   );


Matrix vector::ConfusionMatrix(
   const vector&        vect_true,     // Vektor der wahren Werte
   uint                 label          // Wert des Labels
   );
```

Parameter

vect_true

[in] Vektor mit den wahren Werten.

label

[in] Der Labelwert für die Berechnung der Wahrheitsmatrix oder auch Konfusionsmatrix (confusion matrix).

Rückgabewert

Die Konfusionsmatrix. Wird kein Labelwert angegeben, wird eine Mehrklassen-Konfusionsmatrix zurückgegeben, in der jedes Label mit jedem anderen Label einzeln abgeglichen wird. Wird ein Labelwert angegeben, wird eine 2 x 2-Matrix zurückgegeben, in der das angegebene Label als positiv gilt, während alle anderen Labels negativ sind (ovr, one vs rest).

Hinweis

Die Konfusionsmatrix C ist so beschaffen, dass Cij gleich der Anzahl der Beobachtungen ist, von denen bekannt ist, dass sie zur Gruppe i gehören, und von denen vorhergesagt wird, dass sie zur Gruppe j gehören. Bei der binären Klassifizierung ist die Anzahl der wahren Negativen (TN) gleich C00, der falschen Negativen (FN) gleich C10, der wahren Positiven (TP) gleich C11 und der falschen Positiven (FP) gleich C01.

Mit anderen Worten: Die Matrix kann wie folgt grafisch dargestellt werden:

TN	FP
FN	TP

Die Größen des Vektors der wahren Werte und des Vektors der vorhergesagten Werte sollten gleich sein.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
   vector y_true={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,8,4,2,7,6,8,4,2,3,6};
   vector y_pred={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,2,9,4,9,5,9,2,7,7,0};
   matrix confusion=y_pred.ConfusionMatrix(y_true);
   Print(confusion);
   confusion=y_pred.ConfusionMatrix(y_true,0);
   Print(confusion);
   confusion=y_pred.ConfusionMatrix(y_true,1);
   Print(confusion);
   confusion=y_pred.ConfusionMatrix(y_true,2);
   Print(confusion);


/*
  [[3,0,0,0,0,0,0,0,0,0]
   [0,3,0,0,0,0,0,0,0,0]
   [0,0,1,0,1,0,0,1,0,0]
   [0,0,0,1,0,0,0,1,0,0]
   [0,0,1,0,3,0,0,0,0,1]
   [0,0,0,0,0,2,0,0,0,0]
   [1,0,0,0,0,1,1,0,0,0]
   [0,0,0,0,0,0,0,2,0,1]
   [0,0,1,0,0,0,0,0,0,1]
   [0,0,0,0,0,0,0,0,0,4]]
  [[26,1]
   [0,3]]
  [[27,0]
   [0,3]]
  [[25,2]
   [2,1]]
*/
```

© 2000-2025, MetaQuotes Ltd.

ConfusionMatrixMultilabel

Berechnen der Wahrheits- oder Konfusionsmatrix für jedes Label. Die Methode wird auf den Vektor der vorhergesagten Werte angewendet.

```
uint vector::ConfusionMatrixMultiLabel(
   const vector&       vect_true,     // Vektor der wahren Werte
   matrix&             confusions[]   // Array der berechneten Konfusionsmatritzen
   );
```

Parameter

vect_true

[in] Vektor mit den wahren Werten.

confusions

[out] Ein Array von 2 x 2 Matrizen mit berechneten Konfusionsmatrizen für jedes Label.

Rückgabewert

Größe des Arrays der berechneten Konfusionsmatrizen. Im Falle eines Fehlers wird 0 zurückgegeben.

Hinweis

Das Ergebnis-Array kann dynamisch oder statisch sein. Wenn das Array statisch ist, muss es mindestens so groß sein wie die Anzahl der Klassen.

Die Größen des Vektors der wahren Werte und des Vektors der vorhergesagten Werte sollten gleich sein.

Beispiel:

```
   vector y_true={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,8,4,2,7,6,8,4,2,3,6};
   vector y_pred={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,2,9,4,9,5,9,2,7,7,0};
   matrix label_confusions[12];

   uint   res=y_pred.ConfusionMatrixMultiLabel(y_true,label_confusions);
   Print("res=",res,"  size=",label_confusions.Size());
   for(uint i=0; i<res; i++)
      Print(label_confusions[i]);


/*
  res=10  size=12
  [[26,1]
   [0,3]]
  [[27,0]
   [0,3]]
```

© 2000-2025, MetaQuotes Ltd.

```
   [[25,2]
    [2,1]]
   [[28,0]
    [1,1]]
   [[24,1]
    [2,3]]
   [[27,1]
    [0,2]]
   [[27,0]
    [2,1]]
   [[25,2]
    [1,2]]
   [[28,0]
    [2,0]]
   [[23,3]
    [0,4]]
*/
```

© 2000-2025, MetaQuotes Ltd.

ClassificationMetric

Berechnung der Klassifizierungsmetrik zur Bewertung der Qualität der vorhergesagten Daten im Vergleich zu den wahren Daten. Die Methode wird auf den Vektor der vorhergesagten Werte angewendet.

```
vector vector::ClassificationMetric(
   const vector&                vect_true,      // Vektor der wahren Werte
   ENUM_CLASSIFICATION_METRIC metric          // Typ der Metrik
   );


vector vector::ClassificationMetric(
   const vector&                vect_true,      // Vektor der wahren Werte
   ENUM_CLASSIFICATION_METRIC metric          // Typ der Metrik
   ENUM_AVERAGE_MODE          mode            // Modus der Durchschnittsbildung
   );
```

Parameter

vect_true

[in] Vektor mit den wahren Werten.

metric

[in] Typ der Metrik aus der Enumeration ENUM_CLASSIFICATION_METRIC. Es werden andere Werte als CLASSIFICATION_TOP_K_ACCURACY, CLASSIFICATION_AVERAGE_PRECISION und CLASSIFICATION_ROC_AUC (die in der Methode ClassificationScore genutzt werden) verwendet.

mode

[in] Der Modus der Durchschnittsermittlung aus der Enumeration ENUM_AVERAGE_MODE. Wird für die Metriken CLASSIFICATION_F1, CLASSIFICATION_JACCARD, CLASSIFICATION_PRECISION und CLASSIFICATION_RECALL verwendet.

Rückgabewert

Ein Vektor, der die berechnete Metrik enthält. Im Falle des Mittelungsmodus AVERAGE_NONE enthält der Vektor die Metrikwerte für jede Klasse ohne Mittelwertbildung. (Im Falle der binären Klassifizierung wären dies beispielsweise zwei Metriken für 'false' und 'true').

Hinweis zu den Modi der Mittelwertbildung

AVERAGE_BINARY ist nur für die binäre Klassifizierung sinnvoll.

AVERAGE_MICRO — berechnet die Metriken global durch Zählen der gesamten wahr-positiven, falsch-negativen und falsch-positiven Ergebnisse.

AVERAGE_MACRO — berechnet die Metriken für jedes Label und ermittelt deren ungewichteten Mittelwert. Dabei wird ein Ungleichgewicht das Label nicht berücksichtigt.

AVERAGE_WEIGHTED — berechnet die Metriken für jedes Label und ermittelt ihren Durchschnitt gewichteten durch Unterstützung (der Anzahl der wahren Instanzen für jedes Label). Dies ändert

© 2000-2025, MetaQuotes Ltd.

'macro', um das Label-Ungleichgewicht zu berücksichtigen; es kann zu einem F-Score führen, der nicht zwischen 'precision' und 'recall' liegt.

Beispiel:

```
   vector y_true={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,8,4,2,7,6,8,4,2,3,6};
   vector y_pred={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,2,9,4,9,5,9,2,7,7,0};

   vector accuracy=y_pred.ClassificationMetric(y_true,CLASSIFICATION_ACCURACY);
   Print("accuracy=",accuracy);
   vector balanced=y_pred.ClassificationMetric(y_true,CLASSIFICATION_BALANCED_ACCURACY
   Print("balanced=",balanced);
   Print("");

   vector f1_micro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_F1,AVERAGE_MICRO)
   Print("f1_micro=",f1_micro);
   vector f1_macro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_F1,AVERAGE_MACRO)
   Print("f1_macro=",f1_macro);
   vector f1_weighted=y_pred.ClassificationMetric(y_true,CLASSIFICATION_F1,AVERAGE_WEI
   Print("f1_weighted=",f1_weighted);
   vector f1_none=y_pred.ClassificationMetric(y_true,CLASSIFICATION_F1,AVERAGE_NONE);
   Print("f1_none=",f1_none);
   Print("");

   vector jaccard_micro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_JACCARD,AVER
   Print("jaccard_micro=",jaccard_micro);
   vector jaccard_macro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_JACCARD,AVER
   Print("jaccard_macro=",jaccard_macro);
   vector jaccard_weighted=y_pred.ClassificationMetric(y_true,CLASSIFICATION_JACCARD,A
   Print("jaccard_weighted=",jaccard_weighted);
   vector jaccard_none=y_pred.ClassificationMetric(y_true,CLASSIFICATION_JACCARD,AVERA
   Print("jaccard_none=",jaccard_none);
   Print("");

   vector precision_micro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_PRECISION,
   Print("precision_micro=",precision_micro);
   vector precision_macro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_PRECISION,
   Print("precision_macro=",precision_macro);
   vector precision_weighted=y_pred.ClassificationMetric(y_true,CLASSIFICATION_PRECISI
   Print("precision_weighted=",precision_weighted);
   vector precision_none=y_pred.ClassificationMetric(y_true,CLASSIFICATION_PRECISION,A
   Print("precision_none=",precision_none);
   Print("");

   vector recall_micro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_RECALL,AVERAG
   Print("recall_micro=",recall_micro);
   vector recall_macro=y_pred.ClassificationMetric(y_true,CLASSIFICATION_RECALL,AVERAG
   Print("recall_macro=",recall_macro);
```

© 2000-2025, MetaQuotes Ltd.

```
   vector recall_weighted=y_pred.ClassificationMetric(y_true,CLASSIFICATION_RECALL,AVE
   Print("recall_weighted=",recall_weighted);
   vector recall_none=y_pred.ClassificationMetric(y_true,CLASSIFICATION_RECALL,AVERAGE
   Print("recall_none=",recall_none);
   Print("");

//--- binary classification
   vector y_pred_bin={0,1,0,1,1,0,0,0,1};
   vector y_true_bin={1,0,0,0,1,0,1,1,1};

   vector f1_bin=y_pred_bin.ClassificationMetric(y_true_bin,CLASSIFICATION_F1,AVERAGE_
   Print("f1_bin=",f1_bin);
   vector jaccard_bin=y_pred_bin.ClassificationMetric(y_true_bin,CLASSIFICATION_JACCAR
   Print("jaccard_bin=",jaccard_bin);
   vector precision_bin=y_pred_bin.ClassificationMetric(y_true_bin,CLASSIFICATION_PREC
   Print("precision_bin=",precision_bin);
   vector recall_bin=y_pred_bin.ClassificationMetric(y_true_bin,CLASSIFICATION_RECALL,
   Print("recall_bin=",recall_bin);


/*
  accuracy=[0.6666666666666666]
  balanced=[0.6433333333333333]

  f1_micro=[0.6666666666666666]
  f1_macro=[0.6122510822510823]
  f1_weighted=[0.632049062049062]
  f1_none=[0.8571428571428571,1,0.3333333333333333,0.6666666666666666,0.6666666666666

  jaccard_micro=[0.5]
  jaccard_macro=[0.4921428571428572]
  jaccard_weighted=[0.5056349206349205]
  jaccard_none=[0.75,1,0.2,0.5,0.5,0.6666666666666666,0.3333333333333333,0.4,0,0.5714

  precision_micro=[0.6666666666666666]
  precision_macro=[0.6571428571428571]
  precision_weighted=[0.6706349206349207]
  precision_none=[0.75,1,0.3333333333333333,1,0.75,0.6666666666666666,1,0.5,0,0.571428

  recall_micro=[0.6666666666666666]
  recall_macro=[0.6433333333333333]
  recall_weighted=[0.6666666666666666]
  recall_none=[1,1,0.3333333333333333,0.5,0.6,1,0.3333333333333333,0.6666666666666666,

  f1_bin=[0.4444444444444445]
  jaccard_bin=[0.2857142857142857]
  precision_bin=[0.5]
  recall_bin=[0.4]
*/
```

© 2000-2025, MetaQuotes Ltd.

ClassificationScore

Berechnung der Klassifizierungsmetrik, um die Qualität der vorhergesagten Daten im Vergleich zu den wahren Daten zu bewerten.

Im Gegensatz zu anderen Methoden im Abschnitt „Maschinelles Lernen“ gilt diese Methode für den Vektor der wahren Werte und nicht für den Vektor der vorhergesagten Werte.

```
vector vector::ClassificationScore(
   const matrix&               pred_scores,   // Matrix, die die Wahrscheinlichkeitsver
   ENUM_CLASSIFICATION_METRIC metric         // Typ der Metrik
   ENUM_AVERAGE_MODE           mode          // Modus der Durchschnittsbildung
   );


vector vector::ClassificationScore(
   const matrix&               pred_scores,   // Matrix, die die Wahrscheinlichkeitsver
   ENUM_CLASSIFICATION_METRIC metric         // Typ der Metrik
   int                         param         // zusätzliche Parameter
   );
```

Parameter

pred_scores

[in] Eine Matrix, die eine Reihe von horizontalen Vektoren mit Wahrscheinlichkeiten für jede Klasse enthält. Die Anzahl der Matrixzeilen sollte der Größe des Vektors der wahren Werte entsprechen.

metric

[in] Typ der Metrik aus der Enumeration ENUM_CLASSIFICATION_METRIC. Es werden die Werte CLASSIFICATION_TOP_K_ACCURACY, CLASSIFICATION_AVERAGE_PRECISION und CLASSIFICATION_ROC_AUC verwendet.

mode

[in] Der Modus der Durchschnittsermittlung aus der Enumeration ENUM_AVERAGE_MODE. Wird für die Metriken CLASSIFICATION_AVERAGE_PRECISION und CLASSIFICATION_ROC_AUC verwendet.

param

[in] Im Falle der Metrik CLASSIFICATION_TOP_K_ACCURACY ist anstelle des Mittelungsmodus der ganzzahlige K-Wert anzugeben.

Rückgabewert

Ein Vektor, der die berechnete Metrik enthält. Im Falle des Mittelungsmodus AVERAGE_NONE enthält der Vektor die Metrikwerte für jede Klasse ohne Mittelwertbildung. (Im Falle der binären Klassifizierung wären dies beispielsweise zwei Metriken für 'false' und 'true').

Hinweis zu den Modi der Mittelwertbildung

© 2000-2025, MetaQuotes Ltd.

AVERAGE_BINARY ist nur für die binäre Klassifizierung sinnvoll.

AVERAGE_MICRO - berechnet die Metriken global, indem jedes Element der Indikatormatrix mit Labelwerten als Label betrachtet wird. Die Indikatormatrix mit Labelwerten bezieht sich auf eine Matrix mit einer Reihe von Wahrscheinlichkeiten für jeden Labelwert.

AVERAGE_MACRO — berechnet die Metriken für jedes Label und ermittelt deren ungewichteten Mittelwert. Dabei wird ein Ungleichgewicht das Label nicht berücksichtigt.

AVERAGE_WEIGHTED — berechnet die Metriken für jedes Label und ermittelt ihren Durchschnitt gewichteten durch Unterstützung (der Anzahl der wahren Instanzen für jedes Label).

Hinweis

Im Falle einer binären Klassifikation können wir nicht nur eine n x 2-Matrix eingeben, bei der die erste Spalte die Wahrscheinlichkeiten für ein negatives Label und die zweite Spalte die Wahrscheinlichkeiten für ein positives Label enthält. Dies liegt daran, dass binäre Klassifizierungsmodelle entweder zwei Wahrscheinlichkeiten oder eine Wahrscheinlichkeit für eine positives Label liefern können.

Beispiel:

```
   vector y_true={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,8,4,2,7,6,8,4,2,3,6};
   //vector y_pred={7,2,1,0,4,1,4,9,5,9,0,6,9,0,1,5,9,7,3,4,2,9,4,9,5,9,2,7,7,0};

//--- label scores          0         1         2         3         4         5
   matrix y_scores={{0.000109, 0.000186, 0.000449, 0.000052, 0.000002, 0.000022, 0.000
                    {0.000091, 0.081956, 0.916816, 0.001106, 0.000006, 0.000002, 0.000
                    {0.000108, 0.972863, 0.003600, 0.000021, 0.010479, 0.000015, 0.000
                    {0.925425, 0.000080, 0.002913, 0.000057, 0.000274, 0.000638, 0.063
                    {0.000060, 0.000126, 0.000006, 0.000000, 0.993513, 0.000000, 0.000
                    {0.000016, 0.982124, 0.000045, 0.000002, 0.008445, 0.000001, 0.000
                    {0.000000, 0.000040, 0.000001, 0.000000, 0.989395, 0.000167, 0.000
                    {0.000795, 0.002938, 0.023447, 0.007418, 0.021838, 0.002476, 0.000
                    {0.000091, 0.000226, 0.000038, 0.000007, 0.000048, 0.854910, 0.068
                    {0.000000, 0.000000, 0.000000, 0.000000, 0.003004, 0.000000, 0.000
                    {0.998856, 0.000009, 0.000976, 0.000002, 0.000000, 0.000013, 0.000
                    {0.000178, 0.000446, 0.000326, 0.000033, 0.000193, 0.000071, 0.998
                    {0.000005, 0.000016, 0.000153, 0.000045, 0.004110, 0.000012, 0.000
                    {0.994188, 0.000003, 0.002584, 0.000005, 0.000005, 0.000100, 0.000
                    {0.000173, 0.990569, 0.000792, 0.000040, 0.001798, 0.000035, 0.000
                    {0.000000, 0.000537, 0.000008, 0.005080, 0.000046, 0.992910, 0.000
                    {0.000127, 0.000003, 0.000003, 0.000000, 0.001583, 0.000000, 0.000
                    {0.000001, 0.000012, 0.000072, 0.000020, 0.000000, 0.000000, 0.000
                    {0.000020, 0.000105, 0.001139, 0.901343, 0.002132, 0.083873, 0.000
                    {0.000002, 0.000048, 0.000019, 0.000000, 0.999347, 0.000002, 0.000
                    {0.000059, 0.001344, 0.612502, 0.002749, 0.000229, 0.000678, 0.000
                    {0.000586, 0.000740, 0.001625, 0.000007, 0.269341, 0.000076, 0.016
                    {0.009547, 0.018055, 0.283795, 0.071079, 0.426074, 0.082335, 0.036
                    {0.002506, 0.002545, 0.001148, 0.005659, 0.020416, 0.000112, 0.006
```

© 2000-2025, MetaQuotes Ltd.

```
                     {0.001263, 0.001769, 0.000293, 0.000011, 0.000302, 0.881768, 0.112
                     {0.002904, 0.002909, 0.013421, 0.001461, 0.007519, 0.001251, 0.000
                     {0.000055, 0.001080, 0.893158, 0.000000, 0.104492, 0.000159, 0.001
                     {0.000344, 0.002693, 0.071184, 0.000262, 0.000001, 0.000003, 0.000
                     {0.001404, 0.009375, 0.002638, 0.229189, 0.000064, 0.000896, 0.007
                     {0.491140, 0.000125, 0.000024, 0.000302, 0.000038, 0.034947, 0.473

   vector top_k=y_true.ClassificationScore(y_scores,CLASSIFICATION_TOP_K_ACCURACY,1);
   Print("top 1 accuracy score = ",top_k);
   top_k=y_true.ClassificationScore(y_scores,CLASSIFICATION_TOP_K_ACCURACY,2);
   Print("top 2 accuracy score = ",top_k);
   vector y_true2={0, 1, 2, 2};
   matrix y_score2={{0.5, 0.2, 0.2},  // 0 is in top 2
                    {0.3, 0.4, 0.2},  // 1 is in top 2
                    {0.2, 0.4, 0.3},  // 2 is in top 2
                    {0.7, 0.2, 0.1}}; // 2 isn't in top 2
   top_k=y_true2.ClassificationScore(y_score2,CLASSIFICATION_TOP_K_ACCURACY,2);
   Print("top k = ",top_k);
   Print("");

   vector ap_micro=y_true.ClassificationScore(y_scores,CLASSIFICATION_AVERAGE_PRECISIO
   Print("average precision score micro = ",ap_micro);
   vector ap_macro=y_true.ClassificationScore(y_scores,CLASSIFICATION_AVERAGE_PRECISIO
   Print("average precision score macro = ",ap_macro);
   vector ap_weighted=y_true.ClassificationScore(y_scores,CLASSIFICATION_AVERAGE_PREC
   Print("average precision score weighted = ",ap_weighted);
   vector ap_none=y_true.ClassificationScore(y_scores,CLASSIFICATION_AVERAGE_PRECISION
   Print("average precision score none = ",ap_none);
   Print("");

   vector area_micro=y_true.ClassificationScore(y_scores,CLASSIFICATION_ROC_AUC,AVERAG
   Print("roc auc score micro = ",area_micro);
   vector area_macro=y_true.ClassificationScore(y_scores,CLASSIFICATION_ROC_AUC,AVERAG
   Print("roc auc score macro = ",area_macro);
   vector area_weighted=y_true.ClassificationScore(y_scores,CLASSIFICATION_ROC_AUC,AVE
   Print("roc auc score weighted = ",area_weighted);
   vector area_none=y_true.ClassificationScore(y_scores,CLASSIFICATION_ROC_AUC,AVERAGE
   Print("roc auc score none = ",area_none);
   Print("");

//--- binary classification
   vector y_pred_bin={0,1,0,1,1,0,0,0,1};
   vector y_true_bin={1,0,0,0,1,0,1,1,1};
   vector y_score_true={0.3,0.7,0.1,0.6,0.9,0.0,0.4,0.2,0.8};
   matrix y_score1_bin(y_score_true.Size(),1);
   y_score1_bin.Col(y_score_true,0);
   matrix y_scores_bin={{0.7, 0.3},
                        {0.3, 0.7},
                        {0.9, 0.1},
```

© 2000-2025, MetaQuotes Ltd.

```
                        {0.4, 0.6},
                        {0.1, 0.9},
                        {1.0, 0.0},
                        {0.6, 0.4},
                        {0.8, 0.2},
                        {0.2, 0.8}};

   vector ap=y_true_bin.ClassificationScore(y_scores_bin,CLASSIFICATION_AVERAGE_PRECIS
   Print("average precision score binary = ",ap);
   vector ap2=y_true_bin.ClassificationScore(y_score1_bin,CLASSIFICATION_AVERAGE_PRECI
   Print("average precision score binary = ",ap2);
   vector ap3=y_true_bin.ClassificationScore(y_scores_bin,CLASSIFICATION_AVERAGE_PRECI
   Print("average precision score none = ",ap3);
   Print("");

   vector area=y_true_bin.ClassificationScore(y_scores_bin,CLASSIFICATION_ROC_AUC,AVER
   Print("roc auc score binary = ",area);
   vector area2=y_true_bin.ClassificationScore(y_score1_bin,CLASSIFICATION_ROC_AUC,AVE
   Print("roc auc score binary = ",area2);
   vector area3=y_true_bin.ClassificationScore(y_scores_bin,CLASSIFICATION_ROC_AUC,AVE
   Print("roc auc score none = ",area3);


/*
  top 1 accuracy score = [0.6666666666666666]
  top 2 accuracy score = [1]
  top k = [0.75]

  average precision score micro = [0.8513333333333333]
  average precision score macro = [0.9326666666666666]
  average precision score weighted = [0.9333333333333333]
  average precision score none = [1,1,0.7,1,0.9266666666666666,0.8333333333333333,1,0.

  roc auc score micro = [0.9839506172839506]
  roc auc score macro = [0.9892068783068803]
  roc auc score weighted = [0.9887354497354497]
  roc auc score none = [1,1,0.9506172839506173,1,0.984,0.9821428571428571,1,0.9753086

  average precision score binary = [0.7961904761904761]
  average precision score binary = [0.7961904761904761]
  average precision score none = [0.7678571428571428,0.7961904761904761]

  roc auc score binary = [0.7]
  roc auc score binary = [0.7]
  roc auc score none = [0.7,0.7]
*/
```

© 2000-2025, MetaQuotes Ltd.

PrecisionRecall

Berechnen der Werte, mit denen eine Precision-Recall-Kurve erstellt wird. Ähnlich wie ClassificationScore wird diese Methode auf den Vektor mit wahren Werte angewendet.

```
bool vector::PrecisionRecall(
   const Matrix&                 pred_scores,   // Matrix, die die Wahrscheinlichkeits
   const ENUM_ENUM_AVERAGE_MODE  mode           // Mittelungsmodus
   matrix&                       precision,     // berechnete Präzisionswerte für jede
   matrix&                       recall,        // berechnete „Recall"-Werte für jeder
   matrix&                       thresholds,    // Schwellenwerte in absteigender Reih
   );
```

Parameter

pred_scores

[in] Eine Matrix, die eine Reihe horizontaler Vektoren mit Wahrscheinlichkeiten für jede Klasse enthält. Die Anzahl der Matrixzeilen muss der Größe des Vektors der wahren Werte entsprechen.

mode

[in] Der Modus der Durchschnittsermittlung aus der Enumeration ENUM_AVERAGE_MODE. Es werden nur AVERAGE_NONE, AVERAGE_BINARY und AVERAGE_MICRO verwendet.

precision

[out] Eine Matrix mit den berechneten werten der Präzisionskurve. Wenn keine Mittelung angewendet wird (AVERAGE_NONE), entspricht die Anzahl der Zeilen in der Matrix der Anzahl der Modellklassen.Die Anzahl der Spalten entspricht der Größe des Vektors der wahren Werte (oder der Anzahl der Zeilen in der Wahrscheinlichkeitsverteilungsmatrix pred_score). Bei der Mikromittelung entspricht die Anzahl der Zeilen in der Matrix der Gesamtzahl der Schwellenwerte, ohne Duplikate.

recall

[out] Eine Matrix mit berechneten Werten der „Reacll-"-Kurve.

threshold

[out] Schwellenwertmatrix, die durch Sortieren der Wahrscheinlichkeitsmatrix erhalten wird.

Hinweis

Siehe die Hinweise zur Methode ClassificationScore.

Beispiel

Ein Beispiel für das Sammeln von Statistiken aus dem Modell mnist.onnx (99 % Genauigkeit).

```
//--- Daten für Klassifizierungsmetriken
   vectorf y_true(images);
   vectorf y_pred(images);
   matrixf y_scores(images,10);
//--- Ein- und Ausgabe
   matrixf image(28,28);
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorf result(10);

//--- Test
   for(int test=0; test<images; test++)
     {
      image=test_data[test].image;
      if(!OnnxRun(model,ONNX_DEFAULT,image,result))
        {
         Print("OnnxRun error ",GetLastError());
         break;
        }
      result.Activation(result,AF_SOFTMAX);
     //--- Erheben der Daten
      y_true[test]=(float)test_data[test].label;
      y_pred[test]=(float)result.ArgMax();
      y_scores.Row(result,test);
     }    }
```

Berechnung der Genauigkeit

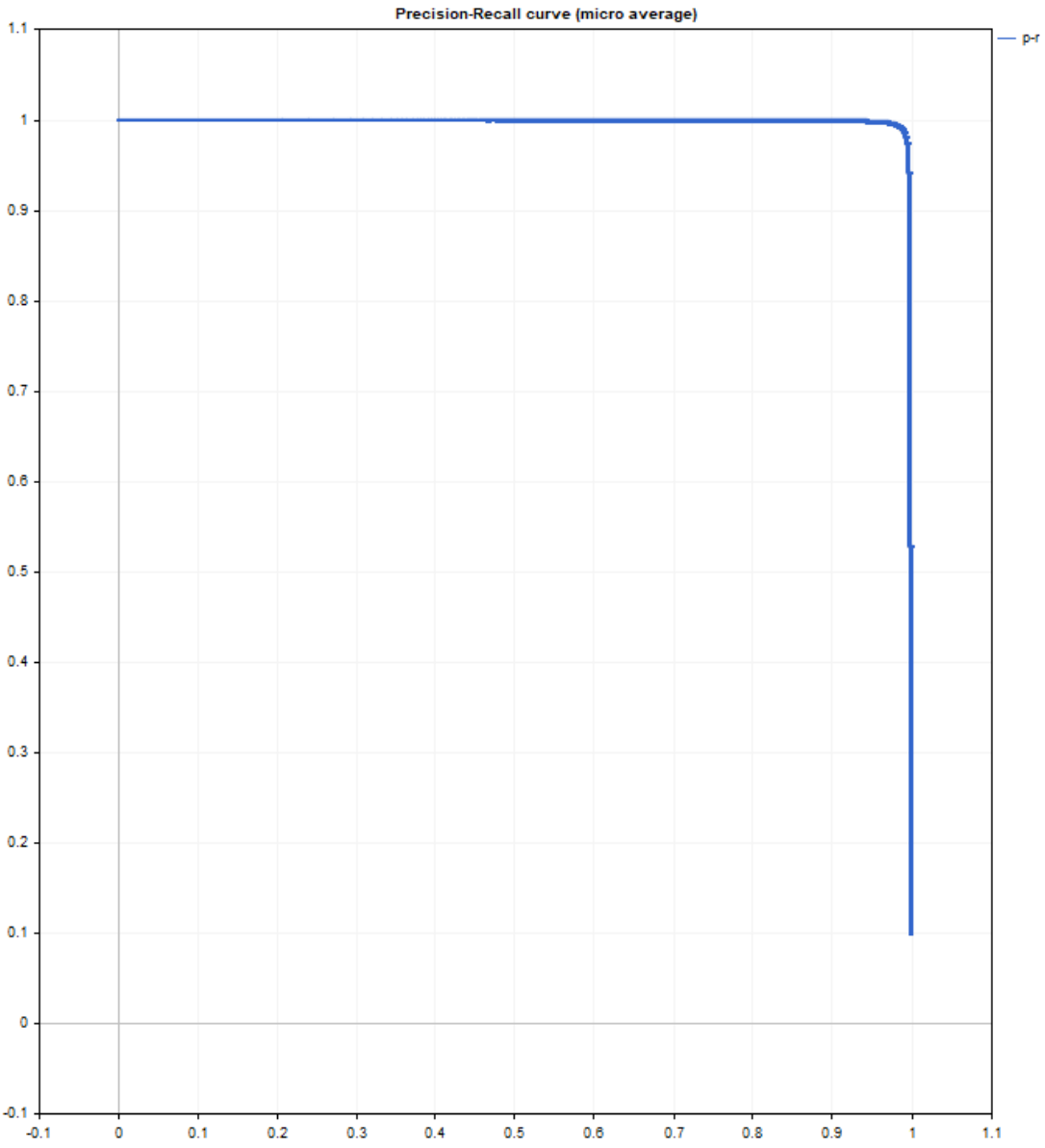
```
   vectorf accuracy=y_pred.ClassificationMetric(y_true,CLASSIFICATION_ACCURACY);
   PrintFormat("accuracy=%f",accuracy[0]);

accuracy=0.989000
```

Ein Beispiel für das Zeichnen des Diagramms von precision-recall, bei dem die Werte von „precision“ auf der y-Achse und die „Recall“-Werte auf der x-Achse aufgetragen werden. Außerdem werden Präzisions- und Recall-Diagramme separat dargestellt, wobei auf der x-Achse Schwellenwerte aufgetragen sind.

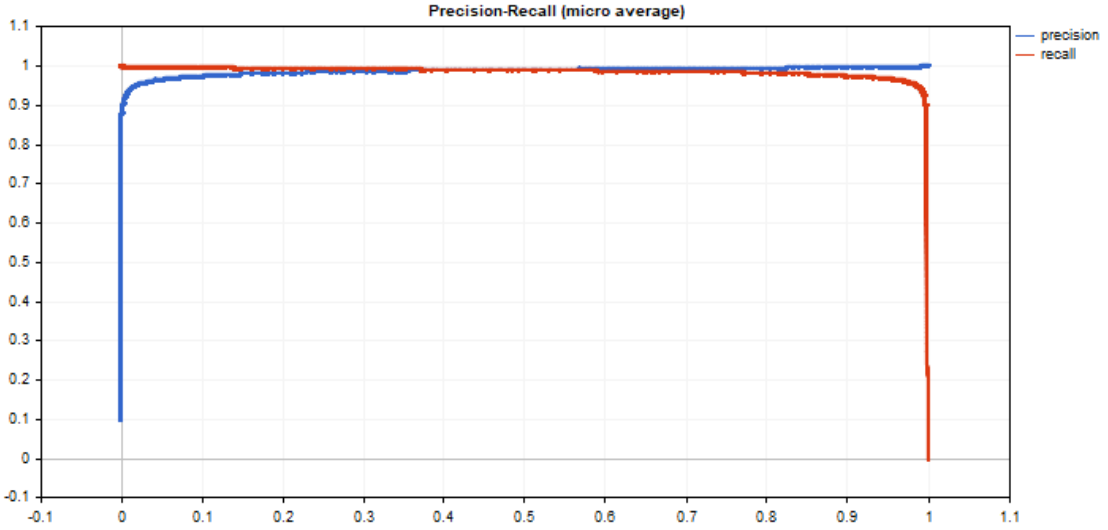
```
   if(y_true.PrecisionRecall(y_scores,AVERAGE_MICRO,mat_precision,mat_recall,mat_thres
     {
      double precision[],recall[],thres[];
      ArrayResize(precision,mat_thres.Cols());
      ArrayResize(recall,mat_thres.Cols());
      ArrayResize(thres,mat_thres.Cols());

      for(uint i=0; i<thres.Size(); i++)
        {
         precision[i]=mat_precision[0][i];
         recall[i]=mat_recall[0][i];
         thres[i]=mat_thres[0][i];
        }
      thres[0]=thres[1]+0.001;

      PlotCurve("Precision-Recall curve (micro average)","p-r","",recall,precision);
      Plot2Curves("Precision-Recall (micro average)","precision","recall",thres,precis
     }
```

© 2000-2025, MetaQuotes Ltd.

Resultierende Kurven:



© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

ReceiverOperatingCharacteristic

Berechnet werden die Werte, um die ROC-Kurve (Receiver Operating Characteristic) zu erstellen. Ähnlich wie ClassificationScore wird diese Methode auf den Vektor mit wahren Werte angewendet.

```
bool vector::ReceiverOperatingCharacteristic(
   const-Matrix&                 pred_scores,    // die Matrix, die die Wahrscheinlichk
   const ENUM_ENUM_AVERAGE_MODE  mode            // Mittelungsmodus
   matrix&                       fpr,            // berechnete Falsch-Positiv-Verhältni
   matrix&                       tpr,            // berechnete echte positive Verhältni
   matrix&                       thresholds,     // Schwellenwerte in absteigender Reih
   );
```

Parameter

pred_scores

[in] Eine Matrix, die eine Reihe horizontaler Vektoren mit Wahrscheinlichkeiten für jede Klasse enthält. Die Anzahl der Matrixzeilen muss der Größe des Vektors der wahren Werte entsprechen.

mode

[in] Der Modus der Durchschnittsermittlung aus der Enumeration ENUM_AVERAGE_MODE. Es werden nur AVERAGE_NONE, AVERAGE_BINARY und AVERAGE_MICRO verwendet.

Verhältnisw

[out] Eine Matrix mit berechneten Werten der Falsch-Positiv-Verhältniskurve. Wenn keine Mittelung angewendet wird (AVERAGE_NONE), entspricht die Anzahl der Zeilen in der Matrix der Anzahl der Modellklassen.Die Anzahl der Spalten entspricht der Größe des Vektors der wahren Werte (oder der Anzahl der Zeilen in der Wahrscheinlichkeitsverteilungsmatrix pred_score). Bei der Mikromittelung entspricht die Anzahl der Zeilen in der Matrix der Gesamtzahl der Schwellenwerte, ohne Duplikate.

tpr

[out] Eine Matrix mit berechneten Werten der echten positiven Verhältniskurve.

threshold

[out] Schwellenwertmatrix, die durch Sortieren der Wahrscheinlichkeitsmatrix erhalten wird.

Hinweis

Siehe die Hinweise zur Methode ClassificationScore.

Beispiel

Ein Beispiel für das Zeichnen von ROC-Diagrammen, bei dem tpr-Werte auf der y-Achse und fpr-Werte auf der x-Achse aufgetragen werden. Außerdem werden fpr- und tpr-Diagramme separat dargestellt, wobei auf der x-Achse Schwellenwerte aufgetragen sind

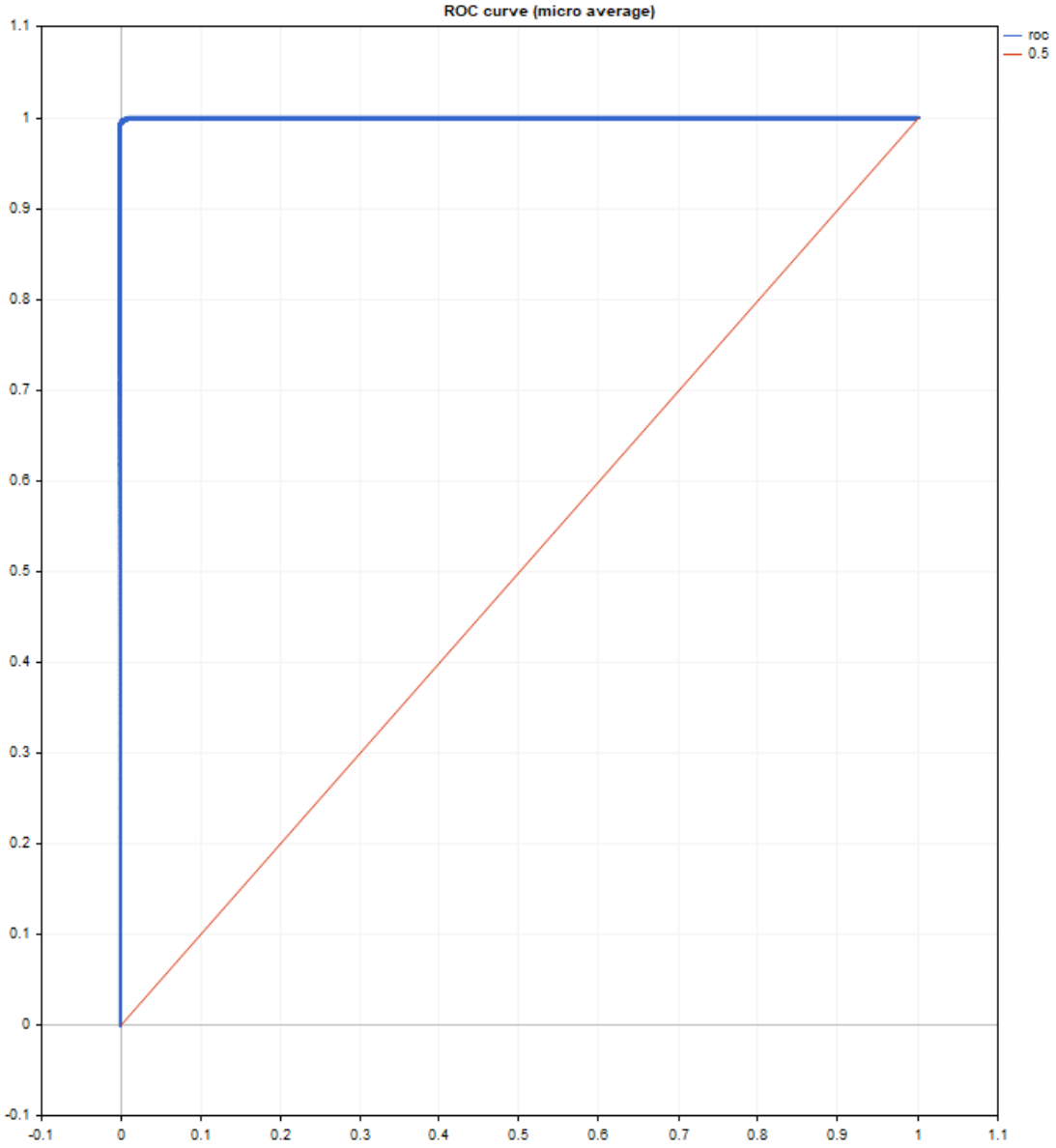
```
   matrixf mat_thres;
   matrixf mat_fpr;
   matrixf mat_tpr;
```

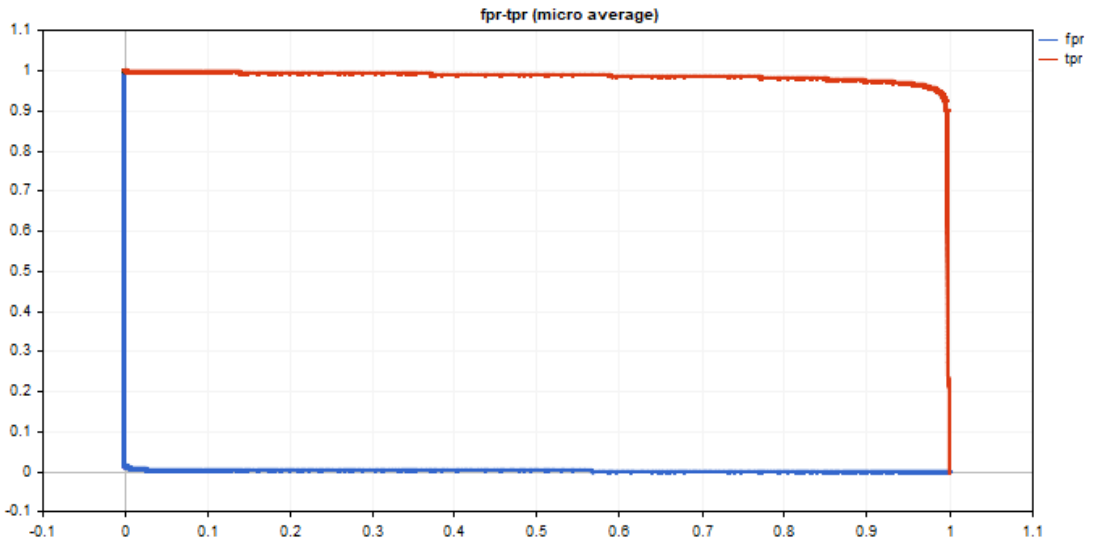
© 2000-2025, MetaQuotes Ltd.

```
if(y_true.ReceiverOperatingCharacteristic(y_scores,AVERAGE_MICRO,mat_fpr,mat_tpr,ma
  {
   double fpr[],tpr[],thres[];
   ArrayResize(fpr,mat_thres.Cols());
   ArrayResize(tpr,mat_thres.Cols());
   ArrayResize(thres,mat_thres.Cols());

   for(uint i=0; i<fpr.Size(); i++)
     {
      fpr[i]=mat_fpr[0][i];
      tpr[i]=mat_tpr[0][i];
      thres[i]=mat_thres[0][i];
     }
   thres[0]=thres[1]+0.001;

   PlotCurve("ROC curve (micro average)","roc","0.5",fpr,tpr);
   Plot2Curves("fpr-tpr (micro average)","fpr","tpr",thres,fpr,tpr);
  }
```

Resultierende Kurven:

© 2000-2025, MetaQuotes Ltd.





© 2000-2025, MetaQuotes Ltd.

Der Ausgabecode des Diagramms ist einfach und basiert auf der Standardbibliothek <Graphics/Graphic.mqh>.

Die Beispiele verwenden die Daten des Modells mnist.onnx. Der Code wird in der Methodenbeschreibung PrecisionRecall dargestellt.

ROC AUC ist nahezu ideal.

```
roc auc score micro = [0.99991]
```

© 2000-2025, MetaQuotes Ltd.

OpenBLAS Methods

OpenBLAS is a high-performance open-source linear algebra library that implements BLAS (Basic Linear Algebra Subprograms) and some LAPACK functions. OpenBLAS is designed to improve computational performance, particularly in matrix and vector operations, which are often used in scientific and engineering tasks such as machine learning, numerical methods, and simulations.

Key features of OpenBLAS:

- Multithreading support: OpenBLAS can efficiently use multiple processor cores for parallel computations, significantly accelerating operations on multiprocessor systems.
- Optimization for processor architectures: OpenBLAS includes optimized builds for various processors such as Intel, AMD, ARM and others. The library automatically detects processor characteristics and selects the most suitable function implementations.
- Extensive BLAS operation support: OpenBLAS implements core BLAS functions, including vector operations (e.g., vector addition and dot product), matrix operations (multiplication), and vector-matrix operations.
- LAPACK compatibility: The library supports LAPACK (Linear Algebra PACKage) functions for more complex linear algebra operations, such as solving systems of linear equations, calculating matrix eigenvalues, and others.
- High performance: Compared to other BLAS libraries, OpenBLAS often shows better results due to hand-optimization for specific processor architectures.

Applications

OpenBLAS is widely used in applications involving numerical computations:

- Training neural networks and other machine learning tasks.
- Scientific computing (e.g. modeling of physical processes).
- Processing and analyzing large amounts of data.

The library is integrated into many popular scientific software packages such as NumPy, SciPy, and TensorFlow, which rely on high-performance linear algebra operations.

OpenBLAS is an excellent choice for those seeking an open-source solution for high-performance computing, particularly when working with large matrices and vectors.

Function	Action
SingularValueDecompositionDC	Singular Value Decomposition, "divide-and-conquer" algorithm. This algorithm is considered the fastest among other SVD algorithms (lapack function GESDD).
SingularValueDecompositionQR	Singular Value Decomposition, QR algorithm. This algorithm is considered a classical SVD algorithm (lapack function GESVD).

© 2000-2025, MetaQuotes Ltd.

Function	Action
SingularValueDecompositionQRPivot	Singular Value Decomposition, QR with pivoting algorithm (lapack function GESVDQ).
SingularValueDecompositionBisect	Singular Value Decomposition, bisection algorithm (lapack function GESVDX).
SingularValueDecompositionJacobiHigh	Singular Value Decomposition, Jacobi high level algorithm (lapack function GEJSV).
SingularValueDecompositionJacobiLow	Singular Value Decomposition, Jacobi low level algorithm (lapack function GESVJ). The method computes small singular values and their singular vectors with much greater accuracy than other SVD routines in certain cases.
SingularValueDecompositionBidiagDC	Singular Value Decomposition, divide-and-conquer algorithm for bidiagonal matrices (lapack function BDSVDX).
SingularValueDecompositionBidiagBisect	Singular Value Decomposition, bisection algorithm for bidiagonal matrices (lapack function BDSVDX).
EigenSolver	Compute eigenvalues and eigenvectors of a regular square matrix using the classical algorithm (lapack function GEEV).
EigenSolver2	Compute generalized eigenvalues and eigenvectors for a pair of ordinary square matrices (lapack function GGEV).
EigenSolverX	Compute eigenvalues and eigenvectors of a regular square matrix in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GEEVX).
EigenSolverShur	Compute eigenvalues, upper triangular matrix in Schur form, and matrix of Schur vectors (lapack function GEES). See also Schur decomposition.
EigenSymmetricDC	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the divide-and-conquer algorithm (lapack functions SYEVD, HEEVD).
EigenSymmetricQR	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex

© 2000-2025, MetaQuotes Ltd.

Function	Action
	conjugate) matrix using the QR algorithm (lapack functions SYEV, HEEV).
EigenSymmetricRobust	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the Multiple Relatively Robust Representations, MRRR algorithm (lapack functions SYEVR, HEEVR).
EigenSymmetricBisect	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the bisection algorithm (lapack functions SYEVX, HEEVX).
SingularSpectrumAnalysisSpectrum	A method function for calculating the relative contributions of spectral components based on their eigenvalues.
SingularSpectrumAnalysisForecast	A method function for calculating reconstructed and predicted data using spectral components of the input time series.
SingularSpectrumAnalysisReconstructComponents	A method function for calculating reconstructed components of the input time series and their contributions.
SingularSpectrumAnalysisReconstructSeries	A method function for calculating the reconstructed time series using the first component_count components.

© 2000-2025, MetaQuotes Ltd.

Singular value decomposition

This section features functions for decomposing a matrix into three components: orthogonal matrices and a diagonal matrix of singular values. SVD is applied to solve various linear algebra problems such as data dimensionality reduction, image compression, solving systems of equations, and data analysis and optimization. The main functions allow you to compute singular values and vectors, reconstruct matrices, and approximate matrices with reduced rank accuracy.

Function	Action
SingularValueDecompositionDC	Singular Value Decomposition, "divide-and-conquer" algorithm. This algorithm is considered the fastest among other SVD algorithms (lapack function GESDD).
SingularValueDecompositionQR	Singular Value Decomposition, QR algorithm. This algorithm is considered a classical SVD algorithm (lapack function GESVD).
SingularValueDecompositionQRPivot	Singular Value Decomposition, QR with pivoting algorithm (lapack function GESVDQ).
SingularValueDecompositionBisect	Singular Value Decomposition, bisection algorithm (lapack function GESVDX).
SingularValueDecompositionJacobiHigh	Singular Value Decomposition, Jacobi high level algorithm (lapack function GEJSV).
SingularValueDecompositionJacobiLow	Singular Value Decomposition, Jacobi low level algorithm (lapack function GESVJ). The method computes small singular values and their singular vectors with much greater accuracy than other SVD routines in certain cases.
SingularValueDecompositionBidiagDC	Singular Value Decomposition, divide-and-conquer algorithm for bidiagonal matrices (lapack function BDSVDX).
SingularValueDecompositionBidiagBisect	Singular Value Decomposition, bisection algorithm for bidiagonal matrices (lapack function BDSVDX).
SingularValueDecompositionBidiagQR	Computes the singular value decomposition of a general matrix that has been reduced to bidiagonal form by the method ReduceToBidiagonal. Lapack function BDSQR.

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionDC

Singular Value Decomposition, "divide-and-conquer" algorithm. This algorithm is considered the fastest among other SVD algorithms (lapack function GESDD).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionDC(
   ENUM_SVD_Z      jobz,     // how to computed
   vector&         S,        // vector of computed singular values
   matrix&         U,        // matrix of computed left vectors U
   matrix&         VT        // transposed matrix of right vectors VT
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionDC(
   ENUM_SVD_Z      jobz,     // how to computed
   vectorf&        S,        // vector of computed singular values
   matrixf&        U,        // matrix of computed left vectors U
   matrixf&        VT        // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionDC(
   ENUM_SVD_Z      jobz,     // how to computed
   vector&         S,        // vector of computed singular values
   matrixc&        U,        // matrix of computed left vectors U
   matrixc&        VT        // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionDC(
   ENUM_SVD_Z      jobz,                  // how to computed
   vectorf&        singular_values,       // vector of computed singular values
   matrixcf&       U,                     // matrix of computed left vectors U
   matrixcf&       VT                     // transposed matrix of right vectors VT
   );
```

Parameters

jobz

[in] ENUM_SVD_Z enumeration value which determines the method for computing left and singular eigenvectors.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

© 2000-2025, MetaQuotes Ltd.

VT

[out] Matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the jobz parameter.

When jobv is set to SVDZ_N, the left and right vectors are not computed. Only singular values are computed.

When jobv is set to SVDZ_A, the full matrices of the U and VT vectors are computed.

When the value is SVDZ_S, truncated matrices of vectors U and VT are computed.

ENUM_SVD_Z

An enumeration defining the way to compute left and right singular vectors.

ID	Description
SVDZ_N	Columns U or rows VT are not computed
SVDZ_A	All M columns of U or all N columns of VT are returned in arrays U and VT
SVDZ_S	The first min(M,N) columns of U or the first min(M,N) columns of VT are returned in arrays U and VT

See also

SingularValueDecompositionQR, SingularValueDecompositionQRPivot

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionQR

Singular Value Decomposition, QR algorithm. Considered a classical SVD algorithm (lapack function GESVD).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionQR(
   ENUM_SVD_Z       jobu,      // how to compute left vectors
   ENUM_SVD_Z       jobv,      // how to compute right vectors
   vector&          S,         // vector of computed singular values
   matrix&          U,         // matrix of computed left vectors U
   matrix&          VT         // transposed matrix of right vectors VT
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionQR(
   ENUM_SVD_Z       jobu,      // how to compute left vectors
   ENUM_SVD_Z       jobv,      // how to compute right vectors
   vectorf&         S,         // vector of computed singular values
   matrixf&         U,         // matrix of computed left vectors U
   matrixf&         VT         // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionQR(
   ENUM_SVD_Z       jobu,      // how to compute left vectors
   ENUM_SVD_Z       jobv,      // how to compute right vectors
   vector           S,         // vector of computed singular values
   matrixc          U,         // matrix of computed left vectors U
   matrixc          VT         // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionQR(
   ENUM_SVD_Z       jobu,                  // how to compute left vectors
   ENUM_SVD_Z       jobv,                  // how to compute right vectors
   vectorf&         singular_values,       // vector of computed singular values
   matrixcf&        u,                     // matrix of computed left vectors U
   matrixcf&        vt                     // transposed matrix of right vectors VT
   );
```

Parameters

jobu

[in] ENUM_SVD_Z enumeration value defining how the left singular vectors should be computed.

jobv

© 2000-2025, MetaQuotes Ltd.

[in] ENUM_SVD_Z enumeration value defining how the right singular vectors should be computed.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

VT

[out] Matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the jobu and jobv parameters.

When set to SVDZ_N, the left (jobu) and right (jobv) vectors are not computed. Singular values are always computed.

When set to SVDZ_A, the full matrices of vectors U (jobu) or VT (jobv) are computed.

With the value SVDZ_S, truncated matrices of vectors U (jobu) or VT (jobv) are computed.

ENUM_SVD_Z

An enumeration defining the way to compute left and right singular vectors.

ID	Description
SVDZ_N	Columns U or rows VT are not computed
SVDZ_A	All M columns of U or all N columns of VT are returned in arrays U and VT
SVDZ_S	The first min(M,N) columns of U or the first min(M,N) columns of VT are returned in arrays U and VT

See also

SingularValueDecompositionDC, SingularValueDecompositionQRPivot

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionQRPivot

Singular Value Decomposition, QR with pivoting algorithm (lapack function GESVDQ).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionQRPivot(
   ENUM_SVDQRP_A   joba,     // computation accuracy level
   ENUM_SVDQRP_P   jobp,     // use row reversal to compute
   ENUM_SVDQRP_R   jobr,     // use triangular matrix R to compute
   ENUM_SVDQRP_U   jobu,     // how to compute left vectors
   ENUM_SVDQRP_V   jobv,     // how to compute right vectors
   vector&         S,        // vector of computed singular values
   matrix&         U,        // matrix of computed left vectors U
   matrix&         VT        // transposed matrix of right vectors VT
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionQRPivot(
   ENUM_SVDQRP_A   joba,     // computation accuracy level
   ENUM_SVDQRP_P   jobp,     // use row reversal to compute
   ENUM_SVDQRP_R   jobr,     // use triangular matrix R to compute
   ENUM_SVDQRP_U   jobu,     // how to compute left vectors
   ENUM_SVDQRP_V   jobv,     // how to compute right vectors
   vectorf&        S,        // vector of computed singular values
   matrixf&        U,        // matrix of computed left vectors U
   matrixf&        VT        // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionQRPivot(
   ENUM_SVDQRP_A   joba,     // computation accuracy level
   ENUM_SVDQRP_P   jobp,     // use row reversal to compute
   ENUM_SVDQRP_R   jobr,     // use triangular matrix R to compute
   ENUM_SVDQRP_U   jobu,     // how to compute left vectors
   ENUM_SVDQRP_V   jobv,     // how to compute right vectors
   vector&         S,        // vector of computed singular values
   matrixc&        U,        // matrix of computed left vectors U
   matrixc&        VT        // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionQRPivot(
   ENUM_SVDQRP_A   joba,                   // computation accuracy level
   ENUM_SVDQRP_P   jobp,                   // use row reversal to compute
   ENUM_SVDQRP_R   jobr,                   // use triangular matrix R to compute
   ENUM_SVDQRP_U   jobu,                   // how to compute left vectors
```

© 2000-2025, MetaQuotes Ltd.

```
   ENUM_SVDQRP_V   jobv,                  // how to compute right vectors
   vectorf&        singular_values,       // vector of computed singular values
   matrixcf&       u,                     // matrix of computed left vectors U
   matrixcf&       vt                     // transposed matrix of right vectors VT
   );
```

Parameters

joba

[in] ENUM_SVDQRP_A enumeration value defining the accuracy level of the SVD computation.

jobp

[in] ENUM_SVDQRP_P enumeration value defining the use of row reversal during the computation process.

jobr

[in] ENUM_SVDQRP_R enumeration value defining whether to transpose the triangular matrix R obtained as a result of the initial QR factorization.

jobu

[in] ENUM_SVDQRP_U enumeration value that determines how the left singular vectors should be computed.

jobv

[in] ENUM_SVDQRP_V enumeration value defining how the right singular vectors should be computed.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

VT

[out] Matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The number of matrix rows must not be less than the number of columns.

If both left and right singular vectors are computed, jobv must be set to SVDQRPV_R when jobu=SVDQRPU_R.

ENUM_SVDQRP_A

An enumeration that specifies the level of accuracy of the SVD computation.

© 2000-2025, MetaQuotes Ltd.

ID	Description
SVDQRP_A	The requested accuracy corresponds to the inverse error limited by epsilon. This is an aggressive level of truncation.
SVDQRPA_M	Similar to SVDQRP_A, but the truncation is softer. This is the average level of truncation.
SVDQRPA_H	High accuracy is required.

ENUM_SVDQRP_P

An enumeration that specifies whether to use row reversal during calculation.

ID	Description
SVDQRPP_P	The rows of A are ordered in descending order. Recommended for numerical reliability.
SVDQRPP_N	No row reversal.

ENUM_SVDQRP_R

An enumeration that specifies whether to transpose the triangular matrix R obtained as a result of the initial QR factorization.

ID	Description
SVDQRPR_T	After the initial rotated QR factorization, GESVD is applied to the transpose R**T of the calculated triangular factor R.
SVDQRPR_N	The triangular factor R is given as input to GESVD.

ENUM_SVDQRP_U

An enumeration defining how left singular vectors should be computed.

ID	Description
SVDQRPU_A	All M left singular vectors are computed.
SVDQRPU_S	The min(M,N) left singular vectors are computed.
SVDQRPU_R	The numerical rank NUMRANK is determined and only the NUMRANK of the left singular vectors is computed.
SVDQRPU_F	N left singular vectors are returned.
SVDQRPU_N	Left singular vectors are not computed.

ENUM_SVDQRP_V

An enumeration defining how right singular vectors should be computed.

© 2000-2025, MetaQuotes Ltd.

ID	Description
SVDQRPV_A	All N right singular vectors are computed.
SVDQRPV_R	The numerical rank NUMRANK is defined and only the NUMRANK of right singular vectors are computed.
SVDQRPV_N	Right singular vectors are not computed.

See also

SingularValueDecompositionDC, SingularValueDecompositionQR

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionBisect

Singular Value Decomposition, bisection algorithm (lapack function GESVDX).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionBisect(
   ENUM_SVD_VECTORS     jobv,      // computation accuracy level
   ENUM_BLAS_RANGE      range,     // subset of computable singular values
   double               lower,     // lower limit of the subset
   double               upper,     // upper limit of the subset
   vector&              S,         // vector of computed singular values
   matrix&              U,         // matrix of computed left vectors U
   matrix&              VT         // transposed matrix of right vectors VT
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionBisect(
   ENUM_SVD_VECTORS     jobv,      // computation accuracy level
   ENUM_BLAS_RANGE      range,     // subset of computable singular values
   double               lower,     // lower limit of the subset
   double               upper,     // upper limit of the subset
   vectorf&             S,         // vector of computed singular values
   matrixf&             U,         // matrix of computed left vectors U
   matrixf&             VT         // ransposed matrix of right vectors VT
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionBisect(
   ENUM_SVD_VECTORS     jobv,      // computation accuracy level
   ENUM_BLAS_RANGE      range,     // subset of computable singular values
   double               lower,     // lower limit of the subset
   double               upper,     // upper limit of the subset
   vector               S,         // vector of computed singular values
   matrixc              U,         // matrix of computed left vectors U
   matrixc              VT         // transposed matrix of right vectors VT
   );
```

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionBisect(
   ENUM_SVD_VECTORS     jobv,              // computation accuracy level
   ENUM_BLAS_RANGE      range,             // subset of computable singular values
   double               lower,             // lower limit of the subset
   double               upper,             // upper limit of the subset
   vectorf&             singular_values,   // vector of computed singular values
   matrixcf&            u,                 // matrix of computed left vectors U
   matrixcf&            vt                 // transposed matrix of right vectors VT
```

© 2000-2025, MetaQuotes Ltd.

```
   );
```

Parameters

jobv

[in] ENUM_SVD_VECTORS enumeration value which determines the method for computing left and singular eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable singular values and vectors.

lower

[in] The lower limit of singular values subset; specified depending on the value of the range parameter.

upper

[in] The upper limit of singular values subset; specified depending on the value of the range parameter.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

VT

[out] Matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobuv* and *range* parameters.

When BLASRANGE_A is set, all singular values are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those singular values (and their vectors) that fall within the range of real values specified by the 'lower' and 'upper' parameters are computed.

With the BLASRANGE_I value, only those singular values (and their vectors) that fall within the range of integer indices specified by the 'lower' and 'upper' parameters are computed. For example, with lower=0 and upper=2, only the first three singular values are computed.

ENUM_SVD_VECTORS

An enumeration defining the way to compute left and right singular vectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
SVDVECTORS_N	Only singular values are computed, without vectors.
SVDVECTORS_U	Left singular vectors are computed.
SVDVECTORS_V	Right singular vectors are computed.
SVDVECTORS_UV	Left and right singular vectors are computed.

ENUM_BLAS_RANGE

An enumeration defining a subset of computable singular values and vectors.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	Singular or eigenvalues from IL to IU will be found

See also

SingularValueDecompositionDC, SingularValueDecompositionQR, SingularValueDecompositionQRPivot

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionJacobiHigh

Singular Value Decomposition, Jacobi high level algorithm (lapack function GEJSV).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionJacobiHigh(
   ENUM_SVDJH_A   joba,            // computation accuracy level
   ENUM_SVDJH_U   jobu,            // how to compute left vectors
   ENUM_SVDJH_V   jobv,            // how to compute right vectors
   ENUM_SVDJH_R   jobr,            // define range of computable singular values
   ENUM_SVDJH_T   jobt,            // define whether to transpose when computing a squ
   ENUM_SVDJH_P   jobp,            // the possibility of structured perturbations to r
   vector&        S,               // vector of computed singular values
   matrix&        U,               // matrix of computed left vectors U
   matrix&        V,               // matrix of computed left vectors V
   vector&        work_results     // additional computation results
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionJacobiHigh(
   ENUM_SVDJH_A   joba,            // computation accuracy level
   ENUM_SVDJH_U   jobu,            // how to compute left vectors
   ENUM_SVDJH_V   jobv,            // how to compute right vectors
   ENUM_SVDJH_R   jobr,            // define range of computable singular values
   ENUM_SVDJH_T   jobt,            // define whether to transpose when computing a squ
   ENUM_SVDJH_P   jobp,            // the possibility of structured perturbations to r
   vectorf&       S,               // vector of computed singular values
   matrixf&       U,               // U matrix of computed left vectors
   matrixf&       V,               // V matrix of computed left vectors
   vector&        work_results     // additional computation results
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionJacobiHigh(
   ENUM_SVDJH_A   joba,            // computation accuracy level
   ENUM_SVDJH_U   jobu,            // how to compute left vectors
   ENUM_SVDJH_V   jobv,            // how to compute right vectors
   ENUM_SVDJH_R   jobr,            // define range of computable singular values
   ENUM_SVDJH_T   jobt,            // define whether to transpose when computing a squ
   ENUM_SVDJH_P   jobp,            // the possibility of structured perturbations to r
   vectorc&       S,               // vector of computed singular values
   matrixc&       U,               // U matrix of computed left vectors
   matrixc&       V,               // V matrix of computed left vectors
   vector&        work_results     // additional computation results
   );
```

© 2000-2025, MetaQuotes Ltd.

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionJacobiHigh(
   ENUM_SVDJH_A    joba,                  // computation accuracy level
   ENUM_SVDJH_U    jobu,                  // how to compute left vectors
   ENUM_SVDJH_V    jobv,                  // how to compute right vectors
   ENUM_SVDJH_R    jobr,                  // define range of computable singular valu
   ENUM_SVDJH_T    jobt,                  // define whether to transpose when computi
   ENUM_SVDJH_P    jobp,                  // the possibility of structured perturbati
   vectorf&        singular_values,       // vector of computed singular values S
   matrixcf&       u,                     // matrix of computed left vectors U
   matrixcf&       v,                     // matrix of computed left vectors V
   vectorf&        work_results           // additional computation results
   );
```

Parameters

joba

[in] ENUM_SVDJH_A enumeration value determining the accuracy level of the SVD computation.

jobu

[in] ENUM_SVDJH_U enumeration value that determines how the left singular vectors should be computed.

jobv

[in] ENUM_SVDJH_V enumeration value that determines how the right singular vectors should be computed.

jobr

[in] ENUM_SVDJH_R enumeration value defining the range of computable values

jobt

[in] ENUM_SVDJH_T enumeration value defining whether to transpose the matrix of it is square. If the matrix is non-square, this parameter is ignored.

jobp

[in] ENUM_SVDJH_P enumeration value defining the possibility of structured perturbations to remove denormalized values.

S

[out] Vector of singular values.

For work(1)/work(2) = one: the singular values of A. During the computation S contains Euclidean column norms of the iterated matrices in the array a.

For work(1)≠work(2): the singular values of A are (work(1)/work(2)) * S(1:n). This factored form is used if sigma_max(A) overflows or if small singular values have been saved from underflow by scaling the input matrix A.

jobr = 'R', some of the singular values may be returned as exact zeros obtained by 'setting to zero' because they are below the numerical rank threshold or are denormalized numbers.

U

© 2000-2025, MetaQuotes Ltd.

[out] Matrix of left singular vectors.

V

[out] Matrix of right singular vectors.

work_results

[out] Vector consisting of 7 statistics obtained as a result of the computation.

work(1) = scale = work(2)/work(1) is the scaling factor such that scale*sva(1:n) are the computed singular values of A. See the description of S.

work(2) = see the description of work(1).

work(3) = sconda is an estimate for the condition number of column equilibrated A. If joba = 'E' or 'G', sconda is an estimate of sqrt(||(R**t * R)**(-1)||_1). It is computed using ?pocon. It holds n**(-1/4) * sconda ≤ ||R**(-1)||_2 ≤ n**(1/4) * sconda, where R is the triangular factor from the QRF of A. However, if R is truncated and the numerical rank is determined to be strictly smaller than n, sconda is returned as -1, indicating that the smallest singular values might be lost.

If full SVD is needed, the following two condition numbers are useful for the analysis of the algorithm. They are provied for a user who is familiar with the details of the method.

work(4) = an estimate of the scaled condition number of the triangular factor in the first QR factorization.

work(5) = an estimate of the scaled condition number of the triangular factor in the second QR factorization.

The following two parameters are computed if jobt = 'T'. They are provided for a user who is familiar with the details of the method.

work(6) = the entropy of A**t*A :: this is the Shannon entropy of diag(A**t*A) / Trace(A**t*A) taken as point in the probability simplex.

work(7) = the entropy of A*A**t.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The number of matrix rows must not be less than the number of columns.

ENUM_SVDJH_A

An enumeration that specifies the level of accuracy of the SVD computation.

ID	Description
SVDJHA_C	Computation as with 'C' with an additional estimate of the condition number. It provides a realistic error bound.
SVDJHA_E	Computation as with SVDJHA_C' with an additional estimate of the condition number. It provides a realistic error bound.

© 2000-2025, MetaQuotes Ltd.

ID	Description
SVDJHA_F	Higher accuracy than the SVDJHA_C option.
SVDJHA_G	Computation as with SVDJHA_F with an additional estimate of the condition number.
SVDJHA_A	Small singular values are the noise and the matrix is treated as numerically rank deficient.
SVDJHA_R	Similar as in SVDJHA_A, but more accuracy.

ENUM_SVDJH_U

An enumeration defining how left singular vectors should be computed.

ID	Description
SVDJHU_U	N columns of U are returned in the array U.
SVDJHU_F	Full set of M left singular vectors is returned in the array U.
SVDJHU_N	U is not computed.

ENUM_SVDJH_V

An enumeration defining how right singular vectors should be computed.

ID	Description
SVDJHV_V	N columns of V are returned in the array V
SVDJHV_J	N columns of V are returned in the array V, but they are computed as the product of Jacobi rotations
SVDJHV_N	V is not computed

ENUM_SVDJH_R

An enumeration that defines the range of values to be computed.

ID	Description
SVDJHR_N	Do not kill small columns of c*A.
SVDJHR_R	RESTRICTED range for sigma(c*A). This option is recommended.

ENUM_SVDJH_T

An enumeration that specifies whether a matrix should be transposed if it is square.

© 2000-2025, MetaQuotes Ltd.

ID	Description
SVDJHT_T	Transpose if entropy test indicates possibly faster convergence of Jacobi process.
SVDJHT_N	Do not use transposition. Do not speculate.

ENUM_SVDJH_P

An enumeration that specifies the possibility of structured perturbations to remove denormalized values.

ID	Description
SVDJHP_P	Introduce perturbation.
SVDJHP_N	Do not perturb.

See also

SingularValueDecompositionDC, SingularValueDecompositionQR

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionJacobiLow

Singular Value Decomposition, Jacobi low level algorithm (lapack function GESVJ). The method computes small singular values and their singular vectors with much greater accuracy than other SVD routines in certain cases.

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionJacobiLow(
   ENUM_SVDJH_U   jobu,            // how to compute left vectors
   ENUM_SVDJH_V   jobv,            // how to compute right vectors
   double         ctol,            // threshold for convergence if jobu='C'
   ulong          mv,              // number of first rows of matrix V if jobv='A'
   vector&        S,               // vector of computed singular values
   matrix&        U,               // matrix of computed left vectors U
   matrix&        V,               // matrix of computed left vectors V
   vector&        work_results     // additional computation results
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionJacobiLow(
   ENUM_SVDJH_U   jobu,            // how to compute left vectors
   ENUM_SVDJH_V   jobv,            // how to compute right vectors
   double         ctol,            // threshold for convergence if jobu='C'
   ulong          mv,              // number of first rows of matrix V if jobv='A'
   vectorf&       S,               // vector of computed singular values
   matrixf&       U,               // U matrix of computed left vectors
   matrixf&       V,               // V matrix of computed left vectors
   vector&        work_results     // additional computation results
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionJacobiLow(
   ENUM_SVDJH_U   jobu,            // how to compute left vectors
   ENUM_SVDJH_V   jobv,            // how to compute right vectors
   double         ctol,            // threshold for convergence if jobu='C'
   ulong          mv,              // number of first rows of matrix V if jobv='A'
   vector&        S,               // vector of computed singular values
   matrixc&       U,               // U matrix of computed left vectors
   matrixc&       V,               // V matrix of computed left vectors
   vector&        work_results     // additional computation results
   );
```

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionJacobiLow(
   ENUM_SVDJH_U    jobu,                  // how to compute left vectors
   ENUM_SVDJH_V    jobv,                  // how to compute right vectors
```

© 2000-2025, MetaQuotes Ltd.

```
   double          ctol,                  // threshold for convergence if jobu='C'
   ulong           mv,                    // number of first rows of matrix V if jobv
   vectorf&        singular_values,       // vector of computed singular values S
   matrixcf&       u,                     // matrix of computed left vectors U
   matrixcf&       v,                     // matrix of computed left vectors V
   vectorf&        work_results           // additional computation results
   );
```

Parameters

jobu

[in] ENUM_SVDJL_U enumeration value that determines how the left singular vectors should be computed.

jobv

[in] ENUM_SVDJL_V enumeration value defining how the right singular vectors should be computed.

ctol

[in] Convergence threshold if jobu=SVDJLU_C. For other values of 'jobu' the parameter is ignored.

mv

[in] Number of rows of matrix V to be computed if jobv=SVDJLV_A. For other values of 'jobv' the parameter is ignored.

S

[out] Vector of singular values.

Depending on the value scale = work(1), where scale is the scaling factor:

if scale = 1, S(1:n) contains the computed singular values of a. During the computation, sva contains the Euclidean column norms of the iterated matrices in the array a.

if scale ≠ 1, the singular values of a are scale*S(1:n), and this factored representation is due to the fact that some of the singular values of a might underflow or overflow.

U

[out] Matrix of left singular vectors.

V

[out] Matrix of right singular vectors (non-transposed).

work_results

[out] Vector consisting of 7 statistics obtained as a result of the computation.

work(1) = scale is the scaling factor such that scale*S(1:n) are the computed singular values of A. See the description of S).

work(2) is the number of the computed nonzero singular value.

work(3) is the number of the computed singular values that are larger than the underflow threshold.

work(4) is the number of sweeps of Jacobi rotations needed for numerical convergence.

© 2000-2025, MetaQuotes Ltd.

work(5) = max_{i.NE.j} |COS(A(:,i),A(:,j))| in the last sweep. This is useful information in cases when ?gesvj did not converge, as it can be used to estimate whether the output is still useful and for post festum analysis.

work(6) is the largest absolute value over all sines of the Jacobi rotation angles in the last sweep. It can be useful in a post festum analysis.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The number of matrix rows must not be less than the number of columns.

ENUM_SVDJL_U

An enumeration defining how left singular vectors should be computed.

ID	Description
SVDJLU_U	The left singular vectors corresponding to the nonzero singular values are computed and returned in the leading columns of A
SVDJLU_C	Analogous to SVDJLU_U, except that user can control the level of numerical orthogonality of the computed left singular vectors.
SVDJLU_N	The matrix U is not computed.

ENUM_SVDJL_V

An enumeration defining how right singular vectors should be computed.

ID	Description
SVDJLV_V	The matrix V is computed
SVDJLV_A	The Jacobi rotations are applied to the MV-by-N matrix V.
SVDJLV_N	The matrix V is not computed.

See also

SingularValueDecompositionDC, SingularValueDecompositionQR

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionBidiagDC

Singular Value Decomposition, divide-and-conquer algorithm for bidiagonal matrices (lapack function BDSDC).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionBidiagDC(
   ENUM_SVDBIDIAG_Z     jobz,        // how to compute left vectors
   ENUM_BLAS_RANGE      range,       // subset of computed singular values
   double               lower,       // lower limit of the subset
   double               upper,       // upper limit of the subset
   vector&              S,           // vector of computed singular values
   matrix&              U,           // U matrix of computed left vectors
   matrix&              VT           // VT transposed matrix of right vectors
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionBidiagDC(
   ENUM_SVDBIDIAG_Z     jobz,        // how to compute left vectors
   ENUM_BLAS_RANGE      range,       // subset of computed singular values
   double               lower,       // lower limit of the subset
   double               upper,       // upper limit of the subset
   vectorf&             S,           // vector of computed singular values
   matrixf&             U,           // U matrix of computed left vectors
   matrixf&             VT           // VT transposed matrix of right vectors
   );
```

Parameters

jobz

[in] ENUM_SVDBIDIAG_Z enumeration value that determines how the left singular vectors should be computed.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable singular values and vectors.

lower

[in] The lower limit of singular values subset; specified depending on the value of the range parameter.

upper

[in] The upper limit of singular values subset; specified depending on the value of the range parameter.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

© 2000-2025, MetaQuotes Ltd.

V

[out] Transposed matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the jobz and range parameters.

When BLASRANGE_A is set, all singular values are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those singular values (and their vectors) that fall within the range of real values specified by the 'lower' and 'upper' parameters are computed.

With the BLASRANGE_I value, only those singular values (and their vectors) that fall within the range of integer indices specified by the 'lower' and 'upper' parameters are computed. For example, with lower=0 and upper=2, only the first three singular values are computed.

A bidiagonal matrix is a square matrix with non-zero main diagonal and one of the sub-diagonals.

Upper bidiagonal matrix

```
[[x, x, 0, 0, 0],
 [0, x, x, 0, 0],
 [0, 0, x, x, 0],
 [0, 0, 0, x, x],
 [0, 0, 0, 0, x]]
```

Lower bidiagonal matrix

```
[[x, 0, 0, 0, 0],
 [x, x, 0, 0, 0],
 [0, x, x, 0, 0],
 [0, 0, x, x, 0],
 [0, 0, 0, x, x]]
```

ENUM_SVDBIDIAG_Z

An enumeration defining how left singular vectors should be computed.

ID	Description
SVDJOBZ_V	Compute singular values and singular vectors.
SVDJOBZ_N	Compute singular values only.

ENUM_BLAS_RANGE

© 2000-2025, MetaQuotes Ltd.

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

See also

SingularValueDecompositionDC, SingularValueDecompositionQR

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionBidiagBisect

Singular Value Decomposition, bisection algorithm for bidiagonal matrices (lapack function BDSVDX).

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionBidiagBisect(
   ENUM_SVDBIDIAG_Z    jobz,       // how to compute left vectors
   ENUM_BLAS_RANGE     range,      // subset of computed singular values
   double              lower,      // lower limit of the subset
   double              upper,      // upper limit of the subset
   vector&             S,          // vector of computed singular values
   matrix&             U,          // U matrix of computed left vectors
   matrix&             VT          // VT transposed matrix of right vectors
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionBidiagBisect(
   ENUM_SVDBIDIAG_Z    jobz,       // how to compute left vectors
   ENUM_BLAS_RANGE     range,      // subset of computed singular values
   double              lower,      // lower limit of the subset
   double              upper,      // upper limit of the subset
   vectorf&            S,          // vector of computed singular values
   matrixf&            U,          // U matrix of computed left vectors
   matrixf&            VT          // VT transposed matrix of right vectors
   );
```

Parameters

jobz

[in] ENUM_SVDBIDIAG_Z enumeration value that determines how the left singular vectors should be computed.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable singular values and vectors.

lower

[in] The lower limit of singular values subset; specified depending on the value of the range parameter.

upper

[in] The upper limit of singular values subset; specified depending on the value of the range parameter.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

© 2000-2025, MetaQuotes Ltd.

V

[out] Transposed matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

When BLASRANGE_A is set, all singular values are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those singular values (and their vectors) that fall within the range of real values specified by the 'lower' and 'upper' parameters are computed.

With the BLASRANGE_I value, only those singular values (and their vectors) that fall within the range of integer indices specified by the 'lower' and 'upper' parameters are computed. For example, with lower=0 and upper=2, only the first three singular values are computed.

A bidiagonal matrix is a square matrix with non-zero main diagonal and one of the sub-diagonals.

Upper bidiagonal matrix

```
[[x, x, 0, 0, 0],
 [0, x, x, 0, 0],
 [0, 0, x, x, 0],
 [0, 0, 0, x, x],
 [0, 0, 0, 0, x]]
```

Lower bidiagonal matrix

```
[[x, 0, 0, 0, 0],
 [x, x, 0, 0, 0],
 [0, x, x, 0, 0],
 [0, 0, x, x, 0],
 [0, 0, 0, x, x]]
```

ENUM_SVDBIDIAG_Z

An enumeration defining how left singular vectors should be computed.

ID	Description
SVDJOBZ_V	Compute singular values and singular vectors.
SVDJOBZ_N	Compute singular values only.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

© 2000-2025, MetaQuotes Ltd.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

See also

SingularValueDecompositionDC, SingularValueDecompositionQR

© 2000-2025, MetaQuotes Ltd.

SingularValueDecompositionBidiagQR

Computes the singular value decomposition of a general matrix that has been reduced to bidiagonal form by the method ReduceToBidiagonal. Lapack function BDSQR.

Computing for type matrix<double>

```
bool  matrix::SingularValueDecompositionBidiagQR(
   matrix&          Q,          // orthogonal matrix Q
   matrix&          PT,         // transposed matrix P
   vector&          S,          // vector of computed singular values
   matrix&          U,          // matrix U of computed left vectors
   matrix&          VT          // transposed matrix V of right vectors
   );
```

Computing for type matrix<float>

```
bool  matrix::SingularValueDecompositionBidiagQR(
   matrix&          Q,          // orthogonal matrix Q
   matrix&          PT,         // transposed matrix P
   vectorf&         S,          // vector of computed singular values
   matrixf&         U,          // matrix U of computed left vectors
   matrixf&         VT          // transposed matrix V of right vectors
   );
```

Computing for type matrix<complex>

```
bool  matrix::SingularValueDecompositionBidiagQR(
   matrix&          Q,          // orthogonal matrix Q
   matrix&          PH,         // Hermitian conjugated matrix P
   vector           S,          // vector of computed singular values
   matrixc          U,          // matrix U of computed left vectors
   matrixc          VH          // Hermitian conjugated matrix V of right vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrix::SingularValueDecompositionBidiagQR(
   matrixf&         Q,          // orthogonal matrix Q
   matrixf&         PH,         // Hermitian conjugated matrix P
   vectorf&         S,          // vector of computed singular values
   matrixcf&        U,          // matrix U of computed left vectors
   matrixcf&        VH          // Hermitian conjugated matrix V of right vectors
   );
```

Parameters

Q

[in] Orthogonal matrix Q produced by method ReflectBidiagonalToQP. If matrix Q has zero size, then left vectors U are not calculated.

PT

© 2000-2025, MetaQuotes Ltd.

[in] Transposed (or hermitian conjugated) matrix P produced by method ReflectBidiagonalToQP. If matrix PT has zero size, then right vectors VT are not calculated.

S

[out] Vector of singular values.

U

[out] Matrix of left singular vectors.

VT

[out] Matrix of right singular vectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

BDSQR computes the singular values and, optionally, the right and/or left singular vectors from the singular value decomposition (SVD) of a N-by-N (upper or lower) bidiagonal matrix B using the implicit zero-shift QR algorithm. The SVD of B has the form

B = Q * S * P**T

where S is the diagonal matrix of singular values, Q is an orthogonal matrix of left singular vectors, and P is an orthogonal matrix of right singular vectors. If left singular vectors are requested, this subroutine actually returns U*Q instead of Q, and, if right singular vectors are requested, this subroutine returns P**T*VT instead of P**T, for given real input matrices U and VT. When U and VT are the orthogonal matrices that reduce a general matrix A to bidiagonal form: A = U*B*VT, as computed by GEBRD, then

A = (U*Q) * S * (P**T*VT)

is the SVD of A.

© 2000-2025, MetaQuotes Ltd.

Eigen Values

The section features functions for computing eigenvalues and eigenvectors. It describes methods for solving standard linear algebra problems using the LAPACK library algorithms. These functions are efficient for matrix analysis, diagonalization, system stabilization, and other tasks.

- **EigenSolver**: The function is designed to compute the eigenvalues and eigenvectors of an arbitrary square matrix using the classical algorithm represented by the GEEV lapack function. This method is applied to a wide range of matrices, allowing the decomposition of matrices into their eigenvalues and eigenvectors.
- **EigenSymmetricDC**: The function for computing eigenvalues and eigenvectors of symmetric or Hermitian matrices using the divide-and-conquer algorithm. The lapack functions SYEVD and HEEVD enable the efficient handling of symmetric or Hermitian matrices, providing faster and more accurate processing of such matrices.

Function	Action
EigenSolver	Compute eigenvalues and eigenvectors of a regular square matrix using the classical algorithm (lapack function GEEV).
EigenSolverX	Compute eigenvalues and eigenvectors of a regular square matrix in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GEEVX).
EigenSolverShur	Compute eigenvalues, upper triangular matrix in Schur form, and matrix of Schur vectors (lapack function GEES). See also Schur decomposition.
EigenSolver2	Compute generalized eigenvalues and eigenvectors for a pair of ordinary square matrices (lapack function GGEV).
EigenSolver2X	Compute generalized eigenvalues and eigenvectors for a pair of regular square matrices in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GGEVX). Both matrices must be the same size.
EigenSolver2Shur	Compute a pair of ordinary square matrices of generalized eigenvalues, generalized eigenvectors, generalized Schur forms, as well as left and right Schur vectors (lapack function GGES).
EigenSolver2Blocked	Compute generalized eigenvalues and eigenvectors for a pair of regular square matrices using a block algorithm (lapack function GGEV3). Both matrices must be the same size. The method parameters are exactly the same as EigenSolver2.

© 2000-2025, MetaQuotes Ltd.

Function	Action
EigenSolver2ShurBlocked	Compute a pair of regular square matrices of generalized eigenvalues, generalized eigenvectors, generalized Schur forms, as well as left and right Schur vectors (lapack function GGES3).
EigenSymmetricDC	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the divide-and-conquer algorithm (lapack functions SYEVD, HEEVD).
EigenSymmetricQR	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the QR algorithm (lapack functions SYEV, HEEV).
EigenSymmetricRobust	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the Multiple Relatively Robust Representations, MRRR algorithm (lapack functions SYEVR, HEEVR).
EigenSymmetricBisect	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the bisection algorithm (lapack functions SYEVX, HEEVX).

© 2000-2025, MetaQuotes Ltd.

General Matrices

Functions for calculating eigenvalues and eigenvectors of a square matrix using classical algorithms. It provides various methods for working with both real and complex matrices, allowing you to solve linear algebra problems with a choice of methods for calculating eigenvectors.

Function	Action
EigenSolver	Compute eigenvalues and eigenvectors of a regular square matrix using the classical algorithm (lapack function GEEV).
EigenSolverX	Compute eigenvalues and eigenvectors of a regular square matrix in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GEEVX).
EigenSolverShur	Compute eigenvalues, upper triangular matrix in Schur form, and matrix of Schur vectors (lapack function GEES). See also Schur decomposition.
EigenSolver2	Compute generalized eigenvalues and eigenvectors for a pair of ordinary square matrices (lapack function GGEV).
EigenSolver2X	Compute generalized eigenvalues and eigenvectors for a pair of regular square matrices in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GGEVX). Both matrices must be the same size.
EigenSolver2Shur	Compute a pair of ordinary square matrices of generalized eigenvalues, generalized eigenvectors, generalized Schur forms, as well as left and right Schur vectors (lapack function GGES).
EigenSolver2Blocked	Compute generalized eigenvalues and eigenvectors for a pair of regular square matrices using a block algorithm (lapack function GGEV3). Both matrices must be the same size. The method parameters are exactly the same as EigenSolver2.
EigenSolver2ShurBlocked	Compute a pair of regular square matrices of generalized eigenvalues, generalized eigenvectors, generalized Schur forms, as well as left and right Schur vectors (lapack function GGES3).

© 2000-2025, MetaQuotes Ltd.

EigenSolver

Compute eigenvalues and eigenvectors of a regular square matrix using the classical algorithm (lapack function GEEV).

Computing for type matrix<double>

```
bool  matrix::EigenSolver(
   ENUM_EIG_VECTORS      jobv,                 // compute left and right eigenvecto
   vectorc&              eigen_values,         // vector of computed eigenvalues
   matrix&               left_eigenvectors,    // matrix of computed left vectors
   matrix&               right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolver(
   ENUM_EIG_VECTORS      jobv,                 // compute left and right eigenvecto
   vectorcf&             eigen_values,         // vector of computed eigenvalues
   matrixf&              left_eigenvectors,    // matrix of computed left vectors
   matrixf&              right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSolver(
   ENUM_EIG_VECTORS      jobv,                 // compute left and right vectors
   vectorc&              eigen_values,         // vector of computed eigenvalues
   matrixc&              left_eigenvectors,    // matrix of computed left vectors
   matrixc&              right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSolver(
   ENUM_EIG_VECTORS      jobv,                 // compute left and right vectors
   vectorcf&             eigen_values,         // vector of computed eigenvalues
   matrixcf&             left_eigenvectors,    // matrix of computed left vectors
   matrixcf&             right_eigenvectors    // matrix of computed right vectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VECTORS enumeration value which determines the method for computing left and right eigenvectors.

EV

[out] Vector of eigenvalues.

left_eigenvectors

[out] Matrix of left eigenvectors.

© 2000-2025, MetaQuotes Ltd.

right_eigenvectors

[out] Matrix of right eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the jobv parameter.

If EIGVECTORS_N is set, the left and right vectors are not computed. Only eigenvalues are computed.

With EIGVECTORS_L, only left eigenvectors are computed, right eigenvectors are not computed.

When EIGVECTORS_R is set, only the right eigenvectors are computed, the left vectors are not computed.

With EIGVECTORS_LR, the left and right eigenvectors are computed, Eigenvalues are always computed.

Real (non-complex) matrices can have a complex solution. Therefore, the vector of eigenvalues must be complex. In case of a complex solution, the error code is set to 4019 (ERR_MATH_OVERFLOW). Otherwise, only the real parts of the complex values of the eigenvalue vector should be used.

ENUM_EIG_VECTORS

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are calculated, without vectors.
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

© 2000-2025, MetaQuotes Ltd.

EigenSolverX

Compute eigenvalues and eigenvectors of a regular square matrix in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GEEVX).

Computing for type matrix<double>

```
bool  matrix::EigenSolverX(
   ENUM_EIG_BALANCE      balance,                // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                   // determines computation of right a
   ENUM_EIG_SENSE        sense,                  // determines computation of recipro
   vectorc&              eigen_values,           // vector of computed eigenvalues
   matrix&               left_eigenvectors,      // matrix of computed left vectors
   matrix&               right_eigenvectors      // matrix of computed right vectors
   matrix&               shur_matrix,            // balanced matrix in Schur form
   long&                 ilo,                    // subscript of balanced matrix
   long&                 ihi,                    // superscript of balanced matrix
   vector&               scale,                  // details of permutations and scali
   double&               ab_norm,                // 1-norm of balanced matrix
   vector&               rconde,                 // vector of reciprocal condition nu
   vector&               rcondv                  // vector of reciprocal condition nu
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolverX(
   ENUM_EIG_BALANCE      balance,                // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                   // determines computation of right a
   ENUM_EIG_SENSE        sense,                  // determines computation of recipro
   vectorcf&             eigen_values,           // vector of computed eigenvalues
   matrixf&              left_eigenvectors,      // matrix of computed left vectors
   matrixf&              right_eigenvectors      // matrix of computed right vectors
   matrixf&              shur_matrix,            // balanced matrix in Schur form
   long&                 ilo,                    // subscript of balanced matrix
   long&                 ihi,                    // superscript of balanced matrix
   vectorf&              scale,                  // details of permutations and scali
   float&                ab_norm,                // 1-norm of balanced matrix
   vectorf&              rconde,                 // vector of reciprocal condition nu
   vectorf&              rcondv                  // vector of reciprocal condition nu
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSolverX(
   ENUM_EIG_BALANCE      balance,                // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                   // determines computation of right a
   ENUM_EIG_SENSE        sense,                  // determines computation of recipro
   vectorc&              eigen_values,           // vector of computed eigenvalues
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixc&              left_eigenvectors,       // matrix of computed left vectors
   matrixc&              right_eigenvectors       // matrix of computed right vectors
   matrixc&              shur_matrix,             // balanced matrix in Schur form
   long&                 ilo,                     // subscript of balanced matrix
   long&                 ihi,                     // superscript of balanced matrix
   vector&               scale,                   // details of permutations and scali
   double&               ab_norm,                 // 1-norm of balanced matrix
   vector&               rconde,                  // vector of reciprocal condition nu
   vector&               rcondv                   // vector of reciprocal condition nu
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSolverX(
   ENUM_EIG_BALANCE      balance,                 // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                    // determines computation of right a
   ENUM_EIG_SENSE        sense,                   // determines computation of recipro
   vectorcf&             eigen_values,            // vector of computed eigenvalues
   matrixcf&             left_eigenvectors,       // matrix of computed left vectors
   matrixcf&             right_eigenvectors       // matrix of computed right vectors
   matrixcf&             shur_matrix,             // balanced matrix in Schur form
   long&                 ilo,                     // subscript of balanced matrix
   long&                 ihi,                     // superscript of balanced matrix
   vectorf&              scale,                   // details of permutations and scali
   float&                ab_norm,                 // 1-norm of balanced matrix
   vectorf&              rconde,                  // vector of reciprocal condition nu
   vectorf&              rcondv                   // vector of reciprocal condition nu
   );
```

Parameters

balance

[in] Value from the ENUM_EIG_BALANCE enumeration which determines the need and method for balancing the input matrix; it is used to improve the conditioning of the eigenvalues and eigenvectors.

jobv

[in] ENUM_EIG_VECTORS enumeration value which determines the method for computing left and right eigenvectors.

sense

[in] Value from the ENUM_EIG_SENSE enumeration determining the need to compute reciprocal condition numbers.

eigen_values

[out] Vector of eigenvalues.

left_eigenvectors

[out] Matrix of left eigenvectors.

right_eigenvectors

© 2000-2025, MetaQuotes Ltd.

[out] Matrix of right eigenvectors.

shur_matrix

[out] Balanced matrix in Schur form; the matrix is not filled if neither left nor right eigenvectors are computed.

ilo

[out] Subscript of the balanced matrix; the matrix is not filled if no balancing is applied.

ihi

[out] Superscript of the balanced matrix; the matrix is not filled if no balancing is applied.

scale

[out] Vector of details of permutations and scaling when balancing the input matrix.

Details of the permutations and scaling factors applied when balancing A.

If P(j) is the index of the row and column interchanged with row and column j, and D(j) is the scaling factor applied to row and column j, then

scale(j) = P(j), for j = 1,...,ilo-1

= D(j), for j = ilo,...,ihi

= P(j) for j = ihi+1,..., n.

The order in which the interchanges are made is n to ihi+1, then 1 to ilo-1.

ab_norm

[out] 1-norm of the balanced matrix (the maximum of the sum of absolute values of elements in any of the matrix columns).

rconde

[out] Vector of reciprocal condition numbers for each eigenvalue; it is computed if the 'sense' parameter is set to 'E' or 'B'.

rcondv

[out] Vector of reciprocal condition numbers for each eigenvector; it is computed if the 'sense' parameter is set to 'V' or 'B.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Input matrix balancing depends on the value of the '*balance*' parameter.

ENUM_EIG_BALANCE

An enumeration defining the need to compute eigenvectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
EIGBALANCE_N	Do not diagonally scale or permute
EIGBALANCE_P	Perform permutations to make the matrix more nearly upper triangular. Do not diagonally scale
EIGBALANCE_S	Diagonally scale the matrix. Do not permute
EIGBALANCE_B	Both diagonally scale and permute

ENUM_EIG_VECTORS

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are computed, without vectors.
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

ENUM_EIG_SENSE

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGSENSE_N	None of reciprocal condition numbers are computed
EIGSENSE_E	Computed for eigenvalues only
EIGSENSE_V	Computed for right eigenvectors only
EIGSENSE_B	Computed for eigenvalues and right eigenvectors

© 2000-2025, MetaQuotes Ltd.

EigenSolverShur

Compute eigenvalues, upper triangular matrix in Schur form, and matrix of Schur vectors (lapack function GEES). See also Schur decomposition.

Computing for type matrix<double>

```
bool  matrix::EigenSolverShur(
   ENUM_EIG_VECTORS      jobvs,                  // method to compute Shur vectors
   vectorc&              eigen_values,           // vector of computed eigenvalues
   matrix&               shur_matrix,            // matrix in Schur form
   matrix&               shur_vectors            // matrix of Schur vectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolverShur(
   ENUM_EIG_VECTORS      jobvs,                  // method to compute Shur vectors
   vectorcf&             eigen_values,           // vector of computed eigenvalues
   matrixf&              shur_matrix,            // matrix in Schur form
   matrixf&              shur_vectors            // matrix of Schur vectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSolverShur(
   ENUM_EIG_VECTORS      jobvs,                  // method to compute Shur vectors
   vectorc&              eigen_values,           // vector of computed eigenvalues
   matrixc&              shur_matrix,            // matrix in Schur form
   matrixc&              shur_vectors            // matrix of Schur vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSolverShur(
   ENUM_EIG_VECTORS      jobvs,                  // method to compute Shur vectors
   vectorcf&             eigen_values,           // vector of computed eigenvalues
   matrixcf&             shur_matrix,            // matrix in Schur form
   matrixcf&             shur_vectors            // matrix of Schur vectors
   );
```

Parameters

jobvs

[in] Value from the ENUM_EIG_SHUR enumeration, which defines the method for computing Shur vectors.

eigen_values

[out] Vector of eigenvalues.

shur_matrix

[out] Upper triangular Schur matrix (Schur form for the input matrix).

© 2000-2025, MetaQuotes Ltd.

shur_vectors

[out] Matrix of Schur vectors; it is not computed if the *jobvs* parameter is set to N.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the *jobvs* parameter values.

Real (non-complex) matrices can have a complex solution. Therefore, the input vector of eigenvalues must be complex. In case of a complex solution, the error code is set to 4019 (ERR_MATH_OVERFLOW). Otherwise, only the real parts of the complex values of the eigenvalue vector should be used.

EigenSolverShur

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGSHUR_N	Schur vectors are not computed
EIGSHUR_V	Schur vectors are computed

© 2000-2025, MetaQuotes Ltd.

EigenSolver2

Compute generalized eigenvalues and eigenvectors for a pair of ordinary square matrices (lapack function GGEV). Both matrices must be the same size.

Computing for type matrix<double>

```
bool  matrix::EigenSolver2(
   matrix&               B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorc&              alpha,                // vector of computed eigenvalues
   vector&               beta,                 // vector of eigenvalue divisors
   matrix&               left_eigenvectors,    // matrix of computed left vectors
   matrix&               right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolver2(
   matrix&               B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorcf&             alpha,                // vector of computed eigenvalues
   vectorf&              beta,                 // vector of eigenvalue divisors
   matrixf&              left_eigenvectors,    // matrix of computed left vectors
   matrixf&              right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSolver2(
   matrixc&              B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorc&              alpha,                // vector of computed eigenvalues
   vectorc&              beta,                 // vector of eigenvalue divisors
   matrixc&              left_eigenvectors,    // matrix of computed left vectors
   matrixc&              right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSolver2(
   matrixcf&             B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorcf&             alpha,                // vector of computed eigenvalues
   vectorcf&             beta,                 // vector of eigenvalue divisors
   matrixcf&             left_eigenvectors,    // matrix of computed left vectors
   matrixcf&             right_eigenvectors    // matrix of computed right vectors
   );
```

Parameters

© 2000-2025, MetaQuotes Ltd.

B

[in] The second matrix in the pair.

jobv

[in] ENUM_EIG_VECTORS enumeration value which determines the method for computing left and right eigenvectors.

alpha

[out] Vector of eigenvalues.

beta

[out] Vector of eigen value divisors.

left_eigenvectors

[out] Matrix of left eigenvectors.

right_eigenvectors

[out] Matrix of right eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the jobv parameter.

ENUM_EIG_VECTORS

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are computed, without vectors.
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

A generalized eigenvalue for a pair of matrices (A,B) is a scalar lambda or a ratio alpha/beta = lambda, such that A - lambda*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

The right eigenvector v(j) corresponding to the eigenvalue lambda(j) of (A,B) satisfies

A * v(j) = lambda(j) * B * v(j).

The left eigenvector u(j) corresponding to the eigenvalue lambda(j) of (A,B) satisfies

u(j)**H * A = lambda(j) * u(j)**H * B .

© 2000-2025, MetaQuotes Ltd.

where u(j)**H is the conjugate-transpose of u(j).

Real (non-complex) matrices can have a complex solution. Therefore, the input vector of eigenvalues must be complex. In case of a complex solution, the error code is set to 4019 (ERR_MATH_OVERFLOW). Otherwise, only the real parts of the complex values of the eigenvalue vector should be used.

© 2000-2025, MetaQuotes Ltd.

EigenSolver2X

Compute generalized eigenvalues and eigenvectors for a pair of regular square matrices in Expert mode, i.e. with the ability to influence the computation algorithm and the ability to obtain accompanying computation data (lapack function GGEVX). Both matrices must be the same size.

Optionally, it also computes a balancing transformation to improve the conditioning of the eigenvalues and eigenvectors (ILO, IHI, LSCALE, RSCALE, ABNRM, and BBNRM), reciprocal condition numbers for the eigenvalues (RCONDE), and reciprocal condition numbers for the right eigenvectors (RCONDV).

Computing for type matrix<double>

```
bool  matrix::EigenSolver2X(
   matrix&               B,                    // second matrix in the pair
   ENUM_EIG_BALANCE      balance,              // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                 // determines computation of right a
   ENUM_EIG_SENSE        sense,                // determines computation of recipro
   vectorc&              alpha,                // vector of computed eigenvalues
   vector&               beta,                 // vector of eigenvalue divisors
   matrix&               left_eigenvectors,    // matrix of computed left vectors
   matrix&               right_eigenvectors    // matrix of computed right vectors
   matrix&               shur_matrix1,         // the first part of the real Schur
   matrix&               shur_matrix2,         // the second part of the real Schur
   long&                 ilo,                  // subscript of balanced matrix
   long&                 ihi,                  // superscript of balanced matrix
   vector&               lscale,               // details of the permutations and s
   vector&               rscale,               // details of the permutations and s
   double&               ab_norm,              // one-norm of balanced input matrix
   double&               bb_norm,              // one-norm of balanced second matri
   vector&               rconde,               // vector of reciprocal condition nu
   vector&               rcondv                // vector of reciprocal condition nu
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolver2X(
   matrixf&              B,                    // second matrix in the pair
   ENUM_EIG_BALANCE      balance,              // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                 // determines computation of right a
   ENUM_EIG_SENSE        sense,                // determines computation of recipro
   vectorcf&             alpha,                // vector of computed eigenvalues
   vectorf&              beta,                 // vector of eigenvalue divisors
   matrixf&              left_eigenvectors,    // matrix of computed left vectors
   matrixf&              right_eigenvectors    // matrix of computed right vectors
   matrixf&              shur_matrix1,         // the first part of the real Schur
   matrixf&              shur_matrix2,         // the second part of the real Schur
   long&                 ilo,                  // subscript of balanced matrix
   long&                 ihi,                  // superscript of balanced matrix
   vectorf&              lscale,               // details of the permutations and s
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorf&              rscale,                 // details of the permutations and s
   float&                ab_norm,                // one-norm of balanced input matrix
   float&                bb_norm,                // one-norm of balanced second matri
   vectorf&              rconde,                 // vector of reciprocal condition nu
   vectorf&              rcondv                  // vector of reciprocal condition nu
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSolver2X(
   matrixc&              B,                      // second matrix in the pair
   ENUM_EIG_BALANCE      balance,                // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                   // determines computation of right a
   ENUM_EIG_SENSE        sense,                  // determines computation of recipro
   vectorc&              alpha,                  // vector of computed eigenvalues
   vectorc&              beta,                   // vector of eigenvalue divisors
   matrixc&              left_eigenvectors,      // matrix of computed left vectors
   matrixc&              right_eigenvectors      // matrix of computed right vectors
   matrixc&              shur_matrix1,           // the first part of the real Schur
   matrixc&              shur_matrix2,           // the second part of the real Schu
   long&                 ilo,                    // subscript of balanced matrix
   long&                 ihi,                    // superscript of balanced matrix
   vector&               lscale,                 // details of the permutations and s
   vector&               rscale,                 // details of the permutations and s
   double&               ab_norm,                // one-norm of balanced input matrix
   double&               bb_norm,                // one-norm of balanced second matri
   vector&               rconde,                 // vector of reciprocal condition nu
   vector&               rcondv                  // vector of reciprocal condition nu
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSolver2X(
   matrixcf&             B,                      // second matrix in the pair
   ENUM_EIG_BALANCE      balance,                // input matrix balancing method
   ENUM_EIG_VECTORS      jobv,                   // determines computation of right a
   ENUM_EIG_SENSE        sense,                  // determines computation of recipro
   vectorcf&             alpha,                  // vector of computed eigenvalues
   vectorf&              beta,                   // vector of eigenvalue divisors
   matrixcf&             left_eigenvectors,      // matrix of computed left vectors
   matrixcf&             right_eigenvectors      // matrix of computed right vectors
   matrixcf&             shur_matrix1,           // the first part of the real Schur
   matrixcf&             shur_matrix2,           // the second part of the real Schu
   long&                 ilo,                    // subscript of balanced matrix
   long&                 ihi,                    // superscript of balanced matrix
   vectorf&              lscale,                 // details of the permutations and s
   vectorf&              rscale,                 // details of the permutations and s
   float&                ab_norm,                // one-norm of balanced input matrix
```

© 2000-2025, MetaQuotes Ltd.

```
   float&                bb_norm,                // one-norm of balanced second matri
   vectorf&              rconde,                 // vector of reciprocal condition nu
   vectorf&              rcondv                  // vector of reciprocal condition nu
   );
```

Parameters

B

[in] The second matrix in the pair.

balance

[in] Value from the ENUM_EIG_BALANCE enumeration which determines the need and method for balancing the input matrix; it is used to improve the conditioning of the eigenvalues and eigenvectors.

jobv

[in] Value from the ENUM_EIG_VECTORS enumeration which determines the method for computing left and right eigenvectors.

sense

[in] Value from the ENUM_EIG_SENSE enumeration which determines the need to compute reciprocal condition numbers.

eigen_values

[out] Vector of eigenvalues.

left_eigenvectors

[out] Matrix of left eigenvectors.

right_eigenvectors

[out] Matrix of right eigenvectors.

shur_matrix1, shur_matrix2

[out] 2 parts of balanced matrix in Schur form; the matrix is not filled if neither left nor right eigenvectors are computed.

ilo

[out] Subscript of the balanced matrix; the matrix is not filled if no balancing is applied.

ihi

[out] Superscript of the balanced matrix; the matrix is not filled if no balancing is applied.

lscale

[out] Vector contains details of the permutations and scaling factors applied to the left side of A and B.

If PL(j) is the index of the row interchanged with row j, and DL(j) is the scaling factor applied to row j, then

lscale(j) = PL(j), for j = 1,..., ilo-1

= DL(j), for j = ilo,...,ihi

= PL(j) for j = ihi+1,..., n.

© 2000-2025, MetaQuotes Ltd.

The order in which the interchanges are made is n to ihi+1, then 1 to ilo-1.

rscale

[out] Vector contains details of the permutations and scaling factors applied to the right side of A and B.

If PR(j) is the index of the column interchanged with column j, and DR(j) is the scaling factor applied to column j, then

rscale(j) = PR(j), for j = 1,..., ilo-1

= DR(j), for j = ilo,...,ihi

= PR(j) for j = ihi+1,..., n.

The order in which the interchanges are made is n to ihi+1, then 1 to ilo-1.

ab_norm

[out] One-norm of the balanced input matrix (the maximum of the sum of absolute values of elements in any of the matrix columns).

bb_norm

[out] One-norm of balanced second matrix B.

rconde

[out] Vector of reciprocal condition numbers for each eigenvalue; it is computed if the 'sense' parameter is set to 'E' or 'B'.

rcondv

[out] Vector of reciprocal condition numbers for each eigenvector; it is computed if the 'sense' parameter is set to 'V' or 'B'.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

Input matrices balancing depends on the value of the *balance* parameter.

ENUM_EIG_BALANCE

An enumeration that specifies whether the matrices should be balanced.

ID	Description
EIGBALANCE_N	Do not diagonally scale or permute.
EIGBALANCE_P	Perform permutations to make the matrix more nearly upper triangular. Do not diagonally scale.
EIGBALANCE_S	Diagonally scale the matrix. Do not permute.
EIGBALANCE_B	Both diagonally scale and permute.

© 2000-2025, MetaQuotes Ltd.

ENUM_EIG_VECTORS

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are computed, without vectors.
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

ENUM_EIG_SENSE

An enumeration determining the need to compute reciprocal condition numbers.

ID	Description
EIGSENSE_N	None of reciprocal condition numbers are computed.
EIGSENSE_E	Computed for eigenvalues only.
EIGSENSE_V	Computed for right eigenvectors only.
EIGSENSE_B	Computed for eigenvalues and right eigenvectors.

© 2000-2025, MetaQuotes Ltd.

EigenSolver2Shur

Compute a pair of ordinary square matrices of generalized eigenvalues, generalized eigenvectors, generalized Schur forms, as well as left and right Schur vectors (lapack function GGES).

Computes the generalized eigenvalues, the generalized real/complex Schur form (S,T), optionally, the left and/or right matrices of Schur vectors (vsl and vsr) for a pair of n-by-n real/complex nonsymmetric matrices (A,B). This gives the generalized Schur factorization:

(A,B) = (vsl*S *vsrH, vsl*T*vsrH)

Optionally, it also orders the eigenvalues so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper quasi-triangular matrix S and the upper triangular matrix T. The leading columns of vsl and vsr then form an orthonormal/unitary basis for the corresponding left and right eigenspaces (deflating subspaces).

Computing for type matrix<double>

```
bool  matrix::EigenSolver2Shur(
   matrix&                 B,         // second matrix in the pair
   ENUM_EIG_VECTORS        jobvs,     // method to compute left and right vectors
   vector&                 alpha,     // vector of computed eigenvalues
   vector&                 beta,      // vector of eigenvalue divisors
   matrix&                 shur_s,    // matrix S in Schur form
   matrix&                 shur_t,    // matrix T in Schur form
   matrix&                 vsl,       // matrix of left Schur vectors vsl
   matrix&                 vsr        // matrix of right Schur vectors vsr
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolver2Shur(
   matrixf&                B,         // second matrix in the pair
   ENUM_EIG_VECTORS        jobvs,     // method to compute left and right vectors
   vectorcf&               alpha,     // vector of computed eigenvalues
   vectorf&                beta,      // vector of eigenvalue divisors
   matrixf&                shur_s,    // matrix S in Schur form
   matrixf&                shur_t,    // matrix T in Schur form
   matrixf&                vsl,       // matrix of left Schur vectors vsl
   matrixf&                vsr        // matrix of right Schur vectors vsr
   );
```

Computing for type matrix<complex>

```
bool  matrix::EigenSolver2Shur(
   matrixc&                B,         // second matrix in the pair
   ENUM_EIG_VECTORS        jobvs,     // method to compute left and right vectors
   vectorc&                alpha,     // vector of computed eigenvalues
   vectorc&                beta,      // vector of eigenvalue divisors
   matrixc&                shur_s,    // matrix S in Schur form
   matrixc&                shur_t,    // matrix T in Schur form
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixc&                vsl,       // matrix of left Schur vectors vsl
   matrixc&                vsr        // matrix of right Schur vectors vsr
   );
```

Вычисления для типа matrix<complexf>

```
bool  matrixcf::EigenSolver2Shur(
   matrixcf&               B,         // second matrix in the pair
   ENUM_EIG_VECTORS        jobvs,     // method to compute left and right vectors
   vectorcf&               alpha,     // vector of computed eigenvalues
   vectorcf&               beta,      // vector of eigenvalue divisors
   matrixcf&               shur_s,    // matrix S in Schur form
   matrixcf&               shur_t,    // matrix T in Schur form
   matrixcf&               vsl,       // matrix of left Schur vectors vsl
   matrixcf&               vsr        // matrix of right Schur vectors vsr
   );
```

Parameters

B

[out] The second matrix in the pair.

jobvs

[in] ENUM_EIG_VECTORS enumeration value which determines the method for computing left and right eigenvectors.

alpha

[out] Vector of eigenvalues.

beta

[out] Vector of eigenvalue divisors.

shur_s

[out] Matrix S, block upper triangular Schur matrix (Schur form for the input matrix).

shur_t

[out] Matrix T, block upper triangular Schur matrix (Schur form for the second matrix in the pair).

vsl

[out] Matrix of left Schur vectors.

vsr

[out] Matrix of right Schur vectors.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

Computation depends on the *jobvs* parameter values.

© 2000-2025, MetaQuotes Ltd.

The second matrix in the pair must be the same size as the first (input) one.

Real (non-complex) matrices can have a complex solution. Therefore, the vector of eigenvalues must be complex. In case of a complex solution, the error code is set to 4019 (ERR_MATH_OVERFLOW). Otherwise, only the real parts of the complex values of the eigenvalue vector should be used.

EigenSolverShur

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are computed, without vectors.
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

© 2000-2025, MetaQuotes Ltd.

EigenSolver2Blocked

Compute generalized eigenvalues and eigenvectors for a pair of regular square matrices using a block algorithm (lapack function GGEV3). Both matrices must be the same size. The method parameters are exactly the same as EigenSolver2.

Computing for type matrix<double>

```
bool  matrix::EigenSolver2Blocked(
   matrix&               B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorc&              alpha,                // vector of computed eigenvalues
   vector&               beta,                 // vector of eigenvalue divisors
   matrix&               left_eigenvectors,    // matrix of computed left vectors
   matrix&               right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSolver2Blocked(
   matrix&               B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorcf&             alpha,                // vector of computed eigenvalues
   vectorf&              beta,                 // vector of eigenvalue divisors
   matrixf&              left_eigenvectors,    // matrix of computed left vectors
   matrixf&              right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<complex>

```
bool  matrix::EigenSolver2Blocked(
   matrixc&              B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorc&              alpha,                // vector of computed eigenvalues
   vectorc&              beta,                 // vector of eigenvalue divisors
   matrixc&              left_eigenvectors,    // matrix of computed left vectors
   matrixc&              right_eigenvectors    // matrix of computed right vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSolver2(
   matrixcf&             B,                    // second matrix in the pair
   ENUM_EIG_VECTORS      jobv,                 // method to compute right and left
   vectorcf&             alpha,                // vector of computed eigenvalues
   vectorcf&             beta,                 // vector of eigenvalue divisors
   matrixcf&             left_eigenvectors,    // matrix of computed left vectors
   matrixcf&             right_eigenvectors    // matrix of computed right vectors
   );
```

Parameters

© 2000-2025, MetaQuotes Ltd.

B

[out] The second matrix in the pair.

jobv

[in] ENUM_EIG_VECTORS enumeration value which determines the method for computing left and right eigenvectors.

alpha

[out] Vector of eigenvalues.

beta

[out] Vector of eigenvalue divisors.

left_eigenvectors

[out] Matrix of left eigenvectors.

righeft_eigenvectors

[out] Matrix of right eigenvectors.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

Computation depends on the value of the jobv parameter.

A generalized eigenvalue for a pair of matrices (A,B) is a scalar lambda or a ratio alpha/beta = lambda, such that A - lambda*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

The right eigenvector v(j) corresponding to the eigenvalue lambda(j) of (A,B) satisfies:

A * v(j) = lambda(j) * B * v(j).

The left eigenvector u(j) corresponding to the eigenvalue lambda(j) of (A,B) satisfies:

u(j)**H * A = lambda(j) * u(j)**H * B .

where u(j)**H is the conjugate-transpose of u(j).

Real (non-complex) matrices can have a complex solution. Therefore, the input vector of eigenvalues must be complex. In case of a complex solution, the error code is set to 4019 (ERR_MATH_OVERFLOW). Otherwise, only the real parts of the complex values of the eigenvalue vector should be used.

ENUM_EIG_VECTORS

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are computed, without vectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

© 2000-2025, MetaQuotes Ltd.

EigenSolver2ShurBlocked

Compute a pair of regular square matrices of generalized eigenvalues, generalized eigenvectors, generalized Schur forms, as well as left and right Schur vectors (lapack function GGES3).

Computes the generalized eigenvalues, the generalized real/complex Schur form (S,T), optionally, the left and/or right matrices of Schur vectors (VSL and VSR) for a pair of n-by-n real/complex nonsymmetric matrices (A,B). This gives the generalized Schur factorization:

(A,B) = (vsl*S *vsrH, vsl*T*vsrH)

Optionally, it also orders the eigenvalues so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper quasi-triangular matrix S and the upper triangular matrix T. The leading columns of vsl and vsr then form an orthonormal/unitary basis for the corresponding left and right eigenspaces (deflating subspaces).

Computing for type matrix<double>

```
bool  matrix::EigenSolver2ShurBlocked(
   matrix&                B,         // second matrix in the pair
   ENUM_EIG_VECTORS       jobvs,     // method to compute left and right vectors
   vectorc&               alpha,     // vector of computed eigenvalues
   vector&                beta,      // vector of eigenvalue divisors
   matrix&                shur_s,    // matrix S in Schur form
   matrix&                shur_t,    // matrix T in Schur form
   matrix&                vsl,       // matrix of left Schur vectors VSL
   matrix&                vsr        // matrix of right Schur vectors VSR
   );
```

Computing for type matrix<float>

```
bool  matrix::EigenSolver2ShurBlocked(
   matrixf&               B,         // second matrix in the pair
   ENUM_EIG_VECTORS       jobvs,     // method to compute left and right vectors
   vectorcf&              alpha,     // vector of computed eigenvalues
   vectorf&               beta,      // vector of eigenvalue divisors
   matrixf&               shur_s,    // matrix S in Schur form
   matrixf&               shur_t,    // matrix T in Schur form
   matrixf&               vsl,       // matrix of left Schur vectors VSL
   matrixf&               vsr        // matrix of right Schur vectors VSR
   );
```

Computing for type matrix<complex>

```
bool  matrix::EigenSolver2ShurBlocked(
   matrixc&               B,         // second matrix in the pair
   ENUM_EIG_VECTORS       jobvs,     // method to compute left and right vectors
   vectorc&               alpha,     // vector of computed eigenvalues
   vectorc&               beta,      // vector of eigenvalue divisors
   matrixc&               shur_s,    // matrix S in Schur form
   matrixc&               shur_t,    // matrix T in Schur form
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixc&                vsl,       // matrix of left Schur vectors VSL
   matrixc&                vsr        // matrix of right Schur vectors VSR
   );
```

Computing for type matrix<complexf>

```
bool  matrix::EigenSolver2ShurBlocked(
   matrixcf&               B,         // second matrix in the pair
   ENUM_EIG_VECTORS        jobvs,     // method to compute left and right vectors
   vectorcf&               alpha,     // vector of computed eigenvalues
   vectorcf&               beta,      // vector of eigenvalue divisors
   matrixcf&               shur_s,    // matrix S in Schur form
   matrixcf&               shur_t,    // matrix T in Schur form
   matrixcf&               vsl,       // matrix of left Schur vectors VSL
   matrixcf&               vsr        // matrix of right Schur vectors VSR
   );
```

Parameters

B

[in] The second matrix in the pair.

jobvs

[in] ENUM_EIG_VECTORS enumeration value which determines the method for computing left and right eigenvectors.

alpha

[out] Vector of eigenvalues.

beta

[out] Vector of eigenvalue divisors.

shur_s

[out] Matrix S, block upper triangular Schur matrix (Schur form for the input matrix).

shur_t

[out] Matrix T, block upper triangular Schur matrix (Schur form for the second matrix in the pair).

vsl

[out] Matrix of left Schur vectors VSL.

vsr

[out] Matrix of right Schur vectors VSR.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

Computation depends on the *jobvs* parameter values.

© 2000-2025, MetaQuotes Ltd.

The second matrix in the pair must be the same size as the first (input) one.

Real (non-complex) matrices can have a complex solution. Therefore, the vector of eigenvalues must be complex. In case of a complex solution, the error code is set to 4019 (ERR_MATH_OVERFLOW). Otherwise, only the real parts of the complex values of the eigenvalue vector should be used.

EigenSolverShur

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVECTORS_N	Only eigenvalues are computed, without vectors.
EIGVECTORS_L	Only left eigenvectors are computed.
EIGVECTORS_R	Only right eigenvectors are computed.
EIGVECTORS_LR	Left and right eigenvectors are computed, eigenvalues are always computed.

© 2000-2025, MetaQuotes Ltd.

Symmetric Matrices

Functions for computing eigenvalues and eigenvectors of symmetric or Hermitian matrices using the divide and conquer algorithm, making the process efficient and fast. These methods can be applied to matrices of different data types, including real and complex numbers.

Function	Action
EigenSymmetricDC	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the divide-and-conquer algorithm (lapack functions SYEVD, HEEVD).
EigenSymmetricQR	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the QR algorithm (lapack functions SYEV, HEEV).
EigenSymmetricRobust	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the Multiple Relatively Robust Representations, MRRR algorithm (lapack functions SYEVR, HEEVR).
EigenSymmetricBisect	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugate) matrix using the bisection algorithm (lapack functions SYEVX, HEEVX).
EigenSymmetricDC2s	Compute all eigenvalues and, optionally, eigenvectors of a real symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal. If eigenvectors are desired, it uses a divide and conquer algorithm (lapack functions SYEVD_2STAGE, HEEVD_2STAGE).
EigenSymmetricQR2s	Compute all eigenvalues and, optionally, eigenvectors of a real symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal (lapack functions SYEV_2STAGE, HEEV_2STAGE).
EigenSymmetricRobust2s	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal then using the Multiple Relatively Robust Representations, MRRR algorithm (lapack functions SYEVR_2STAGE, HEEVR_2STAGE).
EigenSymmetricBisect2s	Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction

© 2000-2025, MetaQuotes Ltd.

Function	Action
	to tridiagonal then using the bisection algorithm (lapack functions SYEVX_2STAGE, HEEVX_2STAGE).
EigenSymmetric2DC	Compute all the eigenvalues, and optionally, the eigenvectors of a generalized symmetric-definite eigenproblem, of the form A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.
EigenSymmetric2QR	Compute all the eigenvalues, and optionally, the eigenvectors of a generalized symmetric-definite eigenproblem, of the form A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.
EigenSymmetric2Bisect	Compute all the eigenvalues, and optionally, the eigenvectors of a generalized symmetric-definite eigenproblem, of the form A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricDC

Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the divide-and-conquer algorithm (lapack functions SYEVD, HEEVD).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricDC(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricDC(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricDC(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricDC(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixcf&             eigen_vectors      // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

Computation depends on the value of the jobv parameter.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated.

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricQR

Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the QR algorithm (lapack functions SYEV, HEEV).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricQR(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricQR(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricQR(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricQR(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixcf&             eigen_vectors      // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

Computation depends on the value of the jobv parameter.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated.

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricRobust

Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the Multiple Relatively Robust Representations, MRRR algorithm (lapack functions SYEVR, HEEVR).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricRobust(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricRobust(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricRobust(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricRobust(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // Upper bound of the subset
   float                 abstol,            // absolute error tolerance
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorf&              eigen_values,       // vector of computed eigenvectors
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

© 2000-2025, MetaQuotes Ltd.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricBisect

Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the bisection algorithm (lapack functions SYEVX, HEEVX).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricBisect(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricBisect(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricBisect(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricBisect(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // Upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

eigen_values

[out] Vector of eigenvalues.

V

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

© 2000-2025, MetaQuotes Ltd.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricDC2s

Compute all eigenvalues and, optionally, eigenvectors of a real symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal. If eigenvectors are desired, it uses a divide and conquer algorithm (lapack functions SYEVD_2STAGE, HEEVD_2STAGE).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricDC2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricDC2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricDC2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricDC2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixcf&             eigen_vectors      // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

Computation depends on the value of the jobv parameter.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated. In the current OpenBLAS implementation, this value is not supported. Attempting to use it will result in error 4003 (ERR_INVALID_PARAMETER).

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated. Not available in this release.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricQR2s

Compute all eigenvalues and, optionally, eigenvectors of a real symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal (lapack functions SYEV_2STAGE, HEEV_2STAGE).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricQR2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricQR2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricQR2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vector&               eigen_values,      // vector of computed eigenvalues
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricQR2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   vectorf&              eigen_values,      // vector of computed eigenvalues
   matrixcf&             eigen_vectors      // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

Computation depends on the value of the jobv parameter.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated. In the current OpenBLAS implementation, this value is not supported. Attempting to use it will result in error 4003 (ERR_INVALID_PARAMETER).

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated. Not available in this release.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricRobust2s

Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal then using the Multiple Relatively Robust Representations, MRRR algorithm (lapack functions SYEVR_2STAGE, HEEVR_2STAGE).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricRobust2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricRobust2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricRobust2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricRobust2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorf&              eigen_values,       // vector of computed eigenvectors
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated. In the current OpenBLAS implementation, this value is not supported. Attempting to use it will result in error 4003 (ERR_INVALID_PARAMETER).

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

© 2000-2025, MetaQuotes Ltd.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated. Not available in this release.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetricBisect2s

Compute eigenvalues and eigenvectors of a symmetric or Hermitian (complex conjugated) matrix using the 2stage technique for the reduction to tridiagonal then using the bisection algorithm (lapack functions SYEVX_2STAGE, HEEVX_2STAGE).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetricBisect2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetricBisect2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetricBisect2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrixc&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetricBisect2s(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorf&              eigen_values,       // vector of computed eigenvectors
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

eigen_values

[out] Vector of eigenvalues.

V

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated. In the current OpenBLAS implementation, this value is not supported. Attempting to use it will result in error 4003 (ERR_INVALID_PARAMETER).

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

© 2000-2025, MetaQuotes Ltd.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated. Not available in this release.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetric2DC

Compute all the eigenvalues, and optionally, the eigenvectors of a generalized symmetric-definite eigenproblem, of the form

A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

Here A and B are assumed to be symmetric (Hermitian) and B is also positive definite. If eigenvectors are desired, it uses a divide-and-conquer algorithm (lapack functions SYGVD, HEGVD).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetric2DC(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrix&               B,                  // second matrix
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors,      // matrix of computed eigenvectors
   matrix&               triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetric2DC(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrixf&              B,                  // second matrix
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors,      // matrix of computed eigenvectors
   matrixf&              triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetric2DC(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrixc&              B,                  // second matrix
   vector&               eigen_values,       // vector of computed eigenvalues
   matrixc&              eigen_vectors,      // matrix of computed eigenvectors
   matrixc&              triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetric2DC(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrixcf&             B,                  // second matrix
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixcf&             eigen_vectors,      // matrix of computed eigenvectors
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixcf&             triangular_factor   // triangular factor from the Cholesky fa
   );
```

Parameters

itype

[in] ENUM_EIGS2_TYPE enumeration value which specified the problem type to be solved : A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

B

[in] Second matrix B. Must be positive definite symmetric (or Hermitian conjugated) matrix.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

triangular_factor

[out] The triangular factor U or L from the Cholesky factorization B = U**T*U or B = L*L**T.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated). Second matrix B must be positive definite symmetric. If the input matrix and second matrix B are triangular, then both must be the same triangular, upper or lower.

ENUM_EIGS2_TYPE

An enumeration that specifies the problem type to be solved.

ID	Description
EIGS2TYPE_1	1: A*x = (lambda)*B*x
EIGS2TYPE_2	2: A*B*x = (lambda)*x
EIGS2TYPE_3	3: B*A*x = (lambda)*x

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetric2QR

Compute all the eigenvalues, and optionally, the eigenvectors of a generalized symmetric-definite eigenproblem, of the form

A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

Here A and B are assumed to be symmetric (Hermitian) and B is also positive definite. This method uses QR algorithm (lapack functions SYGV, HEGV).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetric2QR(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrix&               B,                  // second matrix
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors,      // matrix of computed eigenvectors
   matrix&               triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetric2QR(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrixf&              B,                  // second matrix
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors,      // matrix of computed eigenvectors
   matrixf&              triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetric2QR(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrixc&              B,                  // second matrix
   vector&               eigen_values,       // vector of computed eigenvalues
   matrixc&              eigen_vectors,      // matrix of computed eigenvectors
   matrixc&              triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetric2QR(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   matrixcf&             B,                  // second matrix
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixcf&             eigen_vectors,      // matrix of computed eigenvectors
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixcf&             triangular_factor   // triangular factor from the Cholesky fa
   );
```

Parameters

itype

[in] ENUM_EIGS2_TYPE enumeration value which specified the problem type to be solved : A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

B

[in] Second matrix B. Must be positive definite symmetric (or Hermitian conjugated) matrix.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

triangular_factor

[out] The triangular factor U or L from the Cholesky factorization B = U**T*U or B = L*L**T.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated). Second matrix B must be positive definite symmetric. If the input matrix and second matrix B are triangular, then both must be the same triangular, upper or lower.

ENUM_EIGS2_TYPE

An enumeration that specifies the problem type to be solved.

ID	Description
EIGS2TYPE_1	1: A*x = (lambda)*B*x
EIGS2TYPE_2	2: A*B*x = (lambda)*x
EIGS2TYPE_3	3: B*A*x = (lambda)*x

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenSymmetric2Bisect

Compute all the eigenvalues, and optionally, the eigenvectors of a generalized symmetric-definite eigenproblem, of the form

A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

Here A and B are assumed to be symmetric (Hermitian) and B is also positive definite. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues. This method uses bisection algorithm (lapack functions SYGVX, HEGVX).

Computing for type matrix<double>

```
bool  matrix::EigenSymmetric2Bisect(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,              // subset of eigenvalues to compute
   double                lower,              // lower bound of the subset
   double                upper,              // Upper bound of the subset
   double                abstol,             // absolute error tolerance
   matrix&               B,                  // second matrix
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors,      // matrix of computed eigenvectors
   matrix&               triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenSymmetric2Bisect(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,              // subset of eigenvalues to compute
   float                 lower,              // lower bound of the subset
   float                 upper,              // upper bound of the subset
   float                 abstol,             // absolute error tolerance
   matrixf&              B,                  // second matrix
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors,      // matrix of computed eigenvectors
   matrixf&              triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenSymmetric2Bisect(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,              // subset of eigenvalues to compute
   double                lower,              // lower bound of the subset
   double                upper,              // Upper bound of the subset
   double                abstol,             // absolute error tolerance
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixc&              B,                  // second matrix
   vector&               eigen_values,       // vector of computed eigenvalues
   matrixc&              eigen_vectors,      // matrix of computed eigenvectors
   matrixc&              triangular_factor   // triangular factor from the Cholesky fa
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenSymmetric2Bisect(
   ENUM_EIGS2_TYPE       itype,              // the problem type to be solved
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,              // subset of eigenvalues to compute
   float                 lower,              // lower bound of the subset
   float                 upper,              // Upper bound of the subset
   float                 abstol,             // absolute error tolerance
   matrixcf&             B,                  // second matrix
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixcf&             eigen_vectors,      // matrix of computed eigenvectors
   matrixcf&             triangular_factor   // triangular factor from the Cholesky fa
   );
```

Parameters

itype

[in] ENUM_EIGS2_TYPE enumeration value which specified the problem type to be solved : A*x=(lambda)*B*x, A*Bx=(lambda)*x, or B*A*x=(lambda)*x.

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

© 2000-2025, MetaQuotes Ltd.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

B

[in] Second matrix B. Must be positive definite symmetric (or Hermitian conjugated) matrix.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

triangular_factor

[out] The triangular factor U or L from the Cholesky factorization B = U**T*U or B = L*L**T.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated). Second matrix B must be positive definite symmetric. If the input matrix and second matrix B are triangular, then both must be the same triangular, upper or lower.

ENUM_EIGS2_TYPE

An enumeration that specifies the problem type to be solved.

ID	Description
EIGS2TYPE_1	1: A*x = (lambda)*B*x
EIGS2TYPE_2	2: A*B*x = (lambda)*x
EIGS2TYPE_3	3: B*A*x = (lambda)*x

© 2000-2025, MetaQuotes Ltd.

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

Tridiagonal Matrices

Functions for computing eigenvalues and eigenvectors of symmetric tridiagonal matrices using various algorithms. Each function implements a specific solution method and supports matrix types double and float.

Common Parameters:

- jobv — Determines whether to compute eigenvectors (EIGVALUES_V) or only eigenvalues (EIGVALUES_N).
- range — Specifies the range of computed eigenvalues (BLASRANGE_A, BLASRANGE_V, BLASRANGE_I).
- lower and upper — Lower and upper bounds for computing a subset of the spectrum.
- abstol — Absolute error tolerance.

All functions operate on symmetric tridiagonal matrices and allow selecting the most suitable algorithm depending on performance and accuracy requirements.

Function	Action
EigenTridiagonalDC	Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the divide-and-conquer algorithm (lapack function STEVD).
EigenTridiagonalQR	Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the QR algorithm (lapack function STEV).
EigenTridiagonalRobust	Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the Multiple Relatively Robust Representations, MRRR algorithm (lapack function STEVR).
EigenTridiagonalBisect	Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the bisection algorithm (lapack function STEVX).
EigenTridiagonalQL	Compute all eigenvalues of a symmetric tridiagonal matrix using the Pal-Walker-Kahan variant of the QL or QR algorithm (lapack function STERF).
EigenTridiagonalDCQ	Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the divide-and-conquer algorithm (lapack function STEDC).
EigenTridiagonalQRQ	Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the QR algorithm (lapack function STEQR).
EigenTridiagonalPosDefQ	Compute eigenvalues and eigenvectors of a symmetric positive definite (положительно

© 2000-2025, MetaQuotes Ltd.

Function	Action
	определённая) tridiagonal matrix using the QR algorithm (lapack function PTEQR).

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalDC

Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the divide-and-conquer algorithm (lapack function STEVD).

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalDC(
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalDC(
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the jobv parameter.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated.

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

The input must be a symmetric matrix in the tridiagonal form.

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalQR

Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the QR algorithm (lapack function STEV).

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalQR(
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalQR(
   ENUM_EIG_VALUES       jobv,               // compute eigenvectors or not
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the jobv parameter.

When jobv = EIGVALUES_V, eigenvectors and eigenvalues are calculated.

If EIGVALUES_N is set, eigenvectors are not calculated. Only eigenvalues are computed.

The input must be a symmetric matrix in the tridiagonal form.

ENUM_EIG_VALUES

An enumeration that specifies whether to calculate eigenvectors.

© 2000-2025, MetaQuotes Ltd.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are calculated, without vectors.

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalRobust

Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the Multiple Relatively Robust Representations, MRRR algorithm (lapack function STEVR).

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalRobust(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalRobust(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

© 2000-2025, MetaQuotes Ltd.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input must be a symmetric matrix in the tridiagonal form.

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

ID	Description
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalBisect

Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the bisection algorithm (lapack function STEVX).

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalBisect(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   double                lower,             // lower bound of the subset
   double                upper,             // Upper bound of the subset
   double                abstol,            // absolute error tolerance
   vector&               eigen_values,      // vector of computed eigenvectors
   matrix&               eigen_vectors      // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalBisect(
   ENUM_EIG_VALUES       jobv,              // compute eigenvectors or not
   ENUM_BLAS_RANGE       range,             // subset of eigenvalues to compute
   float                 lower,             // lower bound of the subset
   float                 upper,             // upper bound of the subset
   float                 abstol,            // absolute error tolerance
   vectorf&              eigen_values,      // vector of computed eigenvectors
   matrixf&              eigen_vectors      // matrix of computed eigenvectors
   );
```

Parameters

jobv

[in] ENUM_EIG_VALUES enumeration value which determines the method for computing eigenvectors.

range

[in] ENUM_BLAS_RANGE enumeration value that defines a subset of computable eigenvalues and vectors.

lower

[in] The lower bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

upper

[in] The upper bound of eigenvalues subset; it is specified depending on the value of the '*range*' parameter.

abstol

[in] Absolute error tolerance.

The absolute error tolerance to which each eigenvalue/eigenvector is required.

© 2000-2025, MetaQuotes Ltd.

If jobv = 'V', the eigenvalues and eigenvectors output have residual norms bounded by abstol, and the dot products between different eigenvectors are bounded by abstol.

If abstol < n *eps*|T|, then n *eps*|T| is used instead, where eps is the machine precision, and |T| is the 1-norm of the matrix T. The eigenvalues are computed to an accuracy of eps*|T| irrespective of abstol.

If high relative precision is important, 'abstol' should be set to a safe minimum value X such that 1.0/X does not overflow.

eigen_values

[out] Vector of eigenvalues.

V

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the values of the *jobv* and *range* parameters.

When BLASRANGE_A is set, all eigenvalues are computed, and the lower and upper parameters are ignored.

With the BLASRANGE_V value, only those eigenvalues (and their vectors) are computed, which fall within the range of real values specified by the 'lower' and 'upper' parameters.

With the BLASRANGE_I value, only those eigenvalues (and their vectors) are computed, which fall within the range of integer indices specified by the 'lower' and 'upper' parameters. For example, with lower=0 and upper=2, only the first three eigenvalues are computed.

The input must be a symmetric matrix in the tridiagonal form.

ENUM_EIG_VALUES

An enumeration defining the need to compute eigenvectors.

ID	Description
EIGVALUES_V	Eigenvectors and eigenvalues are calculated.
EIGVALUES_N	Only eigenvalues are computed, without vectors.

ENUM_BLAS_RANGE

An enumeration defining how right singular vectors should be computed.

ID	Description
BLASRANGE_A	All singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

ID	Description
BLASRANGE_V	All singular or eigenvalues in the half-open interval (VL,VU] will be found.
BLASRANGE_I	The IL-th through IU-th singular or eigenvalues will be found.

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalQL

Compute all eigenvalues of a symmetric tridiagonal matrix using the Pal-Walker-Kahan variant of the QL or QR algorithm (lapack function STERF).

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalQL(
   vector&               eigen_values        // vector of computed eigenvalues
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalQL(
   vectorf&              eigen_values        // vector of computed eigenvalues
   );
```

Parameters

eigen_values

[out] Vector of eigenvalues.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The input must be a symmetric matrix in the tridiagonal form.

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalDCQ

Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the divide-and-conquer algorithm (lapack function STEDC). Unlike EigenTridiagonalDC, this method can be used to compute the eigenvectors of the original symmetric matrix. A symmetric matrix can be reduced to tridiagonal form using the ReduceSymmetricToTridiagonal method. The orthogonal matrix Q obtained from this transformation is then used to compute the eigenvectors of the original symmetric matrix.

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalDCQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrix&               Q,                  // orthogonal matrix used in the reductio
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalDCQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixf&              Q,                  // orthogonal matrix used in the reductio
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenTridiagonalDCQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixc&              Q,                  // orthogonal matrix used in the reductio
   vector&               eigen_values,       // vector of computed eigenvalues
   matrixc&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenTridiagonalDCQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixcf&             Q,                  // orthogonal matrix used in the reductio
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

compv

[in] ENUM_EIGTRIDIAG_Z enumeration value which determines the method for computing eigenvectors.

Q

[in] Orthogonal matrix Q produced by method ReflectTridiagonalToQ.

© 2000-2025, MetaQuotes Ltd.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the compv parameter.

When compv = EIGCOMPZ_N, compute eigenvalues only, eigenvectors are not calculated.

If EIGCOMPZ_V is set, eigenvalues are computed and eigenvectors of original symmetric matrix are calculated also.

If EIGCOMPZ_I is set, eigenvalues are computed and eigenvectors of tridiagonal matrix are calculated also.

The input must be a symmetric matrix in the tridiagonal form.

ENUM_EIGTRIDIAG_Z

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGCOMPZ_N	'N': Compute eigenvalues only
EIGCOMPZ_V	'V': Compute eigenvectors of original symmetric matrix also
EIGCOMPZ_I	'I': Compute eigenvectors of tridiagonal matrix also

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalQRQ

Compute eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the QR algorithm (lapack function STEQR). Unlike EigenTridiagonalQR, this method can be used to compute the eigenvectors of the original symmetric matrix. A symmetric matrix can be reduced to tridiagonal form using the ReduceSymmetricToTridiagonal method. The orthogonal matrix Q obtained from this transformation is then used to compute the eigenvectors of the original symmetric matrix.

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalQRQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrix&               Q,                  // orthogonal matrix used in the reductio
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalQRQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixf&              Q,                  // orthogonal matrix used in the reductio
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenTridiagonalQRQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixc&              Q,                  // orthogonal matrix used in the reductio
   vector&               eigen_values,       // vector of computed eigenvalues
   matrixc&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenTridiagonalQRQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixcf&             Q,                  // orthogonal matrix used in the reductio
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

compv

[in] ENUM_EIGTRIDIAG_Z enumeration value which determines the method for computing eigenvectors.

Q

[in] Orthogonal matrix Q produced by method ReflectTridiagonalToQ.

© 2000-2025, MetaQuotes Ltd.

eigen_values

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the compv parameter.

When compv = EIGCOMPZ_N, compute eigenvalues only, eigenvectors are not calculated.

If EIGCOMPZ_V is set, eigenvalues are computed and eigenvectors of original symmetric matrix are calculated also.

If EIGCOMPZ_I is set, eigenvalues are computed and eigenvectors of tridiagonal matrix are calculated also.

The input must be a symmetric matrix in the tridiagonal form.

ENUM_EIGTRIDIAG_Z

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGCOMPZ_N	'N': Compute eigenvalues only
EIGCOMPZ_V	'V': Compute eigenvectors of original symmetric matrix also
EIGCOMPZ_I	'I': Compute eigenvectors of tridiagonal matrix also

© 2000-2025, MetaQuotes Ltd.

EigenTridiagonalPosDefQ

Compute eigenvalues and eigenvectors of a symmetric positive definite tridiagonal matrix using the QR algorithm (lapack function PTEQR). This method can be used to compute the eigenvectors of both the input tridiagonal matrix and the original positive definite symmetric matrix. This is achieved using the orthogonal matrix Q obtained during the reduction to tridiagonal form.

Computing for type matrix<double>

```
bool  matrix::EigenTridiagonalPosDefQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrix&               Q,                  // orthogonal matrix used in the reductio
   vector&               eigen_values,       // vector of computed eigenvalues
   matrix&               eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::EigenTridiagonalPosDefQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixf&              Q,                  // orthogonal matrix used in the reductio
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixf&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::EigenTridiagonalPosDefQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixc&              Q,                  // orthogonal matrix used in the reductio
   vector&               eigen_values,       // vector of computed eigenvalues
   matrixc&              eigen_vectors       // matrix of computed eigenvectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::EigenTridiagonalPosDefQ(
   ENUM_EIGTRIDIAG_Z     compv,              // compute eigenvectors or not
   matrixcf&             Q,                  // orthogonal matrix used in the reductio
   vectorf&              eigen_values,       // vector of computed eigenvalues
   matrixcf&             eigen_vectors       // matrix of computed eigenvectors
   );
```

Parameters

compv

[in] ENUM_EIGTRIDIAG_Z enumeration value which determines the method for computing eigenvectors.

Q

[in] Orthogonal matrix Q produced by method ReflectTridiagonalToQ.

eigen_values

© 2000-2025, MetaQuotes Ltd.

[out] Vector of eigenvalues.

eigen_vectors

[out] Matrix of eigenvectors.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Computation depends on the value of the compv parameter.

When compv = EIGCOMPZ_N, compute eigenvalues only, eigenvectors are not calculated.

If EIGCOMPZ_V is set, eigenvalues are computed and eigenvectors of original symmetric matrix are calculated also.

If EIGCOMPZ_I is set, eigenvalues are computed and eigenvectors of tridiagonal matrix are calculated also.

The input must be a symmetric positive definite tridiagonal matrix.

ENUM_EIGTRIDIAG_Z

An enumeration that specifies whether to calculate eigenvectors.

ID	Description
EIGCOMPZ_N	'N': Compute eigenvalues only
EIGCOMPZ_V	'V': Compute eigenvectors of original symmetric matrix also
EIGCOMPZ_I	'I': Compute eigenvectors of tridiagonal matrix also

© 2000-2025, MetaQuotes Ltd.

Orthogonal Factorizations

OpenBLAS предоставляет ряд процедур для разложения общей прямоугольной матрицы A размера m на n в произведение ортогональной (в комплексном случае — унитарной) и треугольной (или, возможно, трапециевидной) матриц.

Реальная матрица Q называется ортогональной, если $Q^T Q = I$; комплексная матрица Q называется унитарной, если $Q^H Q = I$. Ортогональные и унитарные матрицы обладают важным свойством — они сохраняют евклидову норму вектора:

$||x||_2 = ||Qx||_2$, если Q ортогональна или унитарна

Благодаря этому такие матрицы способствуют численной устойчивости, поскольку не усиливают ошибки округления.

Ортогональные разложения широко применяются при решении задач наименьших квадратов. Их также можно использовать для выполнения предварительных шагов при решении задач на собственные значения или сингулярные значения.

Функция	Выполняемое действие
FactorizationQR	Вычисляет QR-разложение общей матрицы размера m на n: A = Q * R. LAPACK-функция GEQRF.
FactorizationQRNonNeg	Вычисляет QR-разложение общей матрицы размера m на n: A = Q * R, где R — верхнетреугольная матрица с неотрицательными элементами на диагонали. LATPACK-функция GEQRFP.
FactorizationQRPivot	Вычисляет QR-разложение общей матрицы размера m на n с перестановкой столбцов: A * P = Q * R. LAPACK-функция GEQP3.
FactorizationLQ	Computes the LQ factorization of a general m-by-n matrix: A = L * Q. Lapack function GELQF.
FactorizationQL	Computes the QL factorization of a general m-by-n matrix: A = Q * L. Lapack function GEQLF.
FactorizationRQ	Computes the RQ factorization of a general m-by-n matrix: A = R * Q. Lapack function GERQF.

© 2000-2025, MetaQuotes Ltd.

FactorizationQR

Computes the QR factorization of a general m-by-n matrix: A = Q * R. Lapack function GEQRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationQR(
   bool            reduced,      // calculation mode reduced or complete
   matrix&         Q,            // orthogonal matrix Q
   matrix&         R             // upper triangular matrix R
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationQR(
   bool            reduced,      // calculation mode reduced or complete
   matrixf&        Q,            // orthogonal matrix Q
   matrixf&        R             // upper triangular matrix R
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationQR(
   bool            reduced,      // calculation mode reduced or complete
   matrixc&        Q,            // unitary matrix Q
   matrixc&        R             // upper triangular matrix R
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationQR(
   bool            reduced,      // calculation mode reduced or complete
   matrixcf&       Q,            // unitary matrix Q
   matrixcf&       R             // upper triangular matrix R
   );
```

Parameters

reduced

[in] Calculation mode. If reduced is true then matrices Q, R calculated with dimensions (M, K), (K, N). If reduced is false it means complete calculation of matrices Q, R with dimensions (M,M), (M,N).

Q

[out] Orthogonal or unitary matrix Q.

R

[out] Upper triangular matrix R.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

If *reduced* is true

If m >= n, matrix Q is of m-by-n sizes, matrix R is of n-by-n sizes.

If m < n, matrix Q is of m-by-m sizes, matrix R is of m-by-n sizes.

If *reduced* is false, matrix Q is of m-by-m sizes, matrix R is of m-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

FactorizationQRNonNeg

Computes the QR factorization of a general m-by-n matrix: A = Q * R. R is an upper triangular matrix with nonnegative diagonal entries. Lapack function GEQRFP.

Computing for type matrix<double>

```
bool  matrix::FactorizationQRNonNeg(
   bool             reduced,       // calculation mode reduced or complete
   matrix&          Q,             // orthogonal matrix Q
   matrix&          R              // upper triangular matrix R
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationQRNonNeg(
   bool             reduced,       // calculation mode reduced or complete
   matrixf&         Q,             // orthogonal matrix Q
   matrixf&         R              // upper triangular matrix R
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationQRNonNeg(
   bool             reduced,       // calculation mode reduced or complete
   matrixc&         Q,             // unitary matrix Q
   matrixc&         R              // upper triangular matrix R
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationQRNonNeg(
   bool             reduced,       // calculation mode reduced or complete
   matrixcf&        Q,             // unitary matrix Q
   matrixcf&        R              // upper triangular matrix R
   );
```

Parameters

reduced

[in] Calculation mode. If reduced is true then matrices Q, R calculated with dimensions (M, K), (K, N). If reduced is false it means complete calculation of matrices Q, R with dimensions (M,M), (M,N).

Q

[out] Orthogonal or unitary matrix Q.

R

[out] Upper triangular matrix R.

Return Value

© 2000-2025, MetaQuotes Ltd.

Return true if successful, otherwise false in case of an error.

Note

If *reduced* is true

If m >= n, matrix Q is of m-by-n sizes, matrix R is of n-by-n sizes.

If m < n, matrix Q is of m-by-m sizes, matrix R is of m-by-n sizes.

If *reduced* is false, matrix Q is of m-by-m sizes, matrix R is of m-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

FactorizationQRPivot

Computes the QR factorization of a general m-by-n matrix with column pivoting: A * P = Q * R. Lapack function GEQP3.

Computing for type matrix<double>

```
bool  matrix::FactorizationQRPivot(
   bool             reduced,      // calculation mode reduced or complete
   long[]&          jpvt,         // array with predefined permutations
   matrix&          Q,            // orthogonal matrix Q
   matrix&          R,            // upper triangular matrix R
   matrix&          P             // permutation matrix
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationQRPivot(
   bool             reduced,      // calculation mode reduced or complete
   long[]&          jpvt,         // array with predefined permutations
   matrixf&         Q,            // orthogonal matrix Q
   matrixf&         R,            // upper triangular matrix R
   matrixf&         P             // permutation matrix
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationQRPivot(
   bool             reduced,      // calculation mode reduced or complete
   long[]&          jpvt,         // array with predefined permutations
   matrixc&         Q,            // unitary matrix Q
   matrixc&         R,            // upper triangular matrix R
   matrixc&         P             // permutation matrix
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationQRPivot(
   bool             reduced,      // calculation mode reduced or complete
   long[]&          jpvt,         // array with predefined permutations
   matrixcf&        Q,            // unitary matrix Q
   matrixcf&        R,            // upper triangular matrix R
   matrixcf&        P             // permutation matrix
   );
```

Parameters

reduced

[in] Calculation mode. If reduced is true then matrices Q, R calculated with dimensions (M, K), (K, N). If reduced is false it means complete calculation of matrices Q, R with dimensions (M,M), (M,N).

© 2000-2025, MetaQuotes Ltd.

jpvt

[in] Integer array of dimension n. if jpvt(i) ≠ 0, the i-th column of A is moved to the beginning of AP before the computation, and fixed in place during the computation. If jpvt(i) = 0, the i-th column of A is a free column (that is, it may be interchanged during the computation with any other free column). If array has zero size (or not initialized), then all the columns of A assumed to be free.

Q

[out] Orthogonal or unitary matrix Q.

R

[out] Upper triangular matrix R.

P

[out] Permutation matrix P of n-by-n sizes.

Return Value

Return true if successful, otherwise false in case of an error.

Note

If *reduced* is true

If m >= n, matrix Q is of m-by-n sizes, matrix R is of n-by-n sizes.

If m < n, matrix Q is of m-by-m sizes, matrix R is of m-by-n sizes.

If *reduced* is false, matrix Q is of m-by-m sizes, matrix R is of m-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

FactorizationLQ

Computes the LQ factorization of a general m-by-n matrix: A = L * Q. Lapack function GELQF.

Computing for type matrix<double>

```
bool  matrix::FactorizationLQ(
   bool            reduced,      // calculation mode reduced or complete
   matrix&         L,            // lower triangular matrix L
   matrix&         Q             // orthogonal matrix Q
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationLQ(
   bool            reduced,      // calculation mode reduced or complete
   matrixf&        L,            // lower triangular matrix L
   matrixf&        Q             // orthogonal matrix Q
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationLQ(
   bool            reduced,      // calculation mode reduced or complete
   matrixc&        L,            // lower triangular matrix L
   matrixc&        Q             // unitary matrix Q
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationLQ(
   bool            reduced,      // calculation mode reduced or complete
   matrixcf&       L,            // lower triangular matrix L
   matrixcf&       Q             // unitary matrix Q
   );
```

Parameters

reduced

[in] Calculation mode. If reduced is true then matrices L, Q calculated with reduced dimensions (M, K), (K, N). If reduced is false it means complete calculation of matrices L, Q with dimensions (M,N), (N,N).

L

[out] Lower triangular matrix L.

Q

[out] Orthogonal or unitary matrix Q.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

If *reduced* is true

If m <= n, matrix L is of m-by-m sizes, matrix Q is of m-by-n sizes.

If m > n, matrix L is of m-by-n sizes, matrix Q is of n-by-n sizes.

If *reduced* is false, matrix L is of m-by-n sizes, matrix Q is of n-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

FactorizationQL

Computes the QL factorization of a general m-by-n matrix: A = Q * L. Lapack function GEQLF.

Computing for type matrix<double>

```
bool  matrix::FactorizationQL(
   bool            reduced,      // calculation mode reduced or complete
   matrix&         Q,            // orthogonal matrix Q
   matrix&         L             // lower triangular matrix L
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationQL(
   bool            reduced,      // calculation mode reduced or complete
   matrixf&        Q,            // orthogonal matrix Q
   matrixf&        L             // lower triangular matrix L
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationQL(
   bool            reduced,      // calculation mode reduced or complete
   matrixc&        Q,            // unitary matrix Q
   matrixc&        L             // lower triangular matrix L
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationQL(
   bool            reduced,      // calculation mode reduced or complete
   matrixcf&       Q,            // unitary matrix Q
   matrixcf&       L             // lower triangular matrix L
   );
```

Parameters

reduced

[in] Calculation mode. If reduced is true then matrices Q, L calculated with reduced dimensions (M, K), (K, N). If reduced is false it means complete calculation of matrices Q, L with dimensions (M,M), (M,N).

Q

[out] Orthogonal or unitary matrix Q.

L

[out] Lower triangular matrix L.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

If *reduced* is true

If m >= n, matrix Q is of m-by-n sizes, matrix L is of n-by-n sizes.

If m < n, matrix Q is of m-by-m sizes, matrix L is of m-by-n sizes.

If *reduced* is false, matrix Q is of m-by-m sizes, matrix L is of m-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

FactorizationRQ

Computes the RQ factorization of a general m-by-n matrix: A = R * Q. Lapack function GERQF.

Computing for type matrix<double>

```
bool  matrix::FactorizationRQ(
   bool            reduced,      // calculation mode reduced or complete
   matrix&         R,            // upper triangular matrix R
   matrix&         Q             // orthogonal matrix Q
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationRQ(
   bool            reduced,      // calculation mode reduced or complete
   matrixf&        R,            // upper triangular matrix R
   matrixf&        Q             // orthogonal matrix Q
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationRQ(
   bool            reduced,      // calculation mode reduced or complete
   matrixc&        R,            // upper triangular matrix R
   matrixc&        Q             // unitary matrix Q
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationRQ(
   bool            reduced,      // calculation mode reduced or complete
   matrixcf&       R,            // upper triangular matrix R
   matrixcf&       Q             // unitary matrix Q
   );
```

Parameters

reduced

[in] Calculation mode. If reduced is true then matrices R, Q calculated with reduced dimensions (M, K), (K, N). If reduced is false it means complete calculation of matrices L, Q with dimensions (M,N), (N,N).

R

[out] Upper triangular matrix R.

Q

[out] Orthogonal or unitary matrix Q.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

If *reduced* is true

If m <= n, matrix R is of m-by-m sizes, matrix Q is of m-by-n sizes.

If m > n, matrix R is of m-by-n sizes, matrix Q is of n-by-n sizes.

If *reduced* is false, matrix R is of m-by-n sizes, matrix Q is of n-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

Linear Equations

This section provides functions for solving systems of linear equations of the form A·X = B, where A is a square coefficient matrix, B is the right-hand side, and X is the solution matrix or vector. The methods support various data types (double, float, complex, complexf) and are based on LAPACK functions to ensure high precision and performance.

The functions are categorized based on the type of the coefficient matrix A. Each function is overloaded to support both matrix and vector forms for the right-hand side (B) and the solution (X).

Function	Action
LinearEquationsSolution	Computes the solution to the system of linear equations with a square coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n general matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function GESV.
LinearEquationsSolutionTriangular	Computes the solution to the system of linear equations with a square triangular coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n general matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function TRTRS.
LinearEquationsSolutionSy	Computes the solution to the system of linear equations with a symmetric or Hermitian conjugated matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n symmetric or Hermitian conjugated matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack functions SYSV, HESV.
LinearEquationsSolutionComplexSy	Computes the solution to the system of linear equations with a complex symmetric (not Hermitian conjugated!) matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n complex symmetric matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function SYSV.
LinearEquationsSolutionSyPD	Computes the solution to the system of linear equations with a symmetric or Hermitian conjugated positive-definite matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n symmetric or Hermitian conjugated positive-definite matrix, the columns of matrix B are individual right-hand sides,

© 2000-2025, MetaQuotes Ltd.

Function	Action
	and the columns of X are the corresponding solutions. Lapack function POSV.
LinearEquationsSolutionGeTrid	Computes the solution to the system of linear equations with a general (nonsymmetric) tridiagonal coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n general tridiagonal matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function GTSV.
LinearEquationsSolutionSyTridPD	Computes the solution to the system of linear equations with a symmetric tridiagonal positive-definite coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n symmetric tridiagonal matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function PTSV.

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolution

Computes the solution to the system of linear equations with a square coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n general matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function GESV.

Computing for type matrix<double>

```
bool  matrix::LinearEquationsSolution(
   matrix&          B,             // right hand side matrix B
   matrix&          X              // solution matrix X
   );

bool  matrix::LinearEquationsSolution(
   vector&          B,             // right hand side vector B
   vector&          X              // solution vector X
   );
```

Computing for type matrix<float>

```
bool  matrix::LinearEquationsSolution(
   matrixf&         B,             // right hand side matrix B
   matrixf&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolution(
   vectorf&          B,             // right hand side vector B
   vectorf&          X              // solution vector X
   );
```

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolution(
   matrixc&         B,             // right hand side matrix B
   matrixc&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolution(
   vectorc&         B,             // right hand side vector B
   vectorc&         X              // solution vector X
   );
```

Computing for type matrix<complexf>

```
bool  matrix::LinearEquationsSolution(
   matrixcf&        B,             // right hand side matrix B
   matrixcf&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolution(
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorcf&       B,              // right hand side vector B
   vectorcf&       X               // solution vector X
   );
```

Parameters

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolutionTriangular

Computes the solution to the system of linear equations with a square triangular coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n general matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function TRTRS.

Computing for type matrix<double>

```
bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   matrix&             B,            // right hand side matrix B
   matrix&             X             // solution matrix X
   );

bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   vector&             B,            // right hand side vector B
   vector&             X             // solution vector X
   );
```

Computing for type matrix<float>

```
bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   matrixf&            B,            // right hand side matrix B
   matrixf&            X             // solution matrix X
   );

bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   vectorf&            B,            // right hand side vector B
   vectorf&            X             // solution vector X
   );
```

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   matrixc&            B,            // right hand side matrix B
   matrixc&            X             // solution matrix X
   );

bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   vectorc&            B,            // right hand side vector B
   vectorc&            X             // solution vector X
   );
```

Computing for type matrix<complexf>

© 2000-2025, MetaQuotes Ltd.

```
bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   matrixcf&           B,            // right hand side matrix B
   matrixcf&           X             // solution matrix X
   );

bool  matrix::LinearEquationsSolutionTriangular(
   ENUM_EQUATIONS_FORM trans,        // form of the system of equations
   vectorcf&           B,            // right hand side vector B
   vectorcf&           X             // solution vector X
   );
```

Parameters

trans

[in] ENUM_EQUATIONS_FORM enumeration value which specifies the form of the system of equations.

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

ENUM_EQUATIONS_FORM

An enumeration defining which form of the equations' system calculated.

ID	Description
EQUATIONSFORM_N	'N': A * X = B (No transpose)
EQUATIONSFORM_T	'T': A**T * X = B (Transpose)
EQUATIONSFORM_C	'C': A**H * X = B (Conjugate transpose)

In case of real matrices the value EQUATIONSFORM_C assumed as Transpose.

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolutionSy

Computes the solution to the system of linear equations with a symmetric or Hermitian conjugated matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n symmetric or unitary matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack functions SYSV, HESV.

Computing for type matrix<double>

```
bool  matrix::LinearEquationsSolutionSy(
   matrix&          B,             // right hand side matrix B
   matrix&          X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSy(
   vector&          B,             // right hand side vector B
   vector&          X              // solution vector X
   );
```

Computing for type matrix<float>

```
bool  matrix::LinearEquationsSolutionSy(
   matrixf&         B,             // right hand side matrix B
   matrixf&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSy(
   vectorf&          B,             // right hand side vector B
   vectorf&          X              // solution vector X
   );
```

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolutionSy(
   matrixc&         B,             // right hand side matrix B
   matrixc&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSy(
   vectorc&         B,             // right hand side vector B
   vectorc&         X              // solution vector X
   );
```

Computing for type matrix<complexf>

```
bool  matrix::LinearEquationsSolutionSy(
   matrixcf&        B,             // right hand side matrix B
   matrixcf&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSy(
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorcf&       B,              // right hand side vector B
   vectorcf&       X               // solution vector X
   );
```

Parameters

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

The input can be a symmetric (Hermitian conjugated), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolutionComplexSy

Computes the solution to the system of linear equations with a complex symmetric (not Hermitian conjugated!) matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n complex symmetric matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function SYSV.

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolutionComplexSy(
   matrixc&        B,             // right hand side matrix B
   matrixc&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionComplexSy(
   vectorc&        B,             // right hand side vector B
   vectorc&        X              // solution vector X
   );
```

Computing for type matrix<complexf>

```
bool  matrix::LinearEquationsSolutionComplexSy(
   matrixcf&       B,             // right hand side matrix B
   matrixcf&       X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionComplexSy(
   vectorcf&       B,             // right hand side vector B
   vectorcf&       X              // solution vector X
   );
```

Parameters

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

The input can be a complex symmetric, upper triangular or lower triangular matrix. Triangular matrices are assumed to be complex symmetric.

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolutionSyPD

Computes the solution to the system of linear equations with a symmetric or Hermitian conjugated positive-definite matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n symmetric or unitary positive-definite matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function POSV.

Computing for type matrix<double>

```
bool  matrix::LinearEquationsSolutionSyPD(
   matrix&          B,             // right hand side matrix B
   matrix&          X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyPD(
   vector&          B,             // right hand side vector B
   vector&          X              // solution vector X
   );
```

Computing for type matrix<float>

```
bool  matrix::LinearEquationsSolutionSyPD(
   matrixf&         B,             // right hand side matrix B
   matrixf&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyPD(
   vectorf&          B,             // right hand side vector B
   vectorf&          X              // solution vector X
   );
```

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolutionSyPD(
   matrixc&         B,             // right hand side matrix B
   matrixc&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyPD(
   vectorc&         B,             // right hand side vector B
   vectorc&         X              // solution vector X
   );
```

Computing for type matrix<complexf>

```
bool  matrix::LinearEquationsSolutionSyPD(
   matrixcf&        B,             // right hand side matrix B
   matrixcf&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyPD(
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorcf&       B,              // right hand side vector B
   vectorcf&       X               // solution vector X
   );
```

Parameters

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

The input can be a symmetric (Hermitian conjugated), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolutionGeTrid

Computes the solution to the system of linear equations with a general (nonsymmetric) tridiagonal coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n general tridiagonal matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function GTSV.

Computing for type matrix<double>

```
bool  matrix::LinearEquationsSolutionGeTrid(
   matrix&         B,             // right hand side matrix B
   matrix&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionGeTrid(
   vector&         B,             // right hand side vector B
   vector&         X              // solution vector X
   );
```

Computing for type matrix<float>

```
bool  matrix::LinearEquationsSolutionGeTrid(
   matrixf&        B,             // right hand side matrix B
   matrixf&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionGeTrid(
   vectorf&         B,             // right hand side vector B
   vectorf&         X              // solution vector X
   );
```

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolutionGeTrid(
   matrixc&        B,             // right hand side matrix B
   matrixc&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionGeTrid(
   vectorc&        B,             // right hand side vector B
   vectorc&        X              // solution vector X
   );
```

Computing for type matrix<complexf>

```
bool  matrix::LinearEquationsSolutionGeTrid(
   matrixcf&       B,             // right hand side matrix B
   matrixcf&       X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionGeTrid(
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorcf&       B,              // right hand side vector B
   vectorcf&       X               // solution vector X
   );
```

Parameters

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

© 2000-2025, MetaQuotes Ltd.

LinearEquationsSolutionSyTridPD

Computes the solution to the system of linear equations with a symmetric tridiagonal positive-definite coefficient matrix A and multiple right-hand sides. A*X = B, where A is an n-by-n symmetric tridiagonal matrix, the columns of matrix B are individual right-hand sides, and the columns of X are the corresponding solutions. Lapack function PTSV.

Computing for type matrix<double>

```
bool  matrix::LinearEquationsSolutionSyTridPD(
   matrix&          B,             // right hand side matrix B
   matrix&          X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyTridPD(
   vector&          B,             // right hand side vector B
   vector&          X              // solution vector X
   );
```

Computing for type matrix<float>

```
bool  matrix::LinearEquationsSolutionSyTridPD(
   matrixf&         B,             // right hand side matrix B
   matrixf&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyTridPD(
   vectorf&          B,             // right hand side vector B
   vectorf&          X              // solution vector X
   );
```

Computing for type matrix<complex>

```
bool  matrix::LinearEquationsSolutionSyTridPD(
   matrixc&         B,             // right hand side matrix B
   matrixc&         X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyTridPD(
   vectorc&         B,             // right hand side vector B
   vectorc&         X              // solution vector X
   );
```

Computing for type matrix<complexf>

```
bool  matrix::LinearEquationsSolutionSyTridPD(
   matrixcf&        B,             // right hand side matrix B
   matrixcf&        X              // solution matrix X
   );

bool  matrix::LinearEquationsSolutionSyTridPD(
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorcf&       B,              // right hand side vector B
   vectorcf&       X               // solution vector X
   );
```

Parameters

B

[in] Matrix B whose columns are the right-hand sides for the systems of equations. Vector B contains one column of right-hand side.

X

[out] Matrix or vector X with solutions of linear equations system.

Return Value

Return true if successful, otherwise false in case of an error.

Note

Output matrix X has the same sizes as input matrix B. Output vector X has the same size as input vector B.

© 2000-2025, MetaQuotes Ltd.

Factorizations

The Factorizations section contains functions for performing various types of matrix factorizations used in numerical solutions of linear systems, stability analysis, and other linear algebra tasks. These factorizations transform the original matrix into simpler forms, making subsequent computations more efficient. All functions are implemented using LAPACK routines and support the types double, float, complex, and complexf.

The functions in this section are used for:

- Preprocessing matrices when solving systems of linear equations;
- Computing determinants, ranks, and matrix inverses;
- Assessing the stability of numerical methods;
- Solving problems in spectral theory and optimization methods.

Matrix factorization is a critical step in many linear algebra algorithms, and this section provides access to the most efficient and well-established factorization techniques.

Function	Action
FactorizationPLU	Computes an LU factorization of a general M-by-N matrix A using partial pivoting with row interchanges. The factorization has the form A = P * L * U. Lapack function GETRF.
FactorizationPLUGeTrid	Computes an LU factorization of a general (non-symmetric) tridiagonal N-by-N matrix A using elimination with partial pivoting and row interchanges. The factorization has the form A = P * L * U. Lapack function GTTRF.
FactorizationLDL	Computes the factorization of a real symmetric or complex Hermitian matrix A using the Bunch-Kaufman diagonal pivoting method. Lapack functions SYTRF, HETRF.
FactorizationLDLSyTridPD	Forms the factorization of a symmetric positive-definite or, for complex data, Hermitian positive-definite tridiagonal matrix A. Lapack function PTTRF.
FactorizationCholesky	Computes the factorization of a real symmetric or complex Hermitian positive-definite matrix A. Lapack function POTRF.
FactorizationCholeskySyPS	Computes the Cholesky factorization with complete pivoting of a real symmetric (complex Hermitian) positive semidefinite N-by-N matrix. Lapack function PSTRF.

© 2000-2025, MetaQuotes Ltd.

FactorizationPLU

Computes an LU factorization of a general M-by-N matrix A using partial pivoting with row interchanges. The factorization has the form

A = P * L * U

where P is a permutation matrix, L is lower triangular with unit diagonal elements (lower trapezoidal if m > n), and U is upper triangular (upper trapezoidal if m < n). Lapack function GETRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationPLU(
   matrix&          P,             // permutation matrix P
   matrix&          L,             // lower triangular matrix L
   matrix&          U              // upper triangular matrix U
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationPLU(
   matrixf&         P,             // permutation matrix P
   matrixf&         L,             // lower triangular matrix L
   matrixf&         U              // upper triangular matrix U
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationPLU(
   matrixc&         P,             // permutation matrix P
   matrixc&         L,             // lower triangular matrix L
   matrixc&         U              // upper triangular matrix U
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationPLU(
   matrixcf&        P,             // permutation matrix P
   matrixcf&        L,             // lower triangular matrix L
   matrixcf&        U              // upper triangular matrix U
   );
```

Parameters

P

[out] Permutation matrix P.

L

[out] Lower triangular matrix L with unit diagonal elements.

U

[out] Upper triangular matrix U.

© 2000-2025, MetaQuotes Ltd.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

FactorizationPLUGeTrid

Computes an LU factorization of a general (non-symmetric) tridiagonal N-by-N matrix A using elimination with partial pivoting and row interchanges. The factorization has the form

A = P * L * U

where P is a permutation matrix, L is lower triangular with unit diagonal elements, and U is upper triangular. Lapack function GTTRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationPLUGeTrid(
   matrix&          P,             // permutation matrix P
   matrix&          L,             // lower triangular matrix L
   matrix&          U              // upper triangular matrix U
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationPLUGeTrid(
   matrixf&         P,             // permutation matrix P
   matrixf&         L,             // lower triangular matrix L
   matrixf&         U              // upper triangular matrix U
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationPLUGeTrid(
   matrixc&         P,             // permutation matrix P
   matrixc&         L,             // lower triangular matrix L
   matrixc&         U              // upper triangular matrix U
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationPLUGeTrid(
   matrixcf&        P,             // permutation matrix P
   matrixcf&        L,             // lower triangular matrix L
   matrixcf&        U              // upper triangular matrix U
   );
```

Parameters

P

[out] Permutation matrix P.

L

[out] Lower triangular matrix L with unit diagonal elements.

U

[out] Upper triangular matrix U.

© 2000-2025, MetaQuotes Ltd.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

FactorizationLDL

Computes the factorization of a real symmetric or complex Hermitian matrix A using the Bunch-Kaufman diagonal pivoting method. The form of the factorization is:

A = L * D * L**T in case of lower triangular or symmetric matrix A

or

A = U**T * D * U in case of upper triangular matrix A

where L is lower triangular with unit diagonal elements, U is upper triangular with unit diagonal elements. D is a symmetric block-diagonal matrix with 1-by-1 and 2-by-2 diagonal blocks. Lapack functions SYTRF, HETRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationLDL(
   matrix&          L,             // lower or upper triangular matrix
   matrix&          D              // diagonal matrix D
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationLDL(
   matrixf&         L,             // lower or upper triangular matrix
   matrixf&         D              // diagonal matrix D
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationLDL(
   matrixc&         L,             // lower or upper triangular matrix
   matrixc&         D              // diagonal matrix D
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationLDL(
   matrixcf&        L,             // lower or upper triangular matrix
   matrixcf&        D              // diagonal matrix D
   );
```

Parameters

L

[out] Lower or upper triangular matrix with unit diagonal elements.

D

[out] Symmetric block-diagonal matrix D.

Return Value

© 2000-2025, MetaQuotes Ltd.

Return true if successful, otherwise false in case of an error.

Note

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

© 2000-2025, MetaQuotes Ltd.

FactorizationLDLSyTridPD

Forms the factorization of a symmetric positive-definite or, for complex data, Hermitian positive-definite tridiagonal matrix A:

A = L * D * L**T in case of lower triangular or symmetric matrix A

or

A = U**T * D * U in case of upper triangular matrix A

where L is lower triangular with unit diagonal elements, U is upper triangular with unit diagonal elements. D is diagonal matrix. Lapack function PTTRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationLDLSyTridPD(
   matrix&          L,             // lower or upper triangular matrix
   matrix&          D              // diagonal matrix D
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationLDLSyTridPD(
   matrixf&         L,             // lower or upper triangular matrix
   matrixf&         D              // diagonal matrix D
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationLDLSyTridPD(
   matrixc&         L,             // lower or upper triangular matrix
   matrixc&         D              // diagonal matrix D
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationLDLSyTridPD(
   matrixcf&        L,             // lower or upper triangular matrix
   matrixcf&        D              // diagonal matrix D
   );
```

Parameters

L

[out] Lower or upper triangular matrix with unit diagonal elements.

D

[out] Diagonal matrix D.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

Note

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

© 2000-2025, MetaQuotes Ltd.

FactorizationCholesky

Computes the factorization of a real symmetric or complex Hermitian positive-definite matrix A. The factorization has the form:

A = L * L**T in case of lower triangular or symmetric matrix A

or

A = U**T * U in case of upper triangular matrix A

where L is lower triangular, U is upper triangular. Lapack function POTRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationCholesky(
   matrix&          L              // lower or upper triangular matrix
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationCholesky(
   matrixf&         L              // lower or upper triangular matrix
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationCholesky(
   matrixc&         L              // lower or upper triangular matrix
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationCholesky(
   matrixcf&        L              // lower or upper triangular matrix
   );
```

Parameters

L

[out] Lower or upper triangular matrix.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

© 2000-2025, MetaQuotes Ltd.

FactorizationCholeskySyPS

Computes the Cholesky factorization with complete pivoting of a real symmetric (complex Hermitian) positive semidefinite N-by-N matrix. The form of the factorization is:

P**T * A * P = L * L**T in case of lower triangular or symmetric matrix A

or

P**T * A * P = U**T * U in case of upper triangular matrix A

where P is a permutation matrix, L is lower triangular, U is upper triangular. Lapack function PSTRF.

Computing for type matrix<double>

```
bool  matrix::FactorizationCholeskySyPS(
   double          tol           // tolerance
   matrix&         P,            // permutation matrix P
   matrix&         L             // lower or upper triangular matrix
   );
```

Computing for type matrix<float>

```
bool  matrix::FactorizationCholeskySyPS(
   float           tol           // tolerance
   matrixf&        P,            // permutation matrix P
   matrixf&        L             // lower or upper triangular matrix
   );
```

Computing for type matrix<complex>

```
bool  matrix::FactorizationCholeskySyPS(
   double          tol           // tolerance
   matrixc&        P,            // permutation matrix P
   matrixc&        L             // lower or upper triangular matrix
   );
```

Computing for type matrix<complexf>

```
bool  matrix::FactorizationCholeskySyPS(
   float           tol           // tolerance
   matrixcf&       P,            // permutation matrix P
   matrixcf&       L             // lower or upper triangular matrix
   );
```

Parameters

tol

[in] User defined tolerance. If tol < 0, then n*ε*max(A[k,k]), where ε is the machine precision, will be used. The algorithm terminates at the (k-1)st step, if the pivot <=tol.

P

[out] Permutation matrix P.

© 2000-2025, MetaQuotes Ltd.

L

[out] Lower or upper triangular matrix.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The input can be a symmetric (Hermitian), upper triangular or lower triangular matrix. Triangular matrices are assumed to be symmetric (Hermitian conjugated).

© 2000-2025, MetaQuotes Ltd.

Dynamic Mode Decomposition

The Dynamic Mode Decomposition (DMD) functions are designed to compute an evolution operator from data snapshots, enabling efficient analysis of linear dynamics in extended phase spaces. The methods provide memory-efficient representations of evolution operators, with DynamicModeDecompositionQR incorporating QR-based data compression for improved computational efficiency.

Function	Action
DynamicModeDecomposition	Compute the Dynamic Mode Decomposition (DMD) for a pair of data snapshot matrices. Lapack function GEDMD.
DynamicModeDecompositionQR	Compute the Dynamic Mode Decomposition (DMD) for a pair of data snapshot matrices, using a QR factorization based compression of the data. Lapack function GEDMDQ.

The Dynamic mode decomposition (DMD) method is an algorithm for searching for an evolution operator (inverse operator problem solutions) in a finite-dimensional problem solution space (numerical or experimentally obtained) in a set of solutions (slices, "snapshots") in some consecutive moments of time. Expansion of the phase space due to the use of a nonlinear basis (relative to the variables of the problem) allows us to construct a global linear operator describing a linear evolution in the extended "rectifying space" (the Coopman operator) and the Perron-Frobenius operator that is its adjoint one. The DMD method is equivalent to a compressed representation of a linear evolution operator in the form of a product of rectangular matrices, which provides significant savings in the required memory during calculations.

© 2000-2025, MetaQuotes Ltd.

DynamicModeDecomposition

Compute the Dynamic Mode Decomposition(по-русски "разложение по динамическим модам") (DMD) for a pair of data snapshot matrices. Lapack function GEDMD. For the input matrices X and Y such that Y = A*X with an unaccessible matrix A, GEDMD computes a certain number of Ritz pairs of A using the standard Rayleigh-Ritz extraction from a subspace of range(X) that is determined using the leading left singular vectors of X. Optionally, GEDMD returns the residuals of the computed Ritz pairs, the information needed for a refinement of the Ritz vectors, or the eigenvectors of the Exact DMD. Both matrices must be the same size.

Computing for type matrix<double>

```
bool  matrix::DynamicModeDecomposition(
   matrix&               B,                  // second snapshot matrix in the pai
   ENUM_DMD_SCALE        jobs,               // determines whether the initial da
   ENUM_DMD_EIGV         jobz,               // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS    jobr,               // determines whether to compute the
   ENUM_DMD_REFINE       jobf,               // specifies whether to store inform
   ENUM_SVD_ALG          whtsvd,             // allows for a selection of the SVD
   long                  nrnk,               // determines the mode how to comput
   double                tol,                // the tolerance for truncating smal
   vectorc&              eigen_values,       // vector of eigenvalues (Ritz value
   matrix&               left_vectors,       // matrix of left singular vectors
   matrix&               Z                   // matrix of Ritz vectors
   vector&               residuals,          // residuals for the Ritz pairs
   matrix&               res_vectors,        // residual vectors for the Ritz pai
   matrix&               B,                  // computed vectors
   matrix&               W,                  // eigenvectors of the matrix Rayle
   matrix&               S                   // computed vectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::DynamicModeDecomposition(
   matrixf&              B,                  // second snapshot matrix in the pai
   ENUM_DMD_SCALE        jobs,               // determines whether the initial da
   ENUM_DMD_EIGV         jobz,               // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS    jobr,               // determines whether to compute the
   ENUM_DMD_REFINE       jobf,               // specifies whether to store inform
   ENUM_SVD_ALG          whtsvd,             // allows for a selection of the SVD
   long                  nrnk,               // determines the mode how to comput
   float                 tol,                // the tolerance for truncating smal
   vectorcf&             eigen_values,       // vector of eigenvalues (Ritz value
   matrixf&              left_vectors,       // matrix of left singular vectors
   matrixf&              Z                   // matrix of Ritz vectors
   vectorf&              residuals,          // residuals for the Ritz pairs
   matrixf&              res_vectors,        // residual vectors for the Ritz pai
   matrixf&              B,                  // computed vectors
   matrixf&              W,                  // eigenvectors of the matrix Rayle
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixf&              S                       // computed vectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::DynamicModeDecomposition(
   matrixc&              B,                      // second snapshot matrix in the pai
   ENUM_DMD_SCALE        jobs,                   // determines whether the initial da
   ENUM_DMD_EIGV         jobz,                   // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS    jobr,                   // determines whether to compute the
   ENUM_DMD_REFINE       jobf,                   // specifies whether to store inform
   ENUM_SVD_ALG          whtsvd,                 // allows for a selection of the SVD
   long                  nrnk,                   // determines the mode how to comput
   double                tol,                    // the tolerance for truncating smal
   vectorc&              eigen_values,           // vector of eigenvalues (Ritz value
   matrixc&              left_vectors,           // matrix of left singular vectors
   matrixc&              Z                       // matrix of Ritz vectors
   vector&               residuals,              // residuals for the Ritz pairs
   matrixc&              res_vectors,            // residual vectors for the Ritz pai
   matrixc&              B,                      // computed vectors
   matrixc&              W,                      // eigenvectors of the matrix Rayle
   matrixc&              S                       // computed vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::DynamicModeDecomposition(
   matrixcf&             B,                      // second snapshot matrix in the pai
   ENUM_DMD_SCALE        jobs,                   // determines whether the initial da
   ENUM_DMD_EIGV         jobz,                   // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS    jobr,                   // determines whether to compute the
   ENUM_DMD_REFINE       jobf,                   // specifies whether to store inform
   ENUM_SVD_ALG          whtsvd,                 // allows for a selection of the SVD
   long                  nrnk,                   // determines the mode how to comput
   float                 tol,                    // the tolerance for truncating smal
   vectorcf&             eigen_values,           // vector of eigenvalues (Ritz value
   matrixcf&             left_vectors,           // matrix of left singular vectors
   matrixcf&             Z                       // matrix of Ritz vectors
   vectorf&              residuals,              // residuals for the Ritz pairs
   matrixcf&             res_vectors,            // residual vectors for the Ritz pai
   matrixcf&             B,                      // computed vectors
   matrixcf&             W,                      // eigenvectors of the matrix Rayle
   matrixcf&             S                       // computed vectors
   );
```

Parameters

Y

[in] The second snapshot matrix in the pair.

© 2000-2025, MetaQuotes Ltd.

jobs

[in] Value from the ENUM_DMD_SCALE enumeration which determines whether the initial data snapshots are scaled by a diagonal matrix.

jobz

[in] Value from the ENUM_DMD_EIGV enumeration which determines whether the eigenvectors (Koopman modes) will be computed.

jobr

[in] Value from the ENUM_DMD_RESIDUALS enumeration which determines whether to compute the residuals.

jobf

[in] Value from the ENUM_DMD_REFINE enumeration which specifies whether to store information needed for post-processing (e.g. computing refined Ritz vectors).

whtsvd

[in] Value from the ENUM_SVD_ALG enumeration which allows for a selection of the SVD algorithm from the LAPACK library.

nrnk

[in] Value determines the mode how to compute the numerical rank, i.e. how to truncate small singular values of the input matrix X. If

NRNK = -1 :: i-th singular value sigma(i) is truncated if sigma(i) <= TOL*sigma(1). This option is recommended.

NRNK = -2 :: i-th singular value sigma(i) is truncated if sigma(i) <= TOL*sigma(i-1). This option is included for R&D purposes. It requires highly accurate SVD, which may not be feasible.

The numerical rank can be enforced by using positive value of NRNK as follows: 0 < NRNK <= N :: at most NRNK largest singular values will be used. If the number of the computed nonzero singular values is less than NRNK, then only those nonzero values will be used and the actually used dimension is less than NRNK.

tol

[in] The tolerance for truncating small singular values. 0 <= TOL < 1

eigen_values

[out] Vector of eigenvalues of size K. The leading K (K<=N) entries of EIGS contain the computed eigenvalues (Ritz values).

left_vectors

[out] Matrix of left singular vectors. The leading K columns contain a POD basis (POD - Proper Orthogonal Decomposition, правильная ортогональная декомпозиция), i.e. the leading K left singular vectors of the input data matrix X All N columns of contain all left singular vectors of the input matrix X.

Z

[out] Matrix of Ritz vectors. If JOBZ =='V' then Z contains the Ritz vectors. Z(:,i) is an eigenvector of the i-th Ritz value; ||Z(:,i)||_2=1. If JOBZ == 'F', then the Z(:,i)'s are given implicitly as the columns of X(:,1:K)*W(1:K,1:K), i.e. X(:,1:K)*W(:,i) is an eigenvector corresponding to EIGS(i). The columns of W(1:k,1:K) are the computed eigenvectors of the K-by-K Rayleigh quotient.

© 2000-2025, MetaQuotes Ltd.

residuals

[out] Residuals for the K computed Ritz pairs.

res_vectors

[out] Residual vectors for the K computed Ritz pairs.

B

[out] M-by-K matrix (K<=N).

If JOBF =='R', B(1:M,1:K) contains A*U(:,1:K), and can be used for computing the refined vectors; see further details in the provided references.

If JOBF == 'E', B(1:M,1:K) contains A*U(:,1:K)*W(1:K,1:K), which are the vectors from the Exact DMD, up to scaling by the inverse eigenvalues.

If JOBF =='N', then B is not referenced.

W

[out] K-by-K matrix (K<=N). Contains the K computed eigenvectors of the matrix Rayleigh quotient. The Ritz vectors (returned in Z) are the product of X (containing a POD basis for the input matrix X) and W.

S

[out] K-by-K matrix (K<=N). The array S(1:K,1:K) is used for the matrix Rayleigh quotient. This content is overwritten during the eigenvalue decomposition by GEEV.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

The number of matrices rows (M) must not be less than the number of columns (N).

ENUM_DMD_SCALE

An enumeration that determines whether the initial data snapshots are scaled by a diagonal matrix.

ID	Description
DMDSCALE_S	'S': The data snapshots matrices X and Y are multiplied with a diagonal matrix D so that X*D has unit nonzero columns
DMDSCALE_C	'C': The snapshots are scaled as with the 'S' option.If it is found that an i-th column of X is zero vector and the corresponding i-th column of Y is non-zero, then the i-th column of Y is set to zero
DMDSCALE_Y	'Y': The data snapshots matrices X and Y are multiplied with a diagonal matrix D so that Y*D has unit nonzero columns
DMDSCALE_N	'N': No data scaling

© 2000-2025, MetaQuotes Ltd.

ENUM_DMD_EIGV

An enumeration which determines whether the eigenvectors (Koopman modes) will be computed.

ID	Description
DMDEIGV_V	'V': The eigenvectors (Koopman modes) will be computed
DMDEIGV_F	'F': The eigenvectors (Koopman modes) will be returned in factored form
DMDEIGV_N	'N': The eigenvectors are not computed

ENUM_DMD_RESIDUALS

An enumeration which determines whether to compute the residuals.

ID	Description
DMDRESIDUALS_R	'R': The residuals for the computed eigenpairs will be computed
DMDRESIDUALS_N	'N': The residuals are not computed

ENUM_DMD_REFINE

An enumeration which specifies whether to store information needed for post-processing.

ID	Description
DMDREFINE_R	'R': The matrix needed for the refinement of the Ritz vectors is computed and stored in the array B
DMDREFINE_E	'E': The unscaled eigenvectors of the Exact DMD are computed and returned in the array B
DMDREFINE_N	'N': No eigenvector refinement data is computed

ENUM_SVD_ALG

An enumeration selecting the SVD algorythm.

ID	Description
SVDALG_1	1: GESVD (the QR SVD algorithm)
SVDALG_2	2: GESDD (the Divide and Conquer algorithm)

© 2000-2025, MetaQuotes Ltd.

ID	Description
SVDALG_3	3: GESVDQ (the preconditioned QR SVD; this and 4 are the most accurate options)
SVDALG_4	4: GEJSV (the preconditioned Jacobi SVD; this and 3 are the most accurate options)

© 2000-2025, MetaQuotes Ltd.

DynamicModeDecompositionQR

Compute the Dynamic Mode Decomposition(по-русски "декомпозиция динамических мод") (DMD) for a pair of data snapshot matrices, using a QR factorization based compression of the data. Lapack function GEDMDQ. For the input matrices X and Y such that Y = A*X with an unaccessible matrix A, GEDMDQ computes a certain number of Ritz pairs of A using the standard Rayleigh-Ritz extraction from a subspace of range(X) that is determined using the leading left singular vectors of X. Optionally, GEDMDQ returns the residuals of the computed Ritz pairs, the information needed for a refinement of the Ritz vectors, or the eigenvectors of the Exact DMD.

The input M-by-N matrix F. The columns of F are the sequence of data snapshots from a single trajectory, taken at equidistant discrete times. It is assumed that the column norms of F are in the range of the normalized floating point numbers.

Computing for type matrix<double>

```
bool  matrix::DynamicModeDecompositionQR(
   ENUM_DMD_SCALE        jobs,                 // determines whether the initial da
   ENUM_DMDQ_EIGV        jobz,                 // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS    jobr,                 // determines whether to compute the
   ENUM_DMDQ_Q           jobq,                 // specifies whether to explicitly c
   ENUM_DMDQ_R           jobt,                 // specifies whether to return the u
   ENUM_DMD_REFINE       jobf,                 // specifies whether to store inform
   ENUM_SVD_ALG          whtsvd,               // allows for a selection of the SVD
   long                  nrnk,                 // determines the mode how to comput
   double                tol,                  // the tolerance for truncating smal
   vectorc&              eigen_values,         // vector of eigenvalues (Ritz value
   matrix&               Q,                    // Q factor from the QR factorizatio
   matrix&               R,                    // R factor from the QR factorizatio
   matrix&               left_vectors,         // matrix of left singular vectors
   matrix&               Z                     // matrix of Ritz vectors
   vector&               residuals,            // residuals for the Ritz pairs
   matrix&               B,                    // computed vectors
   matrix&               V,                    // eigenvectors of the matrix Rayleig
   matrix&               S                     // computed vectors
   );
```

Computing for type matrix<float>

```
bool  matrixf::DynamicModeDecompositionQR(
   ENUM_DMD_SCALE        jobs,                 // determines whether the initial da
   ENUM_DMDQ_EIGV        jobz,                 // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS    jobr,                 // determines whether to compute the
   ENUM_DMDQ_Q           jobq,                 // specifies whether to explicitly c
   ENUM_DMDQ_R           jobt,                 // specifies whether to return the u
   ENUM_DMD_REFINE       jobf,                 // specifies whether to store inform
   ENUM_SVD_ALG          whtsvd,               // allows for a selection of the SVD
   long                  nrnk,                 // determines the mode how to comput
   float                 tol,                  // the tolerance for truncating smal
```

© 2000-2025, MetaQuotes Ltd.

```
   vectorcf&            eigen_values,          // vector of eigenvalues (Ritz value
   matrixf&             Q,                     // Q factor from the QR factorizatio
   matrixf&             R,                     // R factor from the QR factorizatio
   matrixf&             left_vectors,          // matrix of left singular vectors
   matrixf&             Z                      // matrix of Ritz vectors
   vectorf&             residuals,             // residuals for the Ritz pairs
   matrixf&             B,                     // computed vectors
   matrixf&             V,                     // eigenvectors of the matrix Rayle
   matrixf&             S                      // computed vectors
   );
```

Computing for type matrix<complex>

```
bool  matrixc::DynamicModeDecompositionQR(
   ENUM_DMD_SCALE       jobs,                  // determines whether the initial da
   ENUM_DMDQ_EIGV       jobz,                  // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS   jobr,                  // determines whether to compute the
   ENUM_DMDQ_Q          jobq,                  // specifies whether to explicitly c
   ENUM_DMDQ_R          jobt,                  // specifies whether to return the u
   ENUM_DMD_REFINE      jobf,                  // specifies whether to store inform
   ENUM_SVD_ALG         whtsvd,                // allows for a selection of the SVD
   long                 nrnk,                  // determines the mode how to comput
   double               tol,                   // the tolerance for truncating smal
   vectorc&             eigen_values,          // vector of eigenvalues (Ritz value
   matrixc&             Q,                     // Q factor from the QR factorizatio
   matrixc&             R,                     // R factor from the QR factorizatio
   matrixc&             left_vectors,          // matrix of left singular vectors
   matrixc&             Z                      // matrix of Ritz vectors
   vector&              residuals,             // residuals for the Ritz pairs
   matrixc&             B,                     // computed vectors
   matrixc&             V,                     // eigenvectors of the matrix Rayle
   matrixc&             S                      // computed vectors
   );
```

Computing for type matrix<complexf>

```
bool  matrixcf::DynamicModeDecompositionQR(
   ENUM_DMD_SCALE       jobs,                  // determines whether the initial da
   ENUM_DMDQ_EIGV       jobz,                  // determines whether the eigenvecto
   ENUM_DMD_RESIDUALS   jobr,                  // determines whether to compute the
   ENUM_DMDQ_Q          jobq,                  // specifies whether to explicitly c
   ENUM_DMDQ_R          jobt,                  // specifies whether to return the u
   ENUM_DMD_REFINE      jobf,                  // specifies whether to store inform
   ENUM_SVD_ALG         whtsvd,                // allows for a selection of the SVD
   long                 nrnk,                  // determines the mode how to comput
   float                tol,                   // the tolerance for truncating smal
   vectorcf&            eigen_values,          // vector of eigenvalues (Ritz value
   matrixcf&            Q,                     // Q factor from the QR factorizatio
```

© 2000-2025, MetaQuotes Ltd.

```
   matrixcf&            R,                  // R factor from the QR factorizatio
   matrixcf&            left_vectors,       // matrix of left singular vectors
   matrixcf&            Z                   // matrix of Ritz vectors
   vectorf&             residuals,          // residuals for the Ritz pairs
   matrixcf&            B,                  // computed vectors
   matrixcf&            V,                  // eigenvectors of the matrix Rayle
   matrixcf&            S                   // computed vectors
   );
```

Parameters

jobs

[in] Value from the ENUM_DMD_SCALE enumeration which determines whether the initial data snapshots are scaled by a diagonal matrix. The data snapshots are the columns of F. The leading N-1 columns of F are denoted X and the trailing N-1 columns are denoted Y.

jobz

[in] Value from the ENUM_DMDQ_EIGV enumeration which determines whether the eigenvectors (Koopman modes) will be computed.

jobr

[in] Value from the ENUM_DMD_RESIDUALS enumeration which determines whether to compute the residuals.

jobq

[in] Value from the ENUM_DMDQ_Q enumeration which specifies whether to explicitly compute and return the unitary matrix from the QR factorization of the data snapshot matrix.

jobt

[in] Value from the ENUM_DMDQ_R enumeration which specifies whether to return the upper triangular factor from the QR factorization of the data snapshot matrix.

jobf

[in] Value from the ENUM_DMD_REFINE enumeration which specifies whether to store information needed for post-processing (e.g. computing refined Ritz vectors).

whtsvd

[in] Value from the ENUM_SVD_ALG enumeration which allows for a selection of the SVD algorithm from the LAPACK library.

nrnk

[in] Value determines the mode how to compute the numerical rank, i.e. how to truncate small singular values of the input matrix X. If

NRNK = -1 :: i-th singular value sigma(i) is truncated if sigma(i) <= TOL*sigma(1). This option is recommended.

NRNK = -2 :: i-th singular value sigma(i) is truncated if sigma(i) <= TOL*sigma(i-1). This option is included for R&D purposes. It requires highly accurate SVD, which may not be feasible.

The numerical rank can be enforced by using positive value of NRNK as follows: 0 < NRNK <= N-1 :: at most NRNK largest singular values will be used. If the number of the computed nonzero

© 2000-2025, MetaQuotes Ltd.

singular values is less than NRNK, then only those nonzero values will be used and the actually used dimension is less than NRNK.

tol

[in] The tolerance for truncating small singular values. 0 <= TOL < 1

eigen_values

[out] Vector of eigenvalues of size K. The leading K (K<N) entries of EIGS contain the computed eigenvalues (Ritz values).

Q

[out] Orthogonal matrix/factor of the QR factorization of the initial data snapshots matrix F.

R

[out] N-by-N upper triangular factor from the QR factorization of the data snapshot matrix F.

left_vectors

[out] Matrix of left singular vectors. The leading K columns of X contain the leading K left singular vectors of the hold representations of the leading N-1 snapshots in the orthonormal basis computed in the QR factorization of F. To lift them to the space of the left singular vectors U(:,1:K) of the input data, pre-multiply with the Q factor from the initial QR factorization.

Z

[out] M-by-K matrix

If JOBZ =='V' then Z contains the Ritz vectors. Z(:,i) is an eigenvector of the i-th Ritz value; ||Z(:,i)||_2=1.

If JOBZ == 'F', then the Z(:,i)'s are given implicitly as Z*V, where Z contains orthonormal matrix (the product of Q from the initial QR factorization and the SVD/POD_basis returned by GEDMD in X) and the second factor (the eigenvectors of the Rayleigh quotient) is in the matrix V, as returned by GEDMD. That is, X(:,1:K)*V(:,i) is an eigenvector corresponding to EIGS(i). The columns of V(1:K,1:K) are the computed eigenvectors of the K-by-K Rayleigh quotient.

residuals

[out] Residuals for the K computed Ritz pairs.

B

[out] N-by-K matrix (K<N).

If JOBF =='R', B(1:N,1:K) contains A*U(:,1:K), and can be used for computing the refined vectors.

If JOBF == 'E', B(1:N,1;K) contains A*U(:,1:K)*W(1:K,1:K), which are the vectors from the Exact DMD, up to scaling by the inverse eigenvalues.

In both cases, the content of B can be lifted to the original dimension of the input data by pre-multiplying with the Q factor from the initial QR factorization. Here A denotes a compression of the underlying operator.

If JOBF =='N', then B is not referenced.

V

[out] K-by-K matrix (K<N). Contains the the K eigenvectors of the Rayleigh quotient. The Ritz vectors (returned in Z) are the product of Q from the initial QR factorization.

© 2000-2025, MetaQuotes Ltd.

S

[out] K-by-K matrix (K<N). The array S(1:K,1:K) is used for the matrix Rayleigh quotient. This content is overwritten during the eigenvalue decomposition by GEEV.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

The number of matrices rows (M) must not be less than the number of columns (N).

ENUM_DMD_SCALE

An enumeration that determines whether the initial data snapshots are scaled by a diagonal matrix.

ID	Description
DMDSCALE_S	'S': The data snapshots matrices X and Y are multiplied with a diagonal matrix D so that X*D has unit nonzero columns
DMDSCALE_C	'C': The snapshots are scaled as with the 'S' option.If it is found that an i-th column of X is zero vector and the corresponding i-th column of Y is non-zero, then the i-th column of Y is set to zero
DMDSCALE_Y	'Y': The data snapshots matrices X and Y are multiplied with a diagonal matrix D so that Y*D has unit nonzero columns
DMDSCALE_N	'N': No data scaling

ENUM_DMDQ_EIGV

An enumeration which determines whether the eigenvectors (Koopman modes) will be computed.

ID	Description
DMDQEIGV_V	'V': The eigenvectors (Koopman modes) will be computed
DMDQEIGV_F	'F': The eigenvectors (Koopman modes) will be returned in factored form as the product Z*V
DMDQEIGV_Q	'Q': The eigenvectors (Koopman modes) will be returned in factored form as the product Q*Z
DMDQEIGV_N	'N': The eigenvectors are not computed

ENUM_DMD_RESIDUALS

© 2000-2025, MetaQuotes Ltd.

An enumeration which determines whether to compute the residuals.

ID	Description
DMDRESIDUALS_R	'R': The residuals for the computed eigenpairs will be computed
DMDRESIDUALS_N	'N': The residuals are not computed

ENUM_DMDQ_Q

An enumeration which specifies whether to explicitly compute and return the unitary matrix from the QR factorization.

ID	Description
DMDQQ_Q	'Q': The matrix Q of the QR factorization of the data snapshot matrix is computed
DMDQQ_N	'N': The matrix Q is not explicitly computed

ENUM_DMDQ_R

An enumeration whichspecifies whether to return the upper triangular factor from the QR factorization.

ID	Description
DMDQR_R	'R': The matrix R of the QR factorization of the data snapshot matrix is computed
DMDQR_N	'N': The matrix R is not explicitly computed

ENUM_DMD_REFINE

An enumeration which specifies whether to store information needed for post-processing.

ID	Description
DMDREFINE_R	'R': The matrix needed for the refinement of the Ritz vectors is computed and stored in the array B
DMDREFINE_E	'E': The unscaled eigenvectors of the Exact DMD are computed and returned in the array B
DMDREFINE_N	'N': No eigenvector refinement data is computed

© 2000-2025, MetaQuotes Ltd.

ENUM_SVD_ALG

An enumeration selecting the SVD algorythm.

ID	Description
SVDALG_1	1: GESVD (the QR SVD algorithm)
SVDALG_2	2: GESDD (the Divide and Conquer algorithm)
SVDALG_3	3: GESVDQ (the preconditioned QR SVD; this and 4 are the most accurate options)
SVDALG_4	4: GEJSV (the preconditioned Jacobi SVD; this and 3 are the most accurate options)

© 2000-2025, MetaQuotes Ltd.

Singular Spectrum Analysis

The section contains functions for decomposing a matrix into three components: orthogonal matrices and a diagonal matrix of singular values. SVD is applied to solve various linear algebra problems such as data dimensionality reduction, image compression, solving systems of equations, as well as data analysis and optimization. The main functions allow calculating singular values and vectors, reconstruct matrices, as well as approximate them with reduced rank accuracy.

Function	Action
SingularSpectrumAnalysisSpectrum	A method function for calculating the relative contributions of spectral components based on their eigenvalues.
SingularSpectrumAnalysisForecast	A method function for calculating reconstructed and predicted data using spectral components of the input time series.
SingularSpectrumAnalysisReconstructComponents	A method function for calculating reconstructed components of the input time series and their contributions.
SingularSpectrumAnalysisReconstructSeries	A method function for calculating the reconstructed time series using the first component_count components.

© 2000-2025, MetaQuotes Ltd.

SingularSpectrumAnalysisSpectrum

A method function for calculating the relative contributions of spectral components based on their eigenvalues.

Calculations for vector<double> type

```
bool  vector::SingularSpectrumAnalysisSpectrum(
   ulong   window_length,      // window size for constructing the trajectory matrix
   vector& spectrum            // vector of component contributions to the input serie
   );
```

Calculations for vector<complex> type

```
bool  vector::SingularSpectrumAnalysisSpectrum(
   ulong   window_length,      // window size for constructing the trajectory matrix
   vectorc& spectrum           // vector of component contributions to the input serie
   );
```

Parameters

window_length

[in] Window size for constructing the trajectory matrix, the number of components the input time series should be decomposed into.

spectrum

[out] Vector of component contributions to the input series - eigenvalues of the covariance matrix of the input time series.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

The *window_length* parameter value should be less than the size of the input time series. To construct a full-fledged trajectory matrix, the optimal size is considered to be approximately equal to half the size of the input time series.

© 2000-2025, MetaQuotes Ltd.

SingularSpectrumAnalysisForecast

A method function for calculating reconstructed and predicted data using spectral components of the input time series.

Calculations for vector<double> type

```
bool  vector::SingularSpectrumAnalysisForecast(
   ulong    window_length,         // window size for constructing the trajectory mat
   ulong    component_count,       // number of components used for forecasting
   ulong    forecast_horizon,      // number of points to forecast
   vector&  forecast               // vector consisting of reconstructed and predicte
   );
```

Calculations for vector<complex> type

```
bool  vector::SingularSpectrumAnalysisForecast(
   ulong    window_length,         // window size for constructing the trajectory mat
   ulong    component_count,       // number of components used for forecasting
   ulong    forecast_horizon,      // number of points to forecast
   vectorc& forecast               // vector consisting of reconstructed and predicte
   );
```

Parameters

window_length

[in] Window size for constructing the trajectory matrix, the number of components the input time series should be decomposed into.

component_count

[in] Number of components used for forecasting.

forecast_horizon

[in] Number of points to forecast.

forecast

[out] Combining points reconstructed by *component_count* plus *forecast_horizon* points predicted using the first *component_count* components. Thus, the *forecast* vector has the size of (T+forecast_horizon), where T is the input series length.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

The *window_length* parameter value should be less than the size of the input time series. To construct a full-fledged trajectory matrix, the optimal size is considered to be approximately equal to half the size of the input time series.

© 2000-2025, MetaQuotes Ltd.

SingularSpectrumAnalysisReconstructComponents

A method function for calculating reconstructed components of the input time series and their contributions.

Calculations for vector<double> and matrix<double> types

```
bool  vector::SingularSpectrumAnalysisReconstructComponents(
   ulong    window_length,      // window size for constructing the trajectory matrix
   matrix&  components,         // matrix of reconstructed components
   vector&  contributions       // vector of component contributions to the input seri
   );
```

Calculations for vector<complex> and matrix<complex> types

```
bool  vector::SingularSpectrumAnalysisReconstructComponents(
   ulong    window_length,      // window size for constructing the trajectory matrix
   matrixc& components,         // matrix of reconstructed components
   vectorc& contributions       // vector of component contributions to the input seri
   );
```

Parameters

window_length

[in] Window size for constructing the trajectory matrix, the number of components the input time series should be decomposed into.

components

[out] A matrix of reconstructed components, where each column describes a separate component.

contributions

[out] Vector of component contributions to the input series (eigenvalues of the covariance matrix of the input time series).

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

The *window_length* parameter value should be less than the size of the input time series. To construct a full-fledged trajectory matrix, the optimal size is considered to be approximately equal to half the size of the input time series.

© 2000-2025, MetaQuotes Ltd.

SingularSpectrumAnalysisReconstructSeries

A method function for calculating the reconstructed time series using the first *component_count* components.

Calculations for vector<double> type

```
bool  vector::SingularSpectrumAnalysisReconstructSeries(
   ulong    window_length,       // window size for constructing the trajectory matri
   ulong    component_count,     // number of components used for reconstruction
   vector&  reconstructed        // reconstructed time series
   );
```

Calculations for vector<complex> type

```
bool  vector::SingularSpectrumAnalysisReconstructSeries(
   ulong    window_length,       // window size for constructing the trajectory matri
   ulong    component_count,     // number of components used for reconstruction
   vectorc& reconstructed        // reconstructed time series
   );
```

Parameters

window_length

[in] Window size for constructing the trajectory matrix, the number of components the input time series should be decomposed into.

component_count

[out] Number of components used for time series reconstruction.

reconstructed

[out] Vector containing the reconstructed output series.

Return Value

The function returns 'true' on success or 'false' if an error occurs.

Note

The *window_length* parameter value should be less than the size of the input time series. To construct a full-fledged trajectory matrix, the optimal size is considered to be approximately equal to half the size of the input time series.

© 2000-2025, MetaQuotes Ltd.

Reductions

Functions designed to transform matrices into special forms using orthogonal transformations. These methods are widely used in numerical computations, such as spectral analysis, eigenvalue problems, and singular value decomposition (SVD) of matrices.

These functions enable efficient matrix transformations essential for solving linear algebra problems, including computing Singular Value Decomposition (SVD) and finding eigenvalues of symmetric matrices. The use of LAPACK functions ensures high performance and computational accuracy.

Function	Action
ReduceToBidiagonal	Reduces a general real or complex m-by-n matrix A to upper or lower bidiagonal form B by an orthogonal transformation: Q**T * A * P = B. If m >= n, B is upper bidiagonal; if m < n, B is lower bidiagonal. Lapack function GEBRD.
ReflectBidiagonalToQP	Generates orthogonal matrices Q and P**T (or P**H for complex types) determined by ReduceToBidiagonal method when reducing a real or complex matrix A to bidiagonal form: A = Q * B * P**T. Q and P**T are defined as products of elementary reflectors H(i) or G(i) respectively. Lapack functions ORGBR, UNGBR.
ReduceSymmetricToTridiagonal	Reduces a real symmetric or complex Hermitian matrix A to trdiagonal form B by an orthogonal similarity transformation: Q**T * A * Q = B. Lapack functions SYTRD, HETRD.
ReflectTridiagonalToQ	Generates orthogonal matrix Q which is defined as the product of n-1 elementary reflectors of order n, as returned by ReduceSymmetricToTridiagonal

© 2000-2025, MetaQuotes Ltd.

ReduceToBidiagonal

Reduces a general real or complex m-by-n matrix A to upper or lower bidiagonal form B by an orthogonal transformation: Q**T * A * P = B. If m >= n, B is upper bidiagonal; if m < n, B is lower bidiagonal. Lapack function GEBRD.

Computing for type matrix<double>

```
bool  matrix::ReduceToBidiagonal(
   matrix&          B,             // bidiagonal matrix
   matrix&          reflect_qp,    // q and p-reflectors
   vector&          tau_q,         // scalar factors of the elementary reflectors Q
   vector&          tau_p          // scalar factors of the elementary reflectors P
   );
```

Computing for type matrix<float>

```
bool  matrix::ReduceToBidiagonal(
   matrixf&         B,             // bidiagonal matrix
   matrixf&         reflect_qp,    // q and p-reflectors
   vectorf&         tau_q,         // scalar factors of the elementary reflectors Q
   vectorf&         tau_p          // scalar factors of the elementary reflectors P
   );
```

Computing for type matrix<complex>

```
bool  matrix::ReduceToBidiagonal(
   matrixc&         B,             // bidiagonal matrix
   matrixc&         reflect_qp,    // q and p-reflectors
   vectorc&         tau_q,         // scalar factors of the elementary reflectors Q
   vectorc&         tau_p          // scalar factors of the elementary reflectors P
   );
```

Computing for type matrix<complexf>

```
bool  matrix::ReduceToBidiagonal(
   matrixcf&        B,             // bidiagonal matrix
   matrixcf&        reflect_qp,    // q and p-reflectors
   vectorcf&        tau_q,         // scalar factors of the elementary reflectors Q
   vectorcf&        tau_p          // scalar factors of the elementary reflectors P
   );
```

Parameters

B

[out] Upper or lower bidiagonal matrix.

reflect_qp

[out] Transformed matrix A. If m >= n, the diagonal and the first superdiagonal are overwritten with the upper bidiagonal matrix B; the elements below the diagonal, with the vector tau_q, represent the orthogonal matrix Q as a product of elementary reflectors, and the elements above the first superdiagonal, with the vector tau_p, represent the orthogonal matrix P as a product of

© 2000-2025, MetaQuotes Ltd.

elementary reflectors; if m < n, the diagonal and the first subdiagonal are overwritten with the lower bidiagonal matrix B; the elements below the first subdiagonal, with the vector tau_q, represent the orthogonal matrix Q as a product of elementary reflectors, and the elements above the diagonal, with the vector tau_p, represent the orthogonal matrix P as a product of elementary reflectors.

tau_q

[out] Vector of the scalar factors of the elementary reflectors which represent the orthogonal matrix Q.

tau_p

[out] Vector of the scalar factors of the elementary reflectors which represent the orthogonal matrix P.

Return Value

Return true if successful, otherwise false in case of an error.

Note

The matrices Q and P are represented as products of elementary reflectors:

If m >= n,

Q = H(1) H(2) . . . H(n) and P = G(1) G(2) . . . G(n-1)

Each H(i) and G(i) has the form:

H(i) = I - tauq * v * v**T and G(i) = I - taup * u * u**T

where tauq and taup are scalars, and v and u are vectors;

v(1:i-1) = 0, v(i) = 1, and v(i+1:m) is stored on exit in A(i+1:m,i);

u(1:i) = 0, u(i+1) = 1, and u(i+2:n) is stored on exit in A(i,i+2:n);

tauq is stored in *tau_q*(i) and taup in *tau_p*(i).

If m < n,

Q = H(1) H(2) . . . H(m-1) and P = G(1) G(2) . . . G(m)

Each H(i) and G(i) has the form:

H(i) = I - tauq * v * v**T and G(i) = I - taup * u * u**T

where tauq and taup are scalars, and v and u are vectors;

v(1:i) = 0, v(i+1) = 1, and v(i+2:m) is stored on exit in A(i+2:m,i);

u(1:i-1) = 0, u(i) = 1, and u(i+1:n) is stored on exit in A(i,i+1:n);

tauq is stored in *tau_q*(i) and taup in *tau_p*(i).

The contents of A on exit (output matrix *reflect_qp*) are illustrated by the following examples:

© 2000-2025, MetaQuotes Ltd.

```
m = 6 and n = 5 (m > n):
   (  d   e   u1  u1  u1 )
   (  v1  d   e   u2  u2 )
   (  v1  v2  d   e   u3 )
   (  v1  v2  v3  d   e  )
   (  v1  v2  v3  v4  d  )
   (  v1  v2  v3  v4  v5 )

m = 5 and n = 6 (m < n):
   (  d   u1  u1  u1  u1  u1 )
   (  e   d   u2  u2  u2  u2 )
   (  v1  e   d   u3  u3  u3 )
   (  v1  v2  e   d   u4  u4 )
   (  v1  v2  v3  e   d   u5 )
```

where d and e denote diagonal and off-diagonal elements of B, vi denotes an element of the vector defining H(i), and ui an element of the vector defining G(i).

Matrices Q and P can be produced with ReflectBidiagonalToQP method.

© 2000-2025, MetaQuotes Ltd.

ReflectBidiagonalToQP

Generates orthogonal matrices Q and P**T (or P**H for complex types) determined by ReduceToBidiagonal method when reducing a real or complex matrix A to bidiagonal form: A = Q * B * P**T. Q and P**T are defined as products of elementary reflectors H(i) or G(i) respectively. Lapack functions ORGBR, UNGBR.

As input is used transformed matrix reflect_qp with the same sizes m-by-n as in original matrix A.

Computing for type matrix<double>

```
bool  matrix::ReflectBidiagonalToQP(
   vector&          tau_q,         // scalar factors of the elementary reflectors Q
   vector&          tau_p,         // scalar factors of the elementary reflectors P
   matrix&          Q,             // matrix Q
   matrix&          PT             // transposed matrix P
   );
```

Computing for type matrix<float>

```
bool  matrix::ReflectBidiagonalToQP(
   vectorf&         tau_q,         // scalar factors of the elementary reflectors Q
   vectorf&         tau_p,         // scalar factors of the elementary reflectors P
   matrixf&         Q,             // matrix Q
   matrixf&         PT             // transposed matrix P
   );
```

Computing for type matrix<complex>

```
bool  matrix::ReflectBidiagonalToQP(
   vectorc&         tau_q,         // scalar factors of the elementary reflectors Q
   vectorc&         tau_p,         // scalar factors of the elementary reflectors P
   matrixc&         Q,             // matrix Q
   matrixc&         PH             // hermitian conjugated matrix P
   );
```

Computing for type matrix<complexf>

```
bool  matrix::ReflectBidiagonalToQP(
   vectorcf&        tau_q,         // scalar factors of the elementary reflectors Q
   vectorcf&        tau_p,         // scalar factors of the elementary reflectors P
   matrixcf&        Q,             // matrix Q
   matrixcf&        PH             // hermitian conjugated matrix P
   );
```

Parameters

tau_q

[in] Vector of the scalar factors of the elementary reflectors which represent the orthogonal matrix Q.

tau_p

© 2000-2025, MetaQuotes Ltd.

[in] Vector of the scalar factors of the elementary reflectors which represent the orthogonal matrix P.

Q

[out] Orthogonal matrix Q.

PT (PH)

[out] Transposed (or hermitian conjugated) matrix P.

Return Value

Return true if successful, otherwise false in case of an error.

Note

If m >= n, matrix Q is of m-by-n sizes, matrix PT is of n-by-n sizes.

If m < n, matrix Q is of m-by-m sizes, matrix PT is of m-by-n sizes.

© 2000-2025, MetaQuotes Ltd.

ReduceSymmetricToTridiagonal

Reduces a real symmetric or complex Hermitian matrix A to trdiagonal form B by an orthogonal similarity transformation: Q**T * A * Q = B. Lapack functions SYTRD, HETRD.

Computing for type matrix<double>

```
bool  matrix::ReduceSymmetricToTridiagonal(
   matrix&         B,             // tridiagonal matrix
   matrix&         reflect_q,     // q-reflectors
   vector&         tau_q          // scalar factors of the elementary reflectors Q
   );
```

Computing for type matrix<float>

```
bool  matrix::ReduceSymmetricToTridiagonal(
   matrixf&        B,             // tridiagonal matrix
   matrixf&        reflect_q,     // q-reflectors
   vectorf&        tau_q          // scalar factors of the elementary reflectors Q
   );
```

Computing for type matrix<complex>

```
bool  matrix::ReduceSymmetricToTridiagonal(
   matrixc&        B,             // tridiagonal matrix
   matrixc&        reflect_q,     // q-reflectors
   vectorc&        tau_q          // scalar factors of the elementary reflectors Q
   );
```

Computing for type matrix<complexf>

```
bool  matrix::ReduceSymmetricToTridiagonal(
   matrixcf&       B,             // tridiagonal matrix
   matrixcf&       reflect_q,     // q and p-reflectors
   vectorcf&       tau_q          // scalar factors of the elementary reflectors Q
   );
```

Parameters

B

[out] Symmetric (or Hermitian) tridiagonal matrix.

reflect_q

[out] Upper or lower triangular matrix, it depends on the input matrix A. In upper case the diagonal and first superdiagonal of A are overwritten by the corresponding elements of the tridiagonal matrix B, and the elements above the first superdiagonal, with the array *tau_q*, represent the orthogonal matrix Q as a product of elementary reflectors; in lower case the diagonal and first subdiagonal of A are overwritten by the corresponding elements of the tridiagonal matrix B, and the elements below the first subdiagonal, with the array *tau_q*, represent the orthogonal matrix Q as a product of elementary reflectors.

tau_q

© 2000-2025, MetaQuotes Ltd.

[out] Vector of the scalar factors of the elementary reflectors which represent the orthogonal matrix Q.

Return Value

Return true if successful, otherwise false in case of an error.

Note

In upper case the matrix Q is represented as a product of elementary reflectors

Q = H(n-1) . . . H(2) H(1)

Each H(i) has the form

H(i) = I - tau * v * v**T

where tau is a scalar, and v is a vector with v(i+1:n) = 0 and v(i) = 1; v(1:i-1) is stored on exit in A(1:i-1,i+1), and tau in *tau_q*(i).

In lower case the matrix Q is represented as a product of elementary reflectors

Q = H(1) H(2) . . . H(n-1)

Each H(i) has the form

H(i) = I - tau * v * v**T

where tau is a scalar, and v is a vector with v(1:i) = 0 and v(i+1) = 1; v(i+2:n) is stored on exit in A(i+2:n,i), and tau in *tau_q*(i). */

The contents of A on exit (output matrix *reflect_q*) are illustrated by the following examples with n = 5:

```
      upper case:                          lower case:

      (  d   e   v2  v3  v4 )              (  d                   )
      (      d   e   v3  v4 )              (  e   d               )
      (          d   e   v4 )              (  v1  e   d           )
      (              d   e  )              (  v1  v2  e   d       )
      (                  d  )              (  v1  v2  v3  e   d   )
```

where d and e denote diagonal and off-diagonal elements of B and vi denotes an element of the vector defining H(i).

Matrix Q can be produced with ReflectTridiagonalToQ method.

© 2000-2025, MetaQuotes Ltd.

ReflectTridiagonalToQ

Generates orthogonal matrix Q which is defined as the product of n-1 elementary reflectors of order n, as returned by ReduceSymmetricToTridiagonal:

in upper case Q = H(n-1) . . . H(2) H(1),

in lower case Q = H(1) H(2) . . . H(n-1).

Lapack functions ORGTR, UNGTR.

As input is used transformed matrix reflect_q with the same sizes n-by-n as in original matrix A.

Computing for type matrix<double>

```
bool  matrix::ReflectTridiagonalToQ(
   vector&          tau_q,         // scalar factors of the elementary reflectors Q
   matrix&          Q              // matrix Q
   );
```

Computing for type matrix<float>

```
bool  matrix::ReflectTridiagonalToQ(
   vectorf&         tau_q,         // scalar factors of the elementary reflectors Q
   matrixf&         Q              // matrix Q
   );
```

Computing for type matrix<complex>

```
bool  matrix::ReflectTridiagonalToQ(
   vectorc&         tau_q,         // scalar factors of the elementary reflectors Q
   matrixc&         Q              // matrix Q
   );
```

Computing for type matrix<complexf>

```
bool  matrix::ReflectTridiagonalToQ(
   vectorcf&        tau_q,         // scalar factors of the elementary reflectors Q
   matrixcf&        Q              // matrix Q
   );
```

Parameters

tau_q

[in] Vector of the scalar factors of the elementary reflectors which represent the orthogonal matrix Q.

Q

[out] Orthogonal matrix Q.

Return Value

Return true if successful, otherwise false in case of an error.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Datenverarbeitung

Gruppe von Funktionen, die die Angaben des bestimmten Formats in die Angaben des anderen Formats umwandelt.

Die Funktion NormalizeDoubles(muss besonders betont werden, denn sie gewaehrleistet die erforderliche Genaiugkeit der Preisdarstellung. In Handelsopeartionen können nicht normalisierte Preise, deren Genauigkeit um wenigstens 1 Zeichen die vom Handelsserver angeforderte Genauigkeit überschreitet, nicht verwendet werden.

Funktion	Massnahme
CharToString	Umwandlung des Symbolcodes in die Einsymbolzeile
DoubleToString	Umwandlung des numerishen Wertes in die Textzeile mit der angegebenen Genauigkeit
EnumToString	Umwandlung der Enumerationwert beliebigen Typs in ein String
NormalizeDouble	Runden der Zahl mit dem Fliesspunkt bis der angegebenen Genauigkeit
StringToDouble	Umwandlung der Zeile mit der Symboldarstellung der Zahl in die Zahl des Typs double
StringToInteger	Umwandlung der Zeile mit der Symboldarstellung der Zahl in die Zahl des Typs long
StringToTime	Umwandlung der Zeile mit Zeit und/oder Datum im Format "yyyy.mm.dd [hh:mi]", in eine Zahl des Typs datetime
TimeToString	Umwandlung des Wertes mi t der Zeit in Sekunden, die seit 01.01.1970 vergangen hat, in die Zeile mit Format "yyyy.mm.dd hh:mi"
IntegerToString	Umwandlung des ganzzahligen Wertes in die Zeile der vorgegebenen Laenge
ShortToString	Umwandlung des Symbolcodes (unicode) in die Einsymbolzeile
ShortArrayToString	Kopiert Teil des Feldes in die Zeile
StringToShortArray	Kopiert die Zeile in den gewaehlten Platz des Feldes des Typs ushort
CharArrayToString	Wandelt Symbolcode (ansi) in Einsymbolzeile um
StringToCharArray	Kopiert die Zeile, die aus Unicode in ANSI umgewandelt wurde, in den gewaehlten Platz des Typs uchar
CharArrayToStruct	Kopiert einen Array des Typs uchar in eine POD Struktur
StructToCharArray	Kopiert eine POD Struktur in ein Array des Typs uchar
ColorToARGB	Konvertiert den Typ color in den Typ uint, um Farbdarstellung ARGB zu erhalten.
ColorToString	Wandelt den Farbenwert in die Zeile der Art "R,G,B" um

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
StringToColor	Wandelt Zeile des Typs "R,G,B" oder Zeile mit Farbennamen in den Wert des Typs color um
StringFormat	Wandelt Zahl in die Zeile entsprechend dem vorgegebenen Format um

Sehen Sie auch

Kodeseite Verwenden

© 2000-2025, MetaQuotes Ltd.

CharToString

Umwandlung eines Zeichencodes in eine einzelne Zeichenkette.

```
string  CharToString(
   uchar  char_code      // numerischer Zeichencode
   );
```

Parameter

char_code

[in] ANSI-Code des Zeichens.

Rückgabewert

Eine Zeichenfolge, die ein ANSI-Zeichen enthält.

Beispiel:

```
string ExtStrArray[224];

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- in der Schleife des Zeichencodes 32 (Leerzeichen) bis 255(UCHAR_MAX),
//--- das mit Zeichencodes gefüllte Array, die entsprechend der aktuellen Codepage in
   for(int i=32; i<=UCHAR_MAX; i++)
      ExtStrArray[i-32]=CharToString((uchar)i);

//--- Ausdruck der Kopfzeile und der Tabelle der Zeichen in das Journal
   Print("Table of symbols:");
   ArrayPrint(ExtStrArray,_Digits," | ");
   /*
  Ergebnis:
   Table of symbols:
   [  0] " " | "!" | """ | "#" | "$" | "%" | "&" | "'" | "(" | ")" | "*" | "+" | ","
   [ 25] "9" | ":" | ";" | "<" | "=" | ">" | "?" | "@" | "A" | "B" | "C" | "D" | "E"
   [ 50] "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" | "[" | "\" | "]" | "^"
   [ 75] "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" | "s" | "t" | "u" | "v" | "w"
   [100] "„" | "…" | "†" | "‡" | "€" | "‰" | "Љ" | "‹" | "Њ" | "Ќ" | "Ћ" | "Џ" | "Ђ"
   [125] "ќ" | "ћ" | "џ" | " " | "ў" | "Ў" | "Ј" | "¤" | "Ґ" | "¦" | "§" | "Ё" | "©"
   [150] "Ґ" | "·" | "ё" | "№" | "є" | "»" | "ј" | "S" | "s" | "ї" | "A" | "Б" | "В"
   [175] "П" | "Р" | "С" | "Т" | "У" | "Ф" | "Х" | "Ц" | "Ч" | "Ш" | "Щ" | "Ъ" | "Ы"
   [200] "и" | "й" | "к" | "л" | "м" | "н" | "о" | "п" | "р" | "с" | "т" | "у" | "ф"
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

Siehe auch

StringToCharArray, ShortToString, StringGetCharacter

© 2000-2025, MetaQuotes Ltd.

CharArrayToString

Kopiert wandelt Teil des Arrays des Typs uchar in die Rückgabezeile um.

```
string  CharArrayToString(
   uchar    array[],             // Array
   int      start=0,             // Anfangsposition im Array
   int      count=-1             // Anzahl der Zeichen
   uint     codepage=CP_ACP      // Codeseite
   );
```

Parameter

array[]

[in] Array des Typs uchar.

start=0

[in] Position, aus der Kopieren anfängt. Default-Wert ist 0.

count=-1

[in] Zahl der Arrayelementen für Kopieren. Bestimmt die Länge der sich ergebenden Zeichenkette. Default-Wert ist -1, was bedeutet Kopieren bis zum Arrayende oder bis zur abschließenden 0.

codepage=CP_ACP

[in] Wert der Codeseite. Für die gebräuchlichsten Codepages gibt es entsprechende Konstanten.

Rückgabewert

Zeichenkette.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ein Array der Zeichencodes erstellen
   uchar array[]= { 84, 104, 105, 115,  32, 105, 115,  32,  97,  32,
                   116, 101, 115, 116,  32, 111, 102,  32, 116, 104,
                   101,  32,  67, 104,  97, 114,  65, 114, 114,  97,
                   121,  84, 111,  83, 116, 114, 105, 110, 103,  40,
                    41,  32, 102, 117, 110,  99, 116, 105, 111, 110
                  };
//--- ein in eine Zeichenkette umgewandeltes Array von Codes in das Journal ausdrucken
   Print(CharArrayToString(array));

//--- ein Array mit Codes der Zeichen und abschließenden Null erstellen
   uchar array_z[]= { 84, 104, 105, 115,  32, 105, 115,  32,  97,  32,
                     116, 101, 115, 116,   0,  32, 111, 102,  32, 116,
```

© 2000-2025, MetaQuotes Ltd.

```
                     104, 101,  32,  67, 104,  97, 114,  65, 114, 114,
                      97, 121,  84, 111,  83, 116, 114, 105, 110, 103,
                      40,  41,  32, 102, 117, 110,  99, 116, 105, 111, 110
                    };
//--- das zweite Array von Codes mit der abschließenden Null im Journal ausgeben
   PrintFormat("%s ...", CharArrayToString(array_z));

   /*
  Ergebnis:
   This is a test of the CharArrayToString() function
   This is a test ...
   */
  }
```

Siehe auch

StringToCharArray, ShortArrayToString, Verwenden von Codepages

© 2000-2025, MetaQuotes Ltd.

CharArrayToStruct

Kopiert einen Array des Typs uchar in eine POD Struktur.

```
bool  CharArrayToStruct(
   void&          struct_object,     // Struktur
   const uchar&   char_array[],      // Array
   uint           start_pos=0        // Anfangsposition im Array
   );
```

Parameter

struct_object

[in] Referenz zu einer POD Struktur irgendeines Typs (nur mit einfachen Datentypen).

char_array[]

[in] uchar Typ des Arrays.

start_pos=0

[in] Position im Array, ab der das Kopieren beginnt.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Beispiel:

```
#define DATA_TOTAL 4

MqlRates ExtDataRates[DATA_TOTAL];
uchar    ExtCharArray[];

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Kopieren der Daten der letzten vier Balken in das uchar-Array
   ResetLastError();
   for(int i=0; i<DATA_TOTAL; i++)
     {
      if(!CopyRatesToCharArray(i, ExtCharArray))
         return;
     }

//--- Kopieren aller verfügbaren Daten in das Array der MqlRates-Struktur in einer Sch
   for(int i=0; i<DATA_TOTAL; i++)
     {
      ResetLastError();
      if(!CharArrayToStruct(ExtDataRates[i], ExtCharArray, sizeof(MqlRates)*i))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("CharArrayToStruct() failed. Error code: ", GetLastError());
       return;
      }
     //--- nach erfolgreichem Kopieren der Daten aus dem uchar-Array in die MqlRates-
     //--- werden die in die Struktur kopierten Daten in das Journal ausgedruckt
     MqlRatesPrint(ExtDataRates[i]);
    }

 /*
Ergebnis:
 MqlRates data:
  Time = 2024.02.21 09:00;
  Open = 1.08116;
  High = 1.08141;
  Low = 1.08062;
  Close = 1.08124;
  Tick volume = 5344;
  Spread = 1;
  Real volume = 0

 MqlRates data:
  Time = 2024.02.21 08:00;
  Open = 1.08149;
  High = 1.08153;
  Low = 1.08073;
  Close = 1.08114;
  Tick volume = 3607;
  Spread = 2;
  Real volume = 0

 MqlRates data:
  Time = 2024.02.21 07:00;
  Open = 1.08143;
  High = 1.08165;
  Low = 1.08122;
  Close = 1.08150;
  Tick volume = 2083;
  Spread = 0;
  Real volume = 0

 MqlRates data:
  Time = 2024.02.21 06:00;
  Open = 1.08178;
  High = 1.08190;
  Low = 1.08132;
  Close = 1.08142;
  Tick volume = 1733;
  Spread = 0;
```

© 2000-2025, MetaQuotes Ltd.

```
    Real volume = 0
   */
  }
//+------------------------------------------------------------------+
//| Kopieren der Daten der Balken in das uchar-Array                 |
//| durch den angegebenen Index über die Struktur MqlRates           |
//+------------------------------------------------------------------+
bool CopyRatesToCharArray(const int index, uchar &data_array[])
  {
   MqlRates rates[1];
   uint     start=sizeof(MqlRates);

   ResetLastError();
   if(CopyRates(Symbol(), PERIOD_CURRENT, index, 1, rates)!=1)
     {
      Print("CopyRates() failed. Error code: ", GetLastError());
      return(false);
     }
   if(!StructToCharArray(rates[0], data_array, start*index))
     {
      Print("StructToCharArray() failed. Error code: ", GetLastError());
      return(false);
     }

   return(true);
  }
//+------------------------------------------------------------------+
//| Ausdruck der Felder der Struktur MqlRates in das Journal         |
//+------------------------------------------------------------------+
void MqlRatesPrint(MqlRates &rates)
  {
//--- Erstellen der auszugebenden Zeichenkette
   string text=StringFormat(" Time = %s;\n"
                            " Open = %.*f;\n"
                            " High = %.*f;\n"
                            " Low = %.*f;\n"
                            " Close = %.*f;\n"
                            " Tick volume = %I64u;\n"
                            " Spread = %d;\n"
                            " Real volume = %I64u",
                            TimeToString(rates.time),
                            _Digits, rates.open,
                            _Digits, rates.high,
                            _Digits, rates.low,
                            _Digits, rates.close,
                            rates.tick_volume,
                            rates.spread,
                            rates.real_volume);
//--- Druck der Kopfzeile und der Ausgabezeichenfolge in das Journal
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("MqlRates data:\n", text, "\n");
  }
```

Siehe auch

StringToCharArray, ShortArrayToString, StructToCharArray, Verwenden von Codepages, FileReadStruct, Unions (union), MathSwap

© 2000-2025, MetaQuotes Ltd.

StructToCharArray

Kopiert eine POD Struktur in ein Array des Typs uchar.

```
bool  StructToCharArray(
   const void&  struct_object,      // Struktur
   uchar&       char_array[],       // Array
   uint         start_pos=0         // Anfangsposition im Array
   );
```

Parameter

struct_object

[in] Referenz auf eine POD Struktur irgendeines Typs (nur mit einfachen Datentypen).

char_array[]

[in] uchar Typ des Arrays.

start_pos=0

[in] Position im Array, ab der die kopierten Daten eingetragen werden.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Hinweis

Beim Kopieren werden dynamische Arrays automatisch erweitert (ArrayResize), sollte die Größe nicht ausreichen. Wenn der Array nicht bis zu der benötigten Größe erweitert werden kann, gibt die Funktion einen Fehler zurück.

Beispiel:

```
uchar    ExtCharArray[];
MqlRates ExtRates[1];

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Kopieren der Daten des aktuellen Balkens in die Struktur MqlRates
   if(CopyRates(Symbol(), PERIOD_CURRENT, 0, 1, ExtRates)!=1)
     {
      Print("CopyRates() failed. Error code: ", GetLastError());
      return;
     }

//--- die in der MqlRates-Struktur empfangenen Daten in das Journal drucken
   Print("Data in the structure immediately after receiving it:");
   MqlRatesPrint(ExtRates[0]);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Kopieren der Struktur in das Array vom Typ uchar
   ResetLastError();
   if(!StructToCharArray(ExtRates[0], ExtCharArray))
     {
      Print("StructToCharArray() failed. Error code: ", GetLastError());
      return;
     }

//--- Löschen der Struktur
   ZeroMemory(ExtRates[0]);
//--- die Daten in der Struktur nach dem Löschen ausgeben
   Print("\nData in the structure after ZeroMemory():");
   MqlRatesPrint(ExtRates[0]);

//--- Kopieren der Daten aus dem uchar-Array in die MqlRates-Struktur
   if(!CharArrayToStruct(ExtRates[0], ExtCharArray))
     {
      Print("CharArrayToStruct() failed. Error code: ", GetLastError());
      return;
     }
//--- die Daten in der MqlRates-Struktur ausgeben, nachdem sie aus dem uchar-Array kop
   Print("\nData in the MqlRates structure after copying it from a uchar array:");
   MqlRatesPrint(ExtRates[0]);

   /*
  Ergebnis:

   Data in the structure immediately after receiving it:
   MqlRates data:
    Time = 2024.02.21 07:00;
    Open = 1.08143;
    High = 1.08158;
    Low = 1.08122;
    Close = 1.08137;
    Tick volume = 1341;
    Spread = 4;
    Real volume = 0

   Data in the structure after ZeroMemory():
   MqlRates data:
    Time = 0;
    Open = 0.00000;
    High = 0.00000;
    Low = 0.00000;
    Close = 0.00000;
    Tick volume = 0;
    Spread = 0;
    Real volume = 0
```

© 2000-2025, MetaQuotes Ltd.

```
   Data in the MqlRates structure after copying it from a uchar array:
   MqlRates data:
    Time = 2024.02.21 07:00;
    Open = 1.08143;
    High = 1.08158;
    Low = 1.08122;
    Close = 1.08137;
    Tick volume = 1341;
    Spread = 4;
    Real volume = 0
   */
  }
//+------------------------------------------------------------------+
//| Ausdruck der Felder der Struktur MqlRates in das Journal         |
//+------------------------------------------------------------------+
void MqlRatesPrint(MqlRates &rates)
  {
//--- Erstellen der auszugebenden Zeichenkette
   string text=StringFormat(" Time = %s;\n"
                            " Open = %.*f;\n"
                            " High = %.*f;\n"
                            " Low = %.*f;\n"
                            " Close = %.*f;\n"
                            " Tick volume = %I64u;\n"
                            " Spread = %d;\n"
                            " Real volume = %I64u",
                            TimeToString(rates.time),
                            _Digits, rates.open,
                            _Digits, rates.high,
                            _Digits, rates.low,
                            _Digits, rates.close,
                            rates.tick_volume,
                            rates.spread,
                            rates.real_volume);
//--- Druck der Kopfzeile und der Ausgabezeichenfolge in das Journal
   Print("MqlRates data:\n", text);
  }
```

Siehe auch

StringToCharArray, ShortArrayToString, CharArrayToStruct, Verwenden von Codepages, FileWriteStruct, Unions (union), MathSwap

© 2000-2025, MetaQuotes Ltd.

ColorToARGB

Konvertiert den Typ color in den Typ uint um Farbdarstellung ARGB zu erhalten. Format ARGB wird für Erstellung einer graphischen Ressource, Textausgabe und in der CCanvas-Klasse der Standardbibliothek.

```
uint  ColorToARGB(
   color  clr,          // Die Farbe im Format color, die Sie konvertieren wollen
   uchar  alpha=255     // Alpha-Kanal, der Farbtransparenz verwaltet
   );
```

Optionen

clr

[in] Farbwert in der Variable vom Typ color.

alpha

[in] Wert des Alpha-Kanals, um die Farbe im Format ARGB zu erhalten. Es wird durch den Wert von 0 (Farbe des anlegenden Pixels ändert nicht die Farbe des zugrunde liegenden Pixels) bis 255 (Farbe des anlegenden Pixels voll bedeckt die Farbe des zugrunde liegenden Pixels) angegeben. Farbtransparenz in Prozent wird als (1-alpha/255)*100% berechnet, d.h. je weniger der Alpha-Wert ist, desto mehr transparent die Farbe ist.

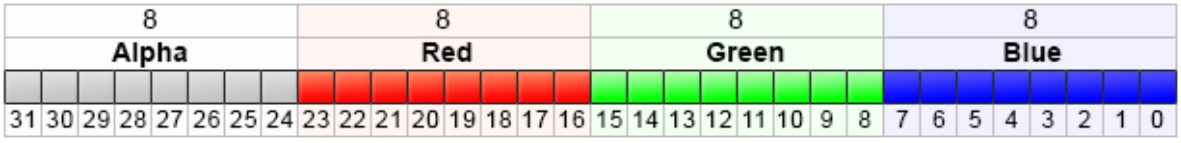
Rückgabewert

Farbdarstellung im Format ARGB, wo in vier Bytes vom Typ uint Werte Alfa, Red, Green, Blue (Alpha, rot, grün, blau) vorgeschrieben sind.

Hinweis

Das Grundformat für Beschreibung der Pixelfarbe auf dem Bildschirm in Computergrafik ist RGB, wo die Farbkomponenten nach den Namen der Primärfarben Rot (Red), Grün (Green) und Blau (Blue) gesetzt werden. Jede Komponente wird durch ein einzelnes Byte, das die Sättigung der Farbe im Bereich von 0 bis 255 (0x00 bis 0xFF hexadezimal) spezifiziert, beschrieben. Da Weiß enthält alle Farben, wird sie als 0xFFFFFF beschrieben, das heißt, jede der drei Komponenten ist im maximalen Wert 0xFF repräsentiert.

Aber in einer Reihe von Aufgaben erforderlich ist, Farbtransparenz anzugeben, um zu beschreiben, wie das Bild aussieht, wenn die Farbe mit einem gewissen Transparenz auf das Bild gelegen ist. In solchen Fällen wird das Konzept der alpha-Kanal (Alpha) benutzt, der als zusätzliche Komponente zum Format RGB eingeführt wird. Schema vom ARGB Format ist unten gezeigt.



ARGB-Werte werden typischerweise in hexadezimal Format angegeben, bei dem jedes Paar von Ziffern den Wert der Kanälen Alpha, Red, Green und Blue repräsentiert. Zum Beispiel ARGB-Farbe 80FFFF00 bedeutet gelb mit Opazität von 50,2%. Am Anfang gibt es 0x80, was 50,2% Alpha-Wert angibt, da es 50,2% vom Wert 0xFF ist; weitere bedeutet erste Paar FF den maximalen Wert der roten Komponente; das nächste Paar FF gibt der gleiche Effekt der grünen Komponente an; und das letzte Paar 00 ist der Mindestwert von Blauanteil (kein blau). Die Zugabe von grün und rot gibt gelb.

© 2000-2025, MetaQuotes Ltd.

Wenn der Alpha-Kanal nicht verwendet wird, kann die Aufnahme zu 6 Ziffern RRGGBB reduziert werden, weshalb die Werte des Alpha-Kanals in den oberen Bits der Integer-Typ uint gespeichert werden.

Je nach dem Kontext können die Hexadezimalzahlen mit dem Präfix '0x' oder '#', beispielsweise 80FFFF00 oder 0x80FFFF00 oder #80FFFF00 aufgezeichnet werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Transparenz angeben
   uchar alpha=0x55;  // Wert x55 bedeutet 55/255=21.6 % von Transparenz
   //--- Konvertierung in ARGB für clrBlue
   PrintFormat("0x%.8X - clrBlue",clrBlue);
   PrintFormat("0x%.8X - clrBlue ARGB with alpha=0x55 (transparency 21.6%%)",ColorToAR
   //--- Konvertierung in ARGB für clrGreen
   PrintFormat("0x%.8X - clrGreen",clrGreen);
   PrintFormat("0x%.8X - clrGreen ARGB with alpha=0x55 (transparency 21.6%%)",ColorToA
   //--- Konvertierung in ARGB für clrRed
   PrintFormat("0x%.8X - clrRed",clrRed);
   PrintFormat("0x%.8X - clrRed ARGB with alpha=0x55 (transparency 21.6%%)",ColorToARG
  }
```

Sehen Sie auch

Ressourcen, ResourceCreate(), TextOut(), Typ color, Typen char, short, int und long

© 2000-2025, MetaQuotes Ltd.

ColorToString

Wandelt Farbenwert in die Zeile der Art "R,G,B" um.

```
string  ColorToString(
   color   color_value,      // Farbenwert
   bool    color_name        // Farbennamen zeigen oder nicht
   );
```

Parameter

color_value

[in] Farbenwert in der Variable des Typs color.

color_name

[in] Zeichen der Notwendigkeit, Farbennamen rückzugeben, falls Farbenwert mit dem Wert einer der vorbestimmten Farbenkonstanten zusammenfaellt.

Rückgabewert

Zeilendarstellung der Farbe als "R,G,B", wo R, G und B Dezimalkonstanten sind, die in dieZ eile umgewandelt sind und den wert im Bereich von 0 bis 255 haben. Wenn der Parameter color_name=true eingestellt wird, wird es versucht, Farbenwert zum Farbennamen zu reduzieren.

Beispiel:

```
   string clr=ColorToString(C'0,255,0'); // gruene Farbe
   Print(clr);

   clr=ColorToString(C'0,255,0',true);   // Farbenkonstante erhalten
   Print(clr);
```

Sehen Sie auch

StringToColor, ColorToARGB

© 2000-2025, MetaQuotes Ltd.

DoubleToString

Umwandeln des numerischen Wertes in die Textzeile.

```
string  DoubleToString(
   double  value,      // Zahl
   int     digits=8    // Anzahl der Dezimalzeichen
   );
```

Parameter

value

[in] Wert mit dem Fliesspunkt.

digits

[in] Format der Genauigkeit. Wenn sich der Wert digits im Bereich von 0 bis 16 befindet,bekommt man die Zeilendarstellung der Zahl mit der angegebenen Zahl der Dezimalzeichen. Wenn sich der Wert digits im Bereich von -1 bis -16 befindet, bekommt man Zeilendarstellung der Zahl im wissenschaftlichen Format mit der angegebenen Anzahl der Dezimalzeichen nach dem Komma. In allen anderen Faellen wird der Zeilenwert der Zahl 8 Dezimalzeichen nach dem Komma enthalten.

Rückgabewert

Zeile mit der Symboldarstellung der Zahl mit der angegebenen Genauigkeit.

Beispiel:

```
   Print("DoubleToString(120.0 + M_PI) : ",DoubleToString(120.0+M_PI));
   Print("DoubleToString(120.0 + M_PI,16) : ",DoubleToString(120.0+M_PI,16));
   Print("DoubleToString(120.0 + M_PI,-16) : ",DoubleToString(120.0+M_PI,-16));
   Print("DoubleToString(120.0 + M_PI,-1) : ",DoubleToString(120.0+M_PI,-1));
   Print("DoubleToString(120.0 + M_PI,-20) : ",DoubleToString(120.0+M_PI,-20));
```

Sehen Sie auch

NormalizeDouble, StringToDouble

© 2000-2025, MetaQuotes Ltd.

EnumToString

Umwandlung der Enumerationwert beliebigen Typs in die Textdarstellung.

```
string  EnumToString(
   any_enum  value      // Enumerationwert beliebigen Typs
   );
```

Parameter

value

[in] Enumerationwert beliebigen Typs.

Rückgabewert

String mit der Textdarstellung der Wert. Um Information über Fehler zu bekommen, rufen Sie die Funktion GetLastError().

Hinweis

Die Funktion kann die folgende folgende Fehlerwerte in die _LastError Funktion einstellen:

- ERR_INTERNAL_ERROR - Fehler der Ausführungsumgebung
- ERR_NOT_ENOUGH_MEMORY - nicht genügend Speicher für die Operation
- ERR_INVALID_PARAMETER - kann nicht den Namen des Enumerationswert zulassen

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
enum interval  // enum benannten Konstanten
  {
   month=1,     // Abstand von einem Monat
   two_months,  // Zwei Monate
   quarter,     // drei Monaten - Quartal
   halfyear=6,  // Halbjahr
   year=12,     // Jahr - 12 Monate
  };
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- das Zeitintervall gleich zu einem Monat
   interval period=month;
   Print(EnumToString(period)+"="+IntegerToString(period));

//--- das Zeitintervall gleich zu einem Quartal (Drei Monate))
   period=quarter;
   Print(EnumToString(period)+"="+IntegerToString(period));

//--- das Zeitintervall gleich zu einem Jahr (12 Monate))
   period=year;
   Print(EnumToString(period)+"="+IntegerToString(period));

//--- prüfen, wie die Auftragsart wird gezeigt
   ENUM_ORDER_TYPE type=ORDER_TYPE_BUY;
   Print(EnumToString(type)+"="+IntegerToString(type));

//--- prüfen, wie eine falsche Wert wird gezeigt
   type=WRONG_VALUE;
   Print(EnumToString(type)+"="+IntegerToString(type));

// Ergebnis:
// month=1
// quarter=3
// year=12
// ORDER_TYPE_BUY=0
// ENUM_ORDER_TYPE::-1=-1
  }
```

Sehen Sie auch

Enumerationen, Input Variablen

© 2000-2025, MetaQuotes Ltd.

IntegerToString

Wandelt den Wert des ganzzahligen Types in die Zeichenkette der angegebenen Länge um und gibt die erhaltene Zeichenkette zurück.

```
string  IntegerToString(
   long     number,              // Zahl
   int      str_len=0,           // Länge der ergebenden Zeichenfolge
   ushort   fill_symbol=' '      // Füllzeichen
   );
```

Parameter

number

[in] Die zu konvertierende Zahl.

str_len=0

[in] Länge der Zeichenfolge. Wenn die Länge der resultierenden Zeichenkette größer als die angegebene Länge ist, wird die Zeichenkette nicht abgeschnitten. Ist die Länge der resultierenden Zeichenkette kleiner als die angegebene Länge, wird vor die resultierende Zeichenkette, links, Füllzeichen angehängt.

fill_symbol=' '

[in] Füll-Zeichen. Default-Wert - Leerzeichen.

Rückgabewert

Zeichenkette.

Beispiel:

```
#define DATA_TOTAL 1001

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Darstellung der Zeichenketten mit dem Indexvielfachen von 100 in der Schleife vo
//--- die Zeichenkette zeigt den Indexwert im Vier-Zeichen-Format an
//--- wenn die Zeichenkette weniger als 4 Zeichen lang ist, werden dem Wert des Schlei
//--- in der Zeichenkette führende Nullen vorangestellt
   for(int i=0; i<DATA_TOTAL; i++)
     {
      if(i%100==0)
         Print("Converted index value: ",IntegerToString(i,4,'0'));
     }
   /*
  Ergebnis:
   Converted index value: 0000
```

© 2000-2025, MetaQuotes Ltd.

```
   Converted index value: 0100
   Converted index value: 0200
   Converted index value: 0300
   Converted index value: 0400
   Converted index value: 0500
   Converted index value: 0600
   Converted index value: 0700
   Converted index value: 0800
   Converted index value: 0900
   Converted index value: 1000
   */
  }
```

Siehe auch

StringToInteger

© 2000-2025, MetaQuotes Ltd.

ShortToString

Wandelt einen Zeichencode (Unicode) in eine einstellige Zeichenkette um und gibt die resultierende Zeichenkette zurück.

```
string  ShortToString(
   ushort  symbol_code      // Zeichencode
   );
```

Parameter

symbol_code

[in] der Zeichencode. Anstelle des Zeichencodes können Sie eine Zeichenkette verwenden, die das Zeichen enthält, oder eine Zeichenkette mit einem Zwei-Byte-Hexadezimalcode, der dem Zeichen aus der Unicode-Tabelle entspricht.

Rückgabewert

Zeichenkette.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ausdruck von 18 Zeichen in einer Schleife, beginnend mit dem Zeichen mit der Uni
   for(int i=0; i<18; i++)
     {
      ushort code=0x23E9+(ushort)i;
      PrintFormat("Unicode number U+%hX: %s",code,ShortToString(code));
     }
   /*
  Ergebnis:
   Unicode number U+23E9: ⏩
   Unicode number U+23EA: ⏪
   Unicode number U+23EB: ⏫
   Unicode number U+23EC: ⏬
   Unicode number U+23ED: ⏭
   Unicode number U+23EE: ⏮
   Unicode number U+23EF: ⏯
   Unicode number U+23F0: ⏰
   Unicode number U+23F1: ⏱
   Unicode number U+23F2: ⏲
   Unicode number U+23F3: ⏳
   Unicode number U+23F4: □
   Unicode number U+23F5: □
   Unicode number U+23F6: □
```

© 2000-2025, MetaQuotes Ltd.

```
   Unicode number U+23F7: ⏷
   Unicode number U+23F8: ⏸
   Unicode number U+23F9: ⏹
   Unicode number U+23FA: ⏺
   */
  }
```

Siehe auch

StringToCharArray, CharToString, StringGetCharacter

© 2000-2025, MetaQuotes Ltd.

ShortArrayToString

Kopiert einen Teil des Arrays in die zurückgegebene Zeichenkette.

```
string  ShortArrayToString(
   ushort  array[],      // Array
   int     start=0,      // Anfangsposition im Array
   int     count=-1      // Anzahl der Zeichen
   );
```

Parameter

array[]

[in] Array des Typs ushort (Analog des Typs wchar_t).

start=0

[in] Position, bei der das Kopieren beginnt. Default-Wet ist 0.

count=-1

[in] Anzahl der Arrayelemente für das Kopieren. Bestimmt die Länge der sich ergebenden Zeichenkette. Default-Wert ist -1, was das Kopieren bis zum Arrayende oder bis zur abschließenden 0 bedeutet.

Rückgabewert

Zeichenkette.

Beispiel:

```
#define ROW_SIZE 16

ushort ExtShortArray[]={0x2190,0x2191,0x2192,0x2193,0x2194,0x2195,0x2196,0x2197,0x2198
                        0x21A0,0x21A1,0x21A2,0x21A3,0x21A4,0x21A5,0x21A6,0x21A7,0x21A8
                        0x21B0,0x21B1,0x21B2,0x21B3,0x21B4,0x21B5,0x21B6,0x21B7,0x21B8
                        0x21C0,0x21C1,0x21C2,0x21C3,0x21C4,0x21C5,0x21C6,0x21C7,0x21C8
                        0x21D0,0x21D1,0x21D2,0x21D3,0x21D4,0x21D5,0x21D6,0x21D7,0x21D8
                        0x21E0,0x21E1,0x21E2,0x21E3,0x21E4,0x21E5,0x21E6,0x21E7,0x21E8
                        0x21F0,0x21F1,0x21F2,0x21F3,0x21F4,0x21F5,0x21F6,0x21F7,0x21F8

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Berechnung der Anzahl der Zeichenketten in der Symboltabelle von 7x16
   int total=(int)ExtShortArray.Size()/ROW_SIZE;
//--- in einer Schleife durch die Tabellenzeilen, werden 7 Gruppen von Pfeilen, 16 Ele
   for(int i=0; i<total; i++)
     {
      int row=i*ROW_SIZE;
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("Arrow set %u: %s",i+1,ShortArrayToString(ExtShortArray,row,ROW_SIZE
     }
  /*
 Ergebnis:
  Arrow set 1: ←↑→↓↔↕□□□□□□□□□□
  Arrow set 2: □□□□□□□□↨□□□□□□□
  Arrow set 3: □□□□□□□□□□□□□□□□
  Arrow set 4: □□□□□□□□□□□□□□□□
  Arrow set 5: □□⇒□⇔□□□□□□□□□□
  Arrow set 6: □□□□□□□□□□□□□□□□
  Arrow set 7: □□□□□□□□□□□□□□□□
  */
 }
```

Siehe auch

StringToShortArray, CharArrayToString, Verwenden von Codepages

© 2000-2025, MetaQuotes Ltd.

TimeToString

Umwandlung des Wertes, der Zeit in Sekunden enthält, die seit 01.01.1970 vergangen ist, in die Zeichenkette mit dem Format "yyyy.mm.dd hh:mi".

```
string  TimeToString(
   datetime  value,                          // Zahl
   int       mode=TIME_DATE|TIME_MINUTES     // Ausgabeformat
   );
```

Parameter

value

[in] Zeit in Sekunden seit 00:00 1. Januar 1970.

mode=TIME_DATE|TIME_MINUTES

[in] Optionale Formate der Datenausgabe. Kann aus einem oder einer Kombination von Flags bestehen:
TIME_DATE führt zum Format " yyyy.mm.dd " ,
TIME_MINUTES führt zum Format " hh:mi ",
TIME_SECONDS führt zum Format " hh:mi:ss ".

Rückgabewert

Zeichenkette.

Beispiel:

```
datetime ExtBarTimeOpen;

//+------------------------------------------------------------------+
//| Initialisierungsfunktion eine nutzerdefinierten Indikators       |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- den Timer auf eine Sekunde einstellen
   EventSetTimer(1);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Nutzerdefinierte Indikator-Deinitialisierungsfunktion            |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   Comment("");
  }
//+------------------------------------------------------------------+
//| Nutzerdefinierte Indikator-Iterationsfunktion                    |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Ermittlung der aktuellen Öffnungszeit des Balkens
   ExtBarTimeOpen=time[rates_total-1];
//--- Rückgabe des Wertes von prev_calculated für den nächsten Aufruf
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Timer Funktion                                                   |
//+------------------------------------------------------------------+
void OnTimer()
  {
//--- Festlegen der Öffnungszeit des vorherigen Balkens
   static datetime bar_open_time=ExtBarTimeOpen;
//--- Zählen der Anzahl der Sekunden, die seit der Öffnung des Balkens vergangen sind
   static int seconds=int(TimeCurrent()-ExtBarTimeOpen);
//--- wenn die vorherige Öffnungszeit nicht gleich der aktuellen ist, handelt es sich
//--- den neuen Eröffnungszeitpunkt dem vorherigen zuweisen, und die Sekunden auf Null
   if(bar_open_time!=ExtBarTimeOpen)
     {
      bar_open_time=ExtBarTimeOpen;
      seconds=0;
     }
//--- Erhöhen und Anpassen der Anzahl der Sekunden, die seit der Eröffnung des Balkens
   seconds++;
   if(seconds>PeriodSeconds(PERIOD_CURRENT))
      seconds=0;
//--- Darstellung der Öffnungszeit des Balkens als yyyy.mm.dd hh:mi
   string bar_time_open=TimeToString(ExtBarTimeOpen);
//--- Darstellung der aktuellen Zeit als yyyy.mm.dd hh:mi:ss
   string time_current=TimeToString(TimeCurrent(),TIME_DATE|TIME_MINUTES|TIME_SECONDS);
//--- Anzahl der verbleibenden Sekunden, bis ein neuer Balken geöffnet wird
   int    sec_left=PeriodSeconds(PERIOD_CURRENT)-seconds;
//--- verbleibende Zeit bis zum Schließen des aktuellen Balkens in hh:mi:ss
   string time_left=TimeToString(sec_left,TIME_MINUTES|TIME_SECONDS);
//--- Erstellen der auszugebenden Zeichenkette
   string txt=StringFormat("Opening time of the current bar: %s\n"+
                           "Time Current: %s\n"+
                           "Seconds have passed since the bar opened: %d\n"+
                           "Approximately seconds left before bar closes: %d\n"+
```

© 2000-2025, MetaQuotes Ltd.

```
                              "Time remaining until bar closes: %s",bar_time_open,time_cu
//--- Anzeige der Eröffnungszeit des Balkens und der aktuellen Zeit,
//--- die Anzahl der Sekunden, die seit der Eröffnung des aktuellen Balkens vergangen
//--- die verbleibende Zeit bis zur Schließung des aktuellen Balkens im Kommentar
   Comment(txt);
   /*
   result on M1:
   Opening time of the current bar: 2024.02.22 18:06
   Time Current: 2024.02.22 18:06:24
   Seconds have passed since the bar opened: 25
   Approximately seconds left before bar closes: 35
   Time remaining until bar closes: 00:00:35

   result on M5:
   Opening time of the current bar: 2024.02.22 18:05
   Time Current: 2024.02.22 18:07:28
   Seconds have passed since the bar opened: 149
   Approximately seconds left before bar closes: 151
   Time remaining until bar closes: 00:02:31

   result on H1:
   Opening time of the current bar: 2024.02.22 18:00
   Time Current: 2024.02.22 18:08:13
   Seconds have passed since the bar opened: 494
   Approximately seconds left before bar closes: 3106
   Time remaining until bar closes: 00:51:46

   result on D1:
   Opening time of the current bar: 2024.02.22 00:00
   Time Current: 2024.02.22 18:11:01
   Seconds have passed since the bar opened: 65462
   Approximately seconds left before bar closes: 20938
   Time remaining until bar closes: 05:48:58
   */
  }
```

Siehe auch

StringToTime, TimeToStruct

© 2000-2025, MetaQuotes Ltd.

NormalizeDouble

Runden der Zahl mit dem Fliesspunkt bis zur angegebenen Genauigkeit.

```
double  NormalizeDouble(
   double   value,      // normalisierte Zahl
   int      digits      // Anzahl der Dezimalzeichen
   );
```

Parameter

value

[in] Wert mit dem Fliesspunkt.

digits

[in] Format der Genauigkeit, Anzahl der Dezimalzeichen (0-8).

Rückgabewert

Wert des Typs double mit der vorgegebenen Genauigkeit.

Hinweis

Berechnete Werte von StopLoss, TakeProfit, und Werte des Eüöffnungspreises der Warteordern, müssen normalisiert werden mit der Genauigkeit, deren Wert durch die Funktion Digits() erhalten werden kann.

Bitte beachten Sie, dass bei der Ausgabe einer normalisierten Zahl ins Journal mit Funktion Print(), kann sie eine größere Anzahl von Nachkommastellen als Sie erwarten, enthalten. Zum Beispiel für:

```
   double a=76.671;              // Normalisierte Zahl mit 3 Nachkommastellen
   Print("Print(76.671)=",a);    // Wir geben es, wie es ist, aus
   Print("DoubleToString(a,8)=",DoubleToString(a,8)); // Wir geben es mit einer vorgeg
```

erhalten sie das folgende im Terminal:

```
   DoubleToString(a,8)=76.67100000
   Print(76.671)=76.67100000000001
```

Beispiel:

```
   double pi=M_PI;
   Print("pi = ",DoubleToString(pi,16));

   double pi_3=NormalizeDouble(M_PI,3);
   Print("NormalizeDouble(pi,3) = ",DoubleToString(pi_3,16))
   ;
   double pi_8=NormalizeDouble(M_PI,8);
   Print("NormalizeDouble(pi,8) = ",DoubleToString(pi_8,16));

   double pi_0=NormalizeDouble(M_PI,0);
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("NormalizeDouble(pi,0) = ",DoubleToString(pi_0,16));
/*
   Ergebnis:
   pi= 3.1415926535897931
   NormalizeDouble(pi,3)= 3.1419999999999999
   NormalizeDouble(pi,8)= 3.1415926499999998
   NormalizeDouble(pi,0)= 3.0000000000000000
*/
```

Sehen Sie auch

DoubleToString, Realtypen (double, float), Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

StringToCharArray

Kopiert eine von Unicode nach Ansi konvertierte Zeichenkette Zeichen für Zeichen an die angegebene Stelle im Array vom Typ uchar. Funktion gibt die Anzahl der kopierten Elemente zurück.

```
int  StringToCharArray(
   string  text_string,          // Quelle der Zeichenketten
   uchar&  array[],              // Array
   int     start=0,              // Anfangsposition im Array
   int     count=-1              // Anzahl der Zeichen
   uint    codepage=CP_ACP       // Codeseite
   );
```

Parameter

text_string

[in] Zeichenkette für Kopieren.

array[]

[out] Array des Typs uchar.

start=0

[in] Position, bei der das Kopieren beginnt. Default-Wert ist 0.

count=-1

[in] Anzahl der Arrayelemente für das Kopieren. Bestimmt die Länge der sich ergebenden Zeichenkette. Default-Wert ist -1, was bedeutet, dass bis zum Arrayende oder bis zur abschließenden 0 kopiert wurde. Die abschließende 0 wird auch in Array-Rezipient kopiert werden, dabei wird gegebenenfalls die Größe des dynamischen Arrays bis zur Zeichenkettengröße vergrößert werden. Wenn die Größe des dynamischen Arrays höher als die Zeichenkettenlänge ist, wird die Arraygröße nicht reduziert werden.

codepage=CP_ACP

[in] Wert der Codeseite. Für die gebräuchlichsten Codepages gibt es entsprechende Konstanten.

Rückgabewert

Anzahl der kopierten Elemente.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- zu konvertierende Zeichenkette
   string text = "This is a test of the StringToCharArray() function";

//--- Konvertierung der Eingabezeichenfolge in ein uchar-Array gemäß der eingestellter
   uchar char_array[];
```

© 2000-2025, MetaQuotes Ltd.

```
   int   copied=StringToCharArray(text, char_array);
   PrintFormat("String length: %u\nNumber of characters copied (with terminal 0): %d\r
               StringLen(text), copied, text);
//--- Druck des resultierenden Arrays in das Journal
   ArrayPrint(char_array, 0, " | ");
   /*
  Ergebnis:
   String length: 50
   Number of characters copied (with terminal 0): 51
   Array of characters for the string 'This is a test of the StringToCharArray() funct
   [ 0]  84 | 104 | 105 | 115 |  32 | 105 | 115 |  32 |  97 |  32 | 116 | 101 | 115 |
   [17]  32 | 116 | 104 | 101 |  32 |  83 | 116 | 114 | 105 | 110 | 103 |  84 | 111 |
   [34]  65 | 114 | 114 |  97 | 121 |  40 |  41 |  32 | 102 | 117 | 110 |  99 | 116 |
   */
  }
```

Siehe auch

CharArrayToString, StringToShortArray, Verwenden von Codepages

© 2000-2025, MetaQuotes Ltd.

StringToColor

Wandelt die Zeile des Typs "R,G,B" oder die Zeile mit dem Farbennamen, in den Wert des Typs color um.

```
color  StringToColor(
   string  color_string      // Zeilendarstellung der Farbe
   );
```

Parameter

color_string

[in] Zeilendarstellung der Farbe des Types "R,G,B"oder Name einer der vorbestimmten Web-Farben.

Rückgabewert

Farbenwert.

Beispiel:

```
   color str_color=StringToColor("0,127,0");
   Print(str_color);
   Print((string)str_color);
//--- verändern wir die Farbe ein bisschen
   str_color=StringToColor("0,128,0");
   Print(str_color);
   Print((string)str_color);
```

Sehen Sie auch

ColorToString, ColorToARGB

© 2000-2025, MetaQuotes Ltd.

StringToDouble

Konvertiert eine Zeichenkette, die die Zeichendarstellung einer Zahl enthält, in eine Zahl vom Typ long (integer).

```
double  StringToDouble(
   string  value      // Zeichenkette
   );
```

Parameter

value

[in] Zeichenkette mit der Zeichendarstellung der Zahl.

Rückgabewert

Wert vom Typ double.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- zu konvertierende Zeichenkette
   string str = "12345.54321";
//--- Konvertierung der Eingabezeichenkette einer reellen Zahl in eine Variable vom Ty
   double converted=StringToDouble(str);
//--- die resultierende Zahl auf 8 Dezimalstellen genau im Journal anzeigen
   PrintFormat("The string '%s' is converted to the real number %.8f", str, converted)
   /*
  Ergebnis:
   The string '12345.54321' is converted to the real number 12345.54321000
   */
  }
```

Siehe auch

NormalizeDouble, Realtypen (double, float), Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

StringToInteger

Umwandlung der Zeichenkette, die Zeichendarstellung der Zahl enthält, darunter des Typs long (ganzzahlig).

```
long  StringToInteger(
   string  value      // Zeichenkette
   );
```

Parameter

value

[in]die Zahl enthaltende Zeichenkette

Rückgabewert

Wert vom Typ long.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- zu konvertierende Zeichenkette
   string str = "12345.54321";
//--- Konvertierung der Eingabezeichenkette einer Zahl in eine Variable vom Typ long
   long converted=StringToInteger(str);
//--- Anzeige der erhaltenen Zahl im Journal, wobei die Nachkommastelle abgeschnitten
   PrintFormat("The string '%s' is converted to the integer number %I64d", str, conver
   /*
  Ergebnis:
   The string '12345.54321' is converted to the integer number 12345
   */
  }
```

Siehe auch

IntegerToString, Realtypen (double, float), Typenreduzierung

© 2000-2025, MetaQuotes Ltd.

StringToShortArray

Kopiert die Zeichenkette zeichenweise an den angegebenen Platz des Arrays vom Typ ushort. Funktion gibt die Anzahl der kopierten Elemente zurück.

```
int  StringToShortArray(
   string  text_string,      // Quelle der Zeichenkette
   ushort& array[],          // Array
   int     start=0,          // Anfangsposition im Array
   int     count=-1          // Anzahl der Zeichen
   );
```

Parameter

text_string

[in] Zeichenkette für Kopieren.

array[]

[out] Array des Typs ushort (Analog des Typs wchar_t).

start=0

[in] Position, bei der das Kopieren beginnt. Default-Wert ist 0.

count=-1

[in] Anzahl der Arrayelemente für das Kopieren. Bestimmt die Länge der sich ergebenden Zeichenkette. Default-Wet ist -1, was bedeutet Kopieren bis zum Arrayende oder bis zur terminalen 0. Die abschließende 0 wird auch in Array-Rezipient kopiert werden, dabei kann die Größe des dynamischen Arrays bis zur Länge der Zeichenkette steigen. Wenn die Größe des dynamischen Arrays mehr als die Zeichenkettenlänge ist, wird die Arraygröße nicht reduziert werden.

Rückgabewert

Anzahl der kopierten Elemente.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Zeichenkette der zu konvertierenden Zeichen
   string text = "Chars: ♡♫☎✈☩☏⌬✉☆☁♕♠®✧❄♣    ";
   ushort short_array[];
//--- Suche nach der Position des Zeichens ":" in der Eingabezeichenkette und Extrakt
   int    pos=StringFind(text, ":");
   string str=(pos<0 ? text : StringSubstr(text, pos+1));
//--- Entfernen von Leerzeichen, Zeilenumbrüche und Tabulatorzeichen links und rechts
   StringTrimLeft(str);
```

© 2000-2025, MetaQuotes Ltd.

```
   StringTrimRight(str);
//--- Kopieren der resultierenden Zeichenkette in das ushort-Array und Ausdrucken das
   int copied=StringToShortArray(str, short_array);
   PrintFormat("String of characters length: %u\n"
               "Number of characters copied (with terminal 0): %d\n"
               "Array of chars for the string '%s':",
               StringLen(str), copied, str);
   ArrayPrint(short_array, 0, " | ");
   /*
  Ergebnis:
   String of characters length: 16
   Number of characters copied (with terminal 0): 17
   Array of chars for the string '♡♫☎☒✣☏❁✉☆⛀♕♠®✧❄♣':
   10084 |  9835 |  9742 |  9992 | 10019 |  9743 |  9728 |  9993 |  9734 |  9729 |  98
   */
  }
```

Siehe auch

ShortArrayToString, StringToCharArray, Verwenden von Codepages

© 2000-2025, MetaQuotes Ltd.

StringToTime

Überträgt eine Zeichenkette, die eine Zeitangabe oder/und ein Datum im Format "yyyy.mm.dd [hh:mi]" enthält in eine Zahl vom Typ datetime.

```
datetime  StringToTime(
   const string  time_string      // Zeichenkette eines Datums
   );
```

Parameter

time_string

[in] Zeichenkette in einer der folgenden Formate:

- "yyyy.mm.dd [hh:mi]"
- "yyyy.mm.dd [hh:mi:ss]"
- "yyyymmdd [hh:mi:ss]"
- "yyyymmdd [hhmiss]"
- "yyyy/mm/dd [hh:mi:ss]"
- "yyyy-mm-dd [hh:mi:ss]"

Rückgabewert

Ein Wert vom Type datetime mit dem Zahlenwert der Sekunden seit dem 01.01.1970.

Hinweis

Jede Folge von Leer- und Tabulatorzeichen zwischen dem Datum und der Uhrzeit zählt als ein einzelnes Leerzeichen, sodass vor dem Aufruf von StringToTime() keine weitere Verarbeitung des time_string erforderlich ist.

Beispiel:

```
//--- Eingabeparameter
input group    "The date can be entered in any of the formats:"
input group    "yyyy.mm.dd [hh:mi], yyyy.mm.dd [hh:mi:ss]"
input group    "yyyymmdd [hh:mi:ss], yyyymmdd [hhmiss]"
input group    "yyyy/mm/dd [hh:mi:ss], yyyy-mm-dd [hh:mi:ss]"
input string   InpDateStr;    // Please enter the date here as a string

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die Umwandlung der in den Eingaben als String eingegebenen Zeit in einen Datumsv
   datetime time=StringToTime(InpDateStr);
//--- Anzeige der eingegebenen Zeichenkette und der erhaltenen Zeit im Journal
```

© 2000-2025, MetaQuotes Ltd.

```
   PrintFormat("Date entered as a string in the form '%s' is converted to datetime in
               InpDateStr, TimeToString(time, TIME_DATE|TIME_MINUTES|TIME_SECONDS));
//--- Erstellen einer vertikalen Linie an der Stelle des übergebenen Zeitstempels und
   if(CreateVLine(time))
      ChartNavigateToTime(time);
   /*
  Ergebnis:
   Date entered as a string in the form '' is converted to datetime in the form '1970.
   Date entered as a string in the form '2024' is converted to datetime in the form '2
   Date entered as a string in the form '202400' is converted to datetime in the form
   Date entered as a string in the form '20240000' is converted to datetime in the for
   Date entered as a string in the form '2024022410' is converted to datetime in the f
   Date entered as a string in the form '20240224 10' is converted to datetime in the
   Date entered as a string in the form '20240224 01' is converted to datetime in the
   Date entered as a string in the form '20240224 0030' is converted to datetime in th
   Date entered as a string in the form '20240224 0100' is converted to datetime in th
   */
  }
//+------------------------------------------------------------------+
//| Erstellen eines vertikalen Linienobjekts                         |
//+------------------------------------------------------------------+
bool CreateVLine(const datetime line_time)
  {
   ResetLastError();

   string name=MQLInfoString(MQL_PROGRAM_NAME)+"_VLINE";
   if(!ObjectCreate(0, name, OBJ_VLINE, 0, line_time, 0))
     {
      Print("ObjectCreate() failed. Error code: ", GetLastError());
      return(false);
     }
   ObjectSetInteger(0, name, OBJPROP_STYLE, STYLE_DOT);
   ObjectSetInteger(0, name, OBJPROP_SELECTABLE, true);

   return(true);
  }
//+------------------------------------------------------------------+
//| Verschieben des Charts auf die angegebene Zeit des Balkens       |
//+------------------------------------------------------------------+
bool ChartNavigateToTime(const datetime time)
  {
   ChartSetInteger(0, CHART_AUTOSCROLL, false);
   ResetLastError();

   int bar=iBarShift(_Symbol, PERIOD_CURRENT, time);
   if(bar<0)
     {
      PrintFormat("%s: iBarShift() failed. Error code: %d", __FUNCTION__, GetLastError
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
      }

   long first=0;
   if(!ChartGetInteger(0, CHART_FIRST_VISIBLE_BAR, 0, first))
     {
      PrintFormat("%s: ChartGetInteger() failed. Error code: %d", __FUNCTION__, GetLas
      return(false);
     }

   return(ChartNavigate(0, CHART_CURRENT_POS, (int)first-bar));
  }
```

Siehe auch

TimeToString, TimeToStruct

© 2000-2025, MetaQuotes Ltd.

StringFormat

Formatiert die erhaltenen Daten und gibt Zeile zurück.

```
string  StringFormat(
   string  format,      // Zeile mit Formatbeschreibung
   ...     ...          // Parameter
   );
```

Parameter

format

[in] Zeile, die Formatierungsverfahren enthält. Regel der Formatierung sind dieselben wie die für die Funktion PrintFormat

...

[in] Parameter, die durch Komma getrennt sind

Zeile.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- String-Variablen
   string output_string;
   string temp_string;
   string format_string;
//--- bereiten wir die Kontraktspezifikation vor
   temp_string=StringFormat("Kontraktspezifikation für %s:\n",_Symbol);
   StringAdd(output_string,temp_string);
//--- Ausgabe des int Wertes
   int digits=(int)SymbolInfoInteger(_Symbol,SYMBOL_DIGITS);
   temp_string=StringFormat("   SYMBOL_DIGITS = %d (Anzahl der Stellen nach dem Komma)
                            digits);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes mit einer Variablen Anzahl nach dem Dezimalkomma
   double point_value=SymbolInfoDouble(_Symbol,SYMBOL_POINT);
   format_string=StringFormat("   SYMBOL_POINT = %%.%df (Wert eines Punktes )\n",
                              digits);
   temp_string=StringFormat(format_string,point_value);
   StringAdd(output_string,temp_string);
//--- Ausgabe des int Wertes
   int spread=(int)SymbolInfoInteger(_Symbol,SYMBOL_SPREAD);
   temp_string=StringFormat("   SYMBOL_SPREAD = %d (aktueller Spread in Punkten)\n",
                            spread);
   StringAdd(output_string,temp_string);
//--- Ausgabe von int
   int min_stop=(int)SymbolInfoInteger(_Symbol,SYMBOL_TRADE_STOPS_LEVEL);
   temp_string=StringFormat("   SYMBOL_TRADE_STOPS_LEVEL = %d (minimaler Abstand in Pu
                            min_stop);
   StringAdd(output_string,temp_string);
//--- Ausgabe von double ohne Bruchteil
   double contract_size=SymbolInfoDouble(_Symbol,SYMBOL_TRADE_CONTRACT_SIZE);
   temp_string=StringFormat("   SYMBOL_TRADE_CONTRACT_SIZE = %.f (Kontraktgröße)\n",
                            contract_size);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes mit standardmäßiger Genauigkeit
   double tick_size=SymbolInfoDouble(_Symbol,SYMBOL_TRADE_TICK_SIZE);
   temp_string=StringFormat("   SYMBOL_TRADE_TICK_SIZE = %f (minimale Preisänderung)\r
                            tick_size);
   StringAdd(output_string,temp_string);
//--- Festlegung der Methode zur Berechnung von Swaps
   int swap_mode=(int)SymbolInfoInteger(_Symbol,SYMBOL_SWAP_MODE);
   string str_swap_mode;
   switch(swap_mode)
     {
      case SYMBOL_SWAP_MODE_DISABLED: str_swap_mode="SYMBOL_SWAP_MODE_DISABLED (keine
      case SYMBOL_SWAP_MODE_POINTS: str_swap_mode="SYMBOL_SWAP_MODE_POINTS (in Punkter
      case SYMBOL_SWAP_MODE_CURRENCY_SYMBOL: str_swap_mode="SYMBOL_SWAP_MODE_CURRENCY_
      case SYMBOL_SWAP_MODE_CURRENCY_MARGIN: str_swap_mode="SYMBOL_SWAP_MODE_CURRENCY_
      case SYMBOL_SWAP_MODE_CURRENCY_DEPOSIT: str_swap_mode="SYMBOL_SWAP_MODE_CURRENCY
      case SYMBOL_SWAP_MODE_INTEREST_CURRENT: str_swap_mode="SYMBOL_SWAP_MODE_INTEREST
      case SYMBOL_SWAP_MODE_INTEREST_OPEN: str_swap_mode="SYMBOL_SWAP_MODE_INTEREST_OP
      case SYMBOL_SWAP_MODE_REOPEN_CURRENT: str_swap_mode="SYMBOL_SWAP_MODE_REOPEN_CUR
      case SYMBOL_SWAP_MODE_REOPEN_BID: str_swap_mode="SYMBOL_SWAP_MODE_REOPEN_BID (du
     }
//--- Ausgabe des string Wertes
   temp_string=StringFormat("   SYMBOL_SWAP_MODE = %s\n",
                            str_swap_mode);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes mit standardmäßiger Genauigkeit
   double swap_long=SymbolInfoDouble(_Symbol,SYMBOL_SWAP_LONG);
```

© 2000-2025, MetaQuotes Ltd.

```
   temp_string=StringFormat("   SYMBOL_SWAP_LONG = %f (Buy Swap)\n",
                            swap_long);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes mit standardmäßiger Genauigkeit
   double swap_short=SymbolInfoDouble(_Symbol,SYMBOL_SWAP_SHORT);
   temp_string=StringFormat("   SYMBOL_SWAP_SHORT = %f (Sell Swap)\n",
                            swap_short);
   StringAdd(output_string,temp_string);
//--- Handelsmodus bestimmen
   int trade_mode=(int)SymbolInfoInteger(_Symbol,SYMBOL_TRADE_MODE);
   string str_trade_mode;
   switch(trade_mode)
     {
      case SYMBOL_TRADE_MODE_DISABLED: str_trade_mode="SYMBOL_TRADE_MODE_DISABLED (Har
      case SYMBOL_TRADE_MODE_LONGONLY: str_trade_mode="SYMBOL_TRADE_MODE_LONGONLY (nur
      case SYMBOL_TRADE_MODE_SHORTONLY: str_trade_mode="SYMBOL_TRADE_MODE_SHORTONLY (n
      case SYMBOL_TRADE_MODE_CLOSEONLY: str_trade_mode="SYMBOL_TRADE_MODE_CLOSEONLY (C
      case SYMBOL_TRADE_MODE_FULL: str_trade_mode="SYMBOL_TRADE_MODE_FULL (keine Begre
     }
//--- Ausgabe des string Wertes
   temp_string=StringFormat("   SYMBOL_TRADE_MODE = %s\n",
                            str_trade_mode);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes in kompakter Form
   double volume_min=SymbolInfoDouble(_Symbol,SYMBOL_VOLUME_MIN);
   temp_string=StringFormat("   SYMBOL_VOLUME_MIN = %g (Mindestvolumen des Trades)\n",
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes
   double volume_step=SymbolInfoDouble(_Symbol,SYMBOL_VOLUME_STEP);
   temp_string=StringFormat("   SYMBOL_VOLUME_STEP = %g (minimaler Schritt der Preisän
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes im
   double volume_max=SymbolInfoDouble(_Symbol,SYMBOL_VOLUME_MAX);
   temp_string=StringFormat("   SYMBOL_VOLUME_MAX = %g (maximales Vopkllumen;
   StringAdd(output_string,temp_string);
//--- Ermittlung der Methode für die Berechnung von Swap
   int calc_mode=(int)SymbolInfoInteger(_Symbol,SYMBOL_TRADE_CALC_MODE);
   string str_calc_mode;
   switch(calc_mode)
     {
      case SYMBOL_CALC_MODE_FOREX:str_calc_mode="SYMBOL_CALC_MODE_FOREX (Forex)";break
      case SYMBOL_CALC_MODE_FUTURES:str_calc_mode="SYMBOL_CALC_MODE_FUTURES (Futures)"
      case SYMBOL_CALC_MODE_CFD:str_calc_mode="SYMBOL_CALC_MODE_CFD (CFD)";break;
      case SYMBOL_CALC_MODE_CFDINDEX:str_calc_mode="SYMBOL_CALC_MODE_CFDINDEX (CFDs au
      case SYMBOL_CALC_MODE_CFDLEVERAGE:str_calc_mode="SYMBOL_CALC_MODE_CFDLEVERAGE (C
      case SYMBOL_CALC_MODE_EXCH_STOCKS:str_calc_mode="SYMBOL_CALC_MODE_EXCH_STOCKS (E
      case SYMBOL_CALC_MODE_EXCH_FUTURES:str_calc_mode="SYMBOL_CALC_MODE_EXCH_FUTURES
      case SYMBOL_CALC_MODE_EXCH_FUTURES_FORTS:str_calc_mode="SYMBOL_CALC_MODE_EXCH_FU
     }
//--- Ausgabe des string Wertes
   temp_string=StringFormat("   SYMBOL_TRADE_CALC_MODE = %s\n",
                            str_calc_mode);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes mit 2 Stellen nach dem Dezimalpounkt
   double margin_initial=SymbolInfoDouble(_Symbol,SYMBOL_MARGIN_INITIAL);
   temp_string=StringFormat("   SYMBOL_MARGIN_INITIAL = %.2f (anfängliche Margin)\n",
                            margin_initial);
   StringAdd(output_string,temp_string);
//--- Ausgabe des double Wertes mit 2 Stellen nach dem Dezimalpunkt
   double margin_maintenance=SymbolInfoDouble(_Symbol,SYMBOL_MARGIN_MAINTENANCE);
   temp_string=StringFormat("   SYMBOL_MARGIN_MAINTENANCE = %.2f (maintenance margin)\
```

© 2000-2025, MetaQuotes Ltd.

```
                              margin_maintenance);
   StringAdd(output_string,temp_string);
//--- Ausgabe des int Wertes
   int freeze_level=(int)SymbolInfoInteger(_Symbol,SYMBOL_TRADE_FREEZE_LEVEL);
   temp_string=StringFormat("   SYMBOL_TRADE_FREEZE_LEVEL = %d (Freeze Level in Punkte
                              freeze_level);
   StringAdd(output_string,temp_string);
   Print(output_string);
   Comment(output_string);
/* das Ergebnis der Ausführung
   Kontraktspezifikation für EURUSD:
     SYMBOL_DIGITS = 5 (Anzahl der Stellen nach dem Komma)
     SYMBOL_POINT = 0.00001 (Punkt)
     SYMBOL_SPREAD = 10 (aktueller Spread in Punkten)
     SYMBOL_TRADE_STOPS_LEVEL = 18 (minimaler Abstand in Punkten für Stop Orders)
     SYMBOL_TRADE_CONTRACT_SIZE = 100000 (Kontraktgröße)
     SYMBOL_TRADE_TICK_SIZE = 0.000010 (minimale Preisänderung)
     SYMBOL_SWAP_MODE = SYMBOL_SWAP_MODE_POINTS (in Punkten)
     SYMBOL_SWAP_LONG = -0.700000 (Swap Long)
     SYMBOL_SWAP_SHORT = -1.000000 (Swap Short)
     SYMBOL_TRADE_MODE = SYMBOL_TRADE_MODE_FULL (keine Begrenzungen für Transaktionen)
     SYMBOL_VOLUME_MIN = 0.01 (Mindestvolumen des Trades)
     SYMBOL_VOLUME_STEP = 0.01 (minimaler Schritt der Änderung des Volumens)
     SYMBOL_VOLUME_MAX = 500 (Höchstvolumen des Trades)
     SYMBOL_TRADE_CALC_MODE = SYMBOL_CALC_MODE_FOREX (Forex)
     SYMBOL_MARGIN_INITIAL = 0.00 (anfängliche Margin)
     SYMBOL_MARGIN_MAINTENANCE = 0.00 (maintenance margin)
     SYMBOL_TRADE_FREEZE_LEVEL = 0 (Freeze Level in Punkten)
*/
  }
```

Sehen Sie auch

PrintFormat, DoubleToString,ColorToString, TimeToString

© 2000-2025, MetaQuotes Ltd.

Mathematische Funktionen

Satz der mathematischen und trigonometrischen Funktionen.

Ursprünglich wurden mathematische Funktionen entwickelt, um relevante Operationen mit skalaren Werten durchzuführen. Ab diesem Build können die meisten Funktionen auf Matrizen und Vektoren angewendet werden. Dazu gehören MathAbs, MathArccos, MathArcsin, MathArctan, MathCeil, MathCos, MathExp, MathFloor, MathLog, MathLog10, MathMod, MathPow, MathRound, MathSin, MathSqrt, MathTan, MathExpm1, MathLog1p, MathArccosh, MathArcsinh, MathArctanh, MathCosh, MathSinh, und MathTanh. Solche Operationen implizieren eine elementweise Behandlung von Matrizen oder Vektoren. Beispiel:

```
//---
  matrix a= {{1, 4}, {9, 16}};
  Print("matrix a=\n",a);
  a=MathSqrt(a);
  Print("MatrSqrt(a)=\n",a);
  /*
   matrix a=
   [[1,4]
    [9,16]]
   MatrSqrt(a)=
   [[1,2]
    [3,4]]
  */
```

Bei MathMod und MathPow kann das zweite Element entweder ein Skalar oder eine Matrix/ein Vektor mit der entsprechenden Größe sein.

Funktion	Massnahme
MathAbs	Gibt absoluten Wert (Modulus) der angegebenen Zahl zurück
MathArccos	Gibt arccos x in Radianen zurück
MathArcsin	Gibt arcsin x in Radianen zurück
MathArctan	Gibt arctg x in Radianen zurück
MathArctan2	Gibt den Winkel (in Bogenmaß), dessen Tangente der Quotient aus zwei angegebenen Zahlen ist, zurück
MathClassify	Rückgabe des Typs einer reellen Zahl
MathCeil	Gibt den ganzzahligen Wert, der der naechste von oben ist, zurück
MathCos	Gibt Kosinus zurück
MathExp	Gibt Exponente der Zahl zurück
MathFloor	Gibt den ganzzahligen Wert, der der naechste von unten ist, zurück
MathLog	Gibt Napierschen Logarithmus zurück
MathLog10	Gibt Dezimallogarithmus zurück

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
MathMax	Gibt maximalen Wert der zwei numerischen Werte
MathMin	Gibt minimalen Wert der zwei numerischen Werte
MathMod	Gibt reellen überbleibsel der Division der zwei Zahlen
MathPow	Erhebt die Basis in die angegebene Potenz
MathRand	Gibt die pseudozufällige Zahl im Bereich von 0 bis 32767 zurück
MathRound	Rundet die Zahl bis die naechste Ganzzahl
MathSin	Gibt sin der Zahl zurück
MathSqrt	Gibt Quadratwurzel zurück
MathSrand	Stellt Anfangszustand für Geberieren der pseudozufälligen Realzahlen
MathTan	Gibt tg der Zahl zurück
MathIsValidNumber	Prüft Richtigkeit der reelle Zahlen
MathExpm1	Gibt den Wert von MathExp(x)-1 zurück
MathLog1p	Gibt den Wert von MathLog(1+x) zurück
MathArccosh	Gibt den Wert des hyperbolischen Arcuscosinus zurück
MathArcsinh	Gibt den Wert des hyperbolischen Arcussinus zurück
MathArctanh	Gibt den Wert des hyperbolischen Arcustangens zurück
MathCosh	Gibt den hyperbolischen Cosinus zurück
MathSinh	Gibt den hyperbolischen Sinus zurück
MathTanh	Gibt den hyperbolischen Tangens zurück
MathSwap	Ändert die Reihenfolge der Bytes folgender Typen ushort/uint/ushort

© 2000-2025, MetaQuotes Ltd.

MathAbs

Gibt absoluten Wert der ihr übergebenen Zahl.

```
double  MathAbs(
   double  value      // Zahl
   );
```

Parameter

value

[in] Zahlenwert.

Rückgabewert

Wert des Typs double, größer als oder gleich 0.

Hinweis

Statt der Funktion MathAbs() kann die Funktion fabs() verwendet werden.

Beispiel:

```
//--- Eingabeparameter
input int      InpValue = -10;   // tragen Sie irgendeinen Wert ein

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermittlung des absoluten Werts der in den Eingaben eingegebenen Zahl
   uint abs_value=MathAbs(InpValue);
//--- Druck der Werte im Journal
   PrintFormat("The entered value %d received the value %u after using the MathAbs() f
  }
```

© 2000-2025, MetaQuotes Ltd.

MathArccos

Gibt den Arkuskosinus-Wert von x im Bereich 0 und □ in Radiant.

```
double  MathArccos(
   double  val     // -1<val<1
   );
```

Parameter

val

[in] Wert val zwischen -1 und 1, dessen Arkuskosinus berechnet werden muss.

Rückgabewert

Arkuskosinus der Zahl in Radiant. Wenn val weniger als -1 oder mehr als 1 ist, gibt die Funktion NaN (unbestimmt, not a number) zurück.

Hinweis

Statt der Funktion MathArccos() kann die Funktion acos() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/1000);
   delta[0]=-1;
//--- Ermittlung von 101 Werten von -1 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Arkuskosinus für jeden Wert des Vektors X
   vector Y=MathArccos(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);
```

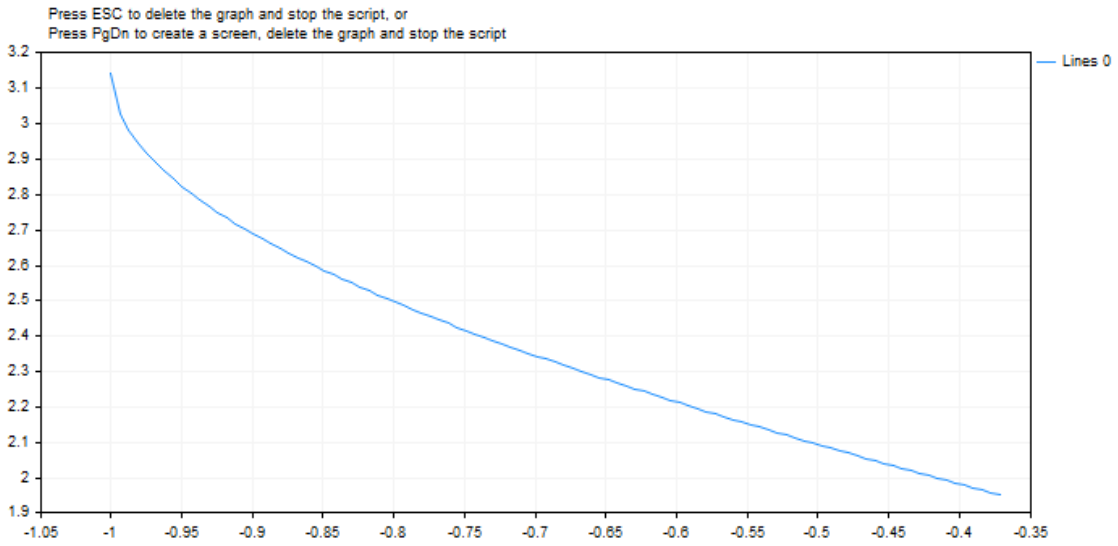
© 2000-2025, MetaQuotes Ltd.

```
//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Screenshot erstellen und speichern des Bilds in einer Datei      |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRIT
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathArcsin

Ermittelt den Arkussinus von x im Bereich von -□/2 bis □/2 Radiant.

```
double  MathArcsin(
   double  val      // -1<value<1
   );
```

Parameter

val

[in] Wert val zwischen -1 und 1, dessen Arkussinus berechnet werden muss.

Rückgabewert

Arcsin val in Radiant im Bereich von -□/2 bis □/2 Radiant. Wenn val kleiner als -1 oder größer als 1, gibt die Funktion NaN (unbestimmt, not a number) zurück.

Hinweis

Statt der Funktion MathArcsin() kann die Funktion asin() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/1000);
   delta[0]=-1;
//--- Ermittlung von 101 Werten von -1 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Sinuswertes für jeden Wert des X-Vektors
   vector Y=MathArcsin(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);
```

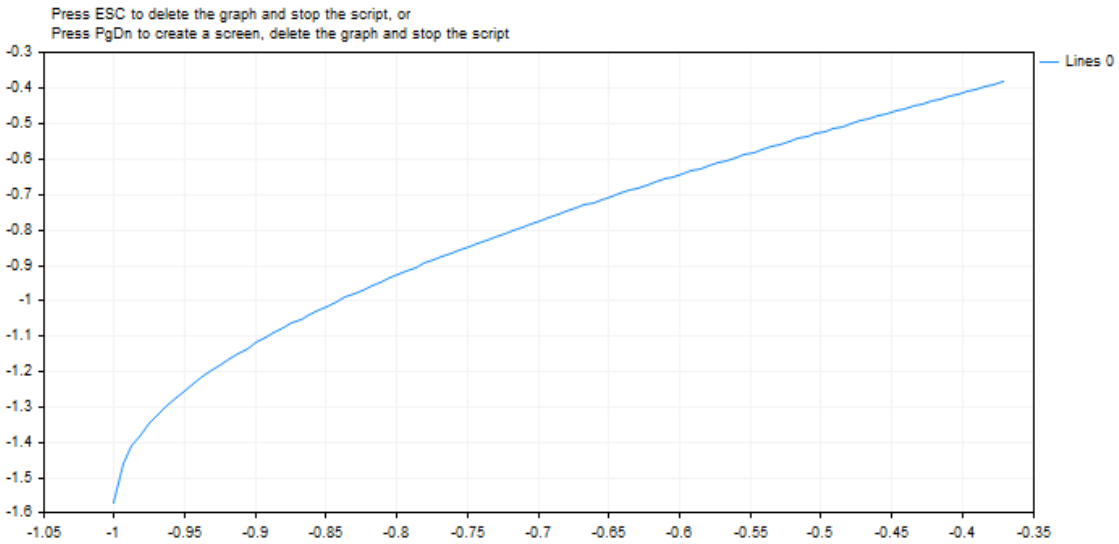
© 2000-2025, MetaQuotes Ltd.

```
//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRIT
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathArctan

Gibt den Arkustangens von x zurück. Wenn x gleich 0 ist, gibt die Funktion 0 zurück.

```
double  MathArctan(
   double  value      // Tangens
   );
```

Parameter

value

[in] Zahl, von der der Arkustangens berechnet werden soll.

Rückgabewert

MathArctan gibt Wert im Bereich von -□/2 bis □/2 Radiant zurück.

Hinweis

Statt der Funktion MathArctan() kann die Funktion atan() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Wertes des Arkustangens für jeden Wert des Vektors X
   vector Y=MathArctan(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

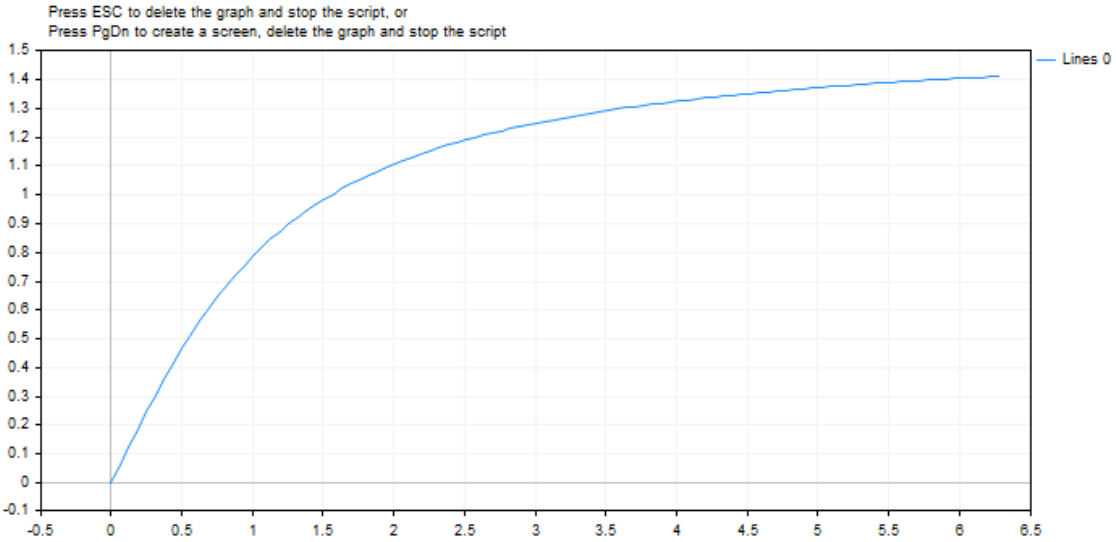
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

MathArctan2

Liefert den Winkel (in Radiant), dessen Wert der Quotient aus zwei angegebenen Zahlen ist.

```
double MathArctan2(
 double y     // Die Y-Koordinate eines Punktes
 double x     // Die X-Koordinate eines Punktes
  );
```

Parameter

y

[in] Wert des Y-Koordinate.

x

[in] Wert der X-Koordinate.

Rückgabewert

MathArctan2 berechnet einen Winkel, θ, im Bereich von -□ bis □ Radiant zurück, sodass MathTan(θ) =y/x.

Bitte beachten Sie folgendes:

- Für (x, y) im Quadranten 1, 0< θ < □/2
- Für (x, y) im Quadranten 2, □/2< θ ≤ □
- Für (x, y) im Quadranten 3, -□ < θ < -□/2
- Für (x, y) im Quadranten 4, -□/2 < θ < 0

Für Punkte an den Grenzen der Quadranten verhält sich der Rückgabewert wie folgt:

- Wenn y gleich 0 und x nicht negativ ist, ist θ = 0.
- Wenn y gleich 0 und x nicht negativ ist, θ = □.
- Wenn y positiv 0 und x gleich 0 ist, θ = □.
- Wenn y negativ 0 und x gleich 0 ist, θ = -□/2.
- Wenn y gleich 0 und x gleich 0 ist, ist θ = 0.

Hinweis

Anstelle der Funktion MathArctan2() können Sie die Funktion atan2() verwenden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

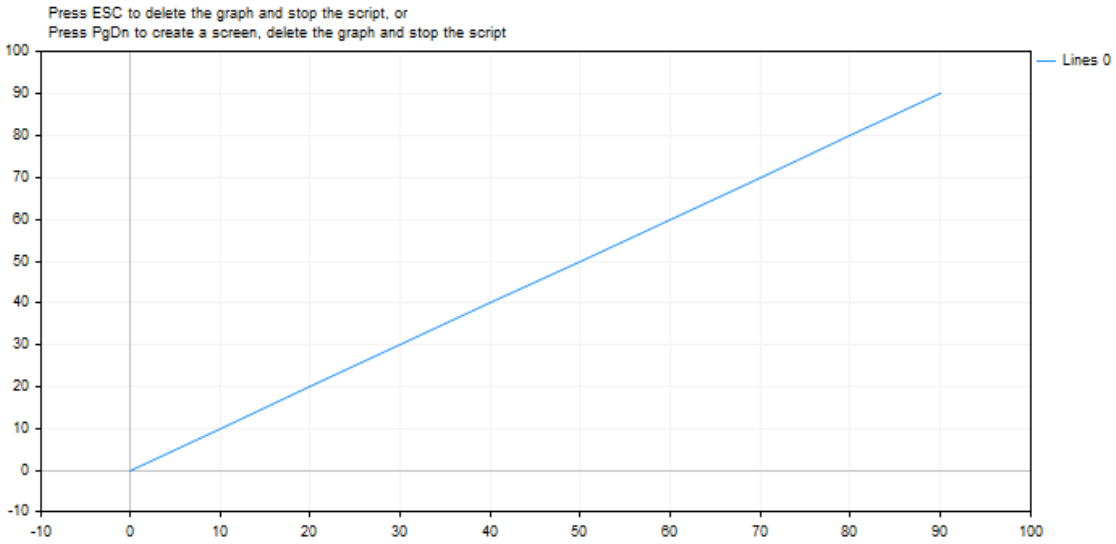
```
void OnStart()
  {
   vector delta=vector::Full(10,10);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Wertes des Arkustangens für jeden Wert des Vektors X
   vector Y=delta.CumSum();

   Print("vector delta = \n",delta);
   Print("vector X = \n",X);
   Print("vector Y = \n",Y);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[];;
   double y_array[];;
   X.Swap(x_array);
   Y.Swap(y_array);

   double array[10];
   for(int i=0; i<10; i++)
     {
      array[i]=MathArctan2(y_array[i],x_array[i]);
     }

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:

© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

MathClassify

Determiniert den Typ einer reellen Zahl und gibt ein Ergebnis als Wert aus der Enumeration ENUM_FP_CLASS zurück.

```
ENUM_FP_CLASS  MathClassify(
   double  value      // reelle Zahl
   );
```

Parameter

value

[in] Die zu prüfende reelle Zahl

Rückgabewert

Ein Wert aus der Enumeration ENUM_FP_CLASS

ENUM_FP_CLASS

ID	Beschreibung
FP_SUBNORMAL	Eine anormale Zahl, die näher Null ist, als die kleinst mögliche normale Zahl DBL_MIN (2.2250738585072014e-308)
FP_NORMAL	Ein normale Zahl im Bereich von 2.2250738585072014e-308 bis 1.7976931348623158e+308
FP_ZERO	Eine positive oder negativ Null
FP_INFINITE	Eine Zahl, die nicht durch einen entsprechenden Typ dargestellt werden kann, Minus- oder Plus-Unendlich
FP_NAN	Das ist keine Zahl

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- test NaN
  double nan=double("nan");
  PrintFormat("Test NaN: %G is %s, MathIsValidNumber(NaN)=%s",
              nan,
              EnumToString(MathClassify(nan)),
              (string)MathIsValidNumber(nan));
//--- test infinity
  double inf=double("inf");
```

© 2000-2025, MetaQuotes Ltd.

```
  PrintFormat("Test Inf: %G is %s, MathIsValidNumber(inf)=%s",
              inf,
              EnumToString(MathClassify(inf)),
              (string)MathIsValidNumber(inf));
//--- test normal value
  double normal=1.2345e6;
  PrintFormat("Test Normal: %G is %s, MathIsValidNumber(normal)=%s",
              normal,
              EnumToString(MathClassify(normal)),
              (string)MathIsValidNumber(normal));
//--- test subnormal value
  double sub_normal=DBL_MIN/2.0;
  PrintFormat("Test Subnormal: %G is %s, MathIsValidNumber(sub_normal)=%s",
              sub_normal,
              EnumToString(MathClassify(sub_normal)),
              (string)MathIsValidNumber(sub_normal));
//--- test zero value
  double zero=0.0/(-1);
  PrintFormat("Test Zero: %G is %s, MathIsValidNumber(zero)=%s",
              zero,
              EnumToString(MathClassify(zero)),
              (string)MathIsValidNumber(zero));
 }
 /*
 Result:
   Test NaN: NAN is FP_NAN, MathIsValidNumber(NaN)=false
   Test Inf: INF is FP_INFINITE, MathIsValidNumber(inf)=false
   Test Normal: 1.2345E+06 is FP_NORMAL, MathIsValidNumber(normal)=true
   Test Subnormal: 1.11254E-308 is FP_SUBNORMAL, MathIsValidNumber(sub_normal)=true
   Test Zero: -0 is FP_ZERO, MathIsValidNumber(zero)=true
*/
//+------------------------------------------------------------------+
```

Siehe auch

Reelle Zahlen (double, float), MathIsValidNumber

© 2000-2025, MetaQuotes Ltd.

MathCeil

Gibt den ganzzahligen Wert zurück, der der nächste von oben ist.

```
double  MathCeil(
   double  val      // Zahl
   );
```

Parameter

val

[in] Numerischer Wert.

Rückgabewert

Der numerische Wert, der die kleinste ganze Zahl darstellt, die größer oder gleich val ist.

Hinweis

Statt der Funktion MathCeil()kann die Funktion ceil() verwendet werden.

Beispiel:

```
#define VALUES_TOTAL 31

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Variablen für die Umwandlung deklarieren
   double value=0;                     // reelle Zahl für MathCeil
   int    ceil_value=0;                // Ergebnis des Funktionsaufrufs
//--- in einer Schleife über die Anzahl der Dezimalschritte einer reellen Zahl
   for(int i=0; i<VALUES_TOTAL; i++)
     {
      //--- erhöhe den Zahlenwert,
      //--- hole den nächstgelegenen ganzzahligen Wert von oben
      //--- und drucke die Kontrollwerte ins Journal
      value+=0.1;
      ceil_value=(int)MathCeil(NormalizeDouble(value,1));
      PrintFormat("value: %.1f, ceil value: %d",value,ceil_value);
      /*
     Ergebnis:
      value: 0.1, ceil value: 1
      value: 0.2, ceil value: 1
      value: 0.3, ceil value: 1
      value: 0.4, ceil value: 1
      value: 0.5, ceil value: 1
      value: 0.6, ceil value: 1
```

© 2000-2025, MetaQuotes Ltd.

```
      value: 0.7, ceil value: 1
      value: 0.8, ceil value: 1
      value: 0.9, ceil value: 1
      value: 1.0, ceil value: 1
      value: 1.1, ceil value: 2
      value: 1.2, ceil value: 2
      value: 1.3, ceil value: 2
      value: 1.4, ceil value: 2
      value: 1.5, ceil value: 2
      value: 1.6, ceil value: 2
      value: 1.7, ceil value: 2
      value: 1.8, ceil value: 2
      value: 1.9, ceil value: 2
      value: 2.0, ceil value: 2
      value: 2.1, ceil value: 3
      value: 2.2, ceil value: 3
      value: 2.3, ceil value: 3
      value: 2.4, ceil value: 3
      value: 2.5, ceil value: 3
      value: 2.6, ceil value: 3
      value: 2.7, ceil value: 3
      value: 2.8, ceil value: 3
      value: 2.9, ceil value: 3
      value: 3.0, ceil value: 3
      value: 3.1, ceil value: 4
      */
   }
}
```

© 2000-2025, MetaQuotes Ltd.

MathCos

Funktion gibt den Kosinus eines Winkels zurück.

```
double  MathCos(
   double  value      // Zahl
   );
```

Parameter

value

[in] Winkel in Radiant.

Rückgabewert

Wert des Typs double im Bereich von -1 bis 1.

Hinweis

Statt der Funktion MathCos()kann die Funktion cos() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Arkuskosinuswert für jeden Wert des Vektors X
   vector Y=MathCos(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

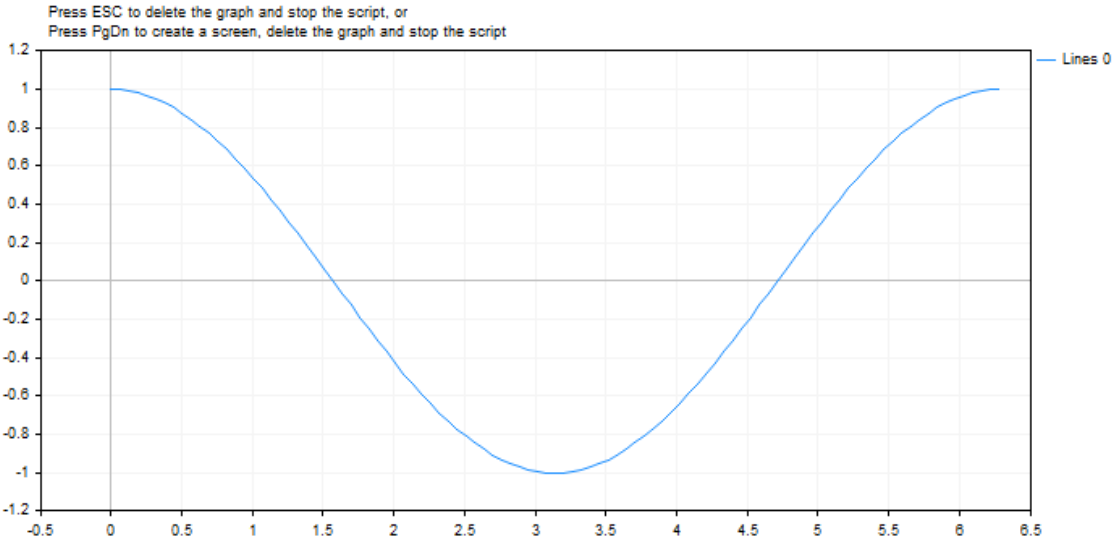
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

MathExp

Gibt den Wert der Zahl e hoch d zurück.

```
double  MathExp(
   double  value      // Exponent für die Zahl e
   );
```

Parameter

value

[in] der Exponent für e.

Rückgabewert

Wert vom Typ double. Beim Bereichsüberschreitung gibt die Funktion INF (unendlich) zurück, bei Bereichsunterschreitung gibt MathExp 0 zurück.

Hinweis

Statt der Funktio MathExp() kann die Funktion exp() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- berechne 9 Werte von 0 bis 8 mit Schritt 1
   vector X(9,VectorArange);
   Print("vector X = \n",X);
//--- Berechnung von "e" (Eulersche Zahl) hoch jedem Wert aus dem Vektor X
   X=MathExp(X);
   Print("MathExp(X) = \n",X);

//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

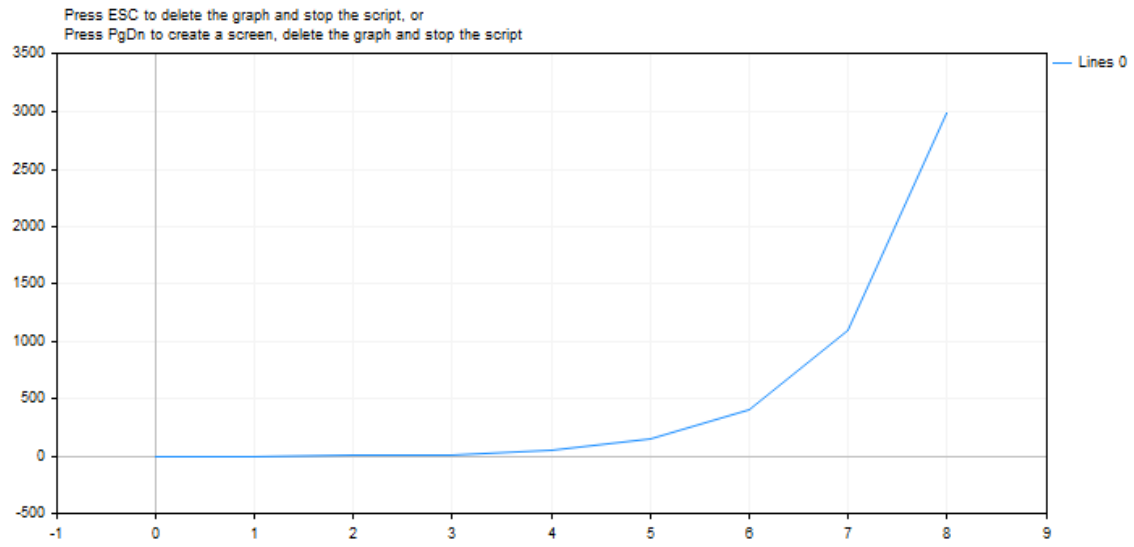
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQLI
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathFloor

Gibt den ganzzahligen Wert zurück, der der nächste von unten ist.

```
double  MathFloor(
   double  val     // Zahl
   );
```

Parameter

val

[in] Numerischer Wert.

Rückgabewert

Der numerische Wert, der die größte ganze Zahl darstellt, die kleiner oder gleich val ist.

Hinweis

Statt der Funktion MathFloor() kann die Funktion floor() verwendet werden.

Beispiel:

```
#define VALUES_TOTAL 31

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Variablen für die Umwandlung deklarieren
   double value=0;                     // reelle Zahl für MathFloor
   int    floor_value=0;               // Ergebnis des Funktionsaufrufs
//--- in einer Schleife über die Anzahl der Dezimalschritte einer reellen Zahl
   for(int i=0; i<VALUES_TOTAL; i++)
     {
      //--- erhöhe den Zahlenwert,
      //--- ermittle die nächstgelegenen, ganzzahligen Wert von unten
      //--- und drucke die Kontrollwerte ins Journal
      value+=0.1;
      floor_value=(int)MathFloor(NormalizeDouble(value,1));
      PrintFormat("value: %.1f, floor value: %d",value,floor_value);
      /*
     Ergebnis:
      value: 0.1, floor value: 0
      value: 0.2, floor value: 0
      value: 0.3, floor value: 0
      value: 0.4, floor value: 0
      value: 0.5, floor value: 0
      value: 0.6, floor value: 0
```

© 2000-2025, MetaQuotes Ltd.

```
      value: 0.7, floor value: 0
      value: 0.8, floor value: 0
      value: 0.9, floor value: 0
      value: 1.0, floor value: 1
      value: 1.1, floor value: 1
      value: 1.2, floor value: 1
      value: 1.3, floor value: 1
      value: 1.4, floor value: 1
      value: 1.5, floor value: 1
      value: 1.6, floor value: 1
      value: 1.7, floor value: 1
      value: 1.8, floor value: 1
      value: 1.9, floor value: 1
      value: 2.0, floor value: 2
      value: 2.1, floor value: 2
      value: 2.2, floor value: 2
      value: 2.3, floor value: 2
      value: 2.4, floor value: 2
      value: 2.5, floor value: 2
      value: 2.6, floor value: 2
      value: 2.7, floor value: 2
      value: 2.8, floor value: 2
      value: 2.9, floor value: 2
      value: 3.0, floor value: 3
      value: 3.1, floor value: 3
      */
     }
```

© 2000-2025, MetaQuotes Ltd.

MathLog

Gibt den natürlichen Logarithmus zurück.

```
double  MathLog(
   double  val      //Zahl zur Bildung des Logarithmus
   );
```

Parameter

val

[in] Werten, dessen Logarithmus berechnet werden soll.

Rückgabewert

Der natürliche Logarithmus des Wertes, falls erfolgreich. Wenn val negativ ist, gibt die Funktion NaN (unbestimmt, not a number) zurück. Wenn val gleich 0 ist, gibt die Funktion INF (unendlich) zurück.

Hinweis

Statt der Funktion MathLog() kann die Funktion log() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- berechne 9 Werte von 0 bis 8 mit Schritt 1
   vector X(9,VectorArange);
   Print("vector X = \n",X);
//--- Berechnung des Logarithmus für jeden Wert des Vektors X
   X=MathLog(X);
   Print("MathLog(X) = \n",X);

//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(y_array,clrDodgerBlue);
```

© 2000-2025, MetaQuotes Ltd.

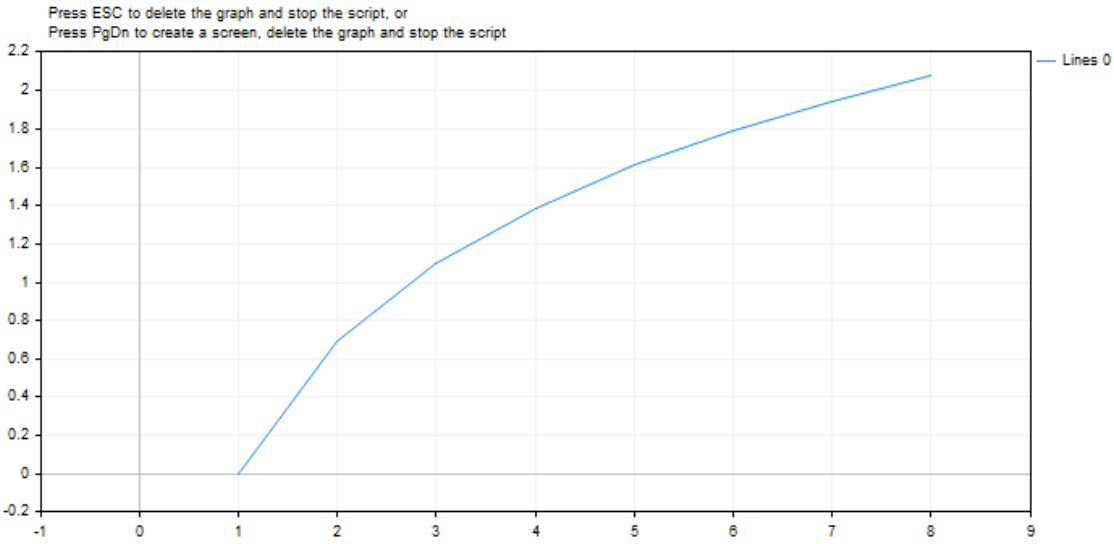
```
//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
   /*
  Ergebnis:
   vector X =
   [0,1,2,3,4,5,6,7,8]
   MathLog(X) =
   [-inf,0,0.6931471805599453,1.09861228866811,1.386294361119891,1.6094379124341,1.791
   */
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:

© 2000-2025, MetaQuotes Ltd.



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathLog

Gibt den Logarithmus einer Zahl zur Basis 10 zurück.

```
double  MathLog10(
   double  val      // Zahl zur Bildung des Logarithmus
   );
```

Parameter

val

[in] Der Wert, dessen Logarithmus zur Basis 10 berechnet werden soll.

Rückgabewert

Der Logarithmus zur Basis 10 von val im Erfolgsfall. Wenn val negativ ist, gibt die Funktion NaN (unbestimmt, not a number) zurück. Wenn val gleich 0 ist, gibt die Funktion INF (unendlich) zurück.

Hinweis

Statt der Funktion MathLog10() kann die Funktion log10() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- berechne 9 Werte von 0 bis 8 mit Schritt 1
   vector X(9,VectorArange);
   Print("vector X = \n",X);
//--- Berechnung des Logarithmus zur Basis 10 für jeden Wert des Vektors X
   X=MathLog10(X);
   Print("MathLog10(X) = \n",X);

//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

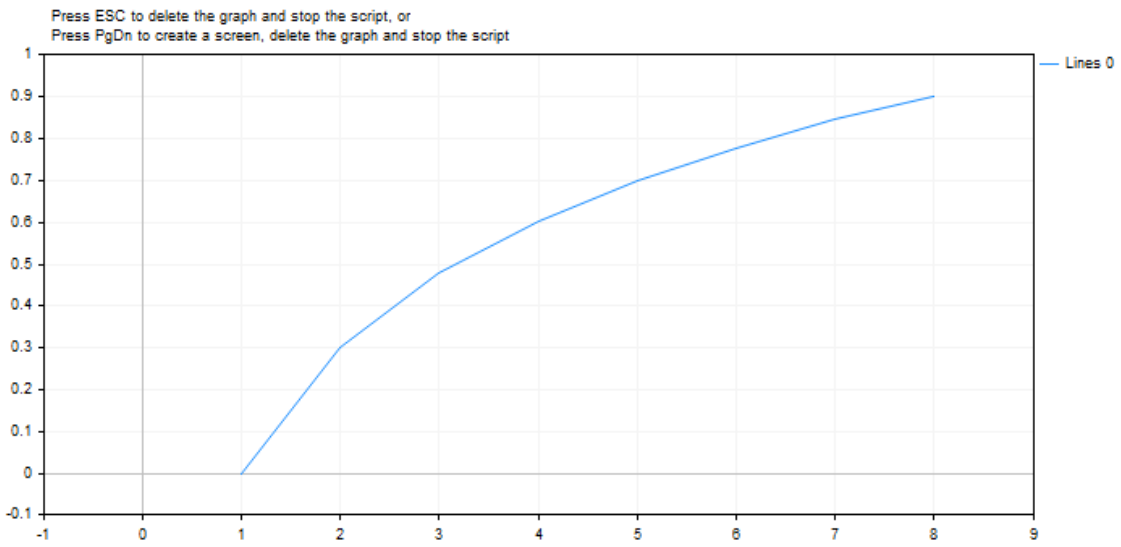
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
   /*
  Ergebnis:
   vector X =
   [0,1,2,3,4,5,6,7,8]
   MathLog10(X) =
   [-inf,0,0.3010299956639812,0.4771212547196624,0.6020599913279624,0.6989700043360189
   */
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQLI
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathMax

Funktion gibt den größeren von zwei numerischen Werten zurück.

```
double  MathMax(
   double  value1,     // erster Wert
   double  value2      // zweiter Wert
   );
```

Parameter

value1

[in] Der erste numerische Wert.

value2

[in] Der zweite numerische Wert.

Rückgabewert

Die größere der beiden Zahlen.

Hinweis

Statt der Funktion MathMax()kann die Funktion fmax() verwendet werden. Funktionen fmax(), fmin(), MathMax(), MathMin() können mit ganzzahligen Typen arbeiten, ohne sie in den Typ double umzuwandeln.

Wenn Werte verschiedener Typen der Funktion übergeben werden, wird der Wert des niedrigen Types automatisch in den des höheren konvertiert. Der Typ des Rückgabewerts entspricht dem des höheren Typs.

Bei der Übergabe desselben Typs wird keine Typenumwandlung durchgeführt.

Beispiel:

```
//--- Eingabeparameter
input int                  InpPeriod = 10;            // Periodenlänge des gleitenden
input ENUM_MA_METHOD       InpMethod = MODE_SMA;      // Glättungsverfahren des gleite
input ENUM_APPLIED_PRICE   InpPrice  = PRICE_CLOSE;   // Berechnung des gleitenden Du

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wenn die Periodenlänge des gleitenden Durchschnitts auf einen Wert kleiner als 1
   int period=(InpPeriod<1 ? 10 : InpPeriod);
//--- Erstellen des Handles für den Indikator Gleitender Durchschnitt
   int handle=iMA(Symbol(),Period(),period,0,InpMethod,InpPrice);
   if(handle==INVALID_HANDLE)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("Failed to create the Moving Average indicator handle. Fehler: ",GetLastEr
      return;
     }
//--- Ermitteln des aktuellen Bid-Preises
   double bid=0;
   ResetLastError();
   if(!SymbolInfoDouble(Symbol(),SYMBOL_BID,bid))
     {
      Print("Failed to get Bid price. Fehler: ",GetLastError());
      return;
     }
//--- Ermitteln des gleitenden Durchschnittswerts für den aktuellen Balken
   double array[1];
   int    copied=CopyBuffer(handle,0,0,1,array);
   if(copied!=1)
     {
      Print("Failed to get Moving Average data. Fehler: ",GetLastError());
      return;
     }
//--- Ermitteln des höheren Preises der beiden (Bid-Preis und Wert des gleitenden Durc
   double max_price=MathMax(bid,array[0]);
   PrintFormat("Bid: %.*f, Moving Average: %.*f, highest price of the two: %.*f",_Digi
   PrintFormat("Bid price %s moving average",(bid>array[0] ? "higher" : bid<array[0] ?
  }
```

© 2000-2025, MetaQuotes Ltd.

MathMin

Funktion gibt kleineren von zwei numerischen Werten zurück.

```
double  MathMin(
   double  value1,      // erster Wert
   double  value2       // zweiter Wert
   );
```

Parameter

value1

[in] Erster numerischer Wert.

value2

[in] Zweiter numerischer Wert.

Rückgabewert

Die kleinere der beiden Zahlen.

Hinweis

Statt der Funktion MathMin() kann die Funktion fmin() verwendet werden. Funktionen fmax(), fmin(), MathMax(), MathMin() können mit ganzzahligen Typen arbeiten, ohne sie in den Typ double umzuwandeln.

Wenn Werte verschiedener Typen der Funktion übergeben werden, wird der Wert des niedrigen Types automatisch in den des höheren konvertiert. Typ des Rückgabewertes entspricht dem höheren Typ.

Bei der Übergabe desselben Typs, wird keine Typenumwandlung durchgeführt.

Beispiel:

```
//--- Eingabeparameter
input uint  InpBars  =  100000;  // gewünschte Anzahl von Balken

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln der Anzahl der verfügbaren Balken auf dem Server
   uint bars_total = Bars(Symbol(),Period());
   if(bars_total==0)
     {
      PrintFormat("Data for timeseries %s %s not yet generated in the terminal. Please
      return;
     }
//--- Ermitteln der Mindestanzahl der Balken aus zwei Werten - aus den auf dem Server
```

© 2000-2025, MetaQuotes Ltd.

```
   int bars = (int)MathMin(bars_total,InpBars);
//--- wenn mehr Balken angefordert werden als auf dem Server verfügbar sind, wird dies
   if(bars_total<InpBars)
      PrintFormat("Number of bars on the server (%u) is less than requested (%u)",bars
//--- Anzeige der Anzahl der für die Arbeit verfügbaren Balken im Journal
   Print("Bars available for work: ",bars);
  }
```

© 2000-2025, MetaQuotes Ltd.

MathMod

Gibt den reellen Rest aus der Division zweier Zahlen zurück.

```
double  MathMod(
   double  value,       // Dividend
   double  value2       // Divisor
   );
```

Parameter

value

[in] Dividend.

value2

[in] Divisor

Rückgabewert

Die Funktion MathMod berechnet den reellen Rest f von val / y, sodass val = i * y + f, wobei i eine ganze Zahl ist, f das gleiche Vorzeichen wie val hat und der Absolutwert von f kleiner als der Absolutwert von y ist.

Hinweis

Statt der Funktion MathMod() kann die Funktion fmod() verwendet werden.

Beispiel:

```
#property script_show_inputs

//--- Eingabeparameter
input double   InpDividentValue  =  10;   // Dividend
input double   InpDivisorValue   =  3;    // Divisor

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermittlung des realen Rests der Division der in den Eingaben eingegebenen Zahler
   double res=MathMod(InpDividentValue,InpDivisorValue);
//--- Druck des Ergebnisses in das Journal
   PrintFormat("Real remainder when dividing %.2f by %.2f = %.2f",InpDividentValue,Inp
  }
```

© 2000-2025, MetaQuotes Ltd.

MathPow

Berechnet die Potenz von der angegebenen Basis.

```
double  MathPow(
   double  base,          // Basis
   double  exponent       // Exponent
   );
```

Parameter

base

[in] Basis.

exponent

[in] Potenzwert.

Rückgabewert

Wert von Basis hoch dem Exponenten.

Hinweis

Statt der Funktion MathPow() kann die Funktion pow() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#property script_show_inputs

#include <Graphics\Graphic.mqh>

//--- Eingabeparameter
input double   InpExponentValue  =  2;     // Exponent

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ermittle 11 Werte von 0 bis 10 mit Schritt 1
   vector X(11,VectorArange);
   Print("vector X = \n",X);
//--- berechne für jeden Basiswert des Vektors X mit dem Exponenten InpExponentValue
   X=MathPow(X,InpExponentValue);
   Print("MathPow(X,",(string)InpExponentValue,") = \n",X);
```

© 2000-2025, MetaQuotes Ltd.

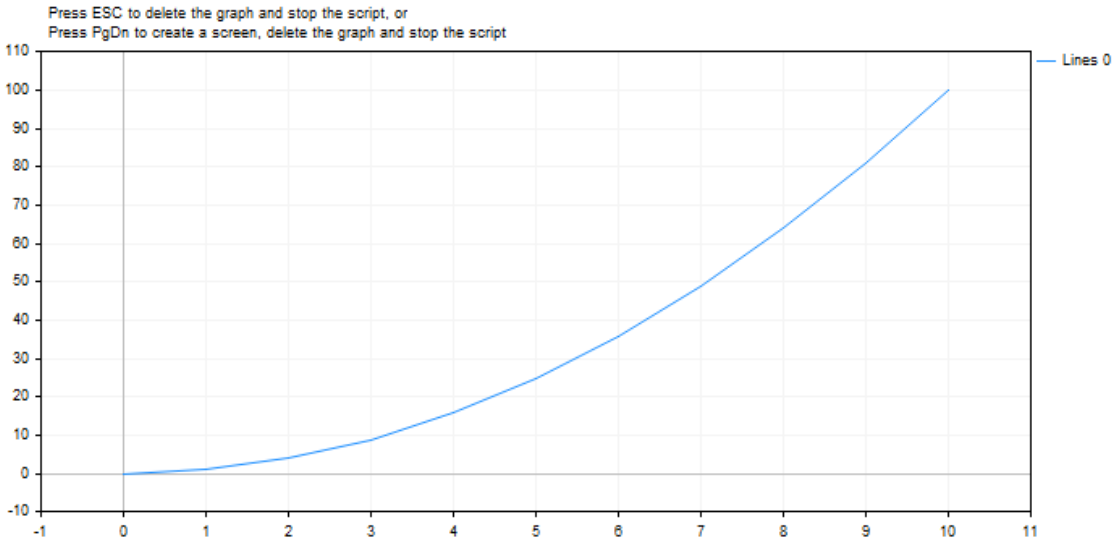
```
//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
   /*
  Ergebnis:
   vector X =
   [0,1,2,3,4,5,6,7,8,9,10]
   MathPow(X,2.0) =
   [0,1,4,9,16,25,36,49,64,81,100]
   */
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:

© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

MathRand

Gibt pseudozufällige Ganzzahl im Bereich von 0 bis 32767 zurück.

```
int  MathRand();
```

Rückgabewert

Ganzzahl im Bereich von 0 bis 32767.

Hinweis

Vor dem zweiten Aufruf der Funktion muss die Funktion MathSrand verwendet werden, um Generator der pseudozufälligen Zahlen in die Anfangsposition einzustellen.

Hinweis

Statt der Funktion MathRand() kann die Funktion rand() verwendet werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- setze einen neuen Ausgangszustand, um bei jedem Start eine Reihe von pseudozufäl
   MathSrand(GetTickCount());
//--- in einer Schleife 10 generierte Pseudo-Zufallszahlen im Journal anzeigen
   for(int i=0; i<10; i++)
     {
      int rand_value=MathRand();
      PrintFormat("Pseudorandom integer №%d: %u",i+1,rand_value);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

MathRound

Gibt einen Wert zurück, der auf die nächste Ganzzahl des angegebenen numerischen Wertes gerundet ist.

```
double  MathRound(
   double  value      // der zu rundende Wert
   );
```

Parameter

value

[in] der zu rundende Wert.

Rückgabewert

Wert, der auf die nächste Ganzzahl gerundet wurde.

Hinweis

Statt der Funktion MathRound() kann die Funktion round() verwendet werden.

Beispiel:

```
#define VALUES_TOTAL 31

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Variablen für die Umwandlung deklarieren
   double value=0;                     // reelle Zahl für das Runden mit MathRound
   int    round_value=0;               // Ergebnisvariable der Rundung
//--- in einer Schleife über die Anzahl der Dezimalschritte einer reellen Zahl
   for(int i=0; i<VALUES_TOTAL; i++)
     {
      //--- Erhöhen der reellen Zahl,
      //--- Erhalt des numerischen Werts, der auf die nächste Ganzzahl gerundet wurde
      //--- und drucke die Kontrollwerte ins Journal
      value+=0.1;
      round_value=(int)MathRound(NormalizeDouble(value,1));
      PrintFormat("value: %.1f, round value: %d",value,round_value);
      /*
     Ergebnis:
      value: 0.1, round value: 0
      value: 0.2, round value: 0
      value: 0.3, round value: 0
      value: 0.4, round value: 0
      value: 0.5, round value: 1
```

© 2000-2025, MetaQuotes Ltd.

```
      value: 0.6, round value: 1
      value: 0.7, round value: 1
      value: 0.8, round value: 1
      value: 0.9, round value: 1
      value: 1.0, round value: 1
      value: 1.1, round value: 1
      value: 1.2, round value: 1
      value: 1.3, round value: 1
      value: 1.4, round value: 1
      value: 1.5, round value: 2
      value: 1.6, round value: 2
      value: 1.7, round value: 2
      value: 1.8, round value: 2
      value: 1.9, round value: 2
      value: 2.0, round value: 2
      value: 2.1, round value: 2
      value: 2.2, round value: 2
      value: 2.3, round value: 2
      value: 2.4, round value: 2
      value: 2.5, round value: 3
      value: 2.6, round value: 3
      value: 2.7, round value: 3
      value: 2.8, round value: 3
      value: 2.9, round value: 3
      value: 3.0, round value: 3
      value: 3.1, round value: 3
      */
   }
}
```

© 2000-2025, MetaQuotes Ltd.

MathSin

Gibt Sinus des angegebenen Winkels zurück.

```
double  MathSin(
   double  value      // Argument in Radiant
   );
```

Parameter

value

[in] Winkel in Radiant.

Rückgabewert

Sinus des Winkels in Radiant. Gibt einen Wert im Bereich von -1 bis 1 zurück.

Hinweis

Statt der Funktion MathSin() kann die Funktion sin() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Sinuswertes für jeden Wert des X-Vektors
   vector Y=MathSin(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

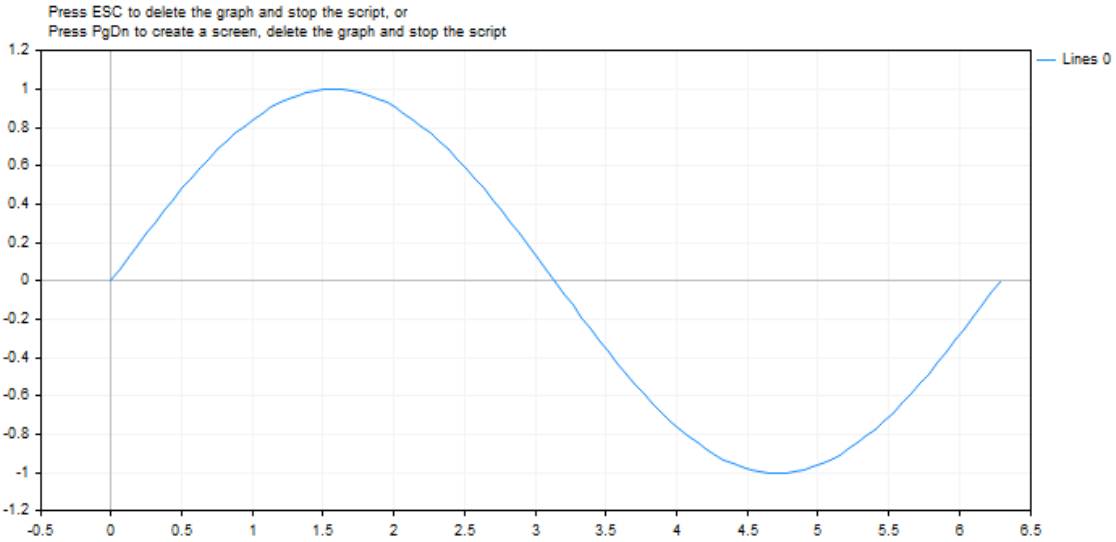
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

MathSqrt

Gibt Quadratwurzel der Zahl zurück.

```
double  MathSqrt(
   double  value      // positive Zahl
   );
```

Parameter

value

[in] Positiver numerischer Wert.

Rückgabewert

Quadratwurzel der Zahl value. Wenn value negativ ist, gibt MathSqrt den Wert NaN (unbestimmt, not a number) zurück.

Hinweis

Statt der Funktion MathSqrt() kann die Funktion sqrt() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ermittle 11 Werte von 0 bis 10 mit Schritt 1
   vector X(11,VectorArange);
   Print("vector X = \n",X);
//--- Berechnung der Quadratwurzel für jeden Wert des Vektors X
   X=MathSqrt(X);
   Print("MathSqrt(X) = \n",X);

//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

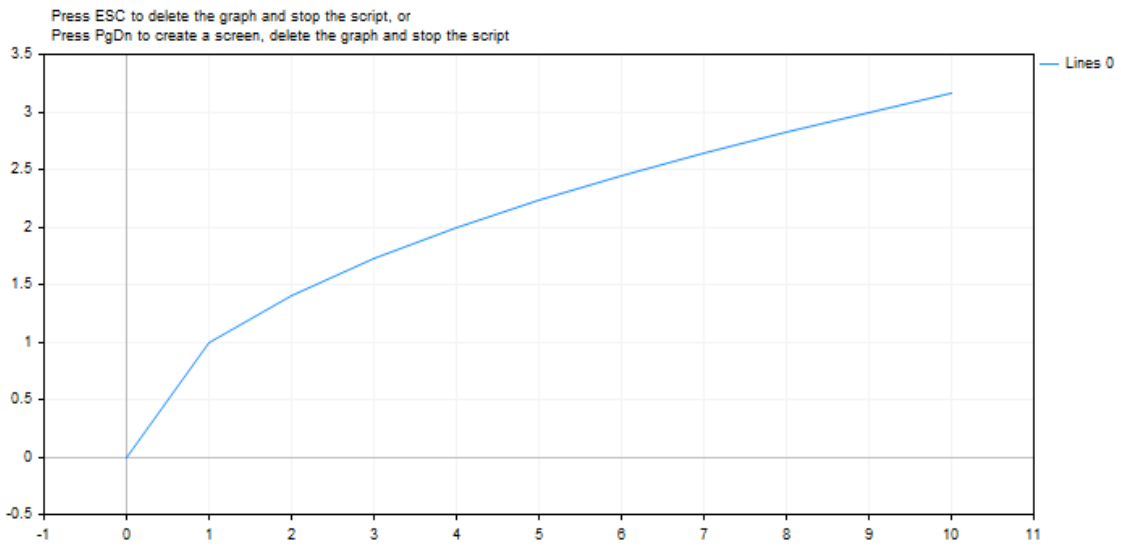
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
   /*
  Ergebnis:
   vector X =
   [0,1,2,3,4,5,6,7,8,9,10]
   MathSqrt(X) =
   [0,1,1.414213562373095,1.732050807568877,2,2.23606797749979,2.449489742783178,2.645
   */
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQLI
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathSrand

Stellt Anfangsposition für Generieren einer Folge der pseudozufälligen Ganzzahlen.

```
void  MathSrand(
   int  seed      // initialisierende Zahl
   );
```

Parameter

seed

[in] Anfangszahl für die Reihe der zufälligen Zahlen.

Rückgabewert

Keinen Rückgabewert.

Hinweis

Funktion MathRand() wird für Erzeugen einer Folge von Pseudo-Zufallszahlen verwendet. Aufruf von MathSrand() mit einer bestimmten Initialisierungszahl ermöglicht es, immer die gleiche Folge von Pseudozufallszahlen zu erhalten.

Zum garantierten Erhalt der nicht wiederkehrenden Abfolge verwenden Sie den Aufruf von MathSrand(GetTickCount()), da der Wert von GetTickCount() sich ab dem Zeitpunkt des Starts des Betriebssystems erhöht und nicht innerhalb von 49 Tage wiederholt, bis der eingebaute Zähler von Millisekunden überläuft. Verwenden von MathSrand(TimeCurrent()) ist nicht geeignet, weil die Funktion TimeCurrent() gibt die Zeit der letzte Tick zurück, die für eine lange Zeit unverändert sein kann, zum Beispiel am Wochenende.

Die Initialisierung des Zufallsgenerators mit MathSrand() für Indikatoren und Expert Advisors wird am besten in der OnInit()-Handler getan; es spart Ihnen den folgenden mehrere Neustarts der Generator in OnTick() und OnCalculate()..

Anstelle der Funktion MathSrand(), srand()-Funktion verwendet werden kann.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator zeigt den zentralen Grenzwertsatz, der besagt:"
#property description "Die Summe von einer ausreichend großen Anzahl von schwach abhär
#property description "mit annähernd gleicher Größe (keiner der Summanden dominiert,"
#property description "oder macht einen entscheidenden Beitrag zu der Summe), hat eine

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- Eigenschaften der graphischen Konstruktion
#property indicator_label1  "Label"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrRoyalBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  5
//--- Variable input
input int      sample_number=10;
//--- Ein Indikator-Puffer für die Erstellung der Verteilung
double         LabelBuffer[];
//--- Zähler von Ticks
double         ticks_counter;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- Bindung des Indikator-Puffers mit dem Array
   SetIndexBuffer(0,LabelBuffer,INDICATOR_DATA);
//--- Den Indikator-Puffer aus der Gegenwart in die Vergangenheit umwenden
   ArraySetAsSeries(LabelBuffer,true);
//--- Den Zufallszahlengenerator initialisieren
   MathSrand(GetTickCount());
//--- Den Zähler von ticks initialisieren
   ticks_counter=0;
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Mit dem Null-Zähler setzen wir den Indikator-Puffer zurück
   if(ticks_counter==0) ArrayInitialize(LabelBuffer,0);
//--- Den Zähler erhöhen
   ticks_counter++;
//--- Wir sollten in regelmäßigen Abständen den Zähler von Ticks zurücksetzen, um die
   if(ticks_counter>100)
     {
      Print("Wir haben die Indikator-Werte zurückgesetzt, beginnen wir Füllen von den
      ticks_counter=0;
     }
//--- Bekommen Wir einen Auszug von Zufallswerten als die Summe der drei Zahlen von 0
   for(int i=0;i<sample_number;i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Berechnung des Index der Zelle, wo die Zufallszahl fällt als die Summe der
      int rand_index=0;
      //--- Bekommen wir drei Zufallszahlen von 0 bis 7
      for(int k=0;k<3;k++)
        {
         //--- Ein Rest der Division durch 7 wird ein Wert von 0 bis 6 zurückgeben
         rand_index+=MathRand()%7;
        }
      //--- Erhöhen wir den Wert in der Zelle Nummer rand_index um 1
      LabelBuffer[rand_index]++;
     }
//--- Verlassen wir den Handler OnCalculate()
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

MathTan

Gibt Tangens der Zahl zurück.

```
double  MathTan(
   double  rad      // Argument in Radiant
   );
```

Parameter

rad

[in] Winkel in Radiant.

Rückgabewert

Tangens der Zahl rad. Wenn rad höher als oder gleich 263 ist oder weniger als oder gleich -263 ist, kommt es zu einem Signifikanzverlust des Ergebnisses und die Funktion gibt eine unbestimmte Zahl zurück.

Hinweis

Statt der Funktion MathTan() kann die Funktion tan() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=-263;
//--- Ermittlung von 101 Werten von -263 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des Tangens für jeden Wert des Vektors X
   vector Y=MathTan(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);
```

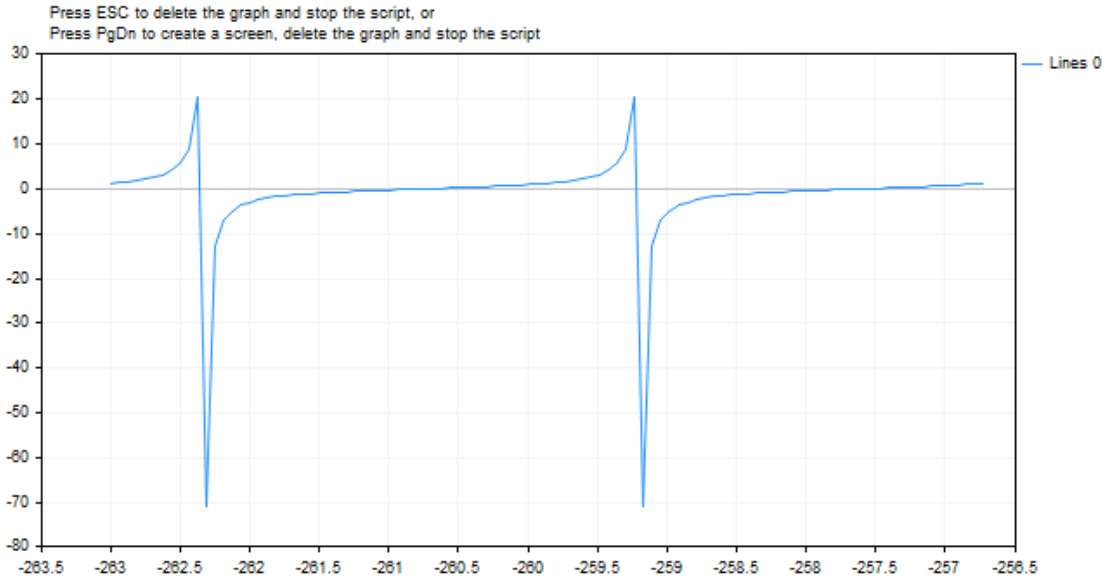
© 2000-2025, MetaQuotes Ltd.

```
//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathIsValidNumber

Prueft die Richtigkeit der Realzahl

```
bool  MathIsValidNumber(
   double  number      // die zu gepruefte Zahl
   );
```

Parameter

number

[in] Gepruefte Zahl.

Rückgabewert

Gibt true zurück, wenn die gepruefte Zahl eine zulaessige Realzahl ist. Wenn die gepruefte Zahl plus oder minus Infinitaet "Nichtzahl " (NaN - not a number) ist, gibt die Funktion false zurück.

Beispiel:

```
   double abnormal=MathArcsin(2.0);
   if(!MathIsValidNumber(abnormal)) Print("Achtung! MathArcsin(2.0) = ",abnormal);
```

Sehen Sie auch

Realtypen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathExpm1

Gibt den Wert von MathExp(x)-1 zurück.

```
double  MathExpm1(
   double  value      // Exponent für die Zahl e
   );
```

Parameter

value

[in] der Exponent für e.

Rückgabewert

Eine Zahl vom Typ double. Beim Bereichsüberschreitung gibt die Funktion INF (unendlich) zurück, bei Bereichsunterschreitung gibt MathExpm1 0 zurück.

Hinweis

Wenn *x* nahe bei Null liegt, gibt die Funktion MathExpm1(x) viel genauere Werte als MathExp(x)-1 zurück.

Statt der Funktion MathExpm1() kann die Funktion expm1() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- berechne 9 Werte von 0 bis 8 mit Schritt 1
   vector X(9,VectorArange);
   Print("vector X = \n",X);
//--- Berechnung von ("e" (Eulersche Zahl) hoch jedem Wert aus dem Vektor X) - 1
   X=MathExpm1(X);
   Print("MathExpm1(X) = \n",X);

//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
```

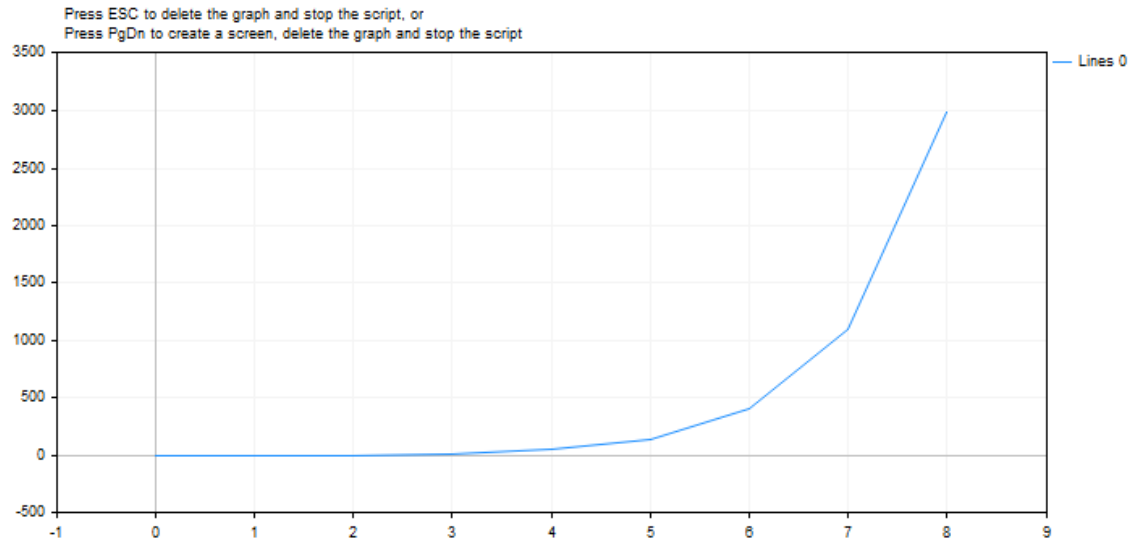
© 2000-2025, MetaQuotes Ltd.

```
   CurvePlot(y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRIT
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathLog1p

Gibt den Wert des Ausdrucks MathLog(1+x) zurück.

```
double  MathLog1p(
   double  value      // Zahl für die Berechnung des Logarithmus
   );
```

Parameter

value

[in] Wert, dessen Logarithmus berechnet werden soll.

Rückgabewert

Der natürliche Logarithmus von (Wert+1) im Erfolgsfall. Wenn value < -1, gibt die Funktion NaN (unbestimmt, not a number) zurück. Wenn value = -1, gibt die Funktion INF (unendlich) zurück.

Hinweis

Wenn der x-Wert nahe bei Null ist, gibt die Funktion MathLog1p(x) viel genauere Werte als MathLog(1+x) zurück.

Statt der Funktion MathLog1p() kann die Funktion log1p() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- berechne 9 Werte von 0 bis 8 mit Schritt 1
   vector X(9,VectorArange);
   Print("vector X = \n",X);
//--- berechne für jeden Vektorwert den Logarithmus für den Ausdruck (1 + X-Vektorwert
   X=MathLog1p(X);
   Print("MathLog1p(X) = \n",X);

//--- Übertragen der berechneten Werte aus dem Vektor in das Array
   double y_array[];
   X.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
```

© 2000-2025, MetaQuotes Ltd.

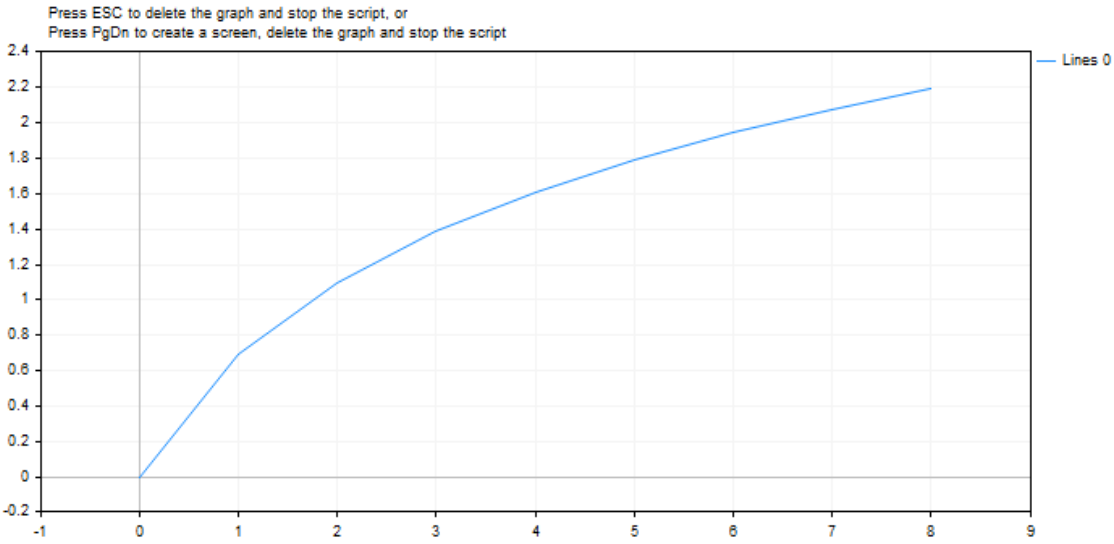
```
   CurvePlot(y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
   /*
  Ergebnis:
   vector X =
   [0,1,2,3,4,5,6,7,8]
   MathLog1p(X) =
   [0,0.6931471805599453,1.09861228866811,1.386294361119891,1.6094379124341,1.7917594
   */
  }
//+------------------------------------------------------------------+
//| Füllen eines Vektors mit 'value' in Schritten von 'step'         |
//+------------------------------------------------------------------+
template<typename T>
void VectorArange(vector<T> &vec,T value=0.0,T step=1.0)
  {
   for(ulong i=0; i<vec.Size(); i++,value+=step)
      vec[i]=value;
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:

© 2000-2025, MetaQuotes Ltd.



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathArccosh

Gibt den Wert des hyperbolischen Arkuskosinus zurück.

```
double  MathArccosh(
   double  value     // 1 <= value < ∞
   );
```

Parameter

value

[in] Der Wert value, dessen hyperbolischer Arkuskosinus berechnet werden soll.

Rückgabewert

Hyperbolischer Arkuskosinus. Wenn value kleiner als +1 ist, gibt die Funktion NaN (undefiniert, not a number) zurück.

Hinweis

Statt der Funktion MathArccosh() kann die Funktion acosh() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/10);
   delta[0]=1;
//--- Ermittlung von 101 Werten von 1 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des hyperbolischen Arkuskosinus für jeden Wert des Vektors X
   vector Y=MathArccosh(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);
```

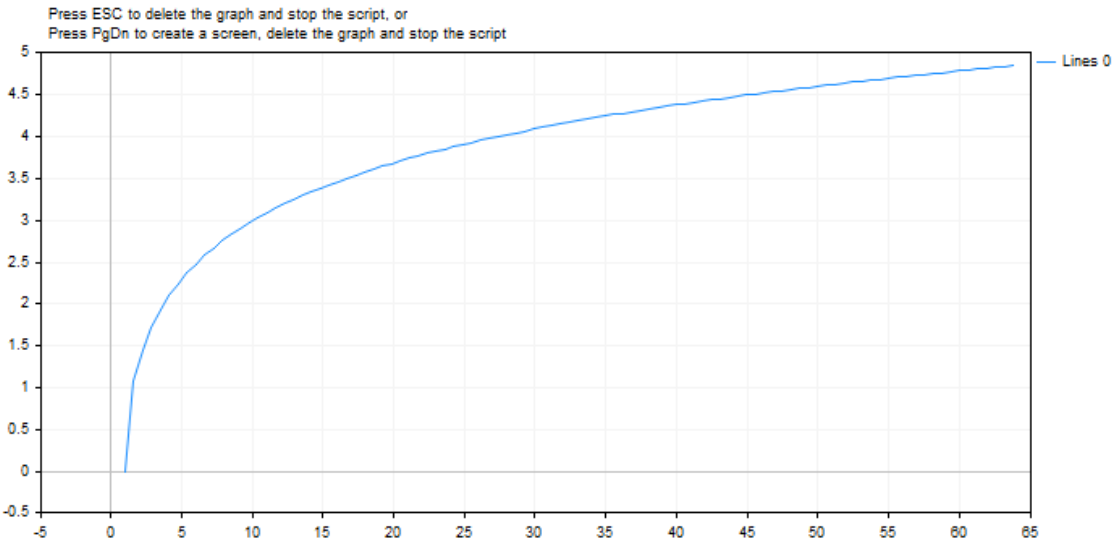
© 2000-2025, MetaQuotes Ltd.

```
//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Screenshot erstellen und speichern des Bilds in einer Datei      |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRIT
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathArcsinh

Gibt den Wert des hyperbolischen Arkussinus zurück.

```
double  MathArcsinh(
   double  value      // -∞ < value < +∞
   );
```

Parameter

val

[in] Der Wert value, dessen Arkussinus berechnet werden muss.

Rückgabewert

Hyperbolischer Arkussinus einer Zahl.

Hinweis

Statt der Funktion MathArcsinh() kann die Funktion asinh() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/10);
   delta[0]=-31;
//--- Ermittlung von 101 Werten von -31 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des hyperbolischen Arkuskosinus für jeden Wert des Vektors X
   vector Y=MathArcsinh(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

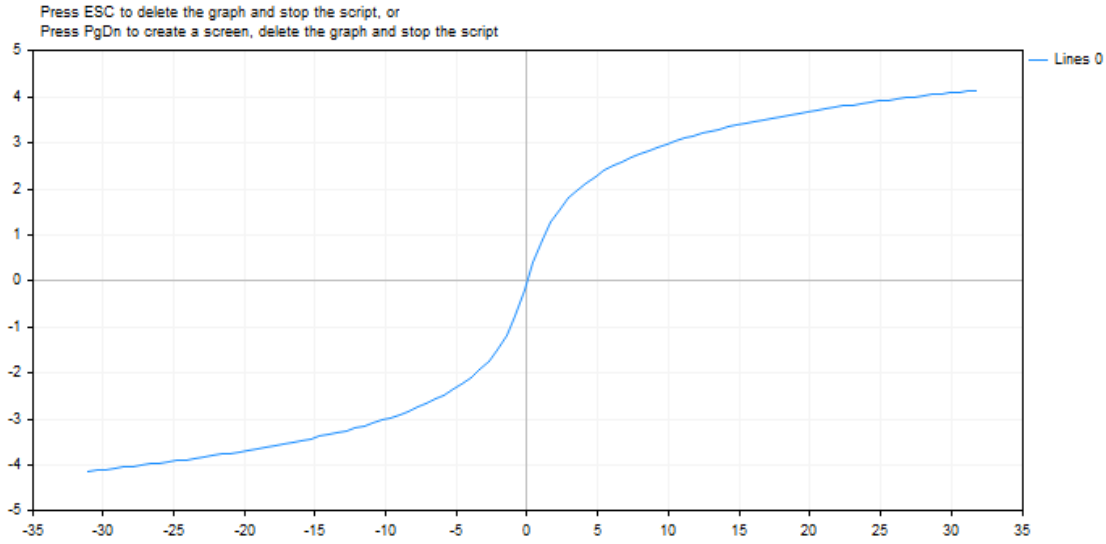
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathArctanh

Gibt den Wert des hyperbolischen Arkustangens zurück.

```
double  MathArctanh(
   double  value      // Wert im Bereich -1 < value < 1
   );
```

Parameter

value

[in] Eine Zahl im Bereich -1 < Wert < 1, die einen Tangens darstellt.

Rückgabewert

Hyperbolischer Arkustangens einer Zahl.

Hinweis

Statt der Funktion MathArctanh() kann die Funktion atanh() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(34,2*M_PI/100);
   delta[0]=-1;
//--- Ermittlung von 34 Werten von -1 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des hyperbolischen Arkustangens für jeden Wert des Vektors X
   vector Y=MathArctanh(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

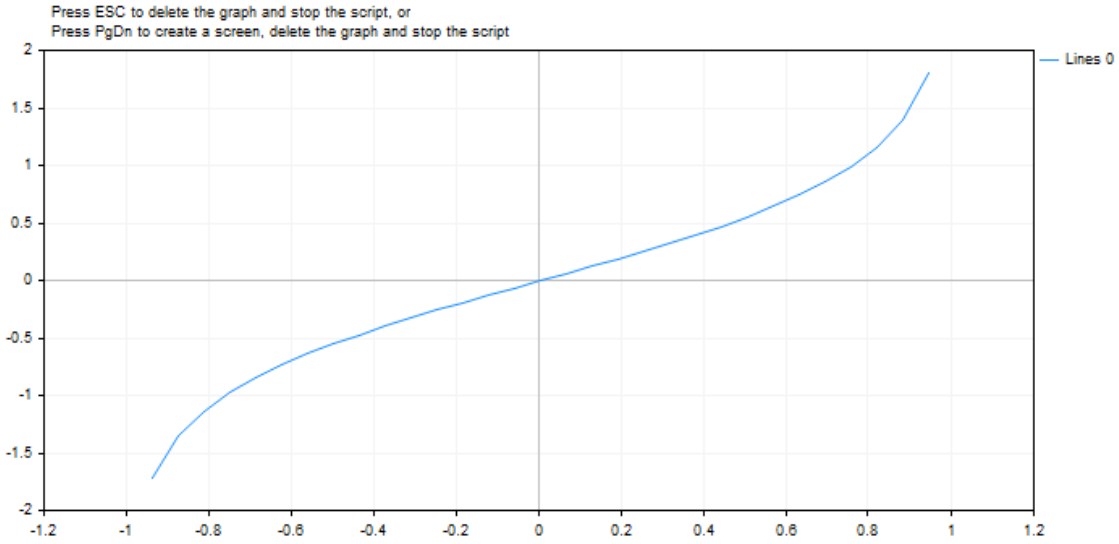
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

MathCosh

Gibt den hyperbolischen Kosinus einer Zahl zurück.

```
double  MathCosh(
   double  value      // die Zahl
   );
```

Parameter

value

[in] Der Wert.

Rückgabewert

Hyperbolischer Kosinus einer Zahl, der zurückgegebene Wert liegt im Bereich von +1 bis plus unendlich.

Hinweis

Statt der Funktion MathCosh() kann die Funktion cosh() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des hyperbolischen Kosinus für jeden Wert des Vektors X
   vector Y=MathCosh(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);
```

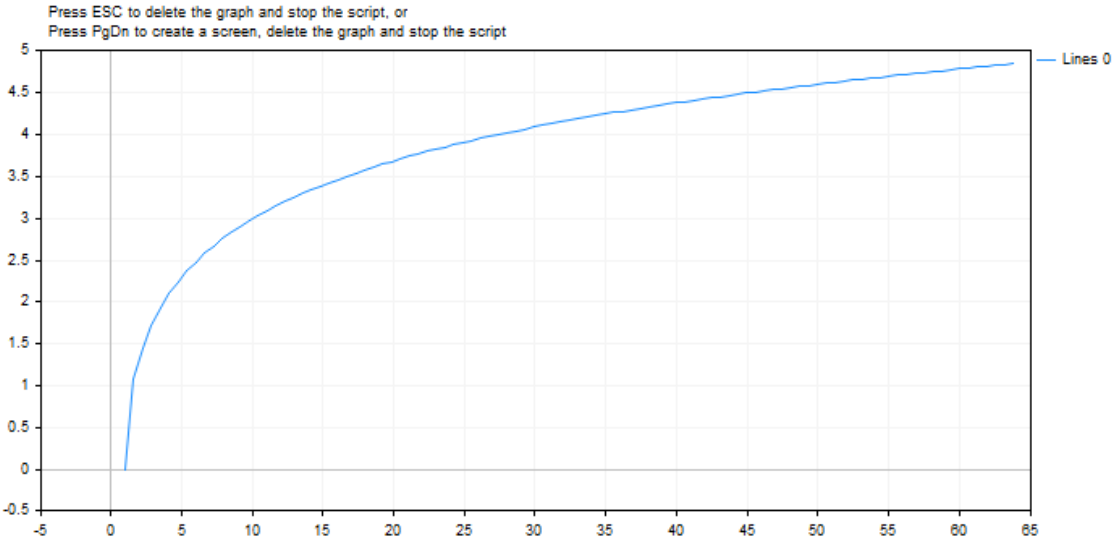
© 2000-2025, MetaQuotes Ltd.

```
//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRIT
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

MathSinh

Gibt den hyperbolischen Sinus einer Zahl zurück.

```
double  MathSinh(
   double  value      // die Zahl
   );
```

Parameter

value

[in] Der Wert.

Rückgabewert

Hyperbolischer Sinus einer Zahl.

Hinweis

Statt der Funktion MathSinh()kann die Funktion sinh() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des hyperbolischen Sinus für jeden Wert des Vektors X
   vector Y=MathSinh(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

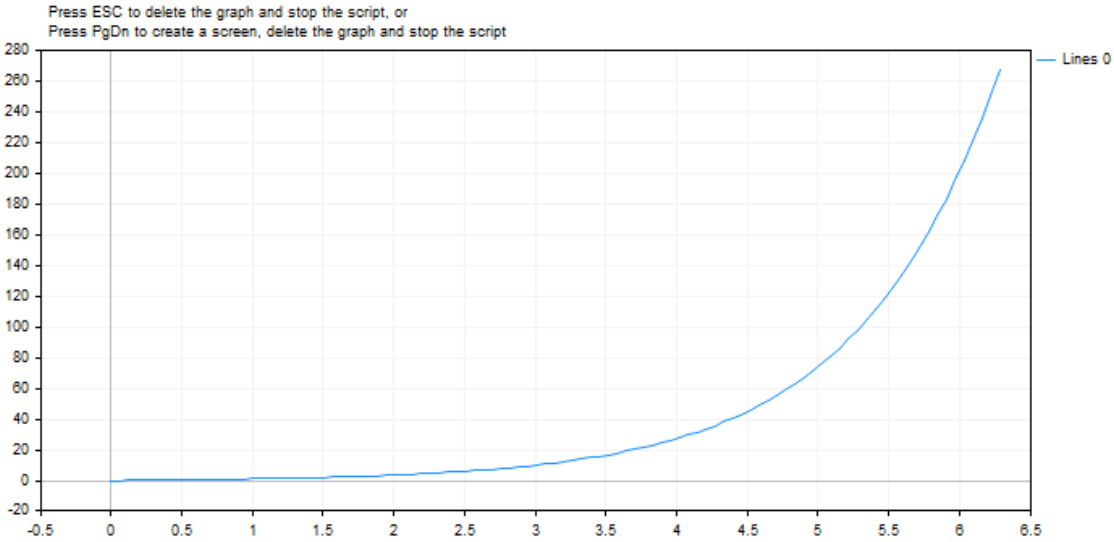
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



© 2000-2025, MetaQuotes Ltd.

MathTanh

Gibt den hyperbolischen Tangens einer Zahl zurück.

```
double  MathTanh(
   double  value      // die Zahl
   );
```

Parameter

value

[in] Der Wert.

Rückgabewert

Der hyperbolische Tangens einer Zahl, der zurückgegebene Wert liegt im Bereich von -1 bis +1.

Hinweis

Statt der Funktion MathTanh() kann die Funktion tanh() verwendet werden.

Beispiel:

```
#define GRAPH_WIDTH  750
#define GRAPH_HEIGHT 350

#include <Graphics\Graphic.mqh>

CGraphic ExtGraph;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   vector delta=vector::Full(101,2*M_PI/100);
   delta[0]=0;
//--- Ermittlung von 101 Werten von 0 bis 2π in der Schrittweite delta
   vector X=delta.CumSum();
//--- Berechnung des hyperbolischen Tangens für jeden Wert des Vektors X
   vector Y=MathTanh(X);

//--- Übertragen der berechneten Werte von Vektoren in Arrays
   double x_array[],y_array[];
   X.Swap(x_array);
   Y.Swap(y_array);

//--- Zeichne die Kurve der berechneten Vektorwerte
   CurvePlot(x_array,y_array,clrDodgerBlue);

//--- warte auf das Drücken der Escape- oder PgDn-Taste, um das Diagramm zu löschen (e
```

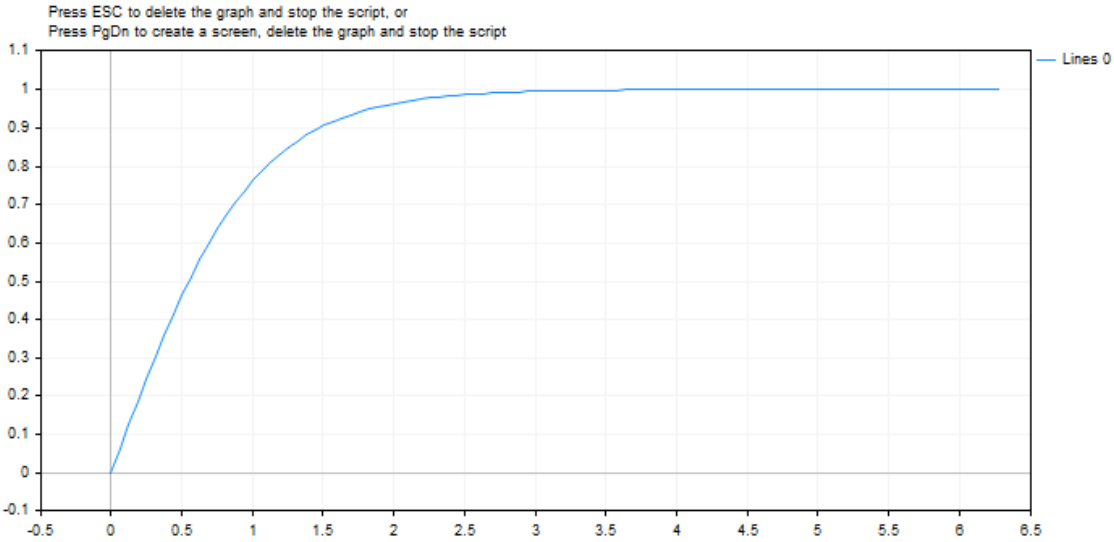
© 2000-2025, MetaQuotes Ltd.

```
   while(!IsStopped())
     {
      if(StopKeyPressed())
         break;
      Sleep(16);
     }

//--- aufräumen
   ExtGraph.Destroy();
  }
//+------------------------------------------------------------------+
//| Wenn ESC gedrückt wird, wird 'true' zurückgeben,                 |
//| bei PgDn entsteht ein Screenshot und 'true' wird zurückgegeben   |
//| Andernfalls wird 'false' zurückgegeben                           |
//+------------------------------------------------------------------+
bool StopKeyPressed()
  {
//--- wenn ESC gedrückt wird, wird 'true' zurückgeben
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)!=0)
      return(true);
//--- wird PgDn gedrückt und ein Bildschirmfoto des Graphen erfolgreich aufgenommen, w
   if(TerminalInfoInteger(TERMINAL_KEYSTATE_PAGEDOWN)!=0 && MakeAndSaveScreenshot(MQL
      return(true);
//--- gib 'false' zurück
   return(false);
  }
//+------------------------------------------------------------------+
//| Erstellen eines Graph-Objekts und Zeichnen einer Kurve           |
//+------------------------------------------------------------------+
void CurvePlot(double &x_array[], double &y_array[], const color colour)
  {
   ExtGraph.Create(ChartID(), "Graphic", 0, 0, 0, GRAPH_WIDTH, GRAPH_HEIGHT);
   ExtGraph.CurveAdd(x_array, y_array, ColorToARGB(colour), CURVE_LINES);
   ExtGraph.IndentUp(30);
   ExtGraph.CurvePlotAll();
   string text1="Press ESC to delete the graph and stop the script, or";
   string text2="Press PgDn to create a screen, delete the graph and stop the script";
   ExtGraph.TextAdd(54, 9, text1, ColorToARGB(clrBlack));
   ExtGraph.TextAdd(54,21, text2, ColorToARGB(clrBlack));
   ExtGraph.Update();
  }
//+------------------------------------------------------------------+
//| Bildschirmfoto machen und speichern des Bildes in einer Datei    |
//+------------------------------------------------------------------+
bool MakeAndSaveScreenshot(const string file_name)
  {
   string file_names[];
   ResetLastError();
   int selected=FileSelectDialog("Save Picture", NULL, "All files (*.*)|*.*", FSD_WRI
```

© 2000-2025, MetaQuotes Ltd.

```
   if(selected<1)
     {
      if(selected<0)
         PrintFormat("%s: FileSelectDialog() function returned error %d", __FUNCTION__
      return false;
     }

   bool res=false;
   if(ChartSetInteger(0,CHART_SHOW,false))
      res=ChartScreenShot(0, file_names[0], GRAPH_WIDTH, GRAPH_HEIGHT);
   ChartSetInteger(0,CHART_SHOW,true);
   return(res);
  }
```

Ergebnis:



Siehe auch

Reelle Typen (double, float)

© 2000-2025, MetaQuotes Ltd.

MathSwap

Ändert die Reihenfolge der Bytes des Typs ushort.

```
ushort  MathSwap(
   ushort  value      // Wert
   );
```

Parameter

Wert

[in] Wert, dessen Bytefolge geändert werden soll.

Rückgabewert

ushort Wert mit umgekehrter Bytefolge.

MathSwap

Ändert die Reihenfolge der Bytes des Typs uint.

```
uint  MathSwap(
   uint   value      // Wert
   );
```

Parameter

Wert

[in] Wert, dessen Bytefolge geändert werden soll.

Rückgabewert

uint Wert mit umgekehrter Bytefolge.

MathSwap

Ändert die Reihenfolge der Bytes des Typs ulong.

```
ulong  MathSwap(
   ulong  value      // Wert
   );
```

Parameter

Wert

[in] Wert, dessen Bytefolge geändert werden soll.

Rückgabewert

ulong Wert mit umgekehrter Bytefolge.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property script_show_inputs

input ulong  InpLongValue  =  1;     // Eingabe des Wertes vom Typ ulong
input uint   InpIntValue   =  2;     // Eingabe des Wertes vom Typ uint
input ushort InpShortValue =  3;     // Eingabe des Wertes vom Typ ushort

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {

//--- Drucken der von MathSwap() eingegebenen und umgewandelten Werte in dezimaler und
   Print(ValueDescription(InpLongValue));
   Print(ValueDescription(InpIntValue));
   Print(ValueDescription(InpShortValue));
   /*
  Ergebnis:
   ulong value: 1
   ulong value: 72057594037927936 using MathSwap()
   binary ulong value: 0000000000000000000000000000000000000000000000000000000000000000
   binary ulong value: 0000000100000000000000000000000000000000000000000000000000000000

   uint value: 2
   uint value: 33554432 using MathSwap()
   binary uint value: 00000000000000000000000000000010
   binary uint value: 00000010000000000000000000000000 using MathSwap()

   ushort value: 3
   ushort value: 768 using MathSwap()
   binary ushort value: 0000000000000011
   binary ushort value: 0000001100000000 using MathSwap()
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe des Textes zur Beschreibung der Variablenwerte          |
//+------------------------------------------------------------------+
template <typename T>
string ValueDescription(T x)
  {
   int    num_bits   = sizeof(T)*8;
   string type_name  = typename(T);
   string bin_x      = NumberToBinaryString(x);
   string bin_swap_x = NumberToBinaryString(MathSwap(x));
   return(StringFormat("%s value: %lld\n%s value: %lld using MathSwap()\nbinary %s va
  }
//+------------------------------------------------------------------+
//| Rückgabe der binären Darstellung einer Zahl als Zeichenkette     |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
template <typename T>
string NumberToBinaryString(T x)
  {
   string res  = "";
   int    i    = -1;
   uchar  size = sizeof(T)*8-1;
   ulong  mask = (ulong)1<<size;
   while(!((x<<++i)& mask));
   for(; i <=size; i++)
      res += !((x<<i)& mask) ? "0" : "1";
   return res;
  }
```

Siehe auch

Netzwerkfunktionen, SocketRead, SocketSend, SocketTlsRead, SocketTlsReadAvailable, SocketTlsSend

© 2000-2025, MetaQuotes Ltd.

Stringfunktionen

Gruppe der Funktionen für Arbeit mit Daten des Typs string.

Funktion	Massnahme
StringAdd	Fuegt ein String in den Platz der angegebenen Unterzeile
StringBufferLen	Gibt die Größe des Puffers zurück, der für den String verteilt wurde
StringCompare	Vergleicht zwei Strings und gibt 1 zurück, wenn der erste String größer als der zweite ist, 0 - wenn sie gleich sind; -1 (minus eins) - wenn der erste String kleiner als der zweite ist
StringConcatenate	Formiert einen String aus übertragenen Parametern
StringFill	Fuellt den angegebenen String durch die gewaehlten Symbole aus
StringFind	Suche des Teilstrings im String
StringGetCharacter	Gibt den Wert des Symboles, das in der angegebenen Position des Strings liegt
StringInit	Initialisiert den String durch die gewaehlten Symbole und stellt die angegebene Stringgroesse bereit
StringLen	Gibt die Zahl der Symbole im String zurück
StringSetLength	StringSetLength legt die Anzahl der Buchstaben einer Zeichenkette fest
StringReplace	Ersetzt alle Substrings im String mit einer bestimmten Abfolge von Symbole
StringReserve	Reserviert die angegebene Größe in dem Puffer einer Zeichenkette im Speicher
StringSetCharacter	Gibt die Kopie des Strings mit dem veränderten Wert des Symboles in der angegebenen Position
StringSplit	Bekommt Teilstrings durch einen angegebenen Trennzeichen aus dem angegebenen String, gibt die Anzahl der Teilstrings zurück
StringSubstr	Extrahiert die Substring aus dem Textstring, die mit der angegebenen Position beginnt
StringToLower	Wandelt alle Symbole des angegebenen Strings in kleine Symbole um
StringToUpper	Wandelt alle Symbole des angegebenen Strings in grosse Symbole um
StringTrimLeft	Entfernt Zeilenvorschub, Spaces und Tabs am Anfang des Strings
StringTrimRight	Entfernt Zeilenvorschub, Spaces und Tabs am Ende des Strings

© 2000-2025, MetaQuotes Ltd.

StringAdd

Fuegt die angegebene Unterzeile zum Ende des Strings.

```
bool  StringAdd(
   string&  string_var,        // String, zu dem zugefuegt wird
   string   add_substring      // String, der zugefuegt wird.
   );
```

Parameter

string_var

[in][out] String, zu dem ein anderer ergaenzt wird.

add_substring

[in] String, der am Ende des Ausgangsstrings ergaenzt wird.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false. Für die Erhaltung des Kodes des Fehlers muss die Funktuon GetLastError() aufgerufen werden.

Beispiel:

```
void OnStart()
  {
   long length=1000000;
   string a="a",b="b",c;
//--- der erste Weg
   uint start=GetTickCount(),stop;
   long i;
   for(i=0;i<length;i++)
     {
      c=a+b;
     }
   stop=GetTickCount();
   Print("time for 'c = a + b' = ",(stop-start)," milliseconds, i = ",i);

//--- der zweite Weg
   start=GetTickCount();
   for(i=0;i<length;i++)
     {
      StringAdd(a,b);
     }
   stop=GetTickCount();
   Print("time for 'StringAdd(a,b)' = ",(stop-start)," milliseconds, i = ",i);

//--- der dritte Weg
   start=GetTickCount();
   a="a"; // initialisieren wir die Variable a erneut
   for(i=0;i<length;i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       StringConcatenate(c,a,b);
      }
   stop=GetTickCount();
   Print("time for 'StringConcatenate(c,a,b)' = ",(stop-start)," milliseconds, i = ",i
  }
```

Sehen Sie auch

StringConcatenate, StringSplit, StringSubstr

© 2000-2025, MetaQuotes Ltd.

StringBufferLen

Gibt die Puffergroesse zurück, der für den String verteilt wurde.

```
int  StringBufferLen(
   string  string_var      // Zeile
   )
```

Parameter

string_var

[in] Zeile.

Rückgabewert

Wert 0 bedeutet, dass der String konstant ist und die Puffergroesse nicht verändert werden kann. -1 bedeutet, dass der String dem Client-Terminal angehört und die Veränderung des Pufferginhalts zu unbestimmten Ergebnissen führen kann.

Beispiel:

```
void OnStart()
  {
   long length=1000;
   string a="a",b="b";
//---
   long i;
   Print("before: StringBufferLen(a) = ",StringBufferLen(a),
         "  StringLen(a) = ",StringLen(a));
   for(i=0;i<length;i++)
     {
      StringAdd(a,b);
     }
   Print("after: StringBufferLen(a) = ",StringBufferLen(a),
         "  StringLen(a) = ",StringLen(a));
  }
```

Sehen Sie auch

StringAdd, StringInit, StringLen, StringFill

© 2000-2025, MetaQuotes Ltd.

StringCompare

Vergleicht zwei Strings gibt das Ergebnis des Vergleichs als Ganzzahl zurück.

```
int  StringCompare(
   const string&  string1,                // erster String des Vergleichs
   const string&  string2,                // zweiter String des Vergleichs
   bool           case_sensitive=true     // Modus der Groß- und Kleinschreibung beir
   );
```

Parametsr

string1

[in] Der erste String.

string2

[in] Der zweite String.

case_sensitive=true

[in] Modus der Groß- und Kleinschreibung. Wenn es true its, dann "A">"a". Wenn es false ist, dann "A"="a". Der Standardwert ist true.

Rückgabewert

- -1 (minus eins), wenn string1<string2
- 0 (null), wenn string1=string2
- 1 (eins), wenn string1>string2

Hinweis

Strings werden Zeichen für Zeichen verglichen, Zeichen werden in alphabetischer Reihenfolge nach der aktuellen Codepage verglichen.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- Was mehr ist, einen Apfel oder ein Haus?
   string s1="Apple";
   string s2="home";

//--- vergleichen, Groß/Kleinschreibung beachtet wird
   int result1=StringCompare(s1,s2);
   if(result1>0) PrintFormat("Vergleichen, Groß/Kleinschreibung beachtet wird: %s > %s
   else
     {
      if(result1<0)PrintFormat("Vergleichen, Groß/Kleinschreibung beachtet wird: %s <
      else PrintFormat("Vergleichen, Groß/Kleinschreibung beachtet wird: %s = %s",s1,s
     }

//--- vergleichen, Groß/Kleinschreibung nicht beachtet wird
   int result2=StringCompare(s1,s2,false);
   if(result2>0) PrintFormat("Vergleichen, Groß/Kleinschreibung nicht beachtet wird: %
   else
     {
      if(result2<0)PrintFormat("Vergleichen, Groß/Kleinschreibung nicht beachtet wird:
      else PrintFormat("Vergleichen, Groß/Kleinschreibung nicht beachtet wird: %s = %s
     }
/* Ergebnis:
     Vergleich, wenn Groß/Kleinschreibung beachtet wird: Apple < home
     Vergleich, wenn Groß/Kleinschreibung nicht beachtet wird: Apple < home
*/
  }
```

Sehen Sie auch

Typ string, CharToString(), ShortToString(), StringToCharArray(), StringToShortArray(), StringGetCharacter(), Kodeseite Verwenden

© 2000-2025, MetaQuotes Ltd.

StringConcatenate

Formiert die Zeichenkette aus den übertragenen Elementen und gibt die Größe der formierten Zeichenkette zurück. Die Parameter können verschiedener Typen sein. Die Anzahl der Parameter darf nicht kleiner als 2 und nicht mehr als 64 sein.

```
int  StringConcatenate(
   string&  string_var,    // Zeichenkette des Formats
   void argument1          // der erste Parameter jew. von einem einfachen Typ
   void argument2          // der zweite Parameter jew. von einem einfachen Typ
   ...                     // folgende Parameter jew. von einfachen Typen
   );
```

Parameter

string_var

[out] die zusammengesetzte Zeichenkette

argumentN

[in] Alle Elemente, die durch Kommas getrennt sein müssen. Möglich sind 2 bis 63 Paramter einfachen Typs.

Rückgabewert

Gibt die Stringlänge zurück, die durch Konkatenanz von Parameter formiert wurde, die in den Typ string umgewandelt wurden. Parameter werden in die Strings nach denselben Regeln umgewandelt, wie in den Funktionen Print() und Comment().

Beispiel:

```
void OnStart()
  {
//--- Deklaration und Definition der an der Verkettung beteiligten Variablen
   string text="";
   string text1="This script shows how the StringConcatenate() function works.\n";
   string text2="This is the second line, at the end of which there is a line break co
   string text3="This is line number ";
   int    num3=3;
   string text31=", the number of which is entered into the function as a separate par
   string textN="\n";
   string text4="This is line number 4, preceded by a separate parameter with a line b
   int    length=StringConcatenate(text, text1, text2, text3, num3, text31, textN, tex
   Print(text, "\nLength of the resulting string = ", length);

   /*
  Ergebnis
   This script shows how the StringConcatenate() function works.
   This is the second line, at the end of which there is a line break control code.
   This is line number 3, the number of which is entered into the function as a separa
   This is line number 4, preceded by a separate parameter with a line break code.
   Line 5 includes a real number: 0.12345
```

© 2000-2025, MetaQuotes Ltd.

```
   Length of the resulting string = 358
   */
  }
```

Siehe auch

StringAdd, StringSplit, StringSubstr

© 2000-2025, MetaQuotes Ltd.

StringFill

Fuellt den angegebenen String mit den angegebenen Symbolen aus.

```
bool  StringFill(
   string&   string_var,       // der zu gefuellte String
   ushort    character         // Symbol, mit dem der String gefuellt wird
   );
```

Parameter

string_var

[in][out] String, der mit dem angegebenen Symbol gefuellt wird.

character

[in] Symbol, mit dem der String gefuellt wird.

Rückgabewert

Im Fall des Erfolges gibt true zurück, andernfalls false. Für Erhaltung des Fehlerkodes muss man die Funktion GetLastError() aufrufen.

Hinweis

Ausfuellung des Strings bedeutet, dass Symbole in String ohne Zwischenoperationen der Erzeugung des neuen Strings und Kopieren eingesetzt werden. Das erlaubt die Zeit der Arbeit mit dem String in der angegebenen Funktion zu reduzieren.

Beispiel:

```
void OnStart()
  {
   string str;
   StringInit(str,20,'_');
   Print("str = ",str);
   StringFill(str,0);
   Print("str = ",str,": StringBufferLen(str) = ", StringBufferLen(str));
  }
// Ergebnis
//   str = ____________________
//   str =  : StringBufferLen(str) = 20
//
```

Sehen Sie auch

StringBufferLen, StringLen, StringInit

© 2000-2025, MetaQuotes Ltd.

StringFind

Suche eines Teils in einer Zeichenkette.

```
int  StringFind(
   string  string_value,        // Zeichenkette, in der gesucht werden soll
   string  match_substring,     // was, was gesucht werden soll
   int     start_pos=0          // ab welcher Position die Suche beginnen soll
   );
```

Parameter

string_value

[in] Zeichenkette in der die Suche durchgeführt wird.

match_substring

[in] Die gesuchte Zeichenfolge.

start_pos=0

[in] Position in der Zeichenkette, ab der die Suche gestartet wird.

Rückgabewert

Gibt die Positionsnummer in der Zeichenkette zurück, ab der der gesuchte Teil beginnt, oder -1, wenn das Gesuchte nicht gefunden wurde.

Beispiel:

```
#define   RESERVE    100

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln der Basiswährung des Symbols und der Gewinnwährung
   string symbol_currency_base  =SymbolInfoString(Symbol(), SYMBOL_CURRENCY_BASE);
   string symbol_currency_profit=SymbolInfoString(Symbol(), SYMBOL_CURRENCY_PROFIT);
   PrintFormat("Symbol Currency Base: %s\nSymbol Currency Profit: %s", symbol_currency

//--- in der Schleife alle auf dem Server verfügbaren Symbole abfragen
   int total=SymbolsTotal(false), pos=-1;
   for(int i=0; i<total; i++)
     {
      //--- den Namen des nächsten Symbols abrufen
      string name=SymbolName(i, false);

      //--- nach einer Teilzeichenkette im Symbolnamen mit dem Namen der Basiswährung
      //--- wenn eine Teilzeichenkette gefunden wird, werden der Symbolname, sein Inde
      pos = StringFind(name, symbol_currency_base);
      if(pos >= 0)
```

© 2000-2025, MetaQuotes Ltd.

```
         PrintFormat("The '%s' symbol at index %u in the list contains the '%s' currer

      //--- Suche nach einer Teilzeichenkette im Symbolnamen mit dem Namen der Gewinnv
      //--- wenn eine Teilzeichenkette gefunden wird, werden der Symbolname, sein Inde
      pos = StringFind(name, symbol_currency_profit);
      if(pos >= 0)
         PrintFormat("The '%s' symbol at index %u in the list contains the '%s' currer
     }

/*
Ergebnis
 StringFind (EURUSD,D1)   Symbol Currency Base: EUR
 StringFind (EURUSD,D1)   Symbol Currency Profit: USD
 The 'EURUSD' symbol at index 0 in the list contains the 'EUR' currency. Substring p
 The 'EURUSD' symbol at index 0 in the list contains the 'USD' currency. Substring p
 The 'GBPUSD' symbol at index 1 in the list contains the 'USD' currency. Substring p
 The 'USDCHF' symbol at index 2 in the list contains the 'USD' currency. Substring p
 The 'USDJPY' symbol at index 3 in the list contains the 'USD' currency. Substring p
 The 'USDCNH' symbol at index 4 in the list contains the 'USD' currency. Substring p
 The 'USDRUB' symbol at index 5 in the list contains the 'USD' currency. Substring p
 The 'AUDUSD' symbol at index 6 in the list contains the 'USD' currency. Substring p
 The 'NZDUSD' symbol at index 7 in the list contains the 'USD' currency. Substring p
 The 'USDCAD' symbol at index 8 in the list contains the 'USD' currency. Substring p
 The 'USDSEK' symbol at index 9 in the list contains the 'USD' currency. Substring p
 The 'USDHKD' symbol at index 10 in the list contains the 'USD' currency. Substring
 The 'USDSGD' symbol at index 11 in the list contains the 'USD' currency. Substring
 The 'USDNOK' symbol at index 12 in the list contains the 'USD' currency. Substring
 The 'USDDKK' symbol at index 13 in the list contains the 'USD' currency. Substring
 The 'USDTRY' symbol at index 14 in the list contains the 'USD' currency. Substring
 The 'USDZAR' symbol at index 15 in the list contains the 'USD' currency. Substring
 The 'USDCZK' symbol at index 16 in the list contains the 'USD' currency. Substring
 The 'USDHUF' symbol at index 17 in the list contains the 'USD' currency. Substring
 The 'USDPLN' symbol at index 18 in the list contains the 'USD' currency. Substring
 The 'USDRUR' symbol at index 19 in the list contains the 'USD' currency. Substring
 The 'EURAUD' symbol at index 27 in the list contains the 'EUR' currency. Substring
 The 'EURCAD' symbol at index 28 in the list contains the 'EUR' currency. Substring
 The 'EURCHF' symbol at index 29 in the list contains the 'EUR' currency. Substring
 The 'EURCZK' symbol at index 30 in the list contains the 'EUR' currency. Substring
 The 'EURDKK' symbol at index 31 in the list contains the 'EUR' currency. Substring
 The 'EURGBP' symbol at index 32 in the list contains the 'EUR' currency. Substring
 The 'EURHKD' symbol at index 33 in the list contains the 'EUR' currency. Substring
 The 'EURHUF' symbol at index 34 in the list contains the 'EUR' currency. Substring
 The 'EURJPY' symbol at index 35 in the list contains the 'EUR' currency. Substring
 The 'EURNOK' symbol at index 36 in the list contains the 'EUR' currency. Substring
 The 'EURNZD' symbol at index 37 in the list contains the 'EUR' currency. Substring
 The 'EURPLN' symbol at index 38 in the list contains the 'EUR' currency. Substring
 The 'EURRUR' symbol at index 39 in the list contains the 'EUR' currency. Substring
 The 'EURRUB' symbol at index 40 in the list contains the 'EUR' currency. Substring
 The 'EURSEK' symbol at index 41 in the list contains the 'EUR' currency. Substring
```

© 2000-2025, MetaQuotes Ltd.

```
   The 'EURTRY' symbol at index 42 in the list contains the 'EUR' currency. Substring
   The 'EURZAR' symbol at index 43 in the list contains the 'EUR' currency. Substring
   The 'XAUUSD' symbol at index 47 in the list contains the 'USD' currency. Substring
   The 'XAUEUR' symbol at index 48 in the list contains the 'EUR' currency. Substring
   The 'XAGUSD' symbol at index 50 in the list contains the 'USD' currency. Substring
   The 'XAGEUR' symbol at index 51 in the list contains the 'EUR' currency. Substring
   The 'USDCRE' symbol at index 53 in the list contains the 'USD' currency. Substring
   The 'XPDUSD' symbol at index 65 in the list contains the 'USD' currency. Substring
   The 'XPTUSD' symbol at index 66 in the list contains the 'USD' currency. Substring
   The 'USDGEL' symbol at index 67 in the list contains the 'USD' currency. Substring
   The 'USDMXN' symbol at index 68 in the list contains the 'USD' currency. Substring
   The 'EURMXN' symbol at index 69 in the list contains the 'EUR' currency. Substring
   The 'USDCOP' symbol at index 75 in the list contains the 'USD' currency. Substring
   The 'USDARS' symbol at index 76 in the list contains the 'USD' currency. Substring
   The 'USDCLP' symbol at index 77 in the list contains the 'USD' currency. Substring
   The 'EURSGD' symbol at index 89 in the list contains the 'EUR' currency. Substring
   The 'USDILS' symbol at index 95 in the list contains the 'USD' currency. Substring
   The 'USDTHB' symbol at index 122 in the list contains the 'USD' currency. Substring
   The 'USDRMB' symbol at index 123 in the list contains the 'USD' currency. Substring
   The 'EURILS' symbol at index 126 in the list contains the 'EUR' currency. Substring
   The 'EURCNH' symbol at index 137 in the list contains the 'EUR' currency. Substring
   The 'USDBRL' symbol at index 139 in the list contains the 'USD' currency. Substring
  */
  }
```

Siehe auch

StringSubstr, StringGetCharacter, StringLen, StringLen

© 2000-2025, MetaQuotes Ltd.

StringGetCharacter

Gibt den Wert des Zeichens zurück, das sich in der angegebenen Position der Zeichenkette befindet.

```
ushort  StringGetCharacter(
   string  string_value,     // Zeichenkette
   int     pos               // Position des Zeichens in der Zeichenkette
   );
```

Parameter

string_value

[in] Zeichenkette.

pos

[in] Positionsnummer in der Zeichenkette. Kann von 0 bis StringLen(text) -1 sein.

Rückgabewert

Code des Zeichens oder 0 bei einem Fehler. </t5>

Beispiel:

```
void OnStart()
  {
//--- alle Kommentare im Chart löschen
   Comment("");
//--- eine Zeichenkette deklarieren, aus der wir den Wert des Zeichencodes ermitteln u
   string message = "The script demonstrates the operation of the StringGetCharacter()
   int    length  = StringLen(message);
//--- eine Variable vom Typ string deklarieren, zu der wir die erhaltenen Zeichen aus
   string text    = "";
//--- in der Schleife durch die Länge des Demo-Strings
   for(int i=0; i<length; i++)
     {
      //--- warte 1/10 Sekunden
      Sleep(100);
      //--- das Zeichen aus der Zeichenkette holen, das sich am Schleifenindex im Demo
      ushort char_code=StringGetCharacter(message, i);
      //--- das Zeichen der auszugebenden Zeichenfolge hinzufügen und die resultierend
      text+=ShortToString(char_code);
      Comment(text);
     }
//--- zwei Sekunden warten und dann den Kommentar aus dem Chart entfernen
   Sleep(2000);
   Comment("");

  /*
  Ergebnis: Der Demo-String erscheint Zeichen für Zeichen auf dem Bildschirm
   The script demonstrates the operation of the StringGetCharacter() function
  */
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Siehe auch

StringSetCharacter, StringBufferLen, StringLen, StringFill, StringInit, StringToCharArray, StringToShortArray

© 2000-2025, MetaQuotes Ltd.

StringInit

Initialisiert den String durch die angegebenen Symbole und gewaehrleistet die angegebene Stringgroesse.

```
bool  StringInit(
   string&    string_var,       // String für die Initialisierung
   int        new_len=0,        // Erforderliche Stringlaenge nach der Initialisierung
   ushort     character=0       // Symbol, mit dem der String gefuellt wird
   );
```

Parameter

string_var

[in][out] String, der initilaisiert oder deinitialisiert werden muss.

new_len=0

[in] Stringlaenge nach der Initialisierug. Wenn die Größe=0, wird der String deinitialisiert, d.h. Stringpuffer wird verteilt und Pufferadresse wird ausgenullt.

character=0

[in] Symbol für Ausfuellung des Strings.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false. Für dier Erhaltung des Fehlerkodes muss die Funktion GetLastError() aufgerufen werden

Bemerkung

Wenn character=0 und Größe new_len>0, wird der Stringpuffer verteilt wird und mit Nullen ausgefuellt wird. Stringgroesse wird gleich 0 sein, denn der ganze Puffer wird mit Endezeichen ausgefuellt werden.

Beispiel:

```
void OnStart()
  {
//---
   string str;
   StringInit(str,200,0);
   Print("str = ",str,": StringBufferLen(str) = ",
         StringBufferLen(str),"  StringLen(str) = ",StringLen(str));
  }
/*  Ergebnis
str = : StringBufferLen(str) = 200   StringLen(str) = 0
*/
```

Sehen Sie auch

StringBufferLen, StringLen

© 2000-2025, MetaQuotes Ltd.

StringLen

Gibt die Anzahl der Zeichen in einer Zeichenkette zurück.

```
int  StringLen(
   string  string_value      // Zeichenkette
   );
```

Parameter

string_value

[in] Zeichenkette für die Berechnung der Länge.

Rückgabewert

Anzahl der Zeichen in einer Zeichenkette ohne Null am Ende.

Beispiel:

```
void OnStart()
  {
//--- die Testzeichenfolge definieren
   string text="123456789012345";
//--- die Anzahl der Zeichen in der Zeichenkette ermitteln
   int str_len=StringLen(text);
//--- die Zeichenkette und die Anzahl der Zeichen in der Zeichenkette im Protokoll aus
   PrintFormat("The StringLen() function returned the value of %d chars in string: '%s

  /*
  Ergebnis
   The StringLen() function returned the value of 15 chars in string: '12345678901234
  */
  }
```

Siehe auch

StringBufferLen, StringTrimLeft, StringTrimRight, StringToCharArray, StringToShortArray

© 2000-2025, MetaQuotes Ltd.

StringSetLength

StringSetLength legt die Anzahl der Zeichen in einer Zeichenkette fest.

```
bool  StringSetLength(
   string&     string_var,       // Zeichenkette
   uint        new_length        // neue Länge der Zeichenkette
   );
```

Parameter

string_var

[in][out] Die Zeichenkette, deren neue Länge festgelegt werden soll.

new_capacity

[in] Gewünschte neue Länge als Zeichenanzahl. Wenn die neue Größe new_length kleiner ist als die aktuelle Länge, werden die überzähligen Zeichen gelöscht.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Funktion StringSetLength() ändert nicht die der Zeichenkette zugeordneten Puffergröße.

Beispiel:

```
void OnStart()
  {
//--- die Zeichenkette definieren
   string text="123456789012345";

//--- eine Zeichenkette und ihre Länge im Protokoll ausgeben
   PrintFormat("Before StringSetLength() the string '%s' has a size of %d characters",

//--- die Größe der Zeichenkette auf 10 Zeichen reduzieren
   StringSetLength(text, 10);

//--- die Zeichenkette und ihre durch StringSetLength() veränderte, neue Länge im Prot
   PrintFormat("After StringSetLength() the string is now '%s', and has a size of %d c

  /*
  Ergebnis
   Before StringSetLength() the string '123456789012345' has a size of 15 characters
   After StringSetLength() the string is now '1234567890', and has a size of 10 charac
  */
  }
```

Siehe auch

StringLen, StringBufferLen, StringReserve StringInit, StringSetCharacter

© 2000-2025, MetaQuotes Ltd.

StringReplace

Ersetzt alle Substrings im String mit einer bestimmten Abfolge von Symbole.

```
int  StringReplace(
   string&         str,              // String, in der Substrings ersetzt werden
   const string    find,             // gewünschter Substring
   const string    replacement       // Substring, der in die gefunden Plätze eingefüg
   );
```

Parameter

str

[in][out] Der String, in dem Sie Ersetzung machen.

find

[in] Der egwünschte Substring zu ersetzen.

replacement

[in] Substring, der in die gefunden Plätze eingefügt wird.

Rückgabewert

Anzahl der Ersetzungen im Falle des Erfolgs, im Falle des Fehlers -1. Für die Erhaltung des Fehlerkodes muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Wenn die Funktion erfolgreich ist, aber nichts ersetzt ist (Substring nicht gefunden), gibt es 0 zurück.

Der Fehler kann durch falsche *str* oder *find* Parameter sein (leer oder nicht initialisierter String, sehen sie StringInit()). Außerdem tritt der Fehler auf, wenn es nicht genug Speicher, um die Ersetzung abzuschließen, gibt.

Beispiel:

```
  string text="The quick brown fox jumped over the lazy dog.";
  int replaced=StringReplace(text,"quick","slow");
  replaced+=StringReplace(text,"brown","black");
  replaced+=StringReplace(text,"fox","bear");
  Print("Replaced: ", replaced,". Result=",text);

//  Ergebnis
//  Replaced: 3. Result=The slow black bear jumped over the lazy dog.
//
```

Sehen Sie auch

StringSetCharacter(), StringSubstr()

© 2000-2025, MetaQuotes Ltd.

StringReserve

Reserviert die angegebene Größe in dem Puffer einer Zeichenkette im Speicher.

```
bool  StringReserve(
   string&     string_var,       // Zeichenkette
   uint        new_capacity      // Puffergröße zum Speichern der Zeichenkette
   );
```

Parameter

string_var

[in][out] Zeichenkette, deren Puffer auf die angegebene Größe geändert werden soll.

new_capacity

[in] geforderte neue Puffergröße der Zeichenkette. Wenn die neue Größe new_capacity kleiner ist als die Länge der Zeichenkette, wird die Größe des Puffers nicht verändert.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - false. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Im Allgemeinen ist die Größe der Zeichenkette nicht gleich der Größe des Puffers, der für die Speicherung der Zeichenkette vorgesehen ist. Beim Erstellen einer Zeichenkette wird der entsprechende Puffer in der Regel mit einer zusätzlichen Reserve versehen. Die StringReserve() Funktion ermöglicht die Verwaltung der Puffergröße und die Angabe der optimalen Größe für zukünftige Operationen.

Im Gegensatz zu StringInit() ändert die Funktion StringReserve() den Zeichenketteninhalt nicht und weist keine Zeichen zu.

Beispiel:

```
void OnStart()
  {
   string s;
//---- Prüfen der Ausführungsgeschwindigkeit, ohne StringReserve zu verwenden.
   ulong t0=GetMicrosecondCount();
   for(int i=0; i< 1024; i++)
      s+=" "+(string)i;
   ulong msc_no_reserve=GetMicrosecondCount()-t0;
   s=NULL;
//--- Jetzt noch einmal das Gleiche mit StringReserve
   StringReserve(s,1024 * 3);
   t0=GetMicrosecondCount();
   for(int i=0; i< 1024; i++)
      s+=" "+(string)i;
   ulong msc_reserve=GetMicrosecondCount()-t0;
//--- Prüfen der Zeit
   Print("Test mit StringReserve dauerte "+(string)msc_reserve+" msc");
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("Test ohne StringReserve dauerte "+(string)msc_no_reserve+" msc");
/* Ergebnis:
    Test mit StringReserve dauerte 50 msc
    Test ohne StringReserve dauerte 121 msc
*/
  }
```

Siehe auch

StringBufferLen, StringSetLength, StringInit, StringSetCharacter

© 2000-2025, MetaQuotes Ltd.

StringSetCharacter

Gibt die Kopie des Strings mit dem veränderten Symbolwert in der angegebenen Position zurück.

```
bool  StringSetCharacter(
   string&   string_var,       // String
   int       pos,              // Position
   ushort    character         // Symbol
   );
```

Parameter

string_var

[in][out] String.

pos

[in] Symbolposition im String. Kann von 0 bis StringLen(text) sein.

character

[in] Symbolkode Unicode.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false. Für die Erhaltung des Kodes des Fehlers muss die Funktuon GetLastError() aufgerufen werden.

Hinweis

Wenn Wert pos weniger als Stringlaenge ist und der Wert des Symbolkodes = 0, wird der String reduziert (aber Puffergroesse, der für den String verteilt wurde, ist unverändert). Stringlaenge wird gleich pos.

Wenn Parameterwert *pos* ist gleich der Stringlaenge, wird das angegebene Symbol zum Stringende hinzugefuegt, und die Stringlaenge wird um 1 vergroessert.

Beispiel:

```
void OnStart()
  {
   string str="0123456789";
   Print("before: str = ",str,",StringBufferLen(str) = ",
         StringBufferLen(str),"  StringLen(str) = ",StringLen(str));
//--- Fuegen wir Nullwert in die Mitte der Zeile
   StringSetCharacter(str,6,0);
   Print("  after: str = ",str,",StringBufferLen(str) = ",
         StringBufferLen(str),"  StringLen(str) =",StringLen(str));
//--- fuegen wir Symbol zum Ende der Zeile hinzu
   int size=StringLen(str);
   StringSetCharacter(str,size,'+');
   Print("addition: str = ",str,",StringBufferLen(str) = ",
         StringBufferLen(str),"  StringLen(str) =",StringLen(str));
  }
/* Ergebnis
```

© 2000-2025, MetaQuotes Ltd.

```
   before: str = 0123456789 ,StringBufferLen(str) = 0   StringLen(str) = 10
    after: str = 012345 ,StringBufferLen(str) = 16   StringLen(str) = 6
   addition: str = 012345+ ,StringBufferLen(str) = 16   StringLen(str) = 7
*/
```

Sehen Sie auch

StringBufferLen, StringLen, StringFill, StringInit, CharToString, ShortToString, CharArrayToString, ShortArrayToString

© 2000-2025, MetaQuotes Ltd.

StringSplit

Bekommt Teilstrings durch ein angegebenes Trennzeichen aus dem angegebenen String, gibt die Anzahl der Teilstrings zurück.

```
int  StringSplit(
   const string   string_value,       // String in der Teilstrings gesucht werden
   const ushort   separator,          // Trennzeichen
   string         & result[]          // Das per Referenz übergegebene Array, um die g
   );
```

Parameter

string_value

[in] Der String, auf den die Teilstrings bekommen werden soll. Der String selbst wird nicht verändert.

pos

[in] Code des Trennzeichen. Um der Code zu bekommen, können Sie die Funktion StringGetCharacter() benutzen.

result[]

[out] Ein Array der Strings, das die gefundenen Teilstrings erhält.

Rückgabewert

Die Anzahl der empfangenen Strings im Array result[]. Wenn das Trennzeichen im übergebenen String nicht gefunden wird, dann wird nur der Source-String in das Array Leitung gelegt werden.

Wenn der String string_value leer oder NULL ist, gibt die Funktion null zurück. Im Falle eines Fehlers, gibt die Funktion -1 zurück. Um die Information über Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Beispiel:

```
string to_split="_Leben_ist_schön_";// String der in Teilstrings unterteilt wird
   string sep="_";                  // Trennzeichen
   ushort u_sep;                    // Code des Trennzeichen
   string result[];                 // Array um Teilstrings zu bekommen
   //--- Code des Trennzeichen erhalten
   u_sep=StringGetCharacter(sep,0);
   //--- Den String in Teilstrings unterteilen
   int k=StringSplit(to_split,u_sep,result);
   //--- Kommentar anzeigen
   PrintFormat("Empfangene Teilstrings: %d. Benutzter Trennzeichen '%s' mit Code %d",k
   //--- Jetzt anzeigen alle Teilstrings
   if(k>0)
     {
      for(int i=0;i<k;i++)
        {
         PrintFormat("result[%d]=\"%s\"",i,result[i]);
        }
     }
```

Sehen Sie auch

StringReplace(), StringSubstr(), StringConcatenate()

© 2000-2025, MetaQuotes Ltd.

StringSubstr

Extrahiert den Teil aus der Zeichenkette, der an der angegebenen Position beginnt.

```
string  StringSubstr(
   string  string_value,      // Zeichenkette
   int     start_pos,         // Startposition
   int     length=-1          // Länge der extrahierten Zeichenkette
   );
```

Parameter

string_value

[in] Zeichenkette, aus der ein Teil extrahiert werden soll.

start_pos

[in] Anfangsposition des Teils. Kann von 0 bis StringLen(text) -1 sein.

length=-1

[in] Die Länge des zu extrahierenden Teils. Wenn der Wert des Parameters -1 ist oder nicht angegeben wurde, wird der Teilstring von der angegebenen Position bis zum Ende der Zeichenkette extrahiert.

Rückgabewert

Falls möglich wird eine Kopie des extrahierten Teilstrings zurückgegeben, andernfalls eine leere Zeichenkette.

Beispiel:

```
void OnStart()
  {
//--- den Namen des aktuellen Symbols abrufen
   string name = Symbol();

//--- das Basissymbol und die Gewinnwährung ermitteln
   string base   = StringSubstr(name, 0, 3);
   string quoted = StringSubstr(name, 3, 3);

//--- die erhaltenen Symbolwährungen im Protokoll ausgeben
   PrintFormat("Symbol: %s. Currency base: %s, currency quoted: %s", name, base, quote

  /*
  Ergebnis
   Symbol: EURUSD. Currency base: EUR, currency quoted: USD
  */
  }
```

Siehe auch

StringSplit, StringFind, StringGetCharacter

© 2000-2025, MetaQuotes Ltd.

StringToLower

Wandelt alle Zeichen der angegebenen Zeichenkette in kleine Buchstaben um.

```
bool  StringToLower(
   string&  string_var      // zu verarbeitende Zeichenkette
   );
```

Parameter

string_var

[in][out] Zeichenkette.

Rückgabewert

Im Erfolgsfall wird true zurückgegeben, anderenfalls false. </t5>

Beispiel:

```
void OnStart()
  {
//--- die Quellzeichenkette in Großbuchstaben definieren
   string text=" - THIS STRING, WRITTEN IN UPPERCASE, MUST BE WRITTEN IN LOWERCASE";
//--- die Quellzeichenkette im Protokoll ausgeben
   Print("Source line:\n", text);
//--- alle Zeichen der Zeichenfolge in Kleinbuchstaben umwandeln und das Ergebnis im P
   if(StringToLower(text))
      Print("The original string after using the StringToLower() function:\n", text);

  /*
  Ergebnis
   Source line:
    - THIS STRING, WRITTEN IN UPPERCASE, MUST BE WRITTEN IN LOWERCASE
   The original string after using the StringToLower() function:
    - this string, written in uppercase, must be written in lowercase
  */
  }
```

Siehe auch

StringToUpper, StringTrimLeft, StringTrimRight

© 2000-2025, MetaQuotes Ltd.

StringToUpper

Glyphosat

```
bool  StringToUpper(
   string&  string_var      // zu verarbeitende Zeichenkette
   );
```

Parameter

string_var

Rückgabewert

Im Erfolgsfall wird true zurückgegeben, anderenfalls false. Zum Erhalt des Fehlerkodes muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

```
void OnStart()
  {
//--- die Quellzeichenkette in Kleinbuchstaben
   string text=" - this string, written in lowercase, must be written in uppercase";
//--- die Quellzeichenkette im Protokoll ausgeben
   Print("Source line:\n", text);
//--- alle Zeichen der Zeichenfolge in Großbuchstaben umwandeln und das Ergebnis im Pr
   if(StringToUpper(text))
      Print("The original string after using the StringToUpper() function:\n", text);

  /*
  Ergebnis
   Source line:
    - this string, written in lowercase, must be written in uppercase
   The original string after using the StringToUpper() function:
    - THIS STRING, WRITTEN IN LOWERCASE, MUST BE WRITTEN IN UPPERCASE
  */
  }
```

Siehe auch

StringToLower, StringTrimLeft, StringTrimRight

© 2000-2025, MetaQuotes Ltd.

StringTrimLeft

Entfernt Zeilenvorschub, Leerzeichen und Tabs im linken Teil der Zeichenkette bis dem ersten bedeutungsvollen Zeichen. Es wird die Zeichenkette selbst modifiziert.

```
int  StringTrimLeft(
   string&  string_var      // zu verkleinernde Zeichenkette
   );
```

Parameter

string_var

[in][out] Zeichenkette, die links reduziert werden soll.

Rückgabewert

Es wird die Anzahl der entfernten Zeichen zurückgegeben.

Beispiel:

```
void OnStart()
  {
//--- Definition der Quellzeichenkette mit sechs Leerzeichen auf der linken Seite
   string text="      All spaces on the left will be removed from this string";
//--- die Quellzeichenkette im Protokoll ausgeben
   PrintFormat("Source line:\n'%s'", text);
//--- entferne alle Leerzeichen auf der linken Seite und gib die Anzahl der entfernter
   int num=StringTrimLeft(text);
   PrintFormat("The StringTrimLeft() function removed %d chars from the left side. Nov

  /*
  Ergebnis
   Source line:
   '      All spaces on the left will be removed from this string'
   The StringTrimLeft() function removed 6 chars from the left side. Now the line lool
   'All spaces on the left will be removed from this string'
  */
  }
```

Siehe auch

StringTrimRight, StringToLower, StringToUpper

© 2000-2025, MetaQuotes Ltd.

StringTrimRight

Entfernt Zeilenvorschub, Leerzeichen und Tabs am rechten Ende der Zeichenkette nach dem letzten bedeutungsvollen Buchstaben. Es wird die Zeichenkette selbst modifiziert.

```
int  StringTrimRight(
   string&  string_var      // zu verkleinernde Zeichenkette
   );
```

Parameter

string_var

[in][out] Zeichenkette, die rechts reduziert werden soll.

Rückgabewert

Es wird die Anzahl der entfernten Zeichen zurückgegeben.

Beispiel:

```
void OnStart()
  {
//--- Definition der Quellzeichenkette mit sechs Leerzeichen am rechten Ende
   string text="All spaces on the right will be removed from this string      ";
//--- die Quellzeichenkette im Protokoll ausgeben
   PrintFormat("Source line:\n'%s'", text);
//--- entferne alle Leerzeichen am rechten Ende und gib die Anzahl der entfernten Zeic
   int num=StringTrimRight(text);
   PrintFormat("The StringTrimRight() function removed %d chars from the right side. N

  /*
  Ergebnis
   Source line:
   'All spaces on the right will be removed from this string      '
   The StringTrimRight() function removed 6 chars from the right side. Now the line lo
   'All spaces on the right will be removed from this string'
  */
  }
```

Siehe auch

StringTrimLeft, StringToLower, StringToUpper

© 2000-2025, MetaQuotes Ltd.

Datum und Zeit

Gruppe von Funktionen, die Arbeit mit den Daten der Art datetime (ganze Zahl, als Sekundenzahl seit 1.Januar 1970 0:00gesehen) gewährleistet.

Um die Zähler und hochauflösende Timer zu arrangieren, verwenden Sie die Funktion GetTickCount(), die den Wert in Millisekunden gibt.

Funktion	Massnahme
TimeCurrent	Bringt den letzten bekannten Serevernamen (Zeit der letzten Kotierung) in datetime Format zurück
TimeTradeServer	Bringt den laufenden Berechnunngszeitrzum des Handelsservers zurück
TimeLocal	Bringt die lokale Computerzeit in datetime Format zurück
TimeGMT	Bringt GMT Zeit in Format datetime mit Rücksicht auf Übergang zur Sommer-/Winterzeit für lokale Zeit des Computers mit dem Client-Terminals
TimeDaylightSavings	Gibt Zeichen des überganges zur Sommer-/Winterzeit zurück
TimeGMTOffset	Gibt die laufende Differenz zwischen Zeit GMT und lokaler Zeit des Computers in Sekunden mit Rücksicht übergang zur Sommer-/Winterzeit
TimeToStruct	Wandelt einen Wert des Typs datetime in eine Variable des Typs der Strukturen MqlDateTime um
StructToTime	Wandelt Variable des Typs der Strukturen MqlDateTime in den Wert des Typs datetime um

© 2000-2025, MetaQuotes Ltd.

TimeCurrent

Gibt die letzte bekannte Serverzeit, die Zeit des Eintreffens des letzten Kurses eines der im Marktübersicht-Fenster ausgewählten Symbole zurück. Im Event Handler OnTick() gibt diese Funktion die Zeit des empfangenen verarbeiteten Ticks zurück. In den anderen Fällen (z.B. Aufruf in den Event Handlern OnInit(), OnDeinit(), OnTimer()usw) das ist die Zeit des Eintreffens des letzten Kurses für jedes Symbol, das in der Marktübersicht verfügbar ist. Die Zeit, die in der Überschrift dieses Fensters angezeigt wird. Der Zeitwert wird auf dem Handelsserver gebildet und hängt nicht von den Zeiteinstellungen auf dem PC ab. Es gibt zwei Varianten der Funktion.

Aufruf ohne Parameter

```
datetime  TimeCurrent();
```

Aufruf mit dem Parameter vom Typ MqlDateTime

```
datetime  TimeCurrent(
   MqlDateTime&  dt_struct      // Variable des Typs der Zeitstruktur
   );
```

Parameter

dt_struct

[out] Variable des Typs der Struktur MqlDateTime.

Rückgabewert

Der Wert vom Typ datetime

Hinweis

Wenn die Variable des Typs der Struktur MqlDateTime als Parameter übergeben wurde, wird sie entsprechend ausgefüllt.

Um einen hochauflösenden Zähler und Timer einzurichten, muss die Funktion GetTickCount() verwendet werden, die die Werte in Millisekunden ausgibt.

Im Strategietester wird die Zeit des letzten Kurses TimeCurrent() in Übereinstimmung mit historischen Daten modelliert.

Beispiel:

```
void OnStart()
  {
//--- Deklarieren der Variable MqlDateTime, die mit Datums-/Zeitdaten gefüllt werden s
   MqlDateTime tm={};
   datetime    time1=TimeCurrent();    // erste Form des Aufrufs: Zeitpunkt des letzte
 Datetime time2=TimeCurrent(tm); // zweite Form des Aufrufs: Zeitstempel des letzten

//--- Ausgabe des Ergebnisses des Empfangs von Datum/Uhrzeit und der Zuweisung der ent
   PrintFormat("Tick time: %s\n- Year: %u\n- Month: %02u\n- Day: %02u\n- Hour: %02u\n-
               (string)time1, tm.year, tm.mon, tm.day, tm.hour, tm.min, tm.sec, tm.day
  /*
  Ergebnis:
```

© 2000-2025, MetaQuotes Ltd.

```
   Tick time: 2024.04.18 15:40:06
   - Year: 2024
   - Month: 04
   - Day: 18
   - Hour: 15
   - Min: 40
   - Sec: 06
   - Day of Year: 108
   - Day of Week: 4 (THURSDAY)
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

TimeTradeServer

Gibt aktuelle Zeit des Handelsservers zurück. Zum Unterschied von der Funktion TimeCurrent(), wird die Zeitberechnung im Client-Terminal durchgeführt und hängt von den Zeitenstellungen im Computer des Nutzers ab. Es gibt 2 Varianten der Funktion.

Aufruf ohne Parameter

```
datetime  TimeTradeServer();
```

Aufruf mit Parameter des Typs MqlDateTime

```
datetime  TimeTradeServer(
   MqlDateTime&  dt_struct      // die Strukturvariable der Zeit
   );
```

Parameter

dt_struct

[out] die Strukturvariable der Zeit MqlDateTime.

Rückgabewert

Zeitwert vom Typs datetime

Hinweis

Wenn als Parameter die Strukturvariable des Typs MqlDateTime übergeben wurde, wird sie entsprechend ausgefüllt.

Um die Zähler und hochauflösende Timer zu arrangieren, verwenden Sie die Funktion GetTickCount(), die den Wert in Millisekunden gibt.

Im Strategietester ist TimeTradeServer() immer gleich der Serverzeit TimeCurrent().

Beispiel:

```
void OnStart()
  {
//--- Deklarieren der Variablen MqlDateTime, die mit Datums-/Zeitdaten gefüllt werden
   MqlDateTime tm={};
   datetime    time_current=TimeCurrent();                  // erste Form des Aufrufs
   datetime    time_server =TimeTradeServer(tm);            // zweite Form des Aufrufs
   int         difference  =int(time_current-time_server);  // Differenz zwischen aktu

//--- Ausgabe der Zeit des letzten Kurses und der geschätzten aktuellen Zeit des Hande
   PrintFormat("Time Current: %s\nTime Trade Server: %s\n- Year: %u\n- Month: %02u\n-
               "- Hour: %02u\n- Min: %02u\n- Sec: %02u\n- Day of Year: %03u\n- Day of
               (string)time_current, (string)time_server, tm.year, tm.mon, tm.day, tm.
               EnumToString((ENUM_DAY_OF_WEEK)tm.day_of_week), difference);
  /*
  Ergebnis:
   Time Current: 2024.04.18 16:10:14
   Time Trade Server: 2024.04.18 16:10:15
```

© 2000-2025, MetaQuotes Ltd.

```
   - Year: 2024
   - Month: 04
   - Day: 18
   - Hour: 16
   - Min: 10
   - Sec: 15
   - Day of Year: 108
   - Day of Week: 4 (THURSDAY)
   Difference between Time Current and Time Trade Server: -1
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

TimeLocal

Gibt lokale Zeit des Computers zurück, auf dem das Client-Terminal läuft. Es gibt 2 Varianten der Funktion.

Aufruf ohne Parameter

```
datetime  TimeLocal();
```

Aufruf mit einem Parameter des Typs MqlDateTime

```
datetime  TimeLocal(
   MqlDateTime&  dt_struct      // die Strukturvariable der Zeit
   );
```

Parameter

dt_struct

[out] die Strukturvariable der Zeit MqlDateTime.

Rückgabewert

Zeitwert des Typs datetime

Hinweis

Wenn Variable des Typs Strukturen MqlDateTime als Parameter übertragenen wurde, wird sie entsprechend ausgefüllt.

Um die Zähler und hochauflösende Timer zu arrangieren, verwenden Sie die Funktion GetTickCount(), die den Wert in Millisekunden gibt.

Im Strategietester ist die lokale Zeit TimeLocal() immer gleich der modellierten Serverzeit TimeCurrent().

Beispiel:

```
void OnStart()
  {
//--- Deklarieren der Variable MqlDateTime, die mit lokalen PC-Zeitdaten gefüllt werde
   MqlDateTime tm={};
   datetime    time1=TimeLocal();   // erste Form des Aufrufs für die Lokalzeit des PC
   datetime    time2=TimeLocal(tm); // zweite Form des Aufrufs für die Lokalzeit des P

//--- Ausgabe der abgerufenen Werte der Lokalzeit des PC und des Füllens der Struktur
   PrintFormat("Local time: %s\n- Year: %u\n- Month: %02u\n- Day: %02u\n- Hour: %02u\r
               (string)time1, tm.year, tm.mon, tm.day, tm.hour, tm.min, tm.sec, tm.day
  /*
  Ergebnis:
   Local time: 2024.04.18 19:44:09
   - Year: 2024
   - Month: 04
   - Day: 18
   - Hour: 19
```

© 2000-2025, MetaQuotes Ltd.

```
   - Min: 44
   - Sec: 09
   - Day of Year: 108
   - Day of Week: 4 (THURSDAY)
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

TimeGMT

Gibt die Zeit GMT zurück, die im Hinblick auf den Übergang zur Sommer- oder Winterzeit je nach der Lokalzeit des Computers, auf dem das eingesetzten Client-Terminal läuft, berechnet wird. Es gibt 2 Varianten der Funktion.

Aufruf ohne Parameter

```
datetime  TimeGMT();
```

Aufruf mit Parameter des Typs MqlDateTime

```
datetime  TimeGMT(
   MqlDateTime&  dt_struct      // die Strukturvariable der Zeit
   );
```

Parameter

dt_struct

[out] die Strukturvariable der Zeit MqlDateTime.

Rückgabewert

Zeitwert vom Typs datetime

Hinweis

Wenn als Parameter die Strukturvariable des Typs MqlDateTime übergeben wurde, wird sie entsprechend ausgefüllt.

Um die Zähler und hochauflösende Timer zu arrangieren, verwenden Sie die Funktion GetTickCount(), die den Wert in Millisekunden gibt.

Im Strategietester ist die TimeGMT() Zeit immer gleich der modellierten Serverzeit TimeTradeServer().

Beispiel:

```
void OnStart()
  {
//--- Deklaration der Variablen MqlDateTime, die mit Datums-/Zeitdaten gefüllt werden
   MqlDateTime tm={};
   datetime    time1=TimeLocal();            // erste Form des Aufrufs: Lokalzeit des
   datetime    time2=TimeGMT(tm);            // zweite Form des Aufrufs: GMT berechnet
   int         shift=int(time1-time2)/3600;  // lokaler Zeit-Offset zu GMT

//--- Ausgabe von Lokalzeit und GMT mit den Daten der gefüllten MqlDateTime-Struktur
   PrintFormat("Time Local: %s\nTime GMT: %s\n- Year: %u\n- Month: %02u\n- Day: %02u\n
               "- Hour: %02u\n- Min: %02u\n- Sec: %02u\n- Day of Year: %03u\n- Day of
               (string)time1, (string)time2, tm.year, tm.mon, tm.day, tm.hour, tm.min,
               EnumToString((ENUM_DAY_OF_WEEK)tm.day_of_week), shift);
  /*
  Ergebnis:
   Time Local: 2024.04.18 19:37:23
```

© 2000-2025, MetaQuotes Ltd.

```
   Time GMT: 2024.04.18 12:37:23
   - Year: 2024
   - Month: 04
   - Day: 18
   - Hour: 12
   - Min: 37
   - Sec: 23
   - Day of Year: 108
   - Day of Week: 4 (THURSDAY)
   Local time offset relative to GMT: +7
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

TimeDaylightSavings

Gibt die Differenz der Sommerzeit zur Normalzeit in Sekunden zurück, nachdem der übergang zur Sommerzeit stattgefunden hat. Hängt von Einstellungen im Computer des Nutzers ab.

```
int  TimeDaylightSavings();
```

Rückgabewert

Nach dem Übergang zur Winterzeit (Normalzeit), wird 0 zurückgegeben.

Beispiel:

```
void OnStart()
  {
//--- liefert die Sommerzeitanpassung in Sekunden
   int sec_dl=TimeDaylightSavings();

//--- Erstellen des Textes erstellen, der den empfangenen Wert beschreibt
   string text=(sec_dl==0 ? "Standard \"winter\" time is used" :
                StringFormat("Daylight saving time has been switched over. The correct

//--- Ausgabe einer Beschreibung der Sommerzeitumstellung in Sekunden im Protokoll
   Print(text);
   /*
   Result for "winter" time:
   Standard "winter" time is used

   Result for "summer" time:
   Daylight saving time has been switched over. The correction is -3600 seconds
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

TimeGMTOffset

Gibt die aktuelle Differenz zwischen der Zeit GMT und der lokalen Zeit des Computers in Sekunden im Hinblick auf übergang zur Winter- oder Sommerzeit zurück. Hängt von den Zeiteinstellungen im Computer des Nutzers ab.

```
int  TimeGMTOffset();
```

Rückgabewert

Wert des Typs int, der den laufenden Unterschied zwischen Zeit GMT und der lokalen Computerzeit TimeLocal() in Sekunden darstellt.

```
TimeGMTOffset() =  TimeGMT() - TimeLocal()
```

Beispiel:

```
void OnStart()
  {
//--- liefert die lokale Zeit, GMT und die Differenz zwischen GMT und lokaler Computer
   datetime time_local=TimeLocal();
   datetime time_gmt  =TimeGMT();
   int      offset    =TimeGMTOffset();

//--- Ausgabe der erhaltenen Zeit- und Offsetwerte im Protokoll
   PrintFormat("Local Time: %s, GMT Time: %s, Seconds Offset: %+d", (string)time_local
  /*
  Ergebnis:
   Local Time: 2024.04.18 19:35:52, GMT Time: 2024.04.18 12:35:52, Seconds Offset: -25
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

TimeToStruct

Konvertiert den Zeitwert des Typs datetime (Anzahl der Sekunden seit 01.01.1970) in die Strukturvariable der Zeit MqlDateTime.

```
bool  TimeToStruct(
   datetime      dt,            // Datum und Zeit
   MqlDateTime&  dt_struct      // die Strukturvariable der Zeit für die Übernahme ein
   );
```

Parameter

dt

[in] Zeitwert für die Konvertierung.

dt_struct

[out] die Strukturvariable der Zeit MqlDateTime.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Informationen über den Fehler zu erhalten, rufen Sie GetLastError() an.

Beispiel:

```
void OnStart()
  {
//--- Ermitteln der letzten bekannten Zeit des Servers, Deklarieren der Datum/Zeit-Str
   datetime    time=TimeCurrent();
   MqlDateTime tm  ={};
   if(!TimeToStruct(time,tm))
      Print("TimeToStruct() failed. Error ", GetLastError());

//--- Ausgabe der erhaltenen Serverzeit und des Ergebnisses des Füllens der MqlDateTim
   PrintFormat("Server time: %s\n- Year: %u\n- Month: %02u\n- Day: %02u\n- Hour: %02u\
               (string)time, tm.year, tm.mon, tm.day, tm.hour, tm.min, tm.sec, tm.day_
  /*
  Ergebnis:
   Server time: 2024.04.18 15:47:27
   - Year: 2024
   - Month: 04
   - Day: 18
   - Hour: 15
   - Min: 47
   - Sec: 27
   - Day of Year: 108
   - Day of Week: 4 (THURSDAY)
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

StructToTime

Wandelt die Variable des Typs Strukturen MqlDateTime in den Wert des Typs datetime um und gibt den empfangenen Wert zurück.

```
datetime  StructToTime(
   MqlDateTime&  dt_struct      // Struktur für Datum und Zeit
   );
```

Parameter

dt_struct

[in] Variable des Typs Strukturen MqlDateTime.

Rückgabewert

Wert des Typs datetime, der Anzahl der Sekunden seit 01.01.1970 enthält.

Beispiel:

```
#property script_show_inputs

input int   InpYear  =  0;    // Jahr
input int   InpMonth =  0;    // Monat
input int   InpDay   =  0;    // Tag
input int   InpHour  =  0;    // Stunde
input int   InpMin   =  0;    // Minute
input int   InpSec   =  0;    // Sekunde

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- die eingegebenen Werte anpassen und sie in Variablen schreiben
   int year =  (InpYear<1970 ? 1970 : InpYear); // Wenn das eingegebene Jahr kleiner a
   int month=  (InpMonth<1 ?  1  :  InpMonth > 12  ?  12 :  InpMonth);
   int day  =  (InpDay  <1 ?  1  :  InpDay   > 31  ?  31 :  InpDay);
   int hour =  (InpHour <0 ?  0  :  InpHour  > 23  ?  23 :  InpHour);
   int min  =  (InpMin  <0 ?  0  :  InpMin   > 59  ?  59 :  InpMin);
   int sec  =  (InpSec  <0 ?  0  :  InpSec   > 59  ?  59 :  InpSec);

//--- Ausgabe der eingegebenen Werte im Protokoll
   PrintFormat("Entered date and time: %04u.%02u.%02u %02u:%02u:%02u", InpYear, InpMor

//--- Ausgabe der angepassten eingegebenen Werte im Protokoll
   PrintFormat("Corrected date and time: %04u.%02u.%02u %02u:%02u:%02u", year, month,

//--- Schreiben der Eingabewerte in die entsprechenden Strukturfelder
   MqlDateTime time_struct={};
   time_struct.year= year;
```

© 2000-2025, MetaQuotes Ltd.

```
   time_struct.mon = month;
   time_struct.day = day;
   time_struct.hour= hour;
   time_struct.min = min;
   time_struct.sec = sec;

//--- das Datum und die Uhrzeit aus der Struktur in eine Variable vom Typ datetime umw
   datetime time = StructToTime(time_struct);

//--- die Ergebnisse der Konvertierung einer Variablen vom Struktur-Typ MqlDateTime a
   Print("Converted date and time: ",TimeToString(time,TIME_DATE|TIME_MINUTES|TIME_SEC
  /*
  Ergebnisse, wenn keine Standardwerte eingegeben werden:
   Entered date and time: 0000.00.00 00:00:00
   Corrected date and time: 1970.01.01 00:00:00
   Converted date and time: 1970.01.01 00:00:00

  Ergebnisse, wenn ein ungültiger Tag im Februar des laufenden Jahres eingegeben wird:
   Entered date and time: 2024.02.31 00:00:00
   Corrected date and time: 2024.02.31 00:00:00
   Converted date and time: 2024.03.02 00:00:00
  */
  }
```

© 2000-2025, MetaQuotes Ltd.

Information über das Konto

Funktionen, die Parameter des laufenden Kontos zurückgeben.

Funktion	Massnahme
AccountInfoDouble	Gibt den wert des Typs double der entsprechenden Eigenschaft des Kontos zurück
AccountInfoInteger	Gibt Wert des ganzzahligen Typs (bool,int oder long) der entsprechenden Eigenschaft des Kontos zurück
AccountInfoString	Gibt den Wert des Typs string der entsprechenden Eigenschaft des Kontos zurück

© 2000-2025, MetaQuotes Ltd.

AccountInfoDouble

Gibt den Wert der entsprechenden Eigenschaft des Kontos zurück.

```
double  AccountInfoDouble(
   ENUM_ACCOUNT_INFO_DOUBLE  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Wert kann einer der Werte ENUM_ACCOUNT_INFO_DOUBLE sein.

Rückgabewert

Wert des Typs double.

Beispiel:

```
void OnStart()
  {
//--- geben wir die ganze zugaengliche Information von der Funktion AccountInfoDouble
   printf("ACCOUNT_BALANCE =  %G",AccountInfoDouble(ACCOUNT_BALANCE));
   printf("ACCOUNT_CREDIT =  %G",AccountInfoDouble(ACCOUNT_CREDIT));
   printf("ACCOUNT_PROFIT =  %G",AccountInfoDouble(ACCOUNT_PROFIT));
   printf("ACCOUNT_EQUITY =  %G",AccountInfoDouble(ACCOUNT_EQUITY));
   printf("ACCOUNT_MARGIN =  %G",AccountInfoDouble(ACCOUNT_MARGIN));
   printf("ACCOUNT_MARGIN_FREE =  %G",AccountInfoDouble(ACCOUNT_MARGIN_FREE));
   printf("ACCOUNT_MARGIN_LEVEL =  %G",AccountInfoDouble(ACCOUNT_MARGIN_LEVEL));
   printf("ACCOUNT_MARGIN_SO_CALL = %G",AccountInfoDouble(ACCOUNT_MARGIN_SO_CALL));
   printf("ACCOUNT_MARGIN_SO_SO = %G",AccountInfoDouble(ACCOUNT_MARGIN_SO_SO));
  }
```

Sehen Sie auch

SymbolInfoDouble, SymbolInfoString, SymbolInfoInteger, PrintFormat

© 2000-2025, MetaQuotes Ltd.

AccountInfoInteger

Gibt den Wert der entsprechenden Eigenschaft des Kontos zurück.

```
long  AccountInfoInteger(
   ENUM_ACCOUNT_INFO_INTEGER  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Wert kann einer der Werte ENUM_ACCOUNT_INFO_INTEGER sein.

Rückgabewert

Wert des Typs long.

Hinweis

Eigenschaft muss eine der Typen bool, int oder long sein.

Beispiel:

```
void OnStart()
  {
//--- Geben wir die ganze zugaengliche Information von der Funktion AccountInfoIntege
   printf("ACCOUNT_LOGIN =  %d",AccountInfoInteger(ACCOUNT_LOGIN));
   printf("ACCOUNT_LEVERAGE =  %d",AccountInfoInteger(ACCOUNT_LEVERAGE));
   bool thisAccountTradeAllowed=AccountInfoInteger(ACCOUNT_TRADE_ALLOWED);
   bool EATradeAllowed=AccountInfoInteger(ACCOUNT_TRADE_EXPERT);
   ENUM_ACCOUNT_TRADE_MODE tradeMode=(ENUM_ACCOUNT_TRADE_MODE)AccountInfoInteger(ACCOU
   ENUM_ACCOUNT_STOPOUT_MODE stopOutMode=(ENUM_ACCOUNT_STOPOUT_MODE)AccountInfoIntege

//--- informieren wir über die Moeglichkeit, Handelsoperationen durchzuführen
   if(thisAccountTradeAllowed)
      Print("Handel für dieses Konto ist erlaubt");
   else
      Print("Handel für dieses Konto ist erlaubt!");

//--- stellen wir fest - ob es moeglich ist, in diesem Konto mit Experten zu handeln
      Print("Handel durch Ratgeber ist für dieses Konto erlaubt");
   else
      Print("Handel durch Ratgeber ist für dieses Konto verboten!");

//--- stellen wir Kontotyp fest
   switch(tradeMode)
     {
      case(ACCOUNT_TRADE_MODE_DEMO):
         Print("Das ist ein Demo-Konto");
         break;
      case(ACCOUNT_TRADE_MODE_CONTEST):
```

© 2000-2025, MetaQuotes Ltd.

```
         Print("Das ist ein Konkurenzkonto");
         break;
      default:Print("Das ist ein tatsaechliches Konto!");
     }

//--- stellen wir Mode der Leveleinstellung fest
   switch(stopOutMode)
     {
      case(ACCOUNT_STOPOUT_MODE_PERCENT):
         Print("Level StopOut wird in Prozenten angegeben");
         break;
      default:Print("Level StopOut wird im Geldausdruck angegeben");
     }
  }
```

Sehen Sie auch

Kontoinformation

© 2000-2025, MetaQuotes Ltd.

AccountInfoString

Gibt den Wert der entsprechenden Eigenschaft des Kontos zurück.

```
string  AccountInfoString(
   ENUM_ACCOUNT_INFO_STRING  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Wert kann einer der Werte ENUM_ACCOUNT_INFO_STRING sein.

Rückgabewert

Wert des Typs string.

Beispiel:

```
void OnStart()
  {
//--- geben wird die ganze zugaengliche Information von der Funktion AccountInfoStrin
   Print("Name des Brokers  = ",AccountInfoString(ACCOUNT_COMPANY));
   Print("Depositenwaehrung  = ",AccountInfoString(ACCOUNT_CURRENCY));
   Print("Name des Kunden = ",AccountInfoString(ACCOUNT_NAME));
   Print("Name des Handelsservers = ",AccountInfoString(ACCOUNT_SERVER));
 ;}
```

Sehen Sie auch

Kontoinformation

© 2000-2025, MetaQuotes Ltd.

Prüfung des Status

Funktionen, die Parameter des laufenden Zustands des Client-Terminals zurückgeben

Funktion	Massname
GetLastError	Gibt den wert des letzten Fehlers zurück
IsStopped	Gibt true zurück, wenn mql5-Programm befielt, das Programm zu beenden
UninitializeReason	Gibt den Kode des Deinitialisierungsgrundes
TerminalInfoInteger	Gibt den Wert des ganzzahligen Types der entsprechenden Eigenschaft der Umwelt des mql5-Programms zurück
TerminalInfoDouble	Gibt den Wert vom Typ double der entsprechenden Eigenschaft der Umgebung eines mql5-Programms zurück
TerminalInfoString	Gibt den Wert desw Typs string der entsprechenden Eigenschaft der Umwelt des mql5-Programms zurück
MQLInfoInteger	Gibt den Wert des ganzzahligen Typs der entsprechenden Eigenschaft des eingesetzten mql5-Programms zurück
MQLInfoString	Gibt den Wert des Typs string der entsprechenden Eigenschaft des eingesetzten mql5-Programms
Symbol	Gibt Symbolname des laufenden Charts.
Period	Gibt den Wert von Timeframe des laufenden Charts zurück
Digits	Gibt die Anzahl der Dezimalzeichen, die Genauigkeit der Berechnung des Preiswertes des laufenden Charts bestimmt
Point	Gibt den Wert des Punktes des laufenden Instrumentes in der Quotationswaehrung zurück.

© 2000-2025, MetaQuotes Ltd.

GetLastError

Gibt Inhalt der Systemvariable _LastError zurück.

```
int  GetLastError();
```

Rückgabewert

Gibt Wert des letzten Fehlers, der während der Durchführung des mql5-Programms entstanden ist, zurück.

Hinweis

Nach Aufruf der Funktion wird der Variable _LastError nicht der Wert Null (kein Fehler) zugewiesen. Um den Wert von _LastError auf Null zurückzusetzen, muss die Funktion ResetLastError() aufgerufen werden.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   MqlRates rates[1]={};   // Darstellung der Daten der aktuellen Bar hier

//--- absichtlich eine Funktion mit ungeeigneten Parametern aufrufen
   int res=CopyRates(NULL, PERIOD_CURRENT, 0, 2, rates);
   if(res!=2)
      PrintFormat("CopyRates() returned %d. LastError %d", res, GetLastError());

//--- Rücksetzen des letzten Fehlercodes vor dem Kopieren der aktuellen Balkendaten ir
   ResetLastError();
//--- wenn die Funktion nicht korrekt funktioniert hat, wird ein Fehlerwert ungleich N
   CopyRates(NULL, PERIOD_CURRENT, 0, 1, rates);
   Print("CopyRates() error ", GetLastError());

//--- Druck des Arrays der erhaltenen Werte
   ArrayPrint(rates);
  }
```

Siehe auch

Rückgabecodes des Handelsservers

© 2000-2025, MetaQuotes Ltd.

IsStopped

Prüft, ob das mql5-Programm beendet werden soll.

```
bool  IsStopped();
```

Rückgabewert

Gibt true zurück, wenn die Systemvariable _StopFlag auf einen Wert ungleich 0 gesetzt wurde. Ein Wert ungleich Null wird der Variablen _StopFlag zugewiesen, wenn sich das mql5-Programm beenden soll. In diesem Fall muss das Programm sofort beendet werden, anderenfalls wird das Programm von außen zwangsläufig in 3 Sekunden beendet.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- in einer Endlosschleife mit einem Stopp-Check
   while(!IsStopped())
     {
      //--- Anzeige der Lokalzeit des PC auf dem Chart
      Comment("Time Local: ", TimeToString(TimeLocal(), TIME_DATE|TIME_MINUTES|TIME_SE
      Sleep(16);
     }
   Print("The StopFlag is set. The program will be stopped.");

//--- aufräumen
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

UninitializeReason

Gibt Code des Grundes der Deinitialisierung zurück.

```
int  UninitializeReason();
```

Rückgabewert

Gibt den Wert der Variable _UninitReason zurück, der vor dem Aufruf der Funktion OnDeinit() zugewiesen wurde. Der Wert hängt vom Grund ab, der zur Deinitialisierung geführt hat.

Beispiel:

```
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- abrufen des Codes für den Deinitialisierungsgrund in der Variablen 'reason'
   int reason=UninitializeReason();
//--- einen Nachrichtenstring mit einem Deinitialisierungsgrund erstellen und die Nach
   string message=StringFormat("%s: Uninitialize reason code: %d (%s)",__FUNCTION__, r
   Print(message);
//--- Erfolg
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- erstelle eine Zeichenkette mit einem Deinitialisierungsgrund aus der Variable 'r
   string message=StringFormat("%s: Uninitialize reason code: %d (%s)",__FUNCTION__, r
   Print(message);
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Codes des Deinitialisierungsgrundes|
//+------------------------------------------------------------------+
string UninitializeReasonDescription(const int reason)
  {
   switch(reason)
     {
      //--- der EA beendete seine Arbeit durch den Aufruf der Funktion ExpertRemove()
      case REASON_PROGRAM :
```

© 2000-2025, MetaQuotes Ltd.

```
        return("Expert Advisor terminated its operation by calling the ExpertRemove()
      //--- das Programm wurde vom Chart entfernt
      case REASON_REMOVE :
        return("Program has been deleted from the chart");
      //--- Rekompilieren des Programms
      case REASON_RECOMPILE :
        return("Program has been recompiled");
      //--- Symbol oder Zeitrahmen des Charts wurde verändert
      case REASON_CHARTCHANGE :
        return("Symbol or chart period has been changed");
      //--- Chart wurde geschlossen
      case REASON_CHARTCLOSE :
        return("Chart has been closed");
      //--- Eingaben wurden vom Nutzer verändert
      case REASON_PARAMETERS :
        return("Input parameters have been changed by a user");
      //--- ein anderes Konto wurde aktiviert oder es gab eine erneute Verbindung zum
      case REASON_ACCOUNT :
        return("Another account has been activated or reconnection to the trade serve
      //--- ein anderes Chart-Template wurde angewendet
      case REASON_TEMPLATE :
        return("A new template has been applied");
      //--- OnInit() gab einen Wert ungleich Null zurück
      case REASON_INITFAILED :
        return("This value means that OnInit() handler has returned a nonzero value");
      //--- Terminal wurde geschlossen
      case REASON_CLOSE :
        return("Terminal has been closed");
     }

//--- Grund der Deinitialisierung ist unbekannt
   return("Unknown reason");
  }
```

© 2000-2025, MetaQuotes Ltd.

TerminalInfoInteger

Gibt Wert der entsprechenden Eigenschaft der Umgebung des mql5-Programms zurück.

```
int  TerminalInfoInteger(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einer der Enumerationswerte ENUM_TERMINAL_INFO_INTEGER sein.

Rückgabewert

Wert des Typs int.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln der Build-Nummer des laufenden Terminals und seiner Eigenschaft "64-Bi
   int  build = TerminalInfoInteger(TERMINAL_BUILD);
   bool x64   = TerminalInfoInteger(TERMINAL_X64);

//--- Ausdruck der erhaltenen Terminaldaten im Journal
   PrintFormat("MetaTrader 5 %s build %d", (x64 ? "x64" : "x32"), build);
   /*
  Ergebnis:
   MetaTrader 5 x64 build 4330
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

TerminalInfoDouble

Gibt den Wert der entsprechenden Eigenschaft der Umgebung des mql5-Programms zurück.

```
double  TerminalInfoDouble(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einer der Werte der Enumeration ENUM_TERMINAL_INFO_DOUBLE sein.

Rückgabewert

Wert vom Typ double.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln der Build-Nummer des laufenden Terminals und seiner Eigenschaft "64-Bi
   int  build = TerminalInfoInteger(TERMINAL_BUILD);
   bool x64   = TerminalInfoInteger(TERMINAL_X64);

//--- Ausdruck der erhaltenen Terminaldaten im Journal
   PrintFormat("MetaTrader 5 %s build %d", (x64 ? "x64" : "x32"), build);
   /*
  Ergebnis:
   MetaTrader 5 x64 build 4330
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

TerminalInfoString

Funktion gibt den Wert der entsprechenden Eigenschaft der Umgebung des mql5-Programms zurück. Die Eigenschaft muss vom Typ string sein.

```
string  TerminalInfoString(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einer der Enumerationswerte ENUM_TERMINAL_INFO_STRING sein.

Rückgabewert

Wert des Typs string.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- abrufen der Daten zu OS und Terminal
   string os_ver  = TerminalInfoString(TERMINAL_OS_VERSION);      // user's OS
   string name    = TerminalInfoString(TERMINAL_NAME);            // terminal name
   string path    = TerminalInfoString(TERMINAL_PATH);            // folder the termir
   string data    = TerminalInfoString(TERMINAL_DATA_PATH);       // folder for storir
   string common  = TerminalInfoString(TERMINAL_COMMONDATA_PATH); // common folder fo

//--- sende die erhaltenen Daten an das Journal
   PrintFormat("OS: %s\nTerminal: %s\n- Path: %s\n- Data path: %s\n- Common Data path:
   /*
   OS: Windows 10 build 19045
   Terminal: MetaTrader 5
   - Path: E:\MetaQuotes\MetaTrader 5
   - Data path: E:\MetaQuotes\MetaTrader 5
   - Common Data path: C:\Users\admin\AppData\Roaming\MetaQuotes\Terminal\Common
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

MQLInfoInteger

Gibt den Wert der entsprechenden Eigenschaft des eingesetzten MQL5-Programms zurück.

```
int  MQLInfoInteger(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einer der Enumerationswerte ENUM_MQL_INFO_INTEGER sein.

Rückgabewert

Wert des Typs int.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermittlung der Werte des freien und verwendeten Speichers des MQL-Programms
   int limit = MQLInfoInteger(MQL_MEMORY_LIMIT);    // maximal möglicher, dynamischer S
   int used  = MQLInfoInteger(MQL_MEMORY_USED);     // Größe des vom MQL5-Programm verw

//--- Senden der erhaltenen Werte an das Journal
   PrintFormat("Maximum possible amount of dynamic memory for MQL5 program: %d Mb\n"+
               "Memory used by MQL5 program: %d Mb", limit, used);
   /*
  Ergebnis
   Maximum possible amount of dynamic memory for MQL5 program: 8388608 Mb
   Memory used by MQL5 program: 2 Mb
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

MQLInfoString

Gibt den Wert der entsprechenden Eigenschaft des eingesetzten MQL5-Programms zurück.

```
string  MQLInfoString(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einer der Enumerationswerte ENUM_MQL_INFO_STRING sein.

Rückgabewert

Wert des Typs string.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der Daten des gestarteten Programms
   string name = MQLInfoString(MQL_PROGRAM_NAME);  // Name des gestarteten MQL5-Progra
   string path = MQLInfoString(MQL_PROGRAM_PATH);  // running program path

//--- sende die erhaltenen Daten an das Journal
   PrintFormat("Name of the running MQL program: '%s'\nPath of the running MQL progran
   /*
  Ergebnis:
   Name of the running MQL program: 'MQLInfoString'
   Path of the running MQL program: E:\MetaQuotes\MetaTrader 5\MQL5\Scripts\MQLInfoStr
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

Symbol

Gibt den Symbolname des aktuellen Charts zurück.

```
string  Symbol();
```

Rückgabewert

Wert der Systemvariable _Symbol, in der der Symbolname des aktuellen Charts aufbewahren wird.

Hinweis

Im Gegensatz zu Expert Advisors, Indikatoren und Skripten sind die Dienste nicht an ein bestimmtes Chart gebunden. Daher gibt in einem Dienst die Funktion Symbol() eine leere Zeichenkette ("") zurück.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des aktuellen Symbolnamens
   string name = Symbol();

//--- sende die erhaltenen Daten an das Journal
   PrintFormat("Current chart symbol name: '%s'", name);
   /*
  Ergebnis
   Current chart symbol name: 'EURUSD'
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

Period

Gibt den Wert Zeitrahmens des aktuellen Charts zurück.

```
ENUM_TIMEFRAMES  Period();
```

Rückgabewert

den Wert der Variable _Period, der dem Zeitrahmen des aktuellen Charts entspricht. Der Wert kann einer aus der Enumeration ENUM_TIMEFRAMES sein.

Hinweis

Im Gegensatz zu Expert Advisors, Indikatoren und Skripten sind die Dienste nicht an ein bestimmtes Chart gebunden. Daher gibt in einem Dienst die Funktion Periode() 0 zurück.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- abrufen des Werts des Zeitrahmens des aktuellen Charts und seiner Beschreibung
   ENUM_TIMEFRAMES period    = Period();
   string          timeframe = StringSubstr(EnumToString(period), 7);

//--- sende die erhaltenen Daten an das Journal
   PrintFormat("Current chart timeframe: %s\nTimeframe value: %s (%d)",
               timeframe, EnumToString(period), period);
   /*
  Ergebnis:
   Current chart timeframe: H4
   Timeframe value: PERIOD_H4 (16388)
   */
  }
```

Siehe auch

PeriodSeconds, Chartperioden, Datum und Zeit, Objektsichtbarkeit

© 2000-2025, MetaQuotes Ltd.

Digits

Gibt die Anzahl der Dezimalzeichen zurück, die die Genauigkeit der Preisberechnung des Symbols des aktuellen Charts angibt.

```
int  Digits();
```

Rückgabewert

Wert der Variable _Digits, in der die Anzahl der Dezimalzeichen aufbewahren wird, die die Genauigkeit der Preisberechnung des Symbols des aktuellen Charts angibt.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- liefert die Anzahl der Dezimalstellen für das aktuelle Diagrammsymbol
   int digits = Digits();

//--- sende die erhaltenen Daten an das Journal
   Print("Number of decimal digits for the current chart symbol: ", digits);
   /*
  Ergebnis:
   Number of decimal digits for the current chart symbol: 5
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

Point

Gibt die Punktgröße des aktuellen Symbols in der Kurswährung zurück.

```
double  Point();
```

Rückgabewert

Wert der Variable _Point, der die Punktgröße des aktuellen Symbols in der Kurswährung zugewiesen ist.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermittlung der aktuellen Punktgröße des Symbols in der Kurswährung
   double point = Point();

//--- sende die erhaltenen Daten an das Journal
   Print("Point size of the current symbol in the quote currency: ", DoubleToString(po
   /*
  Ergebnis:
   Point size of the current symbol in the quote currency: 0.00001
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

Ereignisbehandlung

Die Sprache MQL5 ermöglicht die Behandlung bestimmter vordefinierter Ereignisse. Die Funktionen zur Behandlung dieser Ereignisse sollten in einem MQL5-Programm definiert werden: Funktionsname, Rückgabetyp, eine Reihe von Parametern (falls vorhanden) und deren Typen sollten genau der Beschreibung einer Ereignisbehandlungsfunktion entsprechen.

Die Ereignisbehandlung des Client-Terminals verwendet die Rückgabe- und Parametertypen, um Funktionen zu identifizieren, die ein Ereignis verarbeiten. Wenn eine bestimmte Funktion einige Parameter oder einen Rückgabetyp hat, der nicht den nachfolgenden Beschreibungen entspricht, kann eine solche Funktion nicht zur Behandlung eines Ereignisses verwendet werden.

Funktion	Beschreibung
OnStart	Die Funktion wird aufgerufen, wenn das Ereignis Start auftritt, um Aktionen auszuführen, die im Skript aufgeführt sind.
OnInit	Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis Init eintritt, um ein gestartetes MQL5-Programm zu initialisieren.
OnDeinit	Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis Deinit auftritt, um ein laufendes MQL5-Programm zu de-initialisieren.
OnTick	Die Funktion wird von EAs aufgerufen, wenn das Ereignis NewTick eintritt, um einen neuen Tick abzuarbeiten.
OnCalculate	Die Funktion wird von Indikatoren aufgerufen, wenn das Ereignis Calculate eintritt, um Preisdatenänderungen abzuarbeiten.
OnTimer	Die Funktion wird von Indikatoren und EAs während des periodisches Ereignis Timer aufgerufen, das vom Terminal in festen Zeitintervallen erzeugt wird.
OnTrade	Die Funktion wird von EAs während des Trade Ereignisses aufgerufen, das am Ende einer Handelsoperation auf einem Handelsserver generiert wird.
OnTradeTransaktion	Die Funktion wird von EAs aufgerufen, wenn das Ereignis TradeTransaction auftritt, um die Ergebnisse einer Handelsanfrage zu verarbeiten
OnBookEvent	Die Funktion wird von EAs aufgerufen, wenn das Ereignis BookEvent auftritt, um Änderungen in der Markttiefe zu verarbeiten.
OnChartEvent	Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis ChartEvent auftritt, um Änderungen im Chart zu verarbeiten, die von einem Benutzer oder einem MQL5-Programm vorgenommen wurden.
OnTester	Die Funktion wird von EAs aufgerufen, wenn das Ereignis Tester auftritt, um notwendige Aktionen durchzuführen, nachdem ein EA mit historischen Daten getestet wurde.

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
OnTesterInit	Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterInit auftritt, um notwendige Aktionen vor der Optimierung im Strategie-Tester durchzuführen.
OnTesterDeinit	Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterDeinit nach der EA-Optimierung im Strategie-Tester auftritt.
OnTesterPass	Die Funktion wird von EAs aufgerufen, wenn TesterPass auftritt, um die Ankunft eines neuen Datenrahmens während der EA-Optimierung im Strategie-Tester zu handhaben.

Das Client-Terminal sendet eingehende Ereignisse an entsprechende offene Charts. Außerdem können Ereignisse durch Charts (chart events) oder mql5 Programme (custom events) erzeugt werden. Die Generierung von grafischen Ereignissen zur Objekterstellung/Löschung kann durch Setzen von CHART_EVENT_OBJECT_CREATE und CHART_EVENT_OBJECT_DELETE Chart-Eigenschaften aktiviert oder deaktiviert werden. Jede mql5-Anwendung und jedes Diagramm haben ihre eigene Warteschlange von Ereignissen, in der alle neu eingetroffenen Ereignisse platziert werden.

Ein Programm holt Ereignisse nur aus dem Chart, auf dem es läuft. Alle Ereignisse werden nacheinander in der Reihenfolge ihres Eingangs behandelt. Wenn die Warteschlange (Queue) bereits das Ereignis NewTick enthält oder sich dieses Ereignis in der Verarbeitungsphase befindet, wird das neue Ereignis NewTick nicht in die Warteschlange der mql5-Anwendung hinzugefügt. Wenn sich ChartEvent bereits in einer mql5-Programmwarteschlange befindet oder ein solches Ereignis behandelt wird, dann wird ein neues Ereignis dieses Typs nicht die Warteschlange gestellt. Das Timer-Ereignis wird auf die gleiche Weise verarbeitet - wenn das Ereignis Timer bereits in der Warteschlange ist oder behandelt wird, wird kein neues Timer-Ereignis in eine Warteschlange gestellt.

Ereigniswarteschlangen haben eine begrenzte, aber ausreichende Größe, so dass ein Überlauf der Warteschlange für ein korrekt entwickeltes Programm unwahrscheinlich ist. Wenn die Warteschlange überläuft, werden neue Ereignisse verworfen und nicht in die Warteschlange gestellt.

Es wird dringend empfohlen, keine Endlosschleifen zur Behandlung von Ereignissen zu verwenden. Mögliche Ausnahmen sind Skripte, die ein einzelnes Ereignis Start behandeln.

Bibliotheken behandeln keine Ereignisse.

© 2000-2025, MetaQuotes Ltd.

OnStart

Die Funktion wird aufgerufen, wenn das Ereignis Start eintritt. Diese Funktion wird in einem Skript einmal aufgerufen und ausgeführt. Es gibt zwei Versionen dieser Funktion.

Die Version gibt ein Ergebnis zurück

```
int  OnStart(void);
```

Rückgabewert

Der Rückgabewert vom Typ int wird im Journal-Tab angezeigt.

Der Eintrag "Skript Skript-Name wurde entfernt (Ergebniscode N)" wird im Journal des Terminals geschrieben, nachdem die Skriptausführung abgeschlossen ist. Hier ist N ein Wert, der von der Funktion OnStart() zurückgegeben wird.

Der Eintrag "service Dienst_Name stopped (Ergebniscode N)" wird im Journal des Terminals angelegt, nachdem sich die Ausführung eines Dienstes beendet hat. Hier ist N ein Wert, der von der Funktion OnStart() zurückgegeben wird.

Der Aufruf von OnStart(), der das Ausführungsergebnis zurückgibt, wird zur Verwendung empfohlen, da er nicht nur eine Skript- oder Dienstausführung ermöglicht, sondern auch einen Fehlercode oder andere nützliche Daten zur Analyse des Ergebnisses der Programmausführung zurückgibt.

Die Version ohne Ergebnisrückgabe wird nur aus Kompatibilitätsgründen mit alten Codes belassen. Ein Verwenden wird nicht empfohlen.

```
void  OnStart(void);
```

Hinweis

OnStart() ist die einzige Funktion zur Behandlung von Ereignissen in Skripten und Diensten. An diese Programme werden keine weiteren Ereignisse gesendet. Das Ereignis Start wird wiederum nicht an EAs und nutzerdefinierte Indikatoren übergeben.

Beispielskript

```
//--- Makros für die Arbeit mit Farben
#define XRGB(r,g,b)     (0xFF000000|(uchar(r)<<16)|(uchar(g)<<8)|uchar(b))
#define GETRGB(clr)     ((clr)&0xFFFFFF)
//+------------------------------------------------------------------+
//| Script Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Setzen der Farbe der Abwärtskerzen
   Comment("Set a downward candle color");
   ChartSetInteger(0,CHART_COLOR_CANDLE_BEAR,GetRandomColor());
   ChartRedraw(); // Chart sofort aktualisieren ohne auf einen neuen Tick zu warten
   Sleep(1000);   // Warte 1 Sekunde, um alle Änderungen zu sehen
//--- Setzen der Farbe der Aufwärtskerzen
   Comment("Set an upward candle color");
   ChartSetInteger(0,CHART_COLOR_CANDLE_BULL,GetRandomColor());
```

© 2000-2025, MetaQuotes Ltd.

```
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Hintergrundfarbe
   Comment("Set the background color");
   ChartSetInteger(0,CHART_COLOR_BACKGROUND,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe der Ask-Linie
   Comment("Set color of Ask line");
   ChartSetInteger(0,CHART_COLOR_ASK,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe der Bid-Linie
   Comment("Set color of Bid line");
   ChartSetInteger(0,CHART_COLOR_BID,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe einer Abwärtsbar und des Rahmens der Abwärtskerze
   Comment("Set color of a downward bar and a downward candle frame");
   ChartSetInteger(0,CHART_COLOR_CHART_DOWN,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe der Chartlinie und der Doji-Kerzen
   Comment("Set color of a chart line and Doji candlesticks");
   ChartSetInteger(0,CHART_COLOR_CHART_LINE,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe einer Aufwärtsbar und des Rahmens einer Aufwärtskerze
   Comment("Set color of an upward bar and an upward candle frame");
   ChartSetInteger(0,CHART_COLOR_CHART_UP,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farben der Achsen, des Maßstabes und der Linien OHLC
   Comment("Set color of axes, scale and OHLC line");
   ChartSetInteger(0,CHART_COLOR_FOREGROUND,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Gitterfarbe
   Comment("Set a grid color");
   ChartSetInteger(0,CHART_COLOR_GRID,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe des letzten Preises
   Comment("Set Last price color");
   ChartSetInteger(0,CHART_COLOR_LAST,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farben von Stop-Loss und Take-Profit
   Comment("Set color of Stop Loss and Take Profit order levels");
```

© 2000-2025, MetaQuotes Ltd.

```
   ChartSetInteger(0,CHART_COLOR_STOP_LEVEL,GetRandomColor());
   ChartRedraw();
   Sleep(1000);
//--- Setzen der Farbe des Volumens und des Markteintritts
   Comment("Set color of volumes and market entry levels");
   ChartSetInteger(0,CHART_COLOR_VOLUME,GetRandomColor());
   ChartRedraw();
  }
//+------------------------------------------------------------------+
//| Rückgabe einer zufällig gewählten Farbe                          |
//+------------------------------------------------------------------+
color GetRandomColor()
  {
   color clr=(color)GETRGB(XRGB(rand()%255,rand()%255,rand()%255));
   return clr;
  }
```

Siehe auch

Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

OnInit

Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis Init eintritt. Sie wird auch verwendet zur Initialisierung eines MQL5-Programm verwendet. Es gibt zwei Versionen dieser Funktion.

Die **Version, die das Ergebnis zurück gibt**

```
int  OnInit(void);
```

Rückgabewert

int Ty des Wertes, Null steht für eine erfolgreiche Initialisierung.

Wenn INIT_FAILED zurückgegeben wird, wird der EA zwangsweise aus dem Chart entfernt.

Wenn INIT_FAILED zurückgegeben wird, wird der Indikator nicht aus dem Chart entfernt. Der im Chart verbleibende Indikator ist nicht funktionsfähig — die Ereignisbehandlung im Indikator wird nicht aufgerufen.

Diese Version von OnInit(), die das Ausführungsergebnis zurückgibt, wird zur Verwendung empfohlen, da sie nicht nur die Programminitialisierung erlaubt, sondern auch einen Fehlercode im Falle eines vorzeitigen Programmendes zurückgibt.

Die Version ohne Ergebnisrückgabe wird nur aus Kompatibilitätsgründen mit alten Codes belassen. Ein Verwenden wird nicht empfohlen.

```
void OnInit(void);)
```

Hinweis

Das Init-Ereignis wird unmittelbar nach dem Laden eines EA oder eines Indikators erzeugt. Für Skripte wird das Ereignis nicht erzeugt. Die Funktion OnInit() wird verwendet, um ein MQL5-Programm zu initialisieren. Wenn OnInit() einen Rückgabewert von Typ inthat, bedeutet ein Rückgabewert ungleich Null eine fehlgeschlagene Initialisierung und erzeugt das Ereignis Deinit mit dem Code der Deinitialisierung REASON_INITFAILEDals Ergebnis..

Die Version OnInit() mit der Rückgabe von void bedeutet, dass immer eine erfolgreiche Initialisierung durchgeführt wurde und das zu verwenden wird nicht empfohlen.

Für die Eingaben einer Optimierung eines EAs***wird empfohlen, Werte aus der Enumeration ENUM_INIT_RETCODE als Rückgabewert verwenden. Diese Werte sind für die Festlegung des Optimierungsprozesses einschließlich der Auswahl der am besten geeigneten Testmittel bestimmt. Mit der Funktion TerminalInfoInteger() können während der EA-Initialisierung vor dem Start des Tests Daten zur Agentenkonfiguration und Ressourcen (Anzahl der Kerne, freier Speicherplatz, etc.) abgefragt werden. Basierend auf den erhaltenen Daten können Sie entweder die Verwendung des Test-Agenten erlauben oder die Optimierung des EA unterbinden.

ID	Beschreibung
INIT_SUCCEEDED	Initialisierung war erfolgreich, der Test des EAs kann beginnen. Dieser Code bedeutet dasselbe wie die Null - die EA-Initialisierung im Tester war erfolgreich.

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
INIT_FAILED	Initialisierung ist fehlgeschlagen. Es macht keinen Sinn, den Test trotz des aufgetretenen Fehlers fortzusetzen. Beispielsweise ist es nicht möglich, ein für den EA-Betrieb notwendiges Indikator anzulegen. Die Rückgabe dieses Wertes bedeutet dasselbe wie die Rückgabe des Wertes ungleich Null - EA-Initialisierung im Tester fehlgeschlagen.
INIT_PARAMETERS_INCORRECT	Entwickelt, um einen falschen Satz von Eingabeparametern durch einen Programmierer zu kennzeichnen. In der allgemeinen Optimierungstabelle ist der Ergebnisstring mit diesem Rückgabecode rot markiert. Ein Test für einen solchen Satz von EA-Eingaben wird nicht durchgeführt. Der Agent bleibt bereit, mit neuen Eingaben getestet zu werden. Wenn dieser Wert empfangen wird, gibt der Strategie-Tester diese Aufgabe nicht zur wiederholten Ausführung an andere Agenten weiter.
INIT_AGENT_NOT_SUITABLE	Keine Programmausführungsfehler bei der Initialisierung. Aus irgendwelchen anderen Gründen ist der Agent jedoch nicht für die Durchführung eines Tests geeignet. Zum Beispiel gibt es nicht genügend RAM, kein OpenCL-Unterstützung, etc. Nach der Rückgabe dieses Codes erhält der Agent keine Aufgaben mehr bis zum Ende von dieser Optimierung..

Die Benutzung der Rückgabewerte RINIT_FAILED/INIT_PARAMETERS_INCORRECT von OnInit() im Tester haben einige Besonderheiten, die bei der Optimierung von EAs berücksichtigt werden sollten:

- Die Parameter, für die OnInit() den Wert INIT_PARAMETERS_INCORRECT zurückgibt, gilt als ungeeignet für den Test und wird nicht von der nächsten Population während genetische Optimierung verwendet. Zu viele "verworfene" Parametereinstellungen können bei der Suche nach optimalen EA-Parametern zu falschen Ergebnissen führen. Der Suchalgorithmus geht davon aus, dass die Funktion des Optimierungskriterium keine Lücken in der gesamten Vielzahl der Eingangsparameter aufweist.
- Wenn OnInit() INIT_FAILED zurückgibt, bedeutet das, dass ein Test nicht gestartet werden kann und der EA aus dem Speicher des Agenten entfernt wird. Das EA wird erneut geladen, um den nächsten Durchlauf mit einem neuen Satz von Parametern durchzuführen. Der Start des nächsten Optimierungsdurchgangs dauert dadurch viel länger als bei einem Aufruf von TesterStop().

Beispiele der Verwendung der Funktionen OnInit() von einem EA

```
//--- Eingabeparameter
input int      ma_period=20; // Periodenlänge des gleitenden Durchschnitts

//--- Handle des Indikators, den der EA verwendet
int indicator_handle;
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Prüfen der Gültigkeit von ma_period
   if(ma_period<=0)
     {
      PrintFormat("Invalid ma_period input value: %d",ma_period);
      return (INIT_PARAMETERS_INCORRECT);
     }
//--- Während der Optimierung
   if(MQLInfoInteger(MQL_OPTIMIZATION))
     {
      //--- Prüfung auf ausreichend RAM für den Agenten
      int available_memory_mb=TerminalInfoInteger(TERMINAL_MEMORY_TOTAL);
      if(available_memory_mb<2000)
        {
         PrintFormat("Insufficient memory for the test agent: %d MB",
                     available_memory_mb);
         return (INIT_AGENT_NOT_SUITABLE);
        }
     }
//--- Prüfen des Indikators
   indicator_handle=iCustom(_Symbol,_Period,"My_Indicator",ma_period);
   if(indicator_handle==INVALID_HANDLE)
     {
      PrintFormat("Failed to generate My_Indicator handle. Error code %d",
                  GetLastError());
      return (INIT_FAILED);
     }
//--- EA-Initialisierung war erfolgreich
   return(INIT_SUCCEEDED);
  }
```

Siehe auch

OnDeinit, Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals, Initialization of variables, Creating and deleting objects

© 2000-2025, MetaQuotes Ltd.

OnDeinit

Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis Deinit eintritt. Sie wird auch verwendet, ein laufendes MQL5-Programm zu deinitialisieren.

```
void  OnDeinit(
   const int  reason         // deinitialization reason code
   );
```

Parameter

Ursache

[in] Die Ursache der Deinitialisierung als Zahlencode.

Rückgabewert

Kein Rückgabewert

Hinweis

In den folgenden Fällen wird das Deinit-Ereignis für EAs und Indikatoren generiert:

- Vor einer Neuinitialisierung aufgrund der Änderung des Symbols oder des Zeitrahmens des Charts, auf dem das MQL5-Programm läuft;
- Vor einer Neuinitialisierung durch Änderung der Eingaben;
- Vor dem Entladen eines MQL5-Programms.

Der Parameter *reason* kann die folgenden Werte annehmen:

Konstante	Wert	Beschreibung
REASON_PROGRAM	0	Der EA endete durch den Aufruf der Funktion ExpertRemove()
REASON_REMOVE	1	Das Programm wurde von Chart entfernt
REASON_RECOMPILE	2	Programm wurde neu kompiliert
REASON_CHARTCHANGE	3	Ein Symbol oder ein Zeitrahmen des Charts wurden geändert
REASON_CHARTCLOSE	4	Chart wurde geschlossen
REASON_PARAMETERS	5	Eingaben wurden von Nutzer geändert
REASON_ACCOUNT	6	Ein anderes Konto wurde aktiviert oder die Verbindung zum Handelsserver wurde aufgrund von Änderungen in den Kontoeinstellungen neu hergestellt.
REASON_TEMPLATE	7	Ein anderes Template wurde auf den Chart übertragen
REASON_INITFAILED	8	Die Funktion OnInit() gab einen Wert ungleich Null zurück
REASON_CLOSE	9	Terminal wurde beendet

© 2000-2025, MetaQuotes Ltd.

Der Zahlencode der Deinitialisierung des EAs kann von der Funktion UninitializeReason() oder von der vordefinierten Variablen _UninitReason empfangen werden.

Beispiele der Verwendung der Funktionen OnInit() und OnDeinit() von einem EA

```
input int fake_parameter=3;      // nicht verwendeter Parameter
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Abfrage der Versionsnummer (build) mit dem das Programm kompiliert wurde
   Print(__FUNCTION__," Build #",__MQLBUILD__);
//--- Der Grund eines Reset kann auch in Oninit() erhalten werden
   Print(__FUNCTION__," Deinitialization reason code can be received during the EA res
//--- Der erste Weg, um an den Code der Deinitialisierung zu kommen
   Print(__FUNCTION__," _UninitReason = ",getUninitReasonText(_UninitReason));
//--- Der zweite Weg, um an den Code der Deinitialisierung zu kommen
   Print(__FUNCTION__," UninitializeReason() = ",getUninitReasonText(UninitializeReasc
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Der erste Weg, um an den Code der Deinitialisierung zu kommen
   Print(__FUNCTION__," Deinitialization reason code = ",reason);
//--- Der zweite Weg, um an den Code der Deinitialisierung zu kommen
   Print(__FUNCTION__," _UninitReason = ",getUninitReasonText(_UninitReason));
//--- Der dritte Weg, um an den Code der Deinitialisierung zu kommen
   Print(__FUNCTION__," UninitializeReason() = ",getUninitReasonText(UninitializeReasc
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung der Ursache der Deinitialisierung      |
//+------------------------------------------------------------------+
string getUninitReasonText(int reasonCode)
  {
   string text="";
//---
   switch(reasonCode)
     {
      case REASON_ACCOUNT:
         text="Account was changed";break;
      case REASON_CHARTCHANGE:
         text="Symbol or timeframe was changed";break;
      case REASON_CHARTCLOSE:
         text="Chart was closed";break;
```

© 2000-2025, MetaQuotes Ltd.

```
      case REASON_PARAMETERS:
         text="Input-parameter was changed";break;
      case REASON_RECOMPILE:
         text="Program "+__FILE__+" was recompiled";break;
      case REASON_REMOVE:
         text="Program "+__FILE__+" was removed from chart";break;
      case REASON_TEMPLATE:
         text="New template was applied to chart";break;
      default:text="Another reason";
     }
//---
   return text;
  }
```

Siehe auch

OnInit, Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals, Deinitialisierungsgruende, Sichtbereich und Lebensdauer der Variablen, Erzeugung und Entfernung der Objekte

© 2000-2025, MetaQuotes Ltd.

OnTick

Die Funktion wird von EAs aufgerufen, wenn das Ereignis NewTick eintritt, um einen neuen Preis abzuarbeiten.

```
void  OnTick(void);
```

Rückgabewert

Kein Rückgabewert

Hinweis

Das Ereignis NewTick wird nur für EAs erzeugt, wenn ein neuer Tick für ein Symbol des Charts eintrifft, auf dem der EA läuft. Es macht keinen Sinn, die Funktion OnTick() in einem benutzerdefinierten Indikator oder einem Skript zu definieren, da für sie kein NewTick-Ereignis generiert wird.

Das Ereignis eines Ticks wird nur für EAs generiert, aber das bedeutet nicht, dass EAs die OnTick() unbedingt besitzen müssen, neben OnTick() auch die Ereignisse Timer, BookEvent und ChartEvent für EAs generiert werden.

Alle Ereignisse werden nacheinander in der Reihenfolge ihres Eintreffens behandelt. Wenn die Warteschlange bereits das Ereignis NewTick enthält, oder sich dieses Ereignis in der Verarbeitungsphase befindet, dann wird das neue Ereignis NewTick nicht in die Warteschlange des MQL5-Programms hinzugefügt.

Das Ereignis NewTick wird unabhängig davon generiert, ob der automatische Handel aktiviert ist (Schaltfläche AutoTrading). Ein deaktivierter Autohandel bedeutet nur ein Verbot des Sendens von Handelsanfragen von einem EA. Die Arbeitsweise des EAs wird nicht beeinträchtigt.

Das Deaktivieren des automatischen Handels durch Drücken der Schaltfläche AutoTrading unterbricht nicht die aktuelle Ausführung der Funktion OnTick().

Beispiel des EA mit seiner gesamten Handelslogik in der Funktion OnTick()

```
//+------------------------------------------------------------------+
//|                                                   TradeByATR.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Sample EA trading in the \"explosive\" candle direction"
#property description "\"Explosive\" candle has the body size exceeding k*ATR"
#property description "The \"revers\" parameter reverses the signal direction"

input double lots=0.1;        // Volumen in Lots
input double kATR=3;          // Länge der Signalkerze in ATR
input int    ATRperiod=20;    // Periodenlänge des ATR
input int    holdbars=8;      // Anzahl der Bars, die die Position gehalten werden sol
input int    slippage=10;     // Erlaubter Schlupf
```

© 2000-2025, MetaQuotes Ltd.

```
input bool   revers=false;    // Signal umkehren?
input ulong  EXPERT_MAGIC=0;  // Des EA's Magicnummer
//--- zum Sichern des Handles des Indikators ATR
int atr_handle;
//--- hier werden die letzten Werte des ATR und die Kerzenkörper gesichert
double last_atr,last_body;
datetime lastbar_timeopen;
double trade_lot;
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Initialisierung der Globalen Variablen
   last_atr=0;
   last_body=0;
//--- Setzen des korrekten Volumens
   double min_lot=SymbolInfoDouble(_Symbol,SYMBOL_VOLUME_MIN);
   trade_lot=lots>min_lot? lots:min_lot;
//--- Erstellen des Handles des Indikators ATR
   atr_handle=iATR(_Symbol,_Period,ATRperiod);
   if(atr_handle==INVALID_HANDLE)
     {
      PrintFormat("%s: failed to create iATR, error code %d",__FUNCTION__,GetLastError
      return(INIT_FAILED);
     }
//--- Erfolgreiche Initialisierung des EA
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Information über das Ende der Arbeit des EAs
   Print(__FILE__,": Deinitialization reason code = ",reason);
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- Handelssignal
   static int signal=0; // +1 heißt Kaufsignal, -1 Verkaufssignal
//--- Prüfen und Schließen alter Positionen, die vor mehr als 'holdbars' eröffnet wurd
   ClosePositionsByBars(holdbars,slippage,EXPERT_MAGIC);
//--- Prüfen auf eine neue Bar
   if(isNewBar())
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Prüfen auf ein Signal
      signal=CheckSignal();
     }
//--- Wenn eine Netting-Position eröffnet wurde - warten bis sie geschlossen wurde
   if(signal!=0 && PositionsTotal()>0 && (ENUM_ACCOUNT_MARGIN_MODE)AccountInfoInteger
     {
      signal=0;
      return; // die Ereignisbehandlung von NewTick beenden und kein Markteintritt vo
     }
//--- für ein Hedging-Konto wird jede Position separat gehalten und geschlossen
   if(signal!=0)
     {
      //--- Kaufsignal
      if(signal>0)
        {
         PrintFormat("%s: Buy signal! Revers=%s",__FUNCTION__,string(revers));
         if(Buy(trade_lot,slippage,EXPERT_MAGIC))
            signal=0;
        }
      //--- Verkaufssignal
      if(signal<0)
        {
         PrintFormat("%s: Sell signal! Revers=%s",__FUNCTION__,string(revers));
         if(Sell(trade_lot,slippage,EXPERT_MAGIC))
            signal=0;
        }
     }
//--- Ende der Funktion OnTick
  }
//+------------------------------------------------------------------+
//| Prüfen auf ein neues Handelssignal                               |
//+------------------------------------------------------------------+
int CheckSignal()
  {
//--- 0 beutet klein Signal
   int res=0;
//--- Abfrage des Wertes der ATR der vorletzten kompletten Bar (Indes der Bar ist 2)
   double atr_value[1];
   if(CopyBuffer(atr_handle,0,2,1,atr_value)!=-1)
     {
      last_atr=atr_value[0];
      //--- Datenabfrage der letzten geschlossenen Bar von Array des Typs MqlRates
      MqlRates bar[1];
      if(CopyRates(_Symbol,_Period,1,1,bar)!=-1)
        {
         //--- Berechnen der Körpergröße der letzten, vollständigen Kerze
         last_body=bar[0].close-bar[0].open;
         //--- wenn der Körper der letzten Bar (mit Index 1) den vorherigen ATR-Wert
         if(MathAbs(last_body)>kATR*last_atr)
```

© 2000-2025, MetaQuotes Ltd.

```
            res=last_body>0?1:-1; // positiver Wert der Aufwärtskerze
         }
       else
          PrintFormat("%s: Failed to receive the last bar! Error",__FUNCTION__,GetLastE
      }
    else
       PrintFormat("%s: Failed to receive ATR indicator value! Error",__FUNCTION__,GetL
//--- falls der umgekehrte Handelsmodus aktiviert ist
    res=revers?-res:res;  // Signal umkehren, wenn nötig (Rückgabe von -1 statt 1 und v
//--- Rückgabe des Wertes des Handelssignals
    return (res);
  }
//+------------------------------------------------------------------+
//|  Rückgabe von 'true' wenn eine neue Bar erscheint                |
//+------------------------------------------------------------------+
bool isNewBar(const bool print_log=true)
  {
   static datetime bartime=0; // Sichern der Eröffnungszeit der aktuellen Bar
//--- Abfrage der Eröffnungszeit der Bar Null
   datetime currbar_time=iTime(_Symbol,_Period,0);
//--- Wenn sich die Eröffnungszeit änderte, gibt es eine neue Bar
   if(bartime!=currbar_time)
     {
      bartime=currbar_time;
      lastbar_timeopen=bartime;
      //--- Eintragen der Eröffnungszeit der neuen Bar in das Log
      if(print_log && !(MQLInfoInteger(MQL_OPTIMIZATION)||MQLInfoInteger(MQL_TESTER)))
        {
         //--- Anzeige der Nachricht mit der Eröffnungszeit der neuen bar
         PrintFormat("%s: new bar on %s %s opened at %s",__FUNCTION__,_Symbol,
                     StringSubstr(EnumToString(_Period),7),
                     TimeToString(TimeCurrent(),TIME_SECONDS));
         //--- Datenabfrage beim letzten Tick
         MqlTick last_tick;
         if(!SymbolInfoTick(Symbol(),last_tick))
            Print("SymbolInfoTick() failed, error = ",GetLastError());
         //--- Anzeige der Zeit des letzten Ticks bis zur Millisekunde
         PrintFormat("Last tick was at %s.%03d",
                     TimeToString(last_tick.time,TIME_SECONDS),last_tick.time_msc%1000
        }
      //--- wir haben eine neue Bar
      return (true);
     }
//--- keine neue Bar
   return (false);
  }
//+------------------------------------------------------------------+
//| Kauf zum Marktpreis mit angegebenen Volumen                      |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool Buy(double volume,ulong deviation=10,ulong  magicnumber=0)
  {
//--- Kauf zum Marktpreis
   return (MarketOrder(ORDER_TYPE_BUY,volume,deviation,magicnumber));
  }
//+------------------------------------------------------------------+
//| Verkauf zum Marktpreis mit angegebenen Volumen                   |
//+------------------------------------------------------------------+
bool Sell(double volume,ulong deviation=10,ulong  magicnumber=0)
  {
//--- Verkauf zum Marktpreis
   return (MarketOrder(ORDER_TYPE_SELL,volume,deviation,magicnumber));
  }
//+------------------------------------------------------------------+
//| Position schließen wegen der Haltezeit                           |
//+------------------------------------------------------------------+
void ClosePositionsByBars(int holdtimebars,ulong deviation=10,ulong  magicnumber=0)
  {
   int total=PositionsTotal(); // Anzahl der offenen Positionen
//--- Iterieren über die offenen Position
   for(int i=total-1; i>=0; i--)
     {
      //--- Parameter der Position
      ulong  position_ticket=PositionGetTicket(i);
      string position_symbol=PositionGetString(POSITION_SYMBOL);
      ulong  magic=PositionGetInteger(POSITION_MAGIC);
      datetime position_open=(datetime)PositionGetInteger(POSITION_TIME);
      int bars=iBarShift(_Symbol,PERIOD_CURRENT,position_open)+1;

      //--- wenn die Lebenszeit der Position lang genug ist, und MagicNummer und Symbo
      if(bars>holdtimebars && magic==magicnumber && position_symbol==_Symbol)
        {
         int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS);
         double volume=PositionGetDouble(POSITION_VOLUME);
         ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE)
         string str_type=StringSubstr(EnumToString(type),14);
         StringToLower(str_type); // Kleinschreibung für das korrekte Nachrichtenforma
         PrintFormat("Close position #%I64u %s %s %.2f",
                     position_ticket,position_symbol,str_type,volume);
         //--- Setzen des Auftragsart und Senden der Handelsanfrage
         if(type==POSITION_TYPE_BUY)
            MarketOrder(ORDER_TYPE_SELL,volume,deviation,magicnumber,position_ticket);
         else
            MarketOrder(ORDER_TYPE_BUY,volume,deviation,magicnumber,position_ticket);
        }
     }
  }
//+------------------------------------------------------------------+
//| Vorbereiten und Senden der Handelsanfrage                        |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool MarketOrder(ENUM_ORDER_TYPE type,double volume,ulong slip,ulong magicnumber,ulong
  {
//--- Deklarieren und Initialisieren der Strukturen
   MqlTradeRequest request={};
   MqlTradeResult  result={};
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(type==ORDER_TYPE_BUY)
      price=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
//--- Abfrage der Parameter
   request.action   =TRADE_ACTION_DEAL;                     // Typ der Handelsoperatio
   request.position =pos_ticket;                            // Ticketnummer der zu sch
   request.symbol   =Symbol();                              // Symbol
   request.volume   =volume;                                // Volumen
   request.type     =type;                                  // Auftragsart
   request.price    =price;                                 // Handelspreis
   request.deviation=slip;                                  // erlaubter Schlupf vom
   request.magic    =magicnumber;                           // MagicNumber des Auftra
//--- Senden einer Anfrage
   if(!OrderSend(request,result))
     {
      //--- Datenanzeige im Fehlerfall
      PrintFormat("OrderSend %s %s %.2f at %.5f error %d",
                  request.symbol,EnumToString(type),volume,request.price,GetLastError
      return (false);
     }
//--- Information über eine erfolgreiche Operation
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
   return (true);
  }
```

Siehe auch

Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals, OnTimer, OnBookEvent, OnChartEvent

© 2000-2025, MetaQuotes Ltd.

OnCalculate

Die Funktion wird von Indikatoren aufgerufen, wenn das Ereignis Calculate eintritt, um Preisänderungen abzuarbeiten. Es gibt zwei Versionen dieser Funktion. Aber nur eine kann von einem einzelnen Indikator verwendet werden.

Berechnung auf Basis eines Datenarrays

```
int  OnCalculate(
   const int        rates_total,       // Größe des Preisarrays price[]
   const int        prev_calculated,   // Anzahl der bearbeiteten Bars im letzten Aufi
   const int        begin,             // Index für den Array price[] für den Beginn
   const double&    price[]            // Wertearray für die Berechnung
   );
```

Berechnung auf Basis der aktuellen Zeitreihendaten des Zeitrahmens

```
int  OnCalculate(
   const int        rates_total,       // Umfang der Zeitreihen
   const int        prev_calculated,   // Anzahl der bearbeiteten Bars im letzten Aufi
   const datetime&  time[],            // Array der Zeit
   const double&    open[],            // Open Array
   const double&    high[],            // High Array
   const double&    low[],             // Low Array
   const double&    close[],           // Close Array
   const long&      tick_volume[],     // Tick Volume Array
   const long&      volume[],          // Real Volume Array
   const int&       spread[]           // Spread Array
   );
```

Parameter

rates_total

[in] Anzahl der dem Indikator für die Berechnung zur Verfügung stehenden Elemente der Preis-Arrays oder Eingabeserien. Im zweiten Funktionstyp entspricht der Parameterwert der Anzahl der Bars auf dem Chart, auf dem er gestartet wurde.

prev_calculated

[in] Ist der von OnCalculate() im letzten Aufruf zurückgegebenen Wert. Er wurde entwickelt, um die Bars zu überspringen, die sich seit dem letzten Aufruf dieser Funktion nicht geändert haben.

begin

[in] Indexwert im Array price[], ab dem die aussagekräftigen Daten beginnen. Damit können fehlende oder initiale Daten übersprungen, für die es keine korrekten Werte gibt.

price[]

[in] Array von Werten für Berechnungen. Einer der Zeitreihen mit Preisen, aber auch ein berechneter Indikatorpuffer kann als Preis-Array übergeben werden. Die Art der zur Berechnung übergebenen Daten kann über die Funktion _AppliedTo vordefinierte Variable definiert werden.

time[]

[in] Array mit Eröffnungszeiten der Bars.

© 2000-2025, MetaQuotes Ltd.

open[]

[in] Array mit Eröffnungspreisen der Bars.

high[]

[in] Array mit den Hochs der Bars.

low[]

[in] Array mit den Tiefs der Bars.

close[]

[in] Array mit den Schlusskursen der Bars.

tick_volume[]

[in] Array mit der Tick-Volumina der Bars.

volume[]

[in] Array mit den Handelsvolumina der Bars.

spread[]

[in] Array mit der Spreizung (spread) der Kurse der Bars.

Rückgabewert

Dieser ganzzahlige Rückgabewert wird mit dem Parameter *prev_calculated* dem nächsten Aufruf der Funktion übergeben.

Hinweis

Wenn die Funktion OnCalculate() gleich Null ist, werden im DataWindow des Client-Terminals keine Indikatorwerte angezeigt.

Wenn die Preisdaten seit dem letzten Aufruf der Funktion OnCalculate() geändert wurden (eine umfangreichere Historie wurde geladen oder Lücken in der Historie gefüllt), wird der Wert des Eingabeparameters *prev_calculated* vom Terminal selbst auf Null gesetzt.

Um die Richtung der Indexierung der Zeitreihen *time[]*, *open[]*, *high[]*, *low[]*, *close[]*, *tick_volume[]*, *volume[]* und *spread[]* festzulegen, rufen Sie die Funktion ArrayGetAsSeries() auf. Um nicht von Voreinstellungen abhängig zu sein, rufen Sie die Funktion ArraySetAsSeries() für die Arbeit mit den Arrays auf.

Bei Verwendung des ersten Funktionstyps wird beim Start des Indikators eine notwendige Zeitreihe oder ein Indikator vom Benutzer als Preis-Array[] im Register Parameter ausgewählt. Geben Sie dazu in der Auswahlliste des Feldes "Apply to" field." das gewünschte Element an.

Um die Werte eigener Indikatoren zu erhalten, die als mql5-Programme vorliegen, wird die Funktion iCustom() verwendet. Sie gibt das Handle des Indikator für die weiteren Operationen zurück. Es ist auch möglich, das gewünschte *Preis-Array[]* oder das Handle eines anderen Indikators anzugeben. Dieser Parameter sollte als letzter in der Liste der Eingabevariablen eines benutzerdefinierten Indikators übergeben werden.

Es ist notwendig, die Verbindung zwischen dem Rückgabewert der Funktion OnCalculate() und dem zweiten Eingabeparameter *prev_calculated* zu verwenden. Beim Aufruf der Funktion enthält der Parameter *prev_calculated* den Wert, den die Funktion OnCalculate() beim letzten Aufruf zurückgegeben hat. Dadurch ist es möglich, ressourcenschonende Algorithmen zur Berechnung eines

© 2000-2025, MetaQuotes Ltd.

benutzerdefinierten Indikators zu implementieren, um wiederholte Berechnungen für die Bars zu vermeiden, die sich seit dem letzten Start dieser Funktion nicht geändert haben.

Beispielindikator

```
//+------------------------------------------------------------------+
//|                                            OnCalculate_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Sample Momentum indicator calculation"

//---- Indikatoreinstellungen
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
#property indicator_type1   DRAW_LINE
#property indicator_color1  Blue
//---- Eingaben
input int MomentumPeriod=14; // Calculation period
//---- Indikatorpuffer
double    MomentumBuffer[];
//--- globale Variable zur Speicherung der Berechnungsperioden
int       IntPeriod;
//+------------------------------------------------------------------+
//| Initialisierungsfunktion eine nutzerdefinierten Indikators       |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- Prüfen der Eingabeparameter
   if(MomentumPeriod<0)
     {
      IntPeriod=14;
      Print("Period parameter has an incorrect value. The following value is to be use
     }
   else
      IntPeriod=MomentumPeriod;
//---- Puffer
   SetIndexBuffer(0,MomentumBuffer,INDICATOR_DATA);
//---- Indikatorname zur Anzeige im DataWindow und im Unterfenster
   IndicatorSetString(INDICATOR_SHORTNAME,"Momentum"+"("+string(IntPeriod)+")");
//--- Setzen des Index der Bar, mit der das Zeichnen beginnt
   PlotIndexSetInteger(0,PLOT_DRAW_BEGIN,IntPeriod-1);
//--- setzen von 0.0 als 'empty value', der nicht gezeichnet wird
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Indikatorgenauigkeit für die Anzeige
   IndicatorSetInteger(INDICATOR_DIGITS,2);
  }
//+------------------------------------------------------------------+
//|  Berechnung des Indikators Momentum                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,     // Größe des Arrays price[]
                const int prev_calculated, // Anzahl der vorher bearbeiteten Bars
                const int begin,           // Beginn der signifikanten Bars
                const double &price[])     // Array mit den Werten für die Berechnung
  {
//--- erste Position für die Berechnung
   int StartCalcPosition=(IntPeriod-1)+begin;
//---- bei ungenügenden Daten für die Berechnung
   if(rates_total<StartCalcPosition)
      return(0);  // exit with a zero value - the indicator is not calculated
//--- Korrektur des Beginns der Zeichnung
   if(begin>0)
      PlotIndexSetInteger(0,PLOT_DRAW_BEGIN,StartCalcPosition+(IntPeriod-1));
//--- Beginn der Berechnung, definieren der Startposition
   int pos=prev_calculated-1;
   if(pos<StartCalcPosition)
      pos=begin+IntPeriod;
//--- Hauptschleife der Berechnung
   for(int i=pos;i<rates_total && !IsStopped();i++)
      MomentumBuffer[i]=price[i]*100/price[i-IntPeriod];
//--- OnCalculate Ausführung ist beendet Rückgabe des neuen Wertes von prev_calculated
   return(rates_total);
  }
```

Siehe auch

ArrayGetAsSeries, ArraySetAsSeries, iCustom, Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals, Zugang zu Zeitreihen und Daten der Indikatoren

© 2000-2025, MetaQuotes Ltd.

OnTimer

Die Funktion wird von Indikatoren und EAs während des periodisches Ereignis Timer aufgerufen, das vom Terminal in festen Zeitintervallen erzeugt wird.

```
void  OnTimer(void);
```

Rückgabewert

Kein Rückgabewert

Hinweis

Das Ereignis Timer wird periodisch vom Client-Terminal für einen EA erzeugt, der den Timer mit der Funktion EventSetTimer() aktiviert hat. Normalerweise wird diese Funktion in der Funktion OnInit() aufgerufen. Wenn der EA aufhört zu arbeiten, sollte der Timer mit EventKillTimer(), der normalerweise in der Funktion OnDeinit() aufgerufen wird, eliminiert werden.

Jeder Expert Advisor und jeder Indikator arbeitet mit einem eigenen Timer, der nur Ereignisse von diesem Timer empfängt. Während des Herunterfahrens der mql5-Anwendung wird der Timer zwangsweise entfernt, falls er erstellt wurde, aber nicht mit der Funktion EventKillTimer() gelöscht wurde.

Wenn Sie Timerereignisse häufiger als einmal pro Sekunde empfangen müssen, verwenden Sie EventSetMillisecondTimer() zum Erstellen eines hochauflösenden Timers.

Im Allgemeinen wird bei einer Verkürzung der Zeitspanne des Timers die Testzeit erhöht, da der Handler von Timer-Ereignissen häufiger aufgerufen wird. Im Echtzeitbetrieb werden Timer-Ereignisse aufgrund von Hardware-Beschränkungen nicht mehr als 1 mal in 10-16 Millisekunden generiert.

Für jedes Programm kann nur ein Timer gestartet werden. Jede mql5-Anwendung und jedes Chart haben ihre eigene Warteschlange von Ereignissen, in der alle neu eingetroffenen Ereignisse eingetragen werden. Wenn die Warteschlange (queue) bereits das Ereignis des Timers enthält oder sich dieses Ereignis in der Verarbeitungsphase befindet, wird das neue Timer-Ereignis nicht zur Warteschlange der mql5-Anwendung hinzugefügt.

Beispiel-EA, der die Funktion OnTimer() verwendet

```
//+------------------------------------------------------------------+
//|                                                OnTimer_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Example of using the timer for calculating the trading server t
#property description "It is recommended to run the EA at the end of a trading week be
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Erstellen eines Timers mit der Zeitspanne von 1 Sekunde
   EventSetTimer(1);

//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Löschen des Timers nach dem Ende der Arbeit
   EventKillTimer();

  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
//| Timer Funktion                                                   |
//+------------------------------------------------------------------+
void OnTimer()
  {
//--- Zeitpunkt des OnTimer() bei seinem ersten Aufruf
   static datetime start_time=TimeCurrent();
//--- Des Handelsserver Zeitpunkt beim ersten Aufruf von OnTimer()
   static datetime start_tradeserver_time=0;
//--- Berechnete Serverzeit
   static datetime calculated_server_time=0;
//--- Lokalzeit des PV
   datetime local_time=TimeLocal();
//--- aktuelle geschätzte Serverzeit
   datetime trade_server_time=TimeTradeServer();
//--- Falls die Serverzeit warum auch immer unbekannt ist, beenden vor der Zeit
   if(trade_server_time==0)
      return;
//--- Wenn die erste Zeit des Servers noch nicht gespeichert wurde
   if(start_tradeserver_time==0)
     {
      start_tradeserver_time=trade_server_time;
      //--- Setzen und Berechnen der Werte des Handelsservers
      Print(trade_server_time);
      calculated_server_time=trade_server_time;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
    else
      {
       //--- Erhöhen der ersten Zeit von OnTimer()
       if(start_tradeserver_time!=0)
          calculated_server_time=calculated_server_time+1;;
      }
//---
   string com=StringFormat("                  Start time: %s\r\n",TimeToString(start_t
   com=com+StringFormat("                   Local time: %s\r\n",TimeToString(local_time
   com=com+StringFormat("TimeTradeServer time: %s\r\n",TimeToString(trade_server_time,
   com=com+StringFormat(" EstimatedServer time: %s\r\n",TimeToString(calculated_server
//--- Die Werte aller Zähler anzeigen
   Comment(com);
  }
```

Siehe auch

EventSetTimer, EventSetMillisecondTimer, EventKillTimer, GetTickCount, GetMicrosecondCount, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

OnTrade

Die Funktion wird von EAs aufgerufen, wenn das Ereignis Trade eintritt. Die Funktion reagiert auf Änderungen von den Aufträgen, Positionen und der Handelslisten.

```
void  OnTrade(void);
```

Rückgabewert

Kein Rückgabewert

Hinweis

OnTrade() steht nur Expert Advisor zur Verfügung. Sie kann nicht von Indikatoren oder Scripts verwendet werden, auch wenn man dort eine Funktionen mit dem gleichen Namen und Typ hinzufügt.

Für jede Handelsaktion (Platzieren einer Pending Order, Öffnen/Schließen einer Position, Platzieren von Stopps, Aktivieren von Pending Orders, usw.) wird die Historie der Orders und Trades und/oder die Liste der Positionen und aktuellen Aufträge entsprechend geändert.

Wenn ein Handelsserver einen Auftrag bearbeitet, sendet er dem Terminal eine Nachricht über das eingehende Ereignis Trade. Um relevante Daten über Aufträge und Positionen aus der Historie abzurufen, ist es notwendig, zuerst eine Anfrage an die Handelshistorie mit der Funktion HistorySelect(). zu stellen.

Die Handelsereignisse werden vom Server erstellt und zwar für:

- Veränderte aktive Aufträge,
- Veränderte Positionen,
- Veränderte Transaktionen,
- Veränderte Handelshistorie.

Jedes Ereignis Trade kann als Ergebnis einer oder mehrerer Handelsanfragen auftreten. Handelsanfragen werden mit OrderSend() oder OrderSendAsync() an den Server gesendet. Jede Anfrage kann zu mehreren Handelsereignissen führen. Sie können nicht der Vorstellung vertrauen: "Eine Anfrage - Ein Handelsereignis", da die Verarbeitung von Ereignissen in mehreren Stufen erfolgt und jede Operation den Status von Aufträgen, Positionen und die Handelsgeschichte verändern kann.

Die Funktion OnTrade() wird nach den entsprechenden Aufrufen von OnTradeTransaction() ausgeführt. Im Allgemeinen gibt es keine genaue Korrelation zwischen der Anzahl der Aufrufe von OnTrade() und OnTradeTransaction(). Ein Aufruf von OnTrade() entspricht einem oder mehreren Aufrufen von OnTradeTransaction.

Beispiel-EA, der die Funktion OnTrade() verwendet

```
//+------------------------------------------------------------------+
//|                                                OnTrade_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
```

© 2000-2025, MetaQuotes Ltd.

```
#property version   "1.00"

input    int days=7;             // Umfang der Handelshistorie in Tagen
//--- Setzen der Grenzen für die Handelshistorie in globaler Umgebung
datetime     start;              // Anfangsdatum der Handelshistorie im Cache
datetime     end;                // Enddatum der Handelshistorie im Cache
//--- Globale Zähler
int          orders;             // Anzahl der aktiven Aufträge
int          positions;          // Anzahl der offenen Positionen
int          deals;              // Anzahl der Deals in der Handelshistorie im Cache
int          history_orders;     // Anzahl der Aufträge in der Handelshistorie
bool         started=false;      // Flag der Relevanz des Zählers

//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   end=TimeCurrent();
   start=end-days*PeriodSeconds(PERIOD_D1);
   PrintFormat("Limits of the history to be loaded: start - %s, end - %s",
               TimeToString(start),TimeToString(end));
   InitCounters();
//---
   return(0);
  }
//+------------------------------------------------------------------+
//|  Initialisierung Zähler von Position, Auftrag und Deals          |
//+------------------------------------------------------------------+
void InitCounters()
  {
   ResetLastError();
//--- Laden der Historie
   bool selected=HistorySelect(start,end);
   if(!selected)
     {
      PrintFormat("%s. Failed to load history from %s to %s to cache. Error code: %d",
                  __FUNCTION__,TimeToString(start),TimeToString(end),GetLastError());
      return;
     }
//--- Abfrage des aktuellen Wert
   orders=OrdersTotal();
   positions=PositionsTotal();
   deals=HistoryDealsTotal();
   history_orders=HistoryOrdersTotal();
   started=true;
   Print("Counters of orders, positions and deals successfully initialized");
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
   if(started) SimpleTradeProcessor();
   else InitCounters();
  }
//+------------------------------------------------------------------+
//| Aufruf wenn ein Handelsereignis eintritt                         |
//+------------------------------------------------------------------+
void OnTrade()
  {
   if(started) SimpleTradeProcessor();
   else InitCounters();
  }
//+------------------------------------------------------------------+
//| Beispiel für das Bearbeiten der Änderungen im Handel und Historie|
//+------------------------------------------------------------------+
void SimpleTradeProcessor()
  {
   end=TimeCurrent();
   ResetLastError();
//--- Laden der Handelshistorie des bestimmten Intervalls in den Cache des Programms
   bool selected=HistorySelect(start,end);
   if(!selected)
     {
      PrintFormat("%s. Failed to load history from %s to %s to cache. Error code: %d",
                  __FUNCTION__,TimeToString(start),TimeToString(end),GetLastError());
      return;
     }
//--- Abfrage der aktuellen Wert
   int curr_orders=OrdersTotal();
   int curr_positions=PositionsTotal();
   int curr_deals=HistoryDealsTotal();
   int curr_history_orders=HistoryOrdersTotal();
//--- Prüfen, ob sich die Zahl der aktiven Aufträge verändert hat
   if(curr_orders!=orders)
     {
      //--- Anzahl der aktiven Aufträge hat sich geändert
      PrintFormat("Number of orders has been changed. Previous value is %d, current va
                  orders,curr_orders);
      //--- Aktualisieren des Wertes
      orders=curr_orders;
     }
//--- Ändern der Anzahl der offenen Position
   if(curr_positions!=positions)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Anzahl der offenen Positionen hat sich geändert
      PrintFormat("Number of positions has been changed. Previous value is %d, current
                  positions,curr_positions);
      //--- Aktualisieren des Wertes
      positions=curr_positions;
     }
//--- Veränderung der Anzahl der Deals im Cache der Handelshistorie
   if(curr_deals!=deals)
     {
      //--- Anzahl der Deals in der Handelshistorie wurde geändert
      PrintFormat("Number of deals has been changed. Previous value is %d, current val
                  deals,curr_deals);
      //--- Aktualisieren des Wertes
      deals=curr_deals;
     }
//--- Veränderung der Anzahl der historischen Orders im Cache der Handelshistorie
   if(curr_history_orders!=history_orders)
     {
      //--- Anzahl der historischen Aufträge der Handelshistorie im Cache wurde geände
      PrintFormat("Number of orders in history has been changed. Previous value is %d,
                  history_orders,curr_history_orders);
     //--- Aktualisieren des Wertes
     history_orders=curr_history_orders;
     }
//--- Prüfen, ob es notwendig ist die Grenzen der Handelshistorie im Cache zu verände
   CheckStartDateInTradeHistory();
  }
//+------------------------------------------------------------------+
//|  Ändern des Anfangsdatum zur Anforderung der Handelshistorie     |
//+------------------------------------------------------------------+
void CheckStartDateInTradeHistory()
  {
//--- erstes Intervall, falls jetzt mit der Arbeit begonnen werden soll
   datetime curr_start=TimeCurrent()-days*PeriodSeconds(PERIOD_D1);
//--- Sicherstellen, dass die Anfangsdatum der Handelshistorie nicht mehr als
//--- ein Tag über dem intendierten liegt
   if(curr_start-start>PeriodSeconds(PERIOD_D1))
     {
      //--- Korrektes Anfangsdatum der Historie für den Cache
      start=curr_start;
      PrintFormat("New start limit of the trade history to be loaded: start => %s",
                  TimeToString(start));
      //--- jetzt die Handel neu laden, um das Intervall zu aktualisieren
      HistorySelect(start,end);
      //--- Korrigieren der Zähler der Deals und der Aufträge in der Historie für spät
      history_orders=HistoryOrdersTotal();
      deals=HistoryDealsTotal();
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
/* Sample output:
  Limits of the history to be loaded: start - 2018.07.16 18:11, end - 2018.07.23 18:11
  The counters of orders, positions and deals are successfully initialized
  Number of orders has been changed. Previous value 0, current value 1
  Number of orders has been changed. Previous value 1, current value 0
  Number of positions has been changed. Previous value 0, current value 1
  Number of deals has been changed. Previous value 0, current value 1
  Number of orders in the history has been changed. Previous value 0, current value 1
*/
```

Siehe auch

OrderSend, OrderSendAsync, OnTradeTransaction, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

OnTradeTransaktion

Die Funktion wird in EAs aufgerufen, wenn das Ereignis TradeTransaction eintritt. Die Funktion ist für die Behandlung der Ergebnisse der Ausführung von Handelsanfragen gedacht.

```
void  OnTradeTransaction()
   const MqlTradeTransaction&    trans,     // Struktur der Handelstransaktionen
   const MqlTradeRequest&        request,   // Abfrage der Struktur
   const MqlTradeResult&         result     // Ergebnis der Struktur
   );
```

Parameter

trans

[in] MqlTradeTransaction Typvariable, die eine auf einem Handelskonto getätigte Transaktion beschreibt.

request

[in] MqlTradeRequest Typvariable, die eine Handelsanforderung beschreibt, die zu einer Transaktion führte. Sie enthält nur die Werte für den Typ Transaktion TRADE_TRANSACTION_REQUEST.

result

[in] MqlTradeResult Typvariable, die das Ausführungsergebnis einer Handelsanforderung enthält, die zu einer Transaktion führte. Sie enthält nur die Werte für den Typ Transaktion TRADE_TRANSACTION_REQUEST.

Rückgabewert

Kein Rückgabewert

Hinweis

OnTradeTransaction() wird aufgerufen, um das Ereignis TradeTransaction zu behandeln, das vom Handelsserver an das Terminal in den folgenden Fällen gesendet wird:

- Senden einer Handelsanfrage aus einem MQL5-Programm mit den Funktionen OrderSend()/OrderSendAsync() und deren anschließende Ausführung;
- Manuelles Senden einer Handelsanfrage über die GUI und deren anschließende Ausführung;
- Aktivierung von Pending und Stop Orders auf dem Server;
- Durchführung von Operationen auf der Seite des Handelsservers.

Die Daten zum Transaktionstyp sind im Feld *type* der Variablen *trans* eingetragen. Die Arten von Handelsgeschäften sind beschrieben in der Enumeration ENUM_TRADE_TRANSACTION_TYPE:

- TRADE_TRANSACTION_ORDER_ADD - Hinzufügen eines neuen Auftrages (order)
- TRADE_TRANSACTION_ORDER_UPDATE - Ändern eines bestehenden Auftrags
- TRADE_TRANSACTION_ORDER_DELETE - Löschen eines Auftrags aus der Liste der aktiven
- TRADE_TRANSACTION_DEAL_ADD - Hinzufügen eines Deals zur Historie
- TRADE_TRANSACTION_DEAL_UPDATE - Ändern einer Transaktion (deal) in der Historie
- TRADE_TRANSACTION_DEAL_DELETE - Löschen einer Transaktion aus der Historie
- TRADE_TRANSACTION_HISTORY_ADD - Hinzufügen eines Auftrages zur Historie als Ergebnis einer Ausführung oder einer Stornierung

© 2000-2025, MetaQuotes Ltd.

- TRADE_TRANSACTION_HISTORY_UPDATE - Ändern eines Auftrags in die Auftragshistorie
- TRADE_TRANSACTION_HISTORY_DELETE - Löschen eines Auftrags aus der Auftragsliste
- TRADE_TRANSACTION_POSITION - Änderung einer Position, die nicht auf einer Handelsausführung basiert
- TRADE_TRANSACTION_REQUEST - Benachrichtigung, dass eine Handelsanfrage vom Server bearbeitet wurde und das Ergebnis der Bearbeitung eingegangen ist.

Wenn Transaktionen vom Typ TRADE_TRANSACTION_REQUEST behandelt werden, ist es notwendig, den zweiten und dritten Parameter der Funktion OnTradeTransaction() zu analysieren - *request* und *result* - um zusätzliche Informationen zu erhalten..

Das Senden einer Kaufanfrage führt zu einer Kette von Handelstransaktionen auf einem Handelskonto: 1) Anfrage wird zur Bearbeitung angenommen, 2) ein entsprechender Auftrag für das Konto angelegt, 3) der Auftrag wird dann ausgeführt, 4) der ausgeführte Auftrag wird aus der Liste der aktiven Aufträge entfernt, 5) zur Historie der Aufträge hinzugefügt, 6) die nachfolgende Transaktion wird zur Historie hinzugefügt und 7) eine neue Position wird angelegt. Alle diese Stufen sind Handelstransaktionen. Die Ankunft jeder solchen Transaktion im Terminal ist das Ereignis TradeTransaction. Die Priorität der Ankunft dieser Transaktionen im Terminal ist nicht garantiert. Daher sollten Sie bei der Entwicklung Ihres Handelsalgorithmus nicht erwarten, dass eine Gruppe von Transaktionen nach der anderen eintrifft.

Wenn Transaktionen von der Funktion OnTradeTransaction() des EA verarbeitet werden, verarbeitet das Terminal die eingehenden Handelstransaktionen weiter. So kann sich der Status des Handelskontos im Laufe der OnTradeTransaction() Operation ändern. Zum Beispiel, während ein MQL5-Programm eine neuen Auftrag erteilt, kann er ausgeführt, aus der Liste der offenen Aufträge gelöscht und in die Historie verschoben werden. Das Programm wird über alle diese Ereignisse informiert.

Die Länge der Warteschlange der Transaktionen umfasst 1024 Elemente. Wenn OnTradeTransaction() bei einer weiteren Transaktion zu lange verweilt, können in der Warteschlange die frühere Transaktionen durch neue ersetzt werden.

Ein Aufruf von OnTrade() entspricht einem oder mehreren OnTradeTransaction-Aufrufen. Im Allgemeinen gibt es keine genaue Korrelation zwischen der Anzahl der Aufrufe von OnTrade() und OnTradeTransaction(). Ein Aufruf von OnTrade() entspricht einem oder mehreren Aufrufen von OnTradeTransaction.

Jedes Ergebnis Trade kann als Ergebnis einer oder mehrerer Anfragen an den Handelsserver erscheinen. Handelsanfrage werden mit Trade mittels OrderSend() oder OrderSendAsync() an der Server gesendet. Jede Anfrage kann zu mehreren Handelsereignissen führen. Sie können nicht der Vorstellung vertrauen: "Eine Anfrage - Ein Handelsereignis", da die Verarbeitung von Ereignissen in mehreren Stufen erfolgt kann und jede Operation den Status von Aufträgen, Positionen und die Handelsgeschichte verändern kann.

Beispiel-EA, der die Funktion OnTradeTransaction() verwendet

```
//+------------------------------------------------------------------+
//|                                    OnTradeTransaction_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Sample listener of TradeTransaction events"
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   PrintFormat("LAST PING=%.f ms",
               TerminalInfoInteger(TERMINAL_PING_LAST)/1000.);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
//| TradeTransaction Funktion                                        |
//+------------------------------------------------------------------+
void OnTradeTransaction(const MqlTradeTransaction &trans,
                        const MqlTradeRequest &request,
                        const MqlTradeResult &result)
  {
//---
   static int counter=0;   // Zähler der Aufrufe von OnTradeTransaction()
   static uint lasttime=0; // Zeitpunkt des letzten Aufrufs von OnTradeTransaction()
//---
   uint time=GetTickCount();
//--- wenn die letzte Transaktion vor mehr als 1 Sekunde war
   if(time-lasttime>1000)
     {
      counter=0; // das ist jetzt eine neue Handelsoperation und der Zähler kann zurü
      if(IS_DEBUG_MODE)
         Print(" New trade operation");
     }
   lasttime=time;
   counter++;
   Print(counter,". ",__FUNCTION__);
//--- Ergebnis der Ausführungsanfrage eines Handels
   ulong            lastOrderID   =trans.order;
   ENUM_ORDER_TYPE  lastOrderType =trans.order_type;
   ENUM_ORDER_STATE lastOrderState=trans.order_state;
//--- Name des Symbols der Transaktion
   string trans_symbol=trans.symbol;
//--- Typ der Transaktion
```

© 2000-2025, MetaQuotes Ltd.

```
   ENUM_TRADE_TRANSACTION_TYPE  trans_type=trans.type;
   switch(trans.type)
     {
      case  TRADE_TRANSACTION_POSITION:    // Positionsänderung
        {
         ulong pos_ID=trans.position;
         PrintFormat("MqlTradeTransaction: Position  #%I64u %s modified: SL=%.5f TP=%.
                     pos_ID,trans_symbol,trans.price_sl,trans.price_tp);
        }
      break;
      case TRADE_TRANSACTION_REQUEST:      // Senden einer Handelsanfrage
         PrintFormat("MqlTradeTransaction: TRADE_TRANSACTION_REQUEST");
         break;
      case TRADE_TRANSACTION_DEAL_ADD:     // Hinzufügen eines Deals
        {
         ulong           lastDealID   =trans.deal;
         ENUM_DEAL_TYPE lastDealType =trans.deal_type;
         double         lastDealVolume=trans.volume;
         //--- ID des Deals im internen System - eine Ticketnummer wird vom Börsenplat
         string Exchange_ticket="";
         if(HistoryDealSelect(lastDealID))
            Exchange_ticket=HistoryDealGetString(lastDealID,DEAL_EXTERNAL_ID);
         if(Exchange_ticket!="")
            Exchange_ticket=StringFormat("(Exchange deal=%s)",Exchange_ticket);

         PrintFormat("MqlTradeTransaction: %s deal #%I64u %s %s %.2f lot   %s",EnumToS
                     lastDealID,EnumToString(lastDealType),trans_symbol,lastDealVolume
        }
      break;
      case TRADE_TRANSACTION_HISTORY_ADD: // Hinzufügen eines Auftrages zur Historie
        {
         //--- Auftrags-ID im internen System - eine Ticketnummer wird vom Börsenplatz
         string Exchange_ticket="";
         if(lastOrderState==ORDER_STATE_FILLED)
           {
            if(HistoryOrderSelect(lastOrderID))
               Exchange_ticket=HistoryOrderGetString(lastOrderID,ORDER_EXTERNAL_ID);
            if(Exchange_ticket!="")
               Exchange_ticket=StringFormat("(Exchange ticket=%s)",Exchange_ticket);
           }
         PrintFormat("MqlTradeTransaction: %s order #%I64u %s %s %s   %s",EnumToStrin
                     lastOrderID,EnumToString(lastOrderType),trans_symbol,EnumToStrin
        }
      break;
      default: // andere Transaktionen
        {
         //--- Auftrags-ID im internen System - eine Ticketnummer wird vom Börsenplatz
         string Exchange_ticket="";
         if(lastOrderState==ORDER_STATE_PLACED)
```

© 2000-2025, MetaQuotes Ltd.

```
            {
             if(OrderSelect(lastOrderID))
                Exchange_ticket=OrderGetString(ORDER_EXTERNAL_ID);
             if(Exchange_ticket!="")
                Exchange_ticket=StringFormat("Exchange ticket=%s",Exchange_ticket);
            }
          PrintFormat("MqlTradeTransaction: %s order #%I64u %s %s   %s",EnumToString(tr
                      lastOrderID,EnumToString(lastOrderType),EnumToString(lastOrderSta
        }
       break;
      }
//--- Ticketnummer des Auftrags
   ulong orderID_result=result.order;
   string retcode_result=GetRetcodeID(result.retcode);
   if(orderID_result!=0)
      PrintFormat("MqlTradeResult: order #%d retcode=%s ",orderID_result,retcode_resul
//---
  }
//+------------------------------------------------------------------+
//| Konvertieren eines numerischen Codes in eine Textbeschreibung    |
//+------------------------------------------------------------------+
string GetRetcodeID(int retcode)
  {
   switch(retcode)
     {
      case 10004: return("TRADE_RETCODE_REQUOTE");             break;
      case 10006: return("TRADE_RETCODE_REJECT");              break;
      case 10007: return("TRADE_RETCODE_CANCEL");              break;
      case 10008: return("TRADE_RETCODE_PLACED");              break;
      case 10009: return("TRADE_RETCODE_DONE");                break;
      case 10010: return("TRADE_RETCODE_DONE_PARTIAL");        break;
      case 10011: return("TRADE_RETCODE_ERROR");               break;
      case 10012: return("TRADE_RETCODE_TIMEOUT");             break;
      case 10013: return("TRADE_RETCODE_INVALID");             break;
      case 10014: return("TRADE_RETCODE_INVALID_VOLUME");      break;
      case 10015: return("TRADE_RETCODE_INVALID_PRICE");       break;
      case 10016: return("TRADE_RETCODE_INVALID_STOPS");       break;
      case 10017: return("TRADE_RETCODE_TRADE_DISABLED");      break;
      case 10018: return("TRADE_RETCODE_MARKET_CLOSED");       break;
      case 10019: return("TRADE_RETCODE_NO_MONEY");            break;
      case 10020: return("TRADE_RETCODE_PRICE_CHANGED");       break;
      case 10021: return("TRADE_RETCODE_PRICE_OFF");           break;
      case 10022: return("TRADE_RETCODE_INVALID_EXPIRATION");  break;
      case 10023: return("TRADE_RETCODE_ORDER_CHANGED");       break;
      case 10024: return("TRADE_RETCODE_TOO_MANY_REQUESTS");   break;
      case 10025: return("TRADE_RETCODE_NO_CHANGES");          break;
      case 10026: return("TRADE_RETCODE_SERVER_DISABLES_AT");  break;
      case 10027: return("TRADE_RETCODE_CLIENT_DISABLES_AT");  break;
      case 10028: return("TRADE_RETCODE_LOCKED");              break;
```

© 2000-2025, MetaQuotes Ltd.

```
      case 10029: return("TRADE_RETCODE_FROZEN");              break;
      case 10030: return("TRADE_RETCODE_INVALID_FILL");        break;
      case 10031: return("TRADE_RETCODE_CONNECTION");          break;
      case 10032: return("TRADE_RETCODE_ONLY_REAL");           break;
      case 10033: return("TRADE_RETCODE_LIMIT_ORDERS");        break;
      case 10034: return("TRADE_RETCODE_LIMIT_VOLUME");        break;
      case 10035: return("TRADE_RETCODE_INVALID_ORDER");       break;
      case 10036: return("TRADE_RETCODE_POSITION_CLOSED");     break;
      default:
         return("TRADE_RETCODE_UNKNOWN="+IntegerToString(retcode));
         break;
     }
//---
  }
```

Siehe auch

OrderSend, OrderSendAsync, OnTradeTransaction, Trade request structure, Trade transaction structure, Trade transaction types, Trade operation types, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

OnBookEvent

Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis BookEvent eintritt. Sie dient zur Bearbeitung der Änderungen der Markttiefe.

```
void  OnBookEvent(
   const string&  symbol          // Symbol
   );
```

Parameter

symbol

[in] Name des Symbols für das ein BookEvent eingetreten ist

Rückgabewert

Kein Rückgabewert

Hinweis

Um die Ereignisse BookEvent für ein beliebiges Symbol zu erhalten, abonnieren Sie es einfach mit der Funktion MarketBookAdd(). Um das Abonnement für den Empfang des BookEvent für ein bestimmtes Symbol zu kündigen, rufen Sie die Funktion MarketBookRelease() auf.

Das BookEvent sendet im gesamten Chart. Das bedeutet, dass wen eine Anwendung auf einem Chart BookEvent mit der Funktion MarketBookAdd abboniert hat, alle anderen Indikatoren und EAs, die auf dem gleichen Chart gestartet wurden, auch den Handler von OnBookEvent() haben und diese Ereignisse erhalten. Daher ist es notwendig, einen Symbolnamen zu analysieren, der den Handler OnBookEvent() durch den Parameter *symbol* übergeben wird.

Separate BookEvent-Zähler, sortiert nach Symbolen, stehen für alle Anwendungen zur Verfügung, die auf dem gleichen Chart laufen. Dies bedeutet, dass jedes Chart mehrere Abonnements für verschiedene Symbole haben kann, und für jedes Symbol wird ein Zähler bereitgestellt. Das An- und Abmelden von BookEvent ändert den Abonnementzähler für bestimmte Symbole nur innerhalb eines Charts. Mit anderen Worten, es können zwei benachbarte Charts zum BookEvent für das gleiche Symbol, aber unterschiedliche Abonnementzählerwerte vorhanden sein.

Der anfängliche Zählerstand des Abonnements ist Null. Bei jedem Aufruf von MarketBookAdd() wird der Abonnementzähler für ein bestimmtes Symbol auf dem Chart um eins erhöht (Chartsymbol und Symbol in MarketBookAdd() müssen nicht übereinstimmen). Beim Aufruf von MarketBookRelease() wird der Zähler der Abonnements für ein bestimmtes Symbol im Chart um eins verringert. Die Ereignisse von BookEvent für ein beliebiges Symbol werden innerhalb des Charts gesendet, bis der Zähler gleich Null ist. Daher ist es wichtig, dass jedes MQL5-Programm, das MarketBookAdd() Aufrufe enthält, sich am Ende seiner Arbeit korrekt vom Erhalten von Ereignissen für jedes Symbol mit MarketBookRelease() abmeldet. Um dies zu erreichen, sollte die Anzahl der MarketBookAdd() und MarketBookRelease()-Aufrufe für jeden Aufruf während der gesamten MQL5-Programmlaufzeit gleich sein. Die Verwendung von Flags oder benutzerdefinierten Subskriptionszählern innerhalb des Programms ermöglicht es Ihnen, sicher mit BookEvent-Ereignissen zu arbeiten und verhindert das Deaktivieren von Subskriptionen, um dieses Ereignis in Drittanbieterprogrammen innerhalb derselben Tabelle zu erhalten.

BookEvent Ereignisse werden nie übersprungen und immer in eine Warteschlange gestellt, auch wenn die Behandlung des vorherigen BookEvents noch nicht beendet ist. .

© 2000-2025, MetaQuotes Ltd.

Beispiel

```
//+------------------------------------------------------------------+
//|                                            OnBookEvent_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com/en/articles/2635"
#property version   "1.00"
#property description "Example of measuring the market depth refresh rate using OnBook
#property description "The code is taken from the article https://www.mql5.com/en/arti
//--- Eingabeparameter
input ulong ExtCollectTime   =30;  // Testdauer in Sekunden
input ulong ExtSkipFirstTicks=10;  // Anzahl der Ticks die nach dem Start ignoriert we
//--- Flag des Abonnements des BookEvents
bool book_subscribed=false;
//--- Array der akzeptierten Anfragen der Markttiefe
MqlBookInfo  book[];
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Zeige den Start
   Comment(StringFormat("Waiting for the first %I64u ticks to arrive",ExtSkipFirstTick
   PrintFormat("Waiting for the first %I64u ticks to arrive",ExtSkipFirstTicks);
//--- aktiviere die Verbreitung der Markttiefe
   if(MarketBookAdd(_Symbol))
     {
      book_subscribed=true;
      PrintFormat("%s: MarketBookAdd(%s) function returned true",__FUNCTION__,_Symbol)
     }
   else
      PrintFormat("%s: MarketBookAdd(%s) function returned false! GetLastError()=%d",_
//--- erfolgreiche Initialisierung
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierung des Experten                                   |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Anzeige der Ursache der Deinitialisierung
   Print(__FUNCTION__,": Deinitialization reason code = ",reason);
//--- stornieren des Abonnements der Markttiefe
   if(book_subscribed)
     {
      if(!MarketBookRelease(_Symbol))
```

© 2000-2025, MetaQuotes Ltd.

```
         PrintFormat("%s: MarketBookRelease(%s) returned false! GetLastError()=%d",_Sy
      else
         book_subscribed=false;
     }
//---
  }
//+------------------------------------------------------------------+
//| BookEvent Funktion                                               |
//+------------------------------------------------------------------+
void OnBookEvent(const string &symbol)
  {
   static ulong starttime=0;              // Beginn des Tests
   static ulong tickcounter=0;            // Aktualisierung des Zählers der Markttiefe
//--- Arbeiten mit der Markttiefe, aber nur wenn wir sie selbst abonniert haben
   if(!book_subscribed)
      return;
//--- zählen der Updates für nur ein bestimmtes Symbol
   if(symbol!=_Symbol)
      return;
//--- Ignorieren der ersten Ticks um die Warteschlange zu leeren und zur Vorbereitung
   tickcounter++;
   if(tickcounter<ExtSkipFirstTicks)
      return;
//--- sichern der Startzeit
   if(tickcounter==ExtSkipFirstTicks)
      starttime=GetMicrosecondCount();
//--- Abfrage der Daten der Markttiefe
   MarketBookGet(symbol,book);
//--- wann stoppen?
   ulong endtime=GetMicrosecondCount()-starttime;
   ulong ticks  =1+tickcounter-ExtSkipFirstTicks;
// wie viel Zeit in Microsekunden ist seit Beginn des Tests vergangen?
   if(endtime>ExtCollectTime*1000*1000)
     {
      PrintFormat("%I64u ticks for %.1f seconds: %.1f ticks/sec ",ticks,endtime/1000.0
      ExpertRemove();
      return;
     }
//--- Anzeige des Zählers im Kommentarfeld
   if(endtime>0)
      Comment(StringFormat("%I64u ticks for %.1f seconds: %.1f ticks/sec ",ticks,endti
  }
```

Siehe auch

MarketBookAdd, MarketBookRelease, MarketBookGet, OnTrade, OnTradeTransaction, OnTick, Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

OnChartEvent

Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis ChartEvent eintritt. Die Funktion reagiert auf Änderungen im Chart, die vom Nutzer oder einem MQL5-Programm verursacht werden.

```
void  OnChartEvent()
   const int       id,       // Ereignis ID
   const long&     lparam,   // Ereignisparameter vom Typ long
   const double&   dparam,   // Ereignisparameter vom Typ double
   const string&   sparam    // Ereignisparameter vom Typ string
   );
```

Parameter

id

[in] Ereignis-ID aus der Enumeration ENUM_CHART_EVENT.

lparam

[in] long Typ Ereignisparameter

dparam

[in] double Typ Ereignisparameter

sparam

[in] string Typ Ereignisparameter

Rückgabewert

Kein Rückgabewert

Hinweis

Es gibt 11 Typen von Ereignissen, die mit der vordefinierten Funktion OnChartEvent() behandelt werden können. 65535 IDs von CHARTEVENT_CUSTOM bis einschließlich CHARTEVENT_CUSTOM_LAST werden für benutzerdefinierte Ereignisse bereitgestellt. Um ein benutzerdefiniertes Ereignis zu erzeugen, verwenden Sie die Funktion EventChartCustom().

Kurze Ereignisbeschreibung aus der Enumeration ENUM_CHART_EVENT:

- CHARTEVENT_KEYDOWN - Drücken einer Taste auf der Tastatur, wenn ein Chartfenster im Fokus ist;
- CHARTEVENT_MOUSE_MOVE - Bewegen der Maus und der Maustastenklicks (wenn CHART_EVENT_MOUSE_MOVE=true eines Charts);
- CHARTEVENT_OBJECT_CREATE - erzeugt ein graphisches Objekt (wenn CHART_EVENT_OBJECT_CREATE=true eines Charts);
- CHARTEVENT_OBJECT_CHANGE - Objekteigenschaften über den Eigenschaftsdialog ändern;
- CHARTEVENT_OBJECT_DELETE - ein grafisches Objekt löschen (wenn CHART_EVENT_OBJECT_DELETE=true eines Charts);
- CHARTEVENT_CLICK - Klicken auf ein Chart;
- CHARTEVENT_OBJECT_CLICK - Mausklick auf ein grafisches Objekt eines Charts;
- CHARTEVENT_OBJECT_DRAG - Ziehen eines grafischen Objekts mit der Maus;

© 2000-2025, MetaQuotes Ltd.

- CHARTEVENT_OBJECT_ENDEDIT - Beendet die Bearbeitung von Text im Eingabefeld Editieren eines grafischen Objekts (OBJ_EDIT);
- CHARTEVENT_CHART_CHANGE - ändert ein Chart;
- CHARTEVENT_CUSTOM+n - eigene Ereignis-ID, wobei n im Bereich von 0 bis 65535 liegt. CHARTEVENT_CUSTOM_LAST enthält die letzte akzeptable benutzerdefinierte Ereignis-ID (CHARTEVENT_CUSTOM+65535).

Alle MQL5-Programme arbeiten in anderen Threads als dem Haupt-Thread der Anwendung. Der Hauptanwendungs-Thread ist für die Behandlung aller Windows-Systemmeldungen zuständig und generiert daraus wiederum Windows-Meldungen für die eigene Anwendung. Wenn Sie beispielsweise die Maus über ein Diagramm bewegen (Ereignis WM_MOUSE_MOVE), werden mehrere Systemmeldungen für das spätere Rendern des Anwendungsfensters erzeugt und auch interne Meldungen an Experten und Indikatoren gesendet, die auf dem Diagramm gestartet werden. Es kann vorkommen, dass der Hauptanwendungs-Thread die Systemmeldung WM_PAINT noch nicht verarbeitet hat (und somit das modifizierte Diagramm noch nicht gerendert hat), während ein EA oder ein Indikator das Mausbewegungsereignis bereits empfangen hat. In diesem Fall wird die Chart-Eigenschaft CHART_FIRST_VISIBLE_BAR erst nach der Darstellung des Charts geändert.

Für jeden Ereignistyp haben die Eingänge des OnChartEvent() Befehls bestimmte Werte, die für die Behandlung dieses Ereignisses notwendig sind. Die Tabelle listet Ereignisse und Werte auf, die mit den Parametern übergeben werden.

Ereignis	'id' Parameterwert	'lparam' Parameterwert	'dparam' Parameterwert	'sparam' Parameterwert
Tastendruckereignis	CHARTEVENT_KEYDOWN	gedrückter Tastencode	Die Anzahl der Tastendrücke, die erzeugt wurden, während die Taste im gedrückten Zustand gehalten wurde.	Wert des Strings der Bitmaske, der den Status der Tastaturtasten beschreibt.
Mausereignisse (wenn CHART_EVENT_MOUSE_MOVE=wahr eines Charts)	CHARTEVENT_MOUSE_MOVE	X Koordinate	Y Koordinate	Wert des Strings der Bitmaske, der den Status der Maustasten beschreibt.
Mausrad-Ereignis (wenn CHART_EVENT_MOUSE_WHEEL=true eines Charts)	CHARTEVENT_MOUSE_WHEEL	Zustände der Tasten und Maustasten, X- und Y-Koordinaten des Cursors. Siehe die Beschreibung im Beispiel.	Der Delta-Wert des Mausrads beim Drehen	—
Erstellen von Grafikobjekten	CHARTEVENT_OBJECT_CREATE	—	—	Name des erstellten

© 2000-2025, MetaQuotes Ltd.

Ereignis	'id' Parameterwert	'lparam' Parameterwert	'dparam' Parameterwert	'sparam' Parameterwert
(wenn CHART_EVENT_OBJECT_CREATE =true eines Charts)				Grafikobjektes
Änderungen der Objekteigenschaften über den Eigenschaftendialog	CHARTEVENT_OBJECT_CHANGE	—	—	Name des geänderten Grafikobjekts
Entfernt ein Grafikobjekt (wenn CHART_EVENT_OBJECT_DELETE =true eines Charts)	CHARTEVENT_OBJECT_DELETE	—	—	Name eines entfernten Grafikobjekts
Mausklick auf einem Chart	CHARTEVENT_CLICK	X Koordinate	Y Koordinate	—
Mausklick auf einem Grafikobjekt	CHARTEVENT_OBJECT_CLICK	X Koordinate	Y Koordinate	Name des Grafikobjektes, dessen Ereignis auftrat
Verschieben eines Grafikobjekts mit der Maus	CHARTEVENT_OBJECT_DRAG	—	—	Name des verschobenen Grafikobjekts
Beenden eine Textänderung im Grafikobjekt "Eingabefeld"	CHARTEVENT_OBJECT_ENDEDIT	—	—	Name des Grafikobjekts "Eingabefeld", in dem die Bearbeitung beendet wurde
Ändern der Größe oder der Eigenschaften des Charts über den Eigenschaftendialog	CHARTEVENT_CHART_CHANGE	—	—	—
Benutzerdefiniertes Ereignis mit der Nummer N	CHARTEVENT_CUSTOM+N	Wert, definiert mit der Funktion	Wert, definiert mit der Funktion	Wert, definiert mit der Funktion

© 2000-2025, MetaQuotes Ltd.

Ereignis	'id' Parameterwert	'lparam' Parameterwert	'dparam' Parameterwert	'sparam' Parameterwert
		EventChartCustom()	EventChartCustom()	EventChartCustom()

Beispielchart des 'listener' von Ereignissen:

```
//+------------------------------------------------------------------+
//|                                          OnChartEvent_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Sample chart event listener and custom events generator"
//--- IDs der Servicetasten
#define KEY_NUMPAD_5       12
#define KEY_LEFT           37
#define KEY_UP             38
#define KEY_RIGHT          39
#define KEY_DOWN           40
#define KEY_NUMLOCK_DOWN   98
#define KEY_NUMLOCK_LEFT  100
#define KEY_NUMLOCK_5     101
#define KEY_NUMLOCK_RIGHT 102
#define KEY_NUMLOCK_UP    104
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Anzeige des Wertes von CHARTEVENT_CUSTOM
   Print("CHARTEVENT_CUSTOM=",CHARTEVENT_CUSTOM);
//---
   Print("Launched the EA ",MQLInfoString(MQL5_PROGRAM_NAME));
//--- Setzen des Flags des empfangenen Ereignisses zum Erstellen des Chartobjekts
   ChartSetInteger(ChartID(),CHART_EVENT_OBJECT_CREATE,true);
//--- Setzen des Flags des empfangenen Ereignisses zum Entfernen des Chartobjekts
   ChartSetInteger(ChartID(),CHART_EVENT_OBJECT_DELETE,true);
//--- Aktivieren des Mausrads zum Blättern in den Nachrichten
   ChartSetInteger(0,CHART_EVENT_MOUSE_WHEEL,1);
//--- erzwungenes Update der Eigenschaften des Charts, um die Bereitschaft für die Ere
   ChartRedraw();
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- Tick-Zähler für das Erstellen eine Nutzerereignisses
   static int tick_counter=0;
//--- Teilen der gesammelten Ticks durch diesen Wert
   int simple_number=113;
//---
   tick_counter++;
//--- Senden eines Nutzerereignisses, wenn der Tick-Zähler ein Vielfaches von simple_n
   if(tick_counter%simple_number==0)
     {
      //--- Bilden der ID eines Nutzer-Ereignisses von 0 bis 65535
      ushort custom_event_id=ushort(tick_counter%65535);
      //---  Senden eines Nutzerereignisses mit gültigen Parametern
      EventChartCustom(ChartID(),custom_event_id,tick_counter,SymbolInfoDouble(Symbol
      //--- Eintrag im Log für die Analyse der Ergebnisse des Beispiel
      Print(__FUNCTION__,": Sent a custom event ID=",custom_event_id);
     }
//---
  }
//+------------------------------------------------------------------+
//| ChartEvent Funktion                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
//--- Tastendruck
   if(id==CHARTEVENT_KEYDOWN)
     {
      switch((int)lparam)
        {
         case KEY_NUMLOCK_LEFT:  Print("Pressed KEY_NUMLOCK_LEFT");   break;
         case KEY_LEFT:          Print("Pressed KEY_LEFT");           break;
         case KEY_NUMLOCK_UP:    Print("Pressed KEY_NUMLOCK_UP");     break;
         case KEY_UP:            Print("Pressed KEY_UP");             break;
         case KEY_NUMLOCK_RIGHT: Print("Pressed KEY_NUMLOCK_RIGHT");  break;
         case KEY_RIGHT:         Print("Pressed KEY_RIGHT");          break;
         case KEY_NUMLOCK_DOWN:  Print("Pressed KEY_NUMLOCK_DOWN");   break;
         case KEY_DOWN:          Print("Pressed KEY_DOWN");           break;
         case KEY_NUMPAD_5:      Print("Pressed KEY_NUMPAD_5");       break;
         case KEY_NUMLOCK_5:     Print("Pressed KEY_NUMLOCK_5");      break;
         default:                Print("Pressed unlisted key");
        }
     }
//--- Linksklick auf dem Chart
```

© 2000-2025, MetaQuotes Ltd.

```
   if(id==CHARTEVENT_CLICK)
      Print("Mouse click coordinates on a chart: x = ",lparam,"  y = ",dparam);
//--- Klick auf einem Grafikobjekt
   if(id==CHARTEVENT_OBJECT_CLICK)
      Print("Clicking a mouse button on an object named '"+sparam+"'");
//--- Objekt entfernen
   if(id==CHARTEVENT_OBJECT_DELETE)
      Print("Removed object named ",sparam);
//--- Objekt erstellen
   if(id==CHARTEVENT_OBJECT_CREATE)
      Print("Created object named ",sparam);
//--- Ändern des Objekts
   if(id==CHARTEVENT_OBJECT_CHANGE)
      Print("Changed object named ",sparam);
//--- Objekt verschieben oder die Punkt-Koordinaten haben sich geändert
   if(id==CHARTEVENT_OBJECT_DRAG)
      Print("Changing anchor points of object named ",sparam);
//--- Geänderter Text im Eingabefeld eines editierbaren Grafikobjekts
   if(id==CHARTEVENT_OBJECT_ENDEDIT)
      Print("Changed text in Edit object ",sparam,"  id=",id);
//--- Ereignis einer Mausbewegung
   if(id==CHARTEVENT_MOUSE_MOVE)
      Comment("POINT: ",(int)lparam,",",(int)dparam,"\n",MouseState((uint)sparam));
   if(id==CHARTEVENT_MOUSE_WHEEL)
     {
      //--- Berücksichtigung der Maustasten und -rades von diesem Ereignis
      int flg_keys = (int)(lparam>>32);          // Flag des Zustandes der Tasten Ctrl
      int x_cursor = (int)(short)lparam;         // X Koordinate des Ereignisses des M
      int y_cursor = (int)(short)(lparam>>16);   // Y Koordinate des Ereignisses des M
      int delta    = (int)dparam;                // Gesamtwert des Mausrads, wenn die
      //--- Handhabung des Flags
      string str_keys="";
      if((flg_keys&0x0001)!=0)
         str_keys+="LMOUSE ";
      if((flg_keys&0x0002)!=0)
         str_keys+="RMOUSE ";
      if((flg_keys&0x0004)!=0)
         str_keys+="SHIFT ";
      if((flg_keys&0x0008)!=0)
         str_keys+="CTRL ";
      if((flg_keys&0x0010)!=0)
         str_keys+="MMOUSE ";
      if((flg_keys&0x0020)!=0)
         str_keys+="X1MOUSE ";
      if((flg_keys&0x0040)!=0)
         str_keys+="X2MOUSE ";

      if(str_keys!="")
         str_keys=", keys='"+StringSubstr(str_keys,0,StringLen(str_keys)-1)+"'";
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("%s: X=%d, Y=%d, delta=%d%s",EnumToString(CHARTEVENT_MOUSE_WHEEL),x_
     }
//--- Ereignis der Größenänderung des Charts oder Änderung der Eigenschaften des Chart
   if(id==CHARTEVENT_CHART_CHANGE)
      Print("Changing the chart size or properties");
//--- Nutzerereignis
   if(id>CHARTEVENT_CUSTOM)
      PrintFormat("Custom event ID=%d, lparam=%d, dparam=%G, sparam=%s",id,lparam,dpar
  }
//+------------------------------------------------------------------+
//| Zustand der Maus                                                 |
//+------------------------------------------------------------------+
string MouseState(uint state)
  {
   string res;
   res+="\nML: "   +(((state& 1)== 1)?"DN":"UP");   // Maus, links
   res+="\nMR: "   +(((state& 2)== 2)?"DN":"UP");   // Maus, rechts
   res+="\nMM: "   +(((state&16)==16)?"DN":"UP");   // Maus, mitte
   res+="\nMX: "   +(((state&32)==32)?"DN":"UP");   // Maus, erster X Taste
   res+="\nMY: "   +(((state&64)==64)?"DN":"UP");   // Maus, zweiter X Taste
   res+="\nSHIFT: "+(((state& 4)== 4)?"DN":"UP");   // Shift-Taste
   res+="\nCTRL: " +(((state& 8)== 8)?"DN":"UP");   // Ctrl-Taste
   return(res);
  }
```

Siehe auch

EventChartCustom, Typen der Chartereignisse, Ereignisbearbeiter, Durchführung der Programme, Ereignisse des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

OnTester

Die Funktion wird von EAs aufgerufen, wenn das Ereignis Tester auftritt, um notwendige Aktionen nach dem Test durchzuführen.

```
double  OnTester(void);
```

Rückgabewert

Der Wert des benutzerdefinierten Kriteriums, um auf das Testergebnis zuzugreifen.

Hinweis

Die Funktion OnTester() kann nur beim Testen von EAs verwendet werden und ist in erster Linie für die Berechnung eines Wertes gedacht, der als Kriterium 'Custom max' bei der Optimierung von Eingabeparametern verwendet wird.

Bei der genetischen Optimierung erfolgt die Sortierung innerhalb einer Generation in absteigender Reihenfolge. Dies bedeutet, dass die Ergebnisse mit dem höchsten Wert aus Sicht des Optimierungskriteriums als die besten angesehen werden. Die schlechtesten Werte für eine solche Sortierung werden am Ende platziert und anschließend verworfen. Deshalb beteiligen sie sich nicht an der Bildung der nächsten Generation.

Mit der Funktion OnTester() können Sie also nicht nur Ihre eigenen Testergebnisse erstellen und speichern, sondern auch den Optimierungsprozess steuern, um die besten Parameter der Handelsstrategie zu finden.

Unten ist ein **Beispiel** für die Berechnung einer benutzerdefinierten Kriterienoptimierung. Die Idee ist, die lineare Regression der Saldenkurve zu berechnen. Das wird im Artikel Optimieren einer Strategie unter Verwendung einer Kurve der Salden und dem Vergleich der Ergebnisse mit dem Kriterium "Balance + max Sharpe Ratio" beschrieben.

```
//+------------------------------------------------------------------+
//|                                                OnTester_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Sample EA with the OnTester() handler"
#property description "As a custom optimization criterion, "
#property description "the ratio of the balance graph linear regression"
#property description "divided by the deviation mean-square error is returned"
//--- Einbinden der Klasse mit den Handelsoperationen
#include <Trade\Trade.mqh>
//--- Eingabeparameter des EAs
input double Lots              = 0.1;      // Volumen
input int    Slippage          = 10;       // erlaubter Schlupf
input int    MovingPeriod      = 80;       // Periodenlänge des gleitenden Durchschnit
input int    MovingShift       = 6;        // Versazt des gleitenden Durchschnitts
//--- Globale Variablen
int    IndicatorHandle=0;  // Handle des Indikators
```

© 2000-2025, MetaQuotes Ltd.

```
bool   IsHedging=false;    // Flag des Kontos
CTrade trade;              // für die Durchführung der Handelsoperationen
//---
#define EA_MAGIC 18052018
//+------------------------------------------------------------------+
//| Prüfen der Bedingung für eine Positionseröffnung                 |
//+------------------------------------------------------------------+
void CheckForOpen(void)
  {
   MqlRates rt[2];
//--- Handeln nur zu Beginn einer neuen Bar
   if(CopyRates(_Symbol,_Period,0,2,rt)!=2)
     {
      Print("CopyRates of ",_Symbol," failed, no history");
      return;
     }
//--- Tick-Volumen
   if(rt[1].tick_volume>1)
      return;
//--- Erhalt der Werte des gleitenden Durchschnitts
   double   ma[1];
   if(CopyBuffer(IndicatorHandle,0,1,1,ma)!=1)
     {
      Print("CopyBuffer from iMA failed, no data");
      return;
     }
//--- Prüfen auf ein Signal
   ENUM_ORDER_TYPE signal=WRONG_VALUE;
//--- Kerze eröffnete über, schloss aber unter dem gleitenden Durchschnitt
   if(rt[0].open>ma[0] && rt[0].close<ma[0])
      signal=ORDER_TYPE_BUY;    // Kaufsignal
   else // Kerze eröffnete unter, schloss aber über dem gleitenden Durchschnitt
     {
      if(rt[0].open<ma[0] && rt[0].close>ma[0])
         signal=ORDER_TYPE_SELL;// Verkaufssignal
     }
//--- zusätzliche Prüfungen
   if(signal!=WRONG_VALUE)
     {
      if(TerminalInfoInteger(TERMINAL_TRADE_ALLOWED) && Bars(_Symbol,_Period)>100)
        {
         double price=SymbolInfoDouble(_Symbol,signal==ORDER_TYPE_SELL ? SYMBOL_BID:SY
         trade.PositionOpen(_Symbol,signal,Lots,price,0,0);
        }
     }
//---
  }
//+------------------------------------------------------------------+
//| Prüfen der Bedingung für eine Positionsschließung                |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void CheckForClose(void)
  {
   MqlRates rt[2];
//--- Handeln nur zu Beginn einer neuen Bar
   if(CopyRates(_Symbol,_Period,0,2,rt)!=2)
     {
      Print("CopyRates of ",_Symbol," failed, no history");
      return;
     }
   if(rt[1].tick_volume>1)
      return;
//--- Erhalt der Werte des gleitenden Durchschnitts
   double   ma[1];
   if(CopyBuffer(IndicatorHandle,0,1,1,ma)!=1)
     {
      Print("CopyBuffer from iMA failed, no data");
      return;
     }
//--- Position wurde bereits früher mittels PositionSelect() ausgewählt
   bool signal=false;
   long type=PositionGetInteger(POSITION_TYPE);
//--- Kerze eröffnete über, schloss aber unter dem gleitenden Durchschnitt - Schließen
   if(type==(long)POSITION_TYPE_SELL && rt[0].open>ma[0] && rt[0].close<ma[0])
      signal=true;
//--- Kerze eröffnete unter, schloss aber über dem gleitenden Durchschnitt - Schließen
   if(type==(long)POSITION_TYPE_BUY && rt[0].open<ma[0] && rt[0].close>ma[0])
      signal=true;
//--- zusätzliche Prüfungen
   if(signal)
     {
      if(TerminalInfoInteger(TERMINAL_TRADE_ALLOWED) && Bars(_Symbol,_Period)>100)
         trade.PositionClose(_Symbol,Slippage);
     }
//---
  }
//+------------------------------------------------------------------+
//| Positionsauswahl je nach Kontotyp: Netting oder Hedging          |
//+------------------------------------------------------------------+
bool SelectPosition()
  {
   bool res=false;
//--- Auswahl einer Position für ein Hedging-Konto
   if(IsHedging)
     {
      uint total=PositionsTotal();
      for(uint i=0; i<total; i++)
        {
         string position_symbol=PositionGetSymbol(i);
```

© 2000-2025, MetaQuotes Ltd.

```
         if(_Symbol==position_symbol && EA_MAGIC==PositionGetInteger(POSITION_MAGIC))
           {
            res=true;
            break;
           }
        }
     }
//--- Auswahl einer Position für ein Netting-Konto
   else
     {
      if(!PositionSelect(_Symbol))
         return(false);
      else
         return(PositionGetInteger(POSITION_MAGIC)==EA_MAGIC); //--- Prüfen der Magicn
     }
//--- Berechnungsergebnis
   return(res);
  }
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit(void)
  {
//--- Setzen des Kontotyps: Netting oder Hedging
   IsHedging=((ENUM_ACCOUNT_MARGIN_MODE)AccountInfoInteger(ACCOUNT_MARGIN_MODE)==ACCOU
//--- Initialisieren eines Objekts auf die korrekte Position
   trade.SetExpertMagicNumber(EA_MAGIC);
   trade.SetMarginMode();
   trade.SetTypeFillingBySymbol(Symbol());
   trade.SetDeviationInPoints(Slippage);
//--- Erstellen des Gleitenden Durchschnitts
   IndicatorHandle=iMA(_Symbol,_Period,MovingPeriod,MovingShift,MODE_SMA,PRICE_CLOSE);
   if(IndicatorHandle==INVALID_HANDLE)
     {
      printf("Error creating iMA indicator");
      return(INIT_FAILED);
     }
//--- ok
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick(void)
  {
//--- wenn eine Position bereits eröffnet wurde, prüfen der Bedingung sie zu schließer
   if(SelectPosition())
      CheckForClose();
//--- prüfen der Bedingung die offene Position zu schließen
```

© 2000-2025, MetaQuotes Ltd.

```
   CheckForOpen();
//---
  }
//+------------------------------------------------------------------+
//| Tester Funktion                                                  |
//+------------------------------------------------------------------+
double OnTester()
  {
//--- Nutzerkriterium der Optimierung (je höher desto besser)
   double ret=0.0;
//--- Übertrage der Handelsergebnisse in das Array
   double array[];
   double trades_volume;
   GetTradeResultsToArray(array,trades_volume);
   int trades=ArraySize(array);
//--- Gibt es weniger als 10 Positionen, ist das Testergebnis 0
   if(trades<10)
      return (0);
//--- Durchschnittsergebnis der Positionen
   double average_pl=0;
   for(int i=0;i<ArraySize(array);i++)
      average_pl+=array[i];
   average_pl/=trades;
//--- Anzeige der Nachricht im Einzeltestmodus
   if(MQLInfoInteger(MQL_TESTER) && !MQLInfoInteger(MQL_OPTIMIZATION))
      PrintFormat("%s: Trades=%d, Average profit=%.2f",__FUNCTION__,trades,average_pl)
//--- Berechnen der Linearen Regression für den Saldengraph
   double a,b,std_error;
   double chart[];
   if(!CalculateLinearRegression(array,chart,a,b))
      return (0);
//--- Berechnen des Fehlers der Abweichung des Charts von der Regressionslinie
   if(!CalculateStdError(chart,a,b,std_error))
      return (0);
//--- Berechnen des Verhältnisses des Trendgewinns und der Standardabweichung
   ret=(std_error == 0.0) ? a*trades : a*trades/std_error;
//--- Rückgabe des Wertes des Nutzerkriteriums der Optimierung
   return(ret);
  }
//+------------------------------------------------------------------+
//| Abfrage des Arrays von Gewinn/Verlust der Deals                  |
//+------------------------------------------------------------------+
bool GetTradeResultsToArray(double &pl_results[],double &volume)
  {
//--- Abfrage der kompletten Handelshistorie
   if(!HistorySelect(0,TimeCurrent()))
      return (false);
   uint total_deals=HistoryDealsTotal();
   volume=0;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Setzen die Anfangsgröße des Arrays mit einer Marge - nach der Anzahl der Deals i
   ArrayResize(pl_results,total_deals);
//--- Zähler der Deals, die das Handelsergebnis bestimmen - Gewinn oder Verlust
   int counter=0;
   ulong ticket_history_deal=0;
//--- über alle Deals
   for(uint i=0;i<total_deals;i++)
     {
      //--- Auswahl eines Deals
      if((ticket_history_deal=HistoryDealGetTicket(i))>0)
        {
         ENUM_DEAL_ENTRY deal_entry  =(ENUM_DEAL_ENTRY)HistoryDealGetInteger(ticket_hi
         long            deal_type   =HistoryDealGetInteger(ticket_history_deal,DEAL_T
         double          deal_profit =HistoryDealGetDouble(ticket_history_deal,DEAL_PR
         double          deal_volume =HistoryDealGetDouble(ticket_history_deal,DEAL_VO
         //--- uns interessieren nur die Handelsoperationen
         if((deal_type!=DEAL_TYPE_BUY) && (deal_type!=DEAL_TYPE_SELL))
            continue;
         //--- Nur die Deals, die Gewinn/Verlust festlegen
         if(deal_entry!=DEAL_ENTRY_IN)
           {
            //--- Schreiben des Handelsergebnisses in den Array und Erhöhen des Zähler
            pl_results[counter]=deal_profit;
            volume+=deal_volume;
            counter++;
           }
        }
     }
//--- Setzen der finalen Größe des Arrays
   ArrayResize(pl_results,counter);
   return (true);
  }
//+------------------------------------------------------------------+
//| Berechnen der Linearen Regression y=a*x+b                        |
//+------------------------------------------------------------------+
bool CalculateLinearRegression(double  &change[],double &chartline[],
                               double  &a_coef,double  &b_coef)
  {
//--- Prüfen auf genügend Daten
   if(ArraySize(change)<3)
      return (false);
//--- Erstellen eines Charts mit Akkumulation
   int N=ArraySize(change);
   ArrayResize(chartline,N);
   chartline[0]=change[0];
   for(int i=1;i<N;i++)
      chartline[i]=chartline[i-1]+change[i];
//--- Jetzt folgt die Berechnung des Regressionsverhältnisses
   double x=0,y=0,x2=0,xy=0;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0;i<N;i++)
     {
      x=x+i;
      y=y+chartline[i];
      xy=xy+i*chartline[i];
      x2=x2+i*i;
     }
   a_coef=(N*xy-x*y)/(N*x2-x*x);
   b_coef=(y-a_coef*x)/N;
//---
   return (true);
  }
//+------------------------------------------------------------------+
//|  Berechnen des Fehlerquadrate zum Ermitteln von a und b          |
//+------------------------------------------------------------------+
bool  CalculateStdError(double  &data[],double  a_coef,double  b_coef,double &std_err)
  {
//--- Summe der Fehlerquadrate
   double error=0;
   int N=ArraySize(data);
   if(N<=2)
      return (false);
   for(int i=0;i<N;i++)
      error+=MathPow(a_coef*i+b_coef-data[i],2);
   std_err=MathSqrt(error/(N-2));
//---
   return (true);
  }
```

Siehe auch

Testen von Handelsstrategien, TesterHideIndicators, Arbeit mit Ergebnisse der Optimierung, TesterStatistics, OnTesterInit, OnTesterDeinit, OnTesterPass, MQL_TESTER, MQL_OPTIMIZATION, FileOpen, FileWrite, FileLoad, FileSave

© 2000-2025, MetaQuotes Ltd.

OnTesterInit

Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterInit auftritt, um notwendige Aktionen vor der Optimierung im Strategie-Tester durchzuführen. Es gibt zwei Versionen dieser Funktion.

Die **Version, die das Ergebnis zurück gibt**

```
int  OnTesterInit(void);
```

Rückgabewert

Vom Typ int, Null bedeutet die erfolgreiche Initialisierung eines auf einem Chart gestarteten EAs vor dem Start der Optimierung.

Der Aufruf OnTesterInit(), der das Ausführungsergebnis zurückgibt, wird zur Verwendung empfohlen, da er nicht nur die Programminitialisierung erlaubt, sondern auch einen Fehlercode im Falle eines frühen Optimierungsstopps zurückgibt. Die Rückgabe eines anderen Wertes als INIT_SUCCEEDED (0) bedeutet einen Fehler, es wird keine Optimierung gestartet.

Die Version ohne Ergebnisrückgabe wird nur aus Kompatibilitätsgründen mit alten Codes belassen. Ein Verwenden wird nicht empfohlen.

```
void  OnTesterInit(void);
```

Hinweis

Das Ereignis TesterInit wird generiert, bevor die EA-Optimierung im Strategie-Tester startet. Bei diesem Ereignis wird ein EA mit OnTesterDeInit() oder OnTesterPass() Ereignisbehandler automatisch auf ein separates Terminalchart geladen. Es hat das Symbol und den Zeitrahmen, die im Tester angegeben wurden.

So ein Ereignis erhält die Ereignisse von TesterInit, TesterDeinit und TesterPass, aber nicht Init, Deinit und NewTick. Dementsprechend sollte die notwendige Logik für die Verarbeitung der Ergebnisse jedes Durchlaufs während der Optimierung in den Funktionen OnTesterInit(), OnTesterDeinit() und OnTesterPass() vollständig implementiert werden.

Das Ergebnis jedes einzelnen Durchlaufs während einer Strategieoptimierung kann über einen Frame der Funktion OnTester() mit der Funktion FrameAdd() übergeben werden.

Die Funktion OnTesterInit() wird verwendet, um vor dem Start der Optimierung einen Expert Advisor für weitere zu initiieren, der die Optimierungsergebnisse verarbeitet. Sie wird immer zusammen mit OnTesterDeinit() verwendet.

Die Zeit für die Ausführung von OnTesterInit() ist begrenzt. Wird sie überschritten, wird der EA zwangsweise gestoppt und die Optimierung abgebrochen. Im Tester-Journal wird eine Meldung angezeigt:

```
TesterOnTesterInit dauert zu lange. Der Tester kann nicht initialisiert werden.
```

Das Beispiel wird von OnTick übernommen. Die Funktion OnTesterInit() wurde hinzugefügt, um Optimierungsparameter : zu setzen.

```
//+------------------------------------------------------------------+
//|                                          OnTesterInit_Sample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                 https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Sample EA with the OnTesterInit() handler,"
#property description "in which values and limitations of "
#property description "inputs during optimization are set"

input double lots=0. 1;        // Volumen in Lots
input double kATR=3;           // Länge der Signalkerze in ATR
input int    ATRperiod=20;     // Periodenlänge des ATR
input int    holdbars=8;       // Anzahl der Bars, die die Position gehalten werden sol
input int    slippage=10;      // Erlaubter Schlupf
input bool   revers=false;     // Signal umkehren?
input ulong  EXPERT_MAGIC=0;   // Des EA's Magicnummer
//--- zum Sichern des Handles des Indikators ATR
int atr_handle;
//--- hier werden die letzten Werte des ATR und die Kerzenkörper gesichert
double last_atr,last_body;
datetime lastbar_timeopen;
double trade_lot;
//--- Erinnern der Startzeit der Optimierung
datetime optimization_start;
//--- für die Anzeige der Dauer auf dem Chart nach dem Ende der Optimierung
string report;
//+------------------------------------------------------------------+
//| TesterInit Funktion                                              |
//+------------------------------------------------------------------+
void OnTesterInit()
  {
//--- Setzen der Eingabewerte für die Optimierung
   ParameterSetRange("lots",false,0.1,0,0,0);
   ParameterSetRange("kATR",true,3.0,1.0,0.3,7.0);
   ParameterSetRange("ATRperiod",true,10,15,1,30);
   ParameterSetRange("holdbars",true,5,3,1,15);
   ParameterSetRange("slippage",false,10,0,0,0);
   ParameterSetRange("revers",true,false,false,1,true);
   ParameterSetRange("EXPERT_MAGIC",false,123456,0,0,0);
   Print("Initial values and optimization parameter limitations are set");
//--- Sichern des Beginns der Optimierung
   optimization_start=TimeLocal();
   report=StringFormat("%s: optimization launched at %s",
                       __FUNCTION__,TimeToString(TimeLocal(),TIME_MINUTES|TIME_SECONDS
//--- Anzeige der Nachrichten auf dem Chart und im Journal des Terminal
   Print(report);
   Comment(report);
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| TesterDeinit Funktion                                            |
//+------------------------------------------------------------------+
void OnTesterDeinit()
  {
//--- Dauer der Optimierung
   string log_message=StringFormat("%s: optimization took %d seconds",
                                   __FUNCTION__,TimeLocal()-optimization_start);
   PrintFormat(log_message);
   report=report+"\r\n"+log_message;
   Comment(report);
  }
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Initialisierung der Globalen Variablen
   last_atr=0;
   last_body=0;
//--- Setzen des korrekten Volumens
   double min_lot=SymbolInfoDouble(_Symbol,SYMBOL_VOLUME_MIN);
   trade_lot=lots>min_lot? lots:min_lot;
//--- Erstellen des Handles des Indikators ATR
   atr_handle=iATR(_Symbol,_Period,ATRperiod);
   if(atr_handle==INVALID_HANDLE)
     {
      PrintFormat("%s: failed to create iATR, error code %d",__FUNCTION__,GetLastError
      return(INIT_FAILED);
     }
//--- Erfolgreiche Initialisierung des EA
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- Handelssignal
   static int signal=0; // +1 heißt Kaufsignal, -1 Verkaufssignal
//--- Prüfen und Schließen alter Positionen, die vor mehr als 'holdbars' eröffnet wurd
   ClosePositionsByBars(holdbars,slippage,EXPERT_MAGIC);
//--- Prüfen auf eine neue Bar
   if(isNewBar())
     {
      //--- Prüfen auf ein Signal
      signal=CheckSignal();
     }
//--- Wenn eine Netting-Position eröffnet wurde - warten bis sie geschlossen wurde
```

© 2000-2025, MetaQuotes Ltd.

```
   if(signal!=0 && PositionsTotal()>0 && (ENUM_ACCOUNT_MARGIN_MODE)AccountInfoInteger
     {
      signal=0;
      return; // die Ereignisbehandlung von NewTick beenden und kein Markteintritt vo
     }
//--- für ein Hedging-Konto wird jede Position separat gehalten und geschlossen
   if(signal!=0)
     {
      //--- Kaufsignal
      if(signal>0)
        {
         PrintFormat("%s: Buy signal! Revers=%s",__FUNCTION__,string(revers));
         if(Buy(trade_lot,slippage,EXPERT_MAGIC))
            signal=0;
        }
      //--- Verkaufssignal
      if(signal<0)
        {
         PrintFormat("%s: Sell signal! Revers=%s",__FUNCTION__,string(revers));
         if(Sell(trade_lot,slippage,EXPERT_MAGIC))
            signal=0;
        }
     }
//--- Ende der Funktion OnTick
  }
//+------------------------------------------------------------------+
//| Prüfen auf ein neues Handelssignal                               |
//+------------------------------------------------------------------+
int CheckSignal()
  {
//--- 0 beutet klein Signal
   int res=0;
//--- Abfrage des Wertes der ATR der vorletzten kompletten Bar (Indes der Bar ist 2)
   double atr_value[1];
   if(CopyBuffer(atr_handle,0,2,1,atr_value)!=-1)
     {
      last_atr=atr_value[0];
      //--- Datenabfrage der letzten geschlossenen Bar von Array des Typs MqlRates
      MqlRates bar[1];
      if(CopyRates(_Symbol,_Period,1,1,bar)!=-1)
        {
         //--- Berechnen der Körpergröße der letzten, vollständigen Kerze
         last_body=bar[0].close-bar[0].open;
         //--- wenn der Körper der letzten Bar (mit Index 1) den vorherigen ATR-Wert
         if(MathAbs(last_body)>kATR*last_atr)
            res=last_body>0?1:-1; // positiver Wert der Aufwärtskerze
        }
      else
         PrintFormat("%s: Failed to receive the last bar! Error",__FUNCTION__,GetLastE
```

© 2000-2025, MetaQuotes Ltd.

```
     }
   else
      PrintFormat("%s: Failed to receive ATR indicator value! Error",__FUNCTION__,GetI
//--- falls der umgekehrte Handelsmodus aktiviert ist
   res=revers?-res:res;  // Signal umkehren, wenn nötig (Rückgabe von -1 statt 1 und v
//--- Rückgabe des Wertes des Handelssignals
   return (res);
  }
//+------------------------------------------------------------------+
//|  Rückgabe von 'true' wenn eine neue Bar erscheint                |
//+------------------------------------------------------------------+
bool isNewBar(const bool print_log=true)
  {
   static datetime bartime=0; // Sichern der Eröffnungszeit der aktuellen Bar
//--- Abfrage der Eröffnungszeit der Bar Null
   datetime currbar_time=iTime(_Symbol,_Period,0);
//--- Wenn sich die Eröffnungszeit änderte, gibt es eine neue Bar
   if(bartime!=currbar_time)
     {
      bartime=currbar_time;
      lastbar_timeopen=bartime;
      //--- Eintragen der Eröffnungszeit der neuen Bar in das Log
      if(print_log && !(MQLInfoInteger(MQL_OPTIMIZATION)||MQLInfoInteger(MQL_TESTER)))
        {
         //--- Anzeige der Nachricht mit der Eröffnungszeit der neuen bar
         PrintFormat("%s: new bar on %s %s opened at %s",__FUNCTION__,_Symbol,
                     StringSubstr(EnumToString(_Period),7),
                     TimeToString(TimeCurrent(),TIME_SECONDS));
         //--- Datenabfrage beim letzten Tick
         MqlTick last_tick;
         if(!SymbolInfoTick(Symbol(),last_tick))
            Print("SymbolInfoTick() failed, error = ",GetLastError());
         //--- Anzeige der Zeit des letzten Ticks bis zur Millisekunde
         PrintFormat("Last tick was at %s.%03d",
                     TimeToString(last_tick.time,TIME_SECONDS),last_tick.time_msc%1000
        }
      //--- wir haben eine neue Bar
      return (true);
     }
//--- keine neue Bar
   return (false);
  }
//+------------------------------------------------------------------+
//| Kauf zum Marktpreis mit angegebenen Volumen                      |
//+------------------------------------------------------------------+
bool Buy(double volume,ulong deviation=10,ulong  magicnumber=0)
  {
//--- Kauf zum Marktpreis
   return (MarketOrder(ORDER_TYPE_BUY,volume,deviation,magicnumber));
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Verkauf zum Marktpreis mit angegebenen Volumen                   |
//+------------------------------------------------------------------+
bool Sell(double volume,ulong deviation=10,ulong  magicnumber=0)
  {
//--- Verkauf zum Marktpreis
   return (MarketOrder(ORDER_TYPE_SELL,volume,deviation,magicnumber));
  }
//+------------------------------------------------------------------+
//| Position schließen wegen der Haltezeit                           |
//+------------------------------------------------------------------+
void ClosePositionsByBars(int holdtimebars,ulong deviation=10,ulong  magicnumber=0)
  {
   int total=PositionsTotal(); // Anzahl der offenen Positionen
//--- Iterieren über die offenen Position
   for(int i=total-1; i>=0; i--)
     {
      //--- Parameter der Position
      ulong  position_ticket=PositionGetTicket(i);
      string position_symbol=PositionGetString(POSITION_SYMBOL);
      ulong  magic=PositionGetInteger(POSITION_MAGIC);
      datetime position_open=(datetime)PositionGetInteger(POSITION_TIME);
      int bars=iBarShift(_Symbol,PERIOD_CURRENT,position_open)+1;

      //--- wenn die Lebenszeit der Position lang genug ist, und MagicNummer und Symbo
      if(bars>holdtimebars && magic==magicnumber && position_symbol==_Symbol)
        {
         int    digits=(int)SymbolInfoInteger(position_symbol,SYMBOL_DIGITS);
         double volume=PositionGetDouble(POSITION_VOLUME);
         ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE)
         string str_type=StringSubstr(EnumToString(type),14);
         StringToLower(str_type); // Kleinschreibung für das korrekte Nachrichtenforma
         PrintFormat("Close position #%I64u %s %s %.2f",
                     position_ticket,position_symbol,str_type,volume);
         //--- Setzen des Auftragsart und Senden der Handelsanfrage
         if(type==POSITION_TYPE_BUY)
            MarketOrder(ORDER_TYPE_SELL,volume,deviation,magicnumber,position_ticket);
         else
            MarketOrder(ORDER_TYPE_BUY,volume,deviation,magicnumber,position_ticket);
        }
     }
  }
//+------------------------------------------------------------------+
//| Vorbereiten und Senden der Handelsanfrage                        |
//+------------------------------------------------------------------+
bool MarketOrder(ENUM_ORDER_TYPE type,double volume,ulong slip,ulong magicnumber,ulong
  {
//--- Deklarieren und Initialisieren der Strukturen
```

© 2000-2025, MetaQuotes Ltd.

```
   MqlTradeRequest request={};
   MqlTradeResult  result={};
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(type==ORDER_TYPE_BUY)
      price=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
//--- Abfrage der Parameter
   request.action   =TRADE_ACTION_DEAL;                     // Typ der Handelsoperatio
   request.position =pos_ticket;                            // Ticketnummer der zu sch
   request.symbol   =Symbol();                              // Symbol
   request.volume   =volume;                                // Volumen
   request.type     =type;                                  // Auftragsart
   request.price    =price;                                 // Handelspreis
   request.deviation=slip;                                  // erlaubter Schlupf vom P
   request.magic    =magicnumber;                           // MagicNumber des Auftrag
//--- Senden einer Anfrage
   if(!OrderSend(request,result))
     {
      //--- Datenanzeige im Fehlerfall
      PrintFormat("OrderSend %s %s %.2f at %.5f error %d",
                  request.symbol,EnumToString(type),volume,request.price,GetLastError
      return (false);
     }
//--- Information über eine erfolgreiche Operation
   PrintFormat("retcode=%u  deal=%I64u  order=%I64u",result.retcode,result.deal,result
   return (true);
  }
```

Siehe auch

Testen von Handelsstrategien, Arbeit mit Ergebnisse der Optimierung, OnTesterDeinit, OnTesterPass, ParameterGetRange, ParameterSetRange

© 2000-2025, MetaQuotes Ltd.

OnTesterDeinit

Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterDeinit nach der EA-Optimierung auftritt.

```
void  OnTesterDeinit(void);
```

Rückgabewert

Kein Rückgabewert

Hinweis

Das Ereignis TesterDeinit wird nach dem Ende der EA-Optimierung im Strategie-Tester generiert.

Ein EA mit der Ereignisbehandlung durch OnTesterDeInit() oder OnTesterPass() wird beim Start der Optimierung automatisch auf ein separates Terminalchart geladen. Es hat das Symbol und den Zeitrahmen, die im Tester angegeben wurden. Die Funktion ist für die Endverarbeitung aller Optimierungsergebnisse konzipiert.

Beachten Sie, dass Optimierungsrahmen, die von Testagenten mit der Funktion FrameAdd() gesendet werden, gebündelt geliefert werden können. Daher können nicht alle Frames, sowie TesterPassEreignisse, ankommen und in OnTesterPass() vor dem Ende der Optimierung verarbeitet werden. Wenn Sie alle zurückgestellten Frames in OnTesterDeinit() empfangen wollen, platzieren Sie den Codeblock mit der Funktion FrameNext().

Siehe auch

Testen von Handelsstrategien, Arbeit mit Ergebnisse der Optimierung, TesterStatistics, OnTesterInit, OnTesterPass, ParameterGetRange, ParameterSetRange

© 2000-2025, MetaQuotes Ltd.

OnTesterPass

Die Funktion wird von EAs aufgerufen, wenn TesterPass auftritt, um die Ankunft eines neuen Datenrahmens während der EA-Optimierung im Strategie-Tester zu handhaben.

```
void  OnTesterPass(void);
```

Rückgabewert

Kein Rückgabewert

Hinweis

Das Ereignis TesterPass wird automatisch generiert, wenn ein Frame während der Optimierung des Expert Advisors im Strategietester empfangen wurde.

Ein EA mit der Ereignisbehandlung durch OnTesterDeInit() oder OnTesterPass() wird beim Start der Optimierung automatisch auf ein separates Terminalchart geladen. Es hat das Symbol und den Zeitrahmen, die im Tester angegeben wurden. Mit dieser Funktion kann auf Frames reagiert werde, die vom Testagenten während der Optimierung empfangen wurden. Der Frame mit denn Testergebnissen sollte von der Funktion OnTester() mittels FrameAdd() gesendet werden.

Beachten Sie, dass Optimierungsrahmen, die von Testagenten mit der Funktion FrameAdd() gesendet werden, gebündelt geliefert werden können. Daher können nicht alle Frames, sowie TesterPassEreignisse, ankommen und in OnTesterPass() vor dem Ende der Optimierung verarbeitet werden. Wenn Sie alle zurückgestellten Frames in OnTesterDeinit() empfangen wollen, platzieren Sie den Codeblock mit der Funktion FrameNext().

Nach der Beendigung von OnTesterDeinit(), ist es möglich alle erhaltenen Frames mit den Funktionen FrameFirst()/FrameFilter und FrameNext() zu ordnen.

Siehe auch

Testen von Handelsstrategien, Arbeit mit Ergebnisse der Optimierung, OnTesterInit, OnTesterDeinit, FrameFirst, FrameFilter, FrameNext, FrameInputs

© 2000-2025, MetaQuotes Ltd.

Marktinformation erhalten

Funktionen, um Informationen über Marktstand zu erhalten.

Funktion	Aktion
SymbolsTotal	Gibt die Anzahl zugaenglicher (gewaehlter in MarketWatch oder aller) Symbole zurück
SymbolExist	Prüft, ob ein Symbol mit dem angegebenen Namen existiert
SymbolName	Gibt den Namen des bezeichneten Symbols zurück
SymbolSelect	Waehlt das Symbol im Fenster MarketWatch oder entfernt das Symbol aus dem Fenster
SymbolIsSynchronized	Überprüft ob die Daten des angegebenen Symbol mit Daten auf dem Handelsserversynchronisiert sind
SymbolInfoDouble	Gibt den Wert des Typs double des angegebenen Symbols für die entsprechende Eigenschaft zurück
SymbolInfoInteger	Gibt den Wert des ganzzahligen Typs (long, datetime, int oder bool) des angegebenen Symbols für die entsprechende Eigenschaft zurück
SymbolInfoString	Gibt den Wert des Typs string des angegebenen Symbols für die entsprechende Eigenschaft zurück
SymbolInfoMarginRate	Gibt die Margin je nach Ordertyp und -richtung zurück
SymbolInfoTick	Gibt laufende Preise für das angegebene Symbol in der Variable des Typs MqlTick
SymbolInfoSessionQuote	Ermöglicht die Anfangszeit und die Abschlusszeit der angegebenen Kotierungssession für das angegebene Symbol und für den angegebenen Wochentag
SymbolInfoSessionTrade	Ermöglicht die Anfangszeit und die Abschlusszeit der angegebenen Handelssession für das angegebene Symbol und für den angegebenen Wochentag
MarketBookAdd	Sorgt für Eroeffnung von DOM für das gewaehlte Symbol, und subskribiert auf Nachrichten über DOM Veränderung
MarketBookRelease	Sorgt für Schliessen von DOM für das gewaehlte Symbol und annulliert Subskription auf Nachrichten über DOM Veränderung
MarketBookGet	Gibt Feld des Typs Strukturen MqlBookInfo, das Aufzeichnungen von DOM des angegebenen Symbols enthält

© 2000-2025, MetaQuotes Ltd.

SymbolsTotal

Gibt die Anzahl der im Fenster der Marktübersicht verfügbaren oder aller Symbole zurück.

```
int  SymbolsTotal(
   bool  selected      // true - nur Symbole in der Marktübersicht
   );
```

Parameter

selected

[in] Modus der Anforderung. Kann Werte true oder false haben.

Rückgabewert

Wenn Parameter selected gleich true ist, gibt die Funktion die Anzahl der gewählten Symbole im Fenster der Marktübersicht zurück. Wenn der Wert false ist, wird die Gesamtzahl aller Symbole zurückgegeben.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der Anzahl der verfügbaren Symbole
   int total    = SymbolsTotal(false);       // alle Symbole des Servers
   int selected = SymbolsTotal(true);        // alle Symbole im Fenster der Marktübers

//--- sende die erhaltenen Daten an das Journal
   PrintFormat("Number of available symbols on the server: %d\n"+
               "Number of available symbols selected in MarketWatch: %d", total, selec
   /*
  Ergebnis:
   Number of available symbols on the server: 140
   Number of available symbols selected in MarketWatch: 11
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolExist

Prüft, ob ein Symbol mit dem angegebenen Namen existiert.

```
bool  SymbolExist(
   const string  name,     // Symbolname
   bool&   is_custom       // Eigenschaften des nutzerdefinierten Symbols
   );
```

Parameter

name

[in] Symbolname.

is_custom

[out] Eigenschaften des nutzerdefinierten Symbols, die bei erfolgreicher Ausführung festgelegt werden. Wenn true, ist das erkannte Symbol ein nutzerdefiniertes Symbol.

Rückgabewert

Wenn false, wurde das Symbol unter den standardmäßigen und den nutzerdefinierten Symbolen nicht gefunden.

Beispiel:

```
#define SYMBOL_NAME "GBPUSDn"
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- deklariere das nutzerdefinierte Symbol-Flag und prüfe das Vorhandensein eines Sy
   bool custom = false;
   bool result = SymbolExist(SYMBOL_NAME, custom);

//--- Deklarieren des Standardtextes für den Fehler „Symbol nicht gefunden"
   string text = StringFormat("The symbol '%s' was not found among either the standard

//--- wenn ein Symbol gefunden wurde, wird ein Nachrichtentext erstellt, je nachdem, i
   if(result)
     {
      //--- falls es ein Standardsymbol ist
      if(!custom)
         text = StringFormat("The '%s' symbol is available on the server.", SYMBOL_NAM
      //--- wenn dies ein nutzerdefiniertes Symbol ist
      else
         text = StringFormat("The symbol '%s' was found in the list of custom symbols.
     }

//--- Senden der Nachricht über das Prüfergebnis an das Journal
   Print(text);
```

© 2000-2025, MetaQuotes Ltd.

```
   /*
   result for standard 'GBPUSD' symbol:
   The 'GBPUSD' symbol is available on the server.

   result for custom 'GBPUSDx' symbol:
   The symbol 'GBPUSDx' was found in the list of custom symbols.

   result for missing 'GBPUSDn' symbol:
   The symbol 'GBPUSDn' was not found among either the standard or custom symbols.
   */
  }
```

Siehe auch

SymbolsTotal, SymbolSelect, Custom symbols

© 2000-2025, MetaQuotes Ltd.

SymbolName

Gibt Name des angegebenen Symbols zurück.

```
string  SymbolName(
   int   pos,           // Nummer in der Liste
   bool  selected       // true - nur Symbole in der Marktübersicht
   );
```

Parameter

pos

[in] Nummer des Symbols

selected

[in] Modus der Anforderung. Wenn der Wert true ist, wird das Symbol aus der Liste der im Fenster der Marktübersicht ausgewählten Symbole genommen. Wenn der Wert false ist, wird das Symbol aus der allgemeinen Liste genommen.

Rückgabewert

Name das Symbols als Zeichenkette (string).

Beispiel:

```
#define SYMBOL_NAME "GBPHKD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- das Flag des Serversymbols für das Ergebnis der Suche setzen
   bool found = false;

//--- das Symbol 'SYMBOL_NAME' in der Liste aller Serversymbole finden
   int total = SymbolsTotal(false);
   for(int i=0; i<total; i++)
     {
      //--- Ermitteln eines Symbolnamens in der Liste nach Schleifenindex
      string name = SymbolName(i, false);

      //--- wenn dies das gewünschte Symbol ist, seinen Namen und seine Position in de
      if(name == SYMBOL_NAME)
        {
         PrintFormat("The '%s' symbol was found in the list of server symbols at index
         found = true;
         break;
        }
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn ein Symbol auf dem Server nicht gefunden wird, melden wir dies vor dem Proc
   if(!found)
      PrintFormat("The '%s' symbol was not found on the server.", SYMBOL_NAME);
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolSelect

Wählt das Symbol im Fenster der Marktübersicht oder entfernt das Symbol aus dem Fenster.

```
bool  SymbolSelect(
   string  name,        // Symbolname
   bool    select       // hinzufügen oder entfernen
   );
```

Parameter

name

[in] Symbolname.

select

[in] Schalter. Wenn der Wert false ist, muss das Symbol aus dem Fenster der Marktübersicht entfernt werden, anderenfalls muss das Symbol im Fenster Marktübersicht ausgewählt werden. Symbol kann nicht entfernt werden, wenn es für dieses Symbol offene Charts oder offene Positionen gibt.

Rückgabewert

Im Fall des Misserfolges gibt die Funktion false zurück.

Beispiel:

```
#define SYMBOL_NAME "GBPHKD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Prüfung, ob ein Symbol in den Listen vorhanden ist, wenn nicht, dieses melden ur
   bool custom = false;
   if(!SymbolExist(SYMBOL_NAME, custom))
     {
      PrintFormat("'%s' symbol not found in the lists", SYMBOL_NAME);
      return;
     }

//--- ein Symbol zum Fenster der Marktübersicht hinzufügen
   ResetLastError();
   if(!SymbolSelect(SYMBOL_NAME, true))
     {
      Print("SymbolSelect() failed. Error ", GetLastError());
      return;
     }
//--- wenn ein Symbol erfolgreich zur Liste hinzugefügt wurde, rufen wir seinen Index
   int index = SymbolIndex(SYMBOL_NAME);
   PrintFormat("The '%s' symbol has been added to the MarketWatch list. Symbol index
```

© 2000-2025, MetaQuotes Ltd.

```
//--- jetzt wird das Symbol aus dem Fenster der Marktübersicht entfernt
   ResetLastError();
   if(!SymbolSelect(SYMBOL_NAME, false))
     {
      Print("SymbolSelect() failed. Error ", GetLastError());
      return;
     }
//--- wenn ein Symbol erfolgreich aus der Liste entfernt wurde, ist sein Index im Fens
   index = SymbolIndex(SYMBOL_NAME);
   PrintFormat("The '%s' symbol has been removed from the MarketWatch list. Symbol ind

   /*
  Ergebnis:
   The 'GBPHKD' symbol has been added to the MarketWatch list. Symbol index in the lis
   The 'GBPHKD' symbol has been removed from the MarketWatch list. Symbol index in the
   */
  }
//+------------------------------------------------------------------+
//| Return the symbol index in the Market Watch symbol list          |
//+------------------------------------------------------------------+
int SymbolIndex(const string symbol)
  {
   int total = SymbolsTotal(true);
   for(int i=0; i<total; i++)
     {
      string name = SymbolName(i, true);
      if(name == symbol)
         return i;
     }
   return(WRONG_VALUE);
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolIsSynchronized

Prüft, ob Daten des angegebenen Symbols im Terminal mit Daten im Handelsserver synchronisiert werden oder nicht.

```
bool  SymbolIsSynchronized(
   string  name,        // Symbolname
   );
```

Parameter

name

[in] Symbolname.

Rückgabewert

Wenn die Daten synchronisiert wurden, wird true zurückgegeben, anderenfalls false.

Beispiel:

```
#define SYMBOL_NAME "EURUSD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Flags der Synchronisierung von Daten im Terminal mit den Serverdater
   bool    sync = SymbolIsSynchronized(SYMBOL_NAME);

//--- Erstellen einer Nachricht in Abhängigkeit von dem Flag der Synchronisation
   string text = StringFormat("The data on the '%s' symbol in the terminal is synchror
   if(!sync)
      text = StringFormat("The data for the '%s' symbol in the terminal is not synchr

//--- Senden des erhaltenen Ergebnisses an das Journal
   Print(text);

   /*
   result for synchronized data:
   The data on the 'EURUSD' symbol in the terminal is synchronized with the data on th

   result for data not yet synchronized:
   The data for the 'GBPHKD' symbol in the terminal is not synchronized with the data
   */
  }
```

Siehe auch

SymbolInfoInteger, Datenzugang organisieren

© 2000-2025, MetaQuotes Ltd.

SymbolInfoDouble

Gibt die entsprechende Eigenschaft des angegebenen Symbols zurück. Es gibt zwei Varianten der Funktion.

1. gibt den Wert der Eigenschaft sofort zurück.

```
double  SymbolInfoDouble(
   string                    name,       // Symbol
   ENUM_SYMBOL_INFO_DOUBLE   prop_id     // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich dirchgeführt wird. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  SymbolInfoDouble(
   string                    name,       // Symbol
   ENUM_SYMBOL_INFO_DOUBLE   prop_id,    // Identifikator der Eigenschaft
   double&                   double_var  // hier nehmen wir den wert der Eigenschaft
   );
```

Parameter

name

[in] Symbolname.

prop_id

[in] Identifikator der Symboleigenschaft. Wert kann einer der Enumerationswerte ENUM_SYMBOL_INFO_DOUBLE sein.

double_var

[out] Variable des Typs double, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs double. Im Falle der Nichtausführung können Sie Fehlerinformation durch die Funktion GetLastError() erhalten:

- 5040 - ungültiger string-Parameter für die Angabe des Symbolnamens,
- 4301 - unbekanntes Symbol (Finanzinstrument)
- 4302 - das Symbol nicht in der "Market Watch" ausgewählt (ist in der Liste nicht verfügbar),
- 4303 - ungültiger Identifikator der Symboleigenschaft.

Hinweis

Wenn die Funktion wird verwendet, um Informationen über den letzten Tick zu bekommen, ist es besser SymbolInfoTick() zu verwenden. Es ist möglich, dass seit Verbinden des Terminals an das Handelskonto keine Preise des Symbols empfangen worden. In einem solchen Fall wird der Wert nicht bestimmt.

In den meisten Fällen reicht es aus, SymbolInfoTick() zu verwenden. Mit dieser Funktion können Sie durch einen einzelnen Anruf Werte von Ask, Bid, Last, Volume und Zeit der des letzten Ticks zu bekommen.

© 2000-2025, MetaQuotes Ltd.

Die Funktion bietet Informationen über die Höhe der erhobenen Marge, je nach Art und Richtung des Auftrags SymbolInfoMarginRate().

Beispiel:

```
void OnTick()
  {
//--- erhalten wir Spread aus der Symboleigenschaften
   bool spreadfloat=SymbolInfoInteger(Symbol(),SYMBOL_SPREAD_FLOAT);
   string comm=StringFormat("Spread %s = %I64d Punkte\r\n",
                            spreadfloat?"fliessend":"fixiert",
                            SymbolInfoInteger(Symbol(),SYMBOL_SPREAD));
//--- berechnen wir nun Spread selbsstaendig
   double ask=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
   double bid=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   double spread=ask-bid;
   int spread_points=(int)MathRound(spread/SymbolInfoDouble(Symbol(),SYMBOL_POINT));
   comm=comm+"Berechneter Spread = "+(string)spread_points+" Punkte";
   Comment(comm);
 ;}
```

© 2000-2025, MetaQuotes Ltd.

SymbolInfoInteger

Gibt die entsprechende Eigenschaft des angegebenen Symbols zurück. Es gibt 2 Varianten der Funktion.

1. Gibt den Eigenschaftswert sofort zurück.

```
long  SymbolInfoInteger(
   string                     name,      // Symbol
   ENUM_SYMBOL_INFO_INTEGER   prop_id    // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück abhängig von der erfolgreichen Durchführung der Funktion. Im Erfolgsfall wird der Eigenschaftswert in die Empfangsvariable gestellt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  SymbolInfoInteger(
   string                     name,      // Symbol
   ENUM_SYMBOL_INFO_INTEGER   prop_id,   // Eigenschaften des Identifikatoren
   long&                      long_var   // hier wird der Eigenschaftswert angenommen
   );
```

Parameter

name

[in] Symbolname.

prop_id

[in] Identifikator der Symboleigenschaft. Wert kann einer der Enumerationswerte ENUM_SYMBOL_INFO_INTEGER sein.

long_var

[out] Variable des Typs long, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs long. Im Falle der Nichtausführung können Sie Fehlerinformation durch die Funktion GetLastError() erhalten:

- 5040 - ungültiger string-Parameter für die Angabe des Symbolnamens,
- 4301 - unbekanntes Symbol (Finanzinstrument)
- 4302 - das Symbol nicht in der "Market Watch" ausgewählt (ist in der Liste nicht verfügbar),
- 4303 - ungültiger Identifikator der Symboleigenschaft.

Hinweis

Wenn die Funktion wird verwendet, um Informationen über den letzten Tick zu bekommen, ist es besser SymbolInfoTick() zu verwenden. Es ist möglich, dass seit Verbinden des Terminals an das Handelskonto keine Preise des Symbols empfangen worden. In einem solchen Fall wird der Wert nicht bestimmt.

In den meisten Fällen reicht es aus, SymbolInfoTick() zu verwenden. Mit dieser Funktion können Sie durch einen einzelnen Anruf Werte von Ask, Bid, Last, Volume und Zeit der des letzten Ticks zu bekommen.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
void OnTick()
  {
//--- erhalten wir Spread aus Symboleigenschaften
   bool spreadfloat=SymbolInfoInteger(Symbol(),SYMBOL_SPREAD_FLOAT);
   string comm=StringFormat("Spread %s = %I64d Punkte\r\n",
                            spreadfloat?"fliessend":"fixiert",
                            SymbolInfoInteger(Symbol(),SYMBOL_SPREAD));
//--- berechnen wir Spread selbststaendig
   double ask=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
   double bid=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   double spread=ask-bid;
   int spread_points=(int)MathRound(spread/SymbolInfoDouble(Symbol(),SYMBOL_POINT));
   comm=comm+"Berechneter Spread = "+(string)spread_points+" Punkte";
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolInfoString

Gibt die entsprechende Eigenschaft des angegebenen Symbols zurück. Es gibt 2 Varianten der Funktion.

1. Gibt den Eigenschaftswert sofort zurück.

```
string  SymbolInfoString(
   string                    name,        // Symbol
   ENUM_SYMBOL_INFO_STRING   prop_id      // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird der Eigenschaftswert in die Empfangsvariable gestellt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  SymbolInfoString(
   string                    name,        // Symbol
   ENUM_SYMBOL_INFO_STRING   prop_id,     // Identifikator der Eigenschaft
   string&                   string_var   // ihr wird der Eigenschaftswert zugewiesen
   );
```

Parameter

name

[in] Symbolname.

prop_id

[in] Identifikator der Symboleigenschaft. Wert kann einer der Enumerationswerte ENUM_SYMBOL_INFO_STRING sein .

string_var

[out] Variable des Typs string, der den Wert der angeforderten Eigenschaft zugewiesen wird.

Rückgabewert

Wert des Typs string. Im Falle der Nichtausführung können Sie Fehlerinformation durch die Funktion GetLastError() erhalten:

- 5040 - ungültiger string-Parameter für die Angabe des Symbolnamens,
- 4301 - unbekanntes Symbol (Finanzinstrument)
- 4302 - das Symbol nicht in der "Marktübersicht" ausgewählt (ist in der Liste nicht verfügbar),
- 4303 - ungültiger Identifikator der Symboleigenschaft.

Hinweis

Wenn die Funktion verwendet werden soll, um Informationen über den letzten Tick zu bekommen, ist es besser SymbolInfoTick() zu verwenden. Es ist möglich, dass seit Verbinden des Terminals an das Handelskonto keine Preise des Symbols empfangen wurden. In einem solchen Fall ist der Wert unbestimmt.

In den meisten Fällen reicht es aus, SymbolInfoTick() zu verwenden. Mit dieser Funktion können Sie durch einen einzelnen Aufruf die Werte von Ask, Bid, Last, Volume und Zeit des letzten Ticks erhalten.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#define SYMBOL_NAME "USDJPY"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen von Daten des Typs string des Symbols SYMBOL_NAME
   string currency_base   = SymbolInfoString(SYMBOL_NAME, SYMBOL_CURRENCY_BASE);     //
   string currency_profit = SymbolInfoString(SYMBOL_NAME, SYMBOL_CURRENCY_PROFIT);   //
   string currency_margin = SymbolInfoString(SYMBOL_NAME, SYMBOL_CURRENCY_MARGIN);   //
   string symbol_descript = SymbolInfoString(SYMBOL_NAME, SYMBOL_DESCRIPTION);       //

//--- Erstellen eines Nachrichtentextes mit den erhaltenen Daten
   string text=StringFormat("Symbol %s:\n"+
                            "- Currency Base: %s\n"+
                            "- Currensy Profit: %s\n"+
                            "- Currency Margin: %s\n"+
                            "- Symbol Description: %s",
                            SYMBOL_NAME, currency_base,
                            currency_profit, currency_margin,
                            symbol_descript);

//--- sende die erhaltenen Daten an das Journal
   Print(text);
   /*
   Symbol USDJPY:
   - Currency Base: USD
   - Currensy Profit: JPY
   - Currency Margin: USD
   - Symbol Description: US Dollar vs Yen
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolInfoMarginRate

Gibt die Margensätze in Abhängigkeit von der Auftragsart und -richtung zurück.

```
bool  SymbolInfoMarginRate(
   string             name,                    // Symbol
   ENUM_ORDER_TYPE    order_type,              // Auftragstyp
   double&            initial_margin_rate,     // Anfangsmarge
   double&            maintenance_margin_rate  // Erhaltungsmarge
   );
```

Parameter

name

[in] Name des Symbols.

order_type

[in] Ordertyp.

initial_margin_rate

[out] Variable vom Typ double, der die Anfangsmarge zugewiesen wird. Die Anfangsmarge ist die Höhe des Garantiebetrags für die Ausführung eines Handels mit dem Volumen von 1 Lot der entsprechenden Richtung. Multipliziert man den Koeffizienten mit der Anfangsmarge, erhält man den Betrag, der auf dem Konto reserviert wird, wenn ein Auftrag des angegebenen Typs erteilt wird.

maintenance_margin_rate

[out] Variable vom Typ double für die Erhaltungsmarge. Die Erhaltungsmarge ist der Mindestbetrag für die Aufrechterhaltung einer offenen Position von 1 Lot in der entsprechenden Richtung. Durch Multiplikation des Koeffizienten mit der Erhaltungsmarge erhält man den Betrag, der auf dem Konto reserviert wird, nachdem der Auftrag der angegebenen Art ausgelöst wurde.

Rückgabewert

Gibt true zurück, wenn der Abruf der Eigenschaften erfolgreich ausgeführt wurde, andernfalls false.

Beispiel:

```
#define SYMBOL_NAME Symbol()

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklarieren der Variablen für die Margensätze
   double initial_margin_rate = 0;      // Anfangsmarge
   double maintenance_margin_rate = 0; // Erhaltungsmarge

//--- Abrufen und Senden der Margensätze für SYMBOL_NAME für einen Markt-Kaufaufftrag
   SymbolInfoMarginRatePrint(SYMBOL_NAME, ORDER_TYPE_BUY, initial_margin_rate, mainter
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Abrufen und Senden der Margensätze für SYMBOL_NAME für einen Markt-Verkaufsaufft
   SymbolInfoMarginRatePrint(SYMBOL_NAME, ORDER_TYPE_SELL, initial_margin_rate, mainte

   /*
  Ergebnis:
   Initial margin rate for symbol AFKS for the Buy market order: 0.225600
   Maintenance margin rate for symbol AFKS for the Buy market order: 0.112800
   Initial margin rate for symbol AFKS for the Sell market order: 0.254400
   Maintenance margin rate for symbol AFKS for the Sell market order: 0.127200
   */
  }
//+------------------------------------------------------------------+
//| Sende die Margensätze an das Journal                             |
//+------------------------------------------------------------------+
void SymbolInfoMarginRatePrint(const string symbol, const ENUM_ORDER_TYPE order_type,
  {
//--- Ermitteln der Margensätze des Symbols für den Auftragstyp Marktauftrag
   ResetLastError();
   if(!SymbolInfoMarginRate(symbol, order_type, initial_margin_rate, maintenance_margi
     {
      Print("SymbolInfoMarginRate() failed. Error ", GetLastError());
      return;
     }
//--- Ausdruck der erhaltenen Margensätze
   string type=(order_type==ORDER_TYPE_BUY ? "Buy" : order_type==ORDER_TYPE_SELL ? "Se
   PrintFormat("Initial margin rate for symbol %s for the %s market order: %f\n"+
               "Maintenance margin rate for symbol %s for the %s market order: %f",
               SYMBOL_NAME, type, initial_margin_rate,
               SYMBOL_NAME, type, maintenance_margin_rate);
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolInfoTick

Gibt die aktuellen Preise für das angegebene Symbol in der Variable des Typs MqlTick zurück.

```
bool  SymbolInfoTick(
   string    symbol,      // Symbol
   MqlTick&  tick         // Strukturreferenz
   );
```

Parameter

symbol

[in] Symbolname.

tick

[out] Verweis auf die Struktur des Typs MqlTick, der die aktuelle Zeit und die Preise zugewiesen werden.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Beispiel:

```
#define SYMBOL_NAME "EURUSD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ein Array mit dem Strukturtyp MqlTick der Dimension 1 deklarieren
   MqlTick tick[1]={};

//--- Abrufen der neuesten Preise für das Symbol SYMBOL_NAME mit Hilfe der MqlTick-Str
   if(!SymbolInfoTick(SYMBOL_NAME, tick[0]))
     {
      Print("SymbolInfoTick() failed. Error ", GetLastError());
      return;
     }

//--- sende die erhaltenen Daten an das Journal
   PrintFormat("Latest price data for the '%s' symbol:", SYMBOL_NAME);
   ArrayPrint(tick);
   /*
  Ergebnis:
   Latest price data for the 'EURUSD' symbol:
                    [time]    [bid]    [ask] [last] [volume]     [time_msc] [flags] [volu
   [0] 2024.05.17 23:58:54 1.08685 1.08695 0.0000         0 1715990334319       6
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

SymbolInfoSessionQuote

Ermöglicht den Empfang von Anfangs- und Endzeitpunkt der angegebenen Kurssitzung für ein bestimmtes Symbol und einen bestimmten Wochentag.

```
bool  SymbolInfoSessionQuote(
   string            name,                // Symbolname
   ENUM_DAY_OF_WEEK  day_of_week,         // Wochentag
   uint              session_index,       // Nummer der Notierungssitzung
   datetime&         from,                // Anfang der Notierungssitzung
   datetime&         to                   // Ende der Notierungssitzung
   );
```

Parameter

name

[in] Symbolname.

ENUM_DAY_OF_WEEK

[in] Wochentag, ein Wert aus der Enumeration ENUM_DAY_OF_WEEK.

uint

[in] Ordnungszahl einer Sitzung, deren Anfangs- und Endzeit wir erhalten möchten. Die Indizierung der Sitzungen beginnt mit 0.

from

[out] Uhrzeit des Sitzungsbeginns in Sekunden ab 00 Stunden 00 Minuten, im zurückgegebenen Wert sollte das Datum ignoriert werden.

to

[out] Uhrzeit des Sitzungsendes in Sekunden ab 00 Stunden 00 Minuten, im zurückgegebenen Wert sollte das Datum ignoriert werden.

Rückgabewert

Wenn die Daten für die angegebene Kurssitzung, Symbol und Wochentag erhalten wurden, wird true zurückgegeben, anderenfalls false.

Beispiel:

```
#define SYMBOL_NAME    Symbol()
#define SESSION_INDEX 0

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ausdruck der Kopfzeile mit einem Symbol und dem SESSION_INDEX und
//--- die Ausgabe in einer Schleife von jedem Wochentag von Mo. bis Fr. und der Anfang
   PrintFormat("Symbol %s, Quote session %d:", SYMBOL_NAME, SESSION_INDEX);
   for(int i=MONDAY; i<SATURDAY; i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      SymbolInfoSessionQuotePrint(SYMBOL_NAME, (ENUM_DAY_OF_WEEK)i, SESSION_INDEX);
   /*
  Ergebnis:
   Symbol RU000A103661, Quote session 0:
   - Monday     06:45 - 00:00
   - Tuesday    06:45 - 00:00
   - Wednesday  06:45 - 00:00
   - Thursday   06:45 - 00:00
   - Friday     06:45 - 00:00
   */
  }
//+------------------------------------------------------------------+
//| Sende Anfang und Ende der angegebenen Notierungssitzung          |
//| des angegebenen Symbols und Wochentags an das Journal            |
//+------------------------------------------------------------------+
void SymbolInfoSessionQuotePrint(const string symbol, const ENUM_DAY_OF_WEEK day_of_we
  {
//--- Deklarieren der Variablen, um den Beginn und das Ende der Notierungssitzung aufz
   datetime date_from;  // Anfang der Sitzung
   datetime date_to;    // Ende der Sitzung

//--- Abrufen der Daten aus der Kurssitzung nach Symbol und Wochentag
   if(!SymbolInfoSessionQuote(symbol, day_of_week, session_index, date_from, date_to))
     {
      Print("SymbolInfoSessionQuote() failed. Error ", GetLastError());
      return;
     }

//--- den Namen des Wochentags aus dem Enumerationswert erstellen
   string week_day=EnumToString(day_of_week);
   if(week_day.Lower())
      week_day.SetChar(0, ushort(week_day.GetChar(0)-32));

//--- Daten für die angegebene Notierungssitzung an das Journal senden
   PrintFormat("- %-10s %s - %s", week_day, TimeToString(date_from, TIME_MINUTES), Tir
  }
```

Siehe auch

Information über das Symbol, TimeToStruct, Datumstruktur

© 2000-2025, MetaQuotes Ltd.

SymbolInfoSessionTrade

Ruft Anfang und Ende der angegebenen Handelssitzung für das angegebene Symbol und Wochentag ab.

```
bool  SymbolInfoSessionTrade(
   string            name,                // Symbolname
   ENUM_DAY_OF_WEEK  day_of_week,         // Wochentag
   uint              session_index,       // Nummer der Handelssitzung
   datetime&         from,                // Anfang der Handelssitzung
   datetime&         to                   // Ende der Handelssitzung
   );
```

Parameter

name

[in] Symbolname.

ENUM_DAY_OF_WEEK

[in] Wochentag, ein Wert aus der Enumeration ENUM_DAY_OF_WEEK.

uint

[in] Ordnungszahl einer Sitzung, deren Anfangs- und Endzeit wir erhalten möchten. Die Indexierung der Handelssitzung beginnt mit 0.

from

[out] Anfang der Handelssitzung in Sekunden von 00 Stunden 00 Minuten, im erhaltenen Wert muss das Datum ignoriert werden.

to

[out] Ende der Handelssitzung in Sekunden von 00 Stunden 00 Minuten, im erhaltenen Wert muss das Datum ignoriert werden.

Rückgabewert

Wenn die Daten für die angegebene Handelssitzung, Symbol und Wochentag erhalten wurden, wird true zurückgegeben, anderenfalls false.

Beispiel:

```
#define SYMBOL_NAME    Symbol()
#define SESSION_INDEX 0

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ausdruck der Kopfzeile mit einem Symbol und dem SESSION_INDEX und
//--- die Ausgabe in einer Schleife von jedem Wochentag von Mo bis Fr und der Anfangs-
   PrintFormat("Symbol %s, Trade session %d:", SYMBOL_NAME, SESSION_INDEX);
   for(int i=MONDAY; i<SATURDAY; i++)
      SymbolInfoSessionTradePrint(SYMBOL_NAME, (ENUM_DAY_OF_WEEK)i, SESSION_INDEX);
```

© 2000-2025, MetaQuotes Ltd.

```
   /*
  Ergebnis:
   Symbol GBPUSD, Trade session 0:
   - Monday     00:15 - 23:55
   - Tuesday    00:15 - 23:55
   - Wednesday  00:15 - 23:55
   - Thursday   00:15 - 23:55
   - Friday     00:15 - 23:55
   */
  }
//+------------------------------------------------------------------+
//| Sende die Start- und Endzeiten der angegebenen Handelssitzung    |
//| des angegebenen Symbols und Wochentags an das Journal            |
//+------------------------------------------------------------------+
void SymbolInfoSessionTradePrint(const string symbol, const ENUM_DAY_OF_WEEK day_of_we
  {
//--- Deklarieren der Variablen, um den Beginn und das Ende der Handelssitzung aufzuze
   datetime date_from;  // Anfang der Sitzung
   datetime date_to;    // Ende der Sitzung

//--- Abrufen der Daten aus der Handelssitzung nach Symbol und Wochentag
   if(!SymbolInfoSessionTrade(symbol, day_of_week, session_index, date_from, date_to))
     {
      Print("SymbolInfoSessionTrade() failed. Error ", GetLastError());
      return;
     }

//--- den Namen des Wochentags aus dem Enumerationswert erstellen
   string week_day=EnumToString(day_of_week);
   if(week_day.Lower())
      week_day.SetChar(0, ushort(week_day.GetChar(0)-32));

//--- Daten für die angegebene Handelssitzung an das Journal senden
   PrintFormat("- %-10s %s - %s", week_day, TimeToString(date_from, TIME_MINUTES), Tir
  }
```

Siehe auch

Information über das Instrument, TimeToStruct, Datumstruktur

© 2000-2025, MetaQuotes Ltd.

MarketBookAdd

Ermöglicht die Eröffnung der Markttiefe (Depth of Market, DOM) für ein ausgewähltes Symbol und die Anmeldung zum Erhalt von Benachrichtigungen über DOM-Änderungen.

```
bool  MarketBookAdd(
   string  symbol      // Symbol
   );
```

Parameter

symbol

[in] Symbolname, dessen DOM in dem angegebenen Experten oder Script benutzt werden soll.

Rückgabewert

Wert true im Fall der erfolgreichen Eröffnung, anderenfalls false.

Hinweis

In der Regel muss diese Funktion aus der Funktion OnInit() oder im Klassenkonstrukteur aufgerufen werden. Für Verarbeitung der Eingangsmeldungen muss die Funktion void OnBookEvent(string& symbol) vorhanden sein.

Beispiel:

```
#define   SYMBOL_NAME   "GBPUSD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Öffnen der Markttiefe für das Symbol SYMBOL_NAME
   if(!MarketBookAdd(SYMBOL_NAME))
     {
      PrintFormat("MarketBookAdd(%s) failed. Error ", SYMBOL_NAME, GetLastError());
      return;
     }

//--- Senden der Nachricht über das erfolgreiche Öffnen der Markttiefe an das Journal
   PrintFormat("The MarketBook for the '%s' symbol was successfully opened and a subsc

//--- warte 2 Sekunden
   Sleep(2000);

//--- nach Beendigung, Abmeldung der geöffneten Markttiefe
   ResetLastError();
   if(MarketBookRelease(SYMBOL_NAME))
      PrintFormat("MarketBook for the '%s' symbol was successfully closed", SYMBOL_NAM
   else
      PrintFormat("Error %d occurred when closing MarketBook using the '%s' symbol", G
```

© 2000-2025, MetaQuotes Ltd.

```
 /*
Ergebnis:
 The MarketBook for the 'GBPUSD' symbol was successfully opened and a subscription t
 MarketBook for the 'GBPUSD' symbol was successfully closed
 */
}
```

Siehe auch

DOM Struktur , Strukturen und Klassen

© 2000-2025, MetaQuotes Ltd.

MarketBookRelease

Schließt die Markttiefe (Depth of Market, DOM) für das ausgewählte Symbol und beendet das Abonnement für den Empfang von Benachrichtigungen über die DOM-Änderungen.

```
bool  MarketBookRelease(
   string  symbol      // Symbolname
   );
```

Parameter

symbol

[in] Symbolname.

Rückgabewert

true bei erfolgreichem Schließen, sonst falsch.

Hinweis

Normalerweise muss diese Funktion aus der Funktion OnDeinit() aufgerufen werden, wenn in der Funktion OnInit() die entsprechende Funktion MarketBookAdd() aufgerufen wurde. Oder sie muss vom Destruktor der Klasse aufgerufen werden, wenn die entsprechende Funktion MarketBookAdd() im Konstruktor dieser Klasse aufgerufen wird.

Beispiel:

```
#define SYMBOL_NAME "GBPUSD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Öffnen der Markttiefe für das Symbol SYMBOL_NAME
   if(!MarketBookAdd(SYMBOL_NAME))
     {
      PrintFormat("MarketBookAdd(%s) failed. Error ", SYMBOL_NAME, GetLastError());
      return;
     }

//--- Senden der Nachricht über das erfolgreiche Öffnen der Markttiefe an das Journal
   PrintFormat("The MarketBook for the '%s' symbol was successfully opened and a subsc

//--- warte 2 Sekunden
   Sleep(2000);

//--- nach Beendigung, Abmeldung der geöffneten Markttiefe
//--- sende die Nachricht über die erfolgreiche Abmeldung der Markttiefe oder des Fehl
   ResetLastError();
   if(MarketBookRelease(SYMBOL_NAME))
      PrintFormat("MarketBook for the '%s' symbol was successfully closed", SYMBOL_NAM
```

© 2000-2025, MetaQuotes Ltd.

```
   else
      PrintFormat("Error %d occurred when closing MarketBook using the '%s' symbol", G

   /*
  Ergebnis:
   The MarketBook for the 'GBPUSD' symbol was successfully opened and a subscription t
   MarketBook for the 'GBPUSD' symbol was successfully closed
   */
  }
```

Siehe auch

DOM Struktur, Strukturen und Klassen

© 2000-2025, MetaQuotes Ltd.

MarketBookGet

Gibt Feld der Strukturen MqlBookInfo, das DOM Aufzeichnungen für das gewaehlte Symbol enthält.

```
bool  MarketBookGet(
   string         symbol,      // Symbol
   MqlBookInfo&   book[]       // Referenz auf das Feld
   );
```

Parameter

symbol

[in] Symbolname.

book[]

[out] Verweis auf das Feld der DOM Aufzeichnungen. Feld kann vorlaeufig verteilt werden für ausreichende Zahl der Aufzeichnungen. Wenn das dynamische Feld nicht vorlaeufig im Operativspeicher verteilt wurde, wird das Client-Terminal dieses Feld selbststaendig verteilen.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

DOM muss vorlaeufig durch die Funktion MarketBookAdd() geoeffnet werden.

Beispiel:

```
   MqlBookInfo priceArray[];
   bool getBook=MarketBookGet(NULL,priceArray);
   if(getBook)
     {
      int size=ArraySize(priceArray);
      Print("MarketBookInfo für",Symbol());
      for(int i=0;i<size;i++)
        {
         Print(i+": ",priceArray[i].price
               +"    Volume = "+priceArray[i].volume,
               " type = ",priceArray[i].type);
        }
     }
   else
     {
      Print("Kann nicht Inhalt des Symbols DOM erhalten ",Symbol());
     }
```

Sehen Sie auch

DOM Struktur, Strukturen und Klassen

© 2000-2025, MetaQuotes Ltd.

Funktionen des Wirtschaftskalenders

Dieser Abschnitt beschreibt die Funktionen für das Arbeiten mit dem Wirtschaftskalender, die direkt auf der MetaTrader-Plattform verfügbar sind. Der Wirtschaftskalender ist eine vorgefertigte Enzyklopädie mit Beschreibungen makroökonomischer Indikatoren, deren Veröffentlichungsterminen und Wichtigkeitsmerkmalen. Relevante Werte makroökonomischer Indikatoren werden direkt zum Zeitpunkt der Veröffentlichung an die MetaTrader-Plattform gesendet und in einem Chart als Tags angezeigt, mit denen Sie die erforderlichen Indikatoren nach Ländern, Währungen und Bedeutung visuell verfolgen können.

Alle Funktionen, die mit dem Wirtschaftskalender arbeiten, nutzen die Zeit des Handelsservers (TimeTradeServer). Das bedeutet, dass die Zeit in der Struktur MqlCalendarValue und die Zeitangaben der Funktionen CalendarValueHistoryByEvent/CalendarValueHistory in der Zeitzone des Handelsservers eingestellt sind und nicht in der lokalen Zeitzone des Nutzers.

Die Funktionen des Wirtschaftskalenders ermöglichen die automatische Analyse eingehender Ereignisse nach benutzerdefinierten Kriterien der Wichtigkeit aus der Perspektive der benötigten Länder/Währungen.

Funktion	Aktion
CalendarCountryById	Abrufen der Länderbeschreibung nach seiner ID
CalendarEventById	Abrufen der Ereignisbeschreibung nach seiner ID
CalendarValueById	Abrufen der Ereigniswertbeschreibung nach seiner ID
CalendarCountries	Abrufen des Arrays der Namen der im Kalender verfügbaren Länder
CalendarEventByCountry	Abrufen des Arrays mit den Beschreibungen aller im Kalender verfügbaren Ereignisse nach dem angegebenen Landes-Code
CalendarEventByCurrency	Abrufen des Arrays der Beschreibungen aller im Kalender verfügbaren Ereignisse in einer bestimmten Währung
CalendarValueHistoryByEvent	Abrufen des Arrays der Werte für alle Ereignisse in einem bestimmten Zeitraum mit einem Ereignis-ID
CalendarValueHistory	Abrufen des Arrays der Werte für alle Ereignisse in einem bestimmten Zeitbereich mit der Möglichkeit, nach Land und/oder Währung zu sortieren
CalendarValueLastByEvent	Abrufen des Arrays der Ereigniswerte nach seiner ID seit dem Status der Kalenderdatenbank nach einer angegebenen change_id
CalendarValueLast	Abrufen des Arrays der Werte für alle Ereignisse mit der Möglichkeit, nach Land und/oder Währung seit dem Status der Kalenderdatenbank mit einer angegebenen change_id zu sortieren

© 2000-2025, MetaQuotes Ltd.

CalendarCountryById

Abrufen der Länderbeschreibung nach seiner ID

```
bool  CalendarCountryById(
   const long           country_id,     // Länder-ID
   MqlCalendarCountry&  country         // Variable für die Übernahme der Länderbeschr
   );
```

Parameter

country_id

[in] Country ID (ISO 3166-1).

country

[out] Variable vom Typ MqlCalendarCountry für die Übernahme der Länderbeschreibungen.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 5402 - ERR_CALENDAR_NO_DATA (Land wurde nicht gefunden),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten).

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der Liste der Länder des Kalenders
   MqlCalendarCountry countries[];
   int count=CalendarCountries(countries);
//--- Ergebnisprüfung
   if(count==0)
      PrintFormat("CalendarCountries() lieferte 0! Error %d",GetLastError());
//--- wenn es zwei oder mehr Länder gibt
   if(count>=2)
     {
      MqlCalendarCountry country;
      //--- Abrufen der Länderbeschreibungen nach deren ID
      if(CalendarCountryById(countries[1].id, country))
        {
         //--- Vorbereitung der Länderbeschreibung
         string descr="id = "+IntegerToString(country.id)+"\n";
         descr+=("name = " + country.name+"\n");
         descr+=("code = " + country.code+"\n");
         descr+=("currency = " + country.currency+"\n");
         descr+=("currency_symbol = " + country.currency_symbol+"\n");
```

© 2000-2025, MetaQuotes Ltd.

```
         descr+=("url_name = " + country.url_name);
         //--- Anzeige einer Länderbeschreibung
         Print(descr);
        }
      else
         Print("CalendarCountryById() fehlgeschlagen. Error ",GetLastError());
     }
//---
  }
/*
  Ergebnis:
   id = 999
   name = European Union
   code = EU
   currency = EUR
   currency_symbol = €
   url_name = european-union
*/
```

Siehe auch

CalendarCountries, CalendarEventByCountry

© 2000-2025, MetaQuotes Ltd.

CalendarEventById

Abrufen der Ereignisbeschreibung nach seiner ID.

```
bool  CalendarEventById(
   ulong                event_id,      // Ereignis-ID
   MqlCalendarEvent&    event          // Variable zur Übernahme einer Ereignisbeschrei
   );
```

Parameter

event_id

[in] Ereignis-ID.

event

[out] Variable vom Typ MqlCalendarEvent für die Übernahme der Ereignisbeschreibung.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 5402 - ERR_CALENDAR_NO_DATA (Land wurde nicht gefunden),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten).

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Länder-Code für Deutschland (ISO 3166-1 Alpha-2)
   string germany_code="DE";
//--- Abrufen der Ereignisse Deutschlands
   MqlCalendarEvent events[];
   int events_count=CalendarEventByCountry(germany_code,events);
//--- Anzeige der Ereignisse Deutschlands im Journal
   if(events_count>0)
     {
      PrintFormat("Ereignisse Deutschlands: %d",events_count);
      ArrayPrint(events);
     }
   else
     {
      PrintFormat("Fehler beim Erhalt der Ereignisse für den Länder-Code %s, error %d"
                  germany_code,GetLastError());
      //--- Skript wurde vorzeitig beendet
      return;
     }
//--- Abrufen der Beschreibung des letzten Ereignisses vom Array events[]
```

© 2000-2025, MetaQuotes Ltd.

```
   MqlCalendarEvent event;
   ulong event_id=events[events_count-1].id;
   if(CalendarEventById(event_id,event))
     {
      MqlCalendarCountry country;
      CalendarCountryById(event.country_id,country);
      PrintFormat("Ereignisbeschreibung von event_id=%d erhalten",event_id);
      PrintFormat("Land: %s (Länder-Code = %d)",country.name,event.country_id);
      PrintFormat("Ereignisname: %s",event.name);
      PrintFormat("Ereignis-Code: %s",event.event_code);
      PrintFormat("Ereigniswichtigkeit: %s",EnumToString((ENUM_CALENDAR_EVENT_IMPORTAN
      PrintFormat("Ereignis-Typ: %s",EnumToString((ENUM_CALENDAR_EVENT_TYPE)event.type
      PrintFormat("Ereignis-Sektor: %s",EnumToString((ENUM_CALENDAR_EVENT_SECTOR)event
      PrintFormat("Ereignis-Häufigkeit: %s",EnumToString((ENUM_CALENDAR_EVENT_FREQUENC
      PrintFormat("Art der Ereignisveröffentlichung: %s",EnumToString((ENUM_CALENDAR_E
      PrintFormat("Ereignis-Maßeinheit: %s",EnumToString((ENUM_CALENDAR_EVENT_UNIT)eve
      PrintFormat("Anzahl der Dezimalstellen: %d",event.digits);
      PrintFormat("Ereignis-Multiplikator: %s",EnumToString((ENUM_CALENDAR_EVENT_MULTI
      PrintFormat("Quell-URL: %s",event.source_url);
     }
   else
      PrintFormat("Fehler beim Erhalt der Ereignisbeschreibung für event_d=%s, Fehler
                  event_id,GetLastError());
  }
/*
  Ergebnis:
  Ereignisse Deutschlands: 50
              [id] [type] [sector] [frequency] [time_mode] [country_id] [unit] [importa
   [ 0] 276010001      1        6           2           0          276      1
   [ 1] 276010002      1        6           2           0          276      1
   [ 2] 276010003      1        4           2           0          276      1
   [ 3] 276010004      1        4           2           0          276      1
   ....
   [47] 276500001      1        8           2           0          276      0
   [48] 276500002      1        8           2           0          276      0
   [49] 276500003      1        8           2           0          276      0
  Ereignisbeschreibung für event_id=276500003 erhalten
  Land: Deutschland (Länder-Code = 276)
  Ereignis-Name: Markit Composite PMI
  Ereignis-Code: markit-composite-pmi
   Ereigniswichtigkeit: CALENDAR_IMPORTANCE_MODERATE
   Ereignis-Typ: CALENDAR_TYPE_INDICATOR
   Ereignis-Sektor: CALENDAR_SECTOR_BUSINESS
   Ereignis-Häufigkeit: CALENDAR_FREQUENCY_MONTH
   Art der Ereignisveröffentlichung: CALENDAR_TIMEMODE_DATETIME
   Ereignis-Maßeinheit: CALENDAR_UNIT_NONE
  Anzahl der Dezimalstellen: 1
   Ereignis-Multiplikator: CALENDAR_MULTIPLIER_NONE
   Quell-URL: https://www.markiteconomics.com
```

© 2000-2025, MetaQuotes Ltd.

```
*/
```

Siehe auch

CalendarEventByCountry, CalendarEventByCurrency, CalendarValueById

© 2000-2025, MetaQuotes Ltd.

CalendarValueById

Abrufen der Ereigniswertbeschreibung nach seiner ID.

```
bool  CalendarValueById(
   ulong                value_id,      // Ereigniswert-ID
   MqlCalendarValue&    value          // Variable für die Übernahme des Ereigniswerts
   );
```

Parameter

value_id

[in] Ereigniswert-ID.

Wert

[out] Variable vom Typ MqlCalendarValue für die Übernahme der Ereignisbeschreibung. Beispiele der Behandlung von Kalenderereignisse.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 5402 - ERR_CALENDAR_NO_DATA (Land wurde nicht gefunden),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten).

Hinweis

Alle Funktionen, die mit dem Wirtschaftskalender arbeiten, nutzen die Zeit des Handelsservers (TimeTradeServer). Das bedeutet, dass die Zeit in der Struktur MqlCalendarValue und die Zeitangaben der Funktionen CalendarValueHistoryByEvent/CalendarValueHistory in der Zeitzone des Handelsservers eingestellt sind und nicht in der lokalen Zeitzone des Nutzers.

Die Struktur von MqlCalendarValue bietet Methoden zum Prüfen und Setzen von Werten der Felder actual_value, forecast_value, prev_value und revised_prev_value. Wenn kein Wert angegeben wird, speichert das Feld LONG_MIN (-9223372036854775808).

Bitte beachten Sie, dass die in diesem Feld gespeicherten Werte mit einer Million multipliziert werden. Dies bedeutet, dass, wenn Sie Werte von MqlCalendarValue mithilfe der Funktionen CalendarValueById, CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent und CalendarValueLast abrufen, sollten Sie überprüfen, ob die Feldwerte gleich LONG_MIN sind; wenn in einem Feld ein Wert angegeben ist, sollten Sie den Wert durch 1.000.000 dividieren, um den Wert zu erhalten. Eine andere Methode zum Abrufen der Werte besteht darin, die Werte mit den Funktionen der Struktur MqlCalendarValue zu überprüfen und abzurufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Länder-Code von Japan (ISO 3166-1 Alpha-2)
   string japan_code="JP";
//--- Festlegen der Zeitgrenzen des Intervalls für die Ereignisse
   datetime date_from=D'01.01.2018';  // alle Ereignisse ab 2018
   datetime date_to=0;                // 0 bedeutet, alle bekannten Ereignisse, einsch
//--- Abrufen des Arrays mit den Ereignissen von Japan
   MqlCalendarValue values[];
   int values_count=CalendarValueHistory(values,date_from,date_to,japan_code);
//--- Durchlaufen aller erkannten Ereigniswerte
   if(values_count>0)
     {
      PrintFormat("Anzahl der Ereigniswerte für Japan: %d",values_count);
      //--- Löschen aller "leerer" Werte (actual_value==-9223372036854775808)
      for(int i=values_count-1;i>=0;i--)
        {
         if(values[i].actual_value==-9223372036854775808)
            ArrayRemove(values,i,1);
        }
      PrintFormat("Anzahl der Ereigniswerte nach dem Löschen der Leerwerte: %d",ArrayS
     }
   else
     {
      PrintFormat("Fehler beim Erhalt der Ereignisse für den Länder-Code %s, error %d'
                  japan_code,GetLastError());
      //--- Skript wurde vorzeitig beendet
      return;
     }
//--- Behalten von nicht mehr als 10 Werte im Array values[]
   if(ArraySize(values)>10)
     {
      PrintFormat("Reduzieren und Anzeigen der Liste auf 10 Werte");
      ArrayRemove(values,0,ArraySize(values)-10);
     }
   ArrayPrint(values);

//--- Anzeigen der Ereigniswertbeschreibung auf Basis von value_id
   for(int i=0;i<ArraySize(values);i++)
     {
      MqlCalendarValue value;
      CalendarValueById(values[i].id,value);
      PrintFormat("%d: value_id=%d value=%d impact=%s",
                  i,values[i].id,value.actual_value,EnumToString(ENUM_CALENDAR_EVENT_I
     }
//---
  }
/*
  Ergebnis:
  Anzahl der Ereigniswerte für Japan: 1734
  Anzahl der Ereigniswerte nach dem Löschen der Leerwerte: 1017
```

© 2000-2025, MetaQuotes Ltd.

```
  Reduzieren und Anzeigen der Liste auf 10 Werte
        [id] [event_id]              [time]             [period] [revision] [actual_va
   [0] 56500  392030004 2019.03.28 23:30:00 2019.03.01 00:00:00          0          90
   [1] 56501  392030005 2019.03.28 23:30:00 2019.03.01 00:00:00          0          70
   [2] 56502  392030006 2019.03.28 23:30:00 2019.03.01 00:00:00          0         110
   [3] 56544  392030007 2019.03.28 23:30:00 2019.02.01 00:00:00          0         230
   [4] 56556  392050002 2019.03.28 23:30:00 2019.02.01 00:00:00          0         163
   [5] 55887  392020003 2019.03.28 23:50:00 2019.02.01 00:00:00          0          40
   [6] 55888  392020004 2019.03.28 23:50:00 2019.02.01 00:00:00          0        -180
   [7] 55889  392020002 2019.03.28 23:50:00 2019.02.01 00:00:00          0          20
   [8] 55948  392020006 2019.03.28 23:50:00 2019.02.01 00:00:00          1         140
   [9] 55949  392020007 2019.03.28 23:50:00 2019.02.01 00:00:00          1        -100
  Anzeigen der verkürzten Daten eines Ereigniswertes auf Basis von value_id
   0: value_id=56500 value=900000 impact=CALENDAR_IMPACT_POSITIVE
   1: value_id=56501 value=700000 impact=CALENDAR_IMPACT_NA
   2: value_id=56502 value=1100000 impact=CALENDAR_IMPACT_POSITIVE
   3: value_id=56544 value=2300000 impact=CALENDAR_IMPACT_NEGATIVE
   4: value_id=56556 value=1630000 impact=CALENDAR_IMPACT_POSITIVE
   5: value_id=55887 value=400000 impact=CALENDAR_IMPACT_NEGATIVE
   6: value_id=55888 value=-1800000 impact=CALENDAR_IMPACT_POSITIVE
   7: value_id=55889 value=200000 impact=CALENDAR_IMPACT_NEGATIVE
   8: value_id=55948 value=1400000 impact=CALENDAR_IMPACT_POSITIVE
   9: value_id=55949 value=-1000000 impact=CALENDAR_IMPACT_NEGATIVE
*/
```

Siehe auch

CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent, CalendarValueLast

© 2000-2025, MetaQuotes Ltd.

CalendarCountries

Abrufen des im Kalender verfügbaren Arrays der Ländernamen.

```
int  CalendarCountries(
 MqlCalendarCountry& countries[]   // Array mit der Liste der Beschreibungen der Lände
   );
```

Parameter

countries[]

[out] Ein Array von Typ MqlCalendarCountry für die Übernahme aller Beschreibungen der Kalenderländer.

Rückgabewert

Anzahl der empfangenen Beschreibungen. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- 5400 - ERR_CALENDAR_MORE_DATA (Array ist zu klein, um die Beschreibungen aller Länder zu übernehmen, es wurden nur diejenigen übernommen, die eingetragen werden konnten).

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der Liste der Länder des Kalenders
   MqlCalendarCountry countries[];
   int count=CalendarCountries(countries);
//--- Anzeigen des Arrays im Journal
   if(count>0)
      ArrayPrint(countries);
   else
      PrintFormat("CalendarCountries() lieferte 0! Error %d",GetLastError());
/*
  Ergebnis:
        [id]           [name] [code] [currency] [currency_symbol]       [url_name] [re
   [ 0]    0 "Worldwide"      "WW"   "ALL"      ""                "worldwide"
   [ 1]  999 "European Union" "EU"   "EUR"      "€"               "european-union"
   [ 2]  840 "United States"  "US"   "USD"      "$"               "united-states"
   [ 3]  124 "Canada"         "CA"   "CAD"      "$"               "canada"
   [ 4]   36 "Australia"      "AU"   "AUD"      "$"               "australia"
   [ 5]  554 "New Zealand"    "NZ"   "NZD"      "$"               "new-zealand"
   [ 6]  392 "Japan"          "JP"   "JPY"      "Ŧ"               "japan"
   [ 7]  156 "China"          "CN"   "CNY"      "Ŧ"               "china"
   [ 8]  826 "United Kingdom" "GB"   "GBP"      "J"               "united-kingdom"
```

© 2000-2025, MetaQuotes Ltd.

```
   [ 9]  756 "Switzerland"    "CH"   "CHF"      "₣"              "switzerland"
   [10]  276 "Germany"        "DE"   "EUR"      "€"              "germany"
   [11]  250 "France"         "FR"   "EUR"      "€"              "france"
   [12]  380 "Italy"          "IT"   "EUR"      "€"              "italy"
   [13]  724 "Spain"          "ES"   "EUR"      "€"              "spain"
   [14]   76 "Brazil"         "BR"   "BRL"      "R$"             "brazil"
   [15]  410 "South Korea"    "KR"   "KRW"      "₩"              "south-korea"
*/
  }
```

Siehe auch

CalendarEventByCountry, CalendarCountryById

© 2000-2025, MetaQuotes Ltd.

CalendarEventByCountry

Abrufen des Arrays mit den Beschreibungen aller im Kalender verfügbaren Ereignisse nach dem angegebenen Landes-Code.

```
int  CalendarEventByCountry(
   string               country_code,     //Code-Name des Landes (ISO 3166-1 alpha-2)
   MqlCalendarEvent&    events[]          // Variable zur Übernahme des Arrays der Bes
   );
```

Parameter

country_code

[in] Code-Name des Landes (ISO 3166-1 alpha-2)

events[]

[out] Array vom Typ MqlCalendarEvent für die Übernahme der Beschreibungen aller Ereignisse des angegebenen Landes.

Rückgabewert

Anzahl der empfangenen Beschreibungen. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 4004 - ERR_NOT_ENOUGH_MEMORY (nicht genügend Speicherplatz für die Ausführung der Anforderung),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- Fehler bei der Ausführung von ArrayResize()

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Länder-Code der EU (ISO 3166-1 Alpha-2)
   string EU_code="EU";
//--- Abrufen der Ereignisse der EU
   MqlCalendarEvent events[];
   int events_count=CalendarEventByCountry(EU_code,events);
//--- Anzeige der Ereignisse der EU im Journal
   if(events_count>0)
     {
      PrintFormat("EU Ereignisse: %d",events_count);
      ArrayPrint(events);
     }
//---
  }
/*
  Ergebnis:
```

© 2000-2025, MetaQuotes Ltd.

```
  EU events: 56
              [id] [type]  [country_id] [unit] [importance] [multiplier] [digits] [ever
   [ 0] 999010001      0           999      0             2            0         0  "ECB
   [ 1] 999010002      0           999      0             2            0         0  "ECB
   [ 2] 999010003      0           999      0             3            0         0  "ECB
   [ 3] 999010004      0           999      0             3            0         0  "ECB
   [ 4] 999010005      0           999      0             2            0         0  "ECB
   [ 5] 999010006      1           999      1             3            0         2  "ECB
   [ 6] 999010007      1           999      1             3            0         2  "ECB
   [ 7] 999010008      0           999      0             2            0         0  "ECB
   [ 8] 999010009      1           999      2             2            3         3  "ECB
   [ 9] 999010010      0           999      0             2            0         0  "ECB
   [10] 999010011      0           999      0             2            0         0  "ECB
    ...

*/
```

Siehe auch

CalendarCountries, CalendarCountryById

© 2000-2025, MetaQuotes Ltd.

CalendarEventByCurrency

Abrufen des Arrays der Beschreibungen aller im Kalender verfügbaren Ereignisse in einer bestimmten Währung.

```
int  CalendarEventByCurrency(
   const string         currency,     // Währungs-Code des Landes
   MqlCalendarEvent&    events[]      // Variable zur Übernahme des Arrays der Beschre
   );
```

Parameter

currency

[in] Währungs-Code des Landes.

events[]

[out] Array vom Typ MqlCalendarEvent für die Übernahme der Beschreibungen aller Ereignisse der angegebenen Währung.

Rückgabewert

Anzahl der empfangenen Beschreibungen. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 4004 - ERR_NOT_ENOUGH_MEMORY (nicht genügend Speicherplatz für die Ausführung der Anforderung),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- Fehler bei der Ausführung von ArrayResize()

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Deklarieren des Arrays zur Übernahme der Ereignisse des Wirtschaftskalenders
      MqlCalendarEvent events[];
//--- Abrufen der Ereignisse der Währung der EU
      int count = CalendarEventByCurrency("EUR",events);
      Print("count = ", count);
//--- 10 Ereignisse sind für das aktuelle Beispiel ausreichend
      if(count>10)
         ArrayResize(events,10);
//--- Anzeige der Ereignisse im Journal
      ArrayPrint(events);
  }
/*
  Ergebnis:
             [id] [type] [country_id] [unit] [importance]
```

© 2000-2025, MetaQuotes Ltd.

```
   [0] 999010001       0           999       0           2 "https://www.ecb.europa.eu/ho
   [1] 999010002       0           999       0           2 "https://www.ecb.europa.eu/ho
   [2] 999010003       0           999       0           3 "https://www.ecb.europa.eu/ho
   [3] 999010004       0           999       0           3 "https://www.ecb.europa.eu/ho
   [4] 999010005       0           999       0           2 "https://www.ecb.europa.eu/ho
   [5] 999010006       1           999       1           3 "https://www.ecb.europa.eu/ho
   [6] 999010007       1           999       1           3 "https://www.ecb.europa.eu/ho
   [7] 999010008       0           999       0           2 "https://www.ecb.europa.eu/ho
   [8] 999010009       1           999       2           2 "https://www.ecb.europa.eu/ho
   [9] 999010010       0           999       0           2 "https://www.ecb.europa.eu/ho
*/
```

Siehe auch

CalendarEventById, CalendarEventByCountry

© 2000-2025, MetaQuotes Ltd.

CalendarValueHistoryByEvent

Abrufen des Arrays der Werte für alle Ereignisse in einem bestimmten Zeitraum mit einem Ereignis-ID.

```
bool  CalendarValueHistoryByEvent(
   ulong              event_id,          // Ereignis-ID
   MqlCalendarValue&  values[],          // Array der Wertbeschreibungen
   datetime           datetime_from,     // Anfangszeitpunkt des Zeitraums
   datetime           datetime_to=0      // Endzeitpunkt des Zeitraums
   );
```

Parameter

event_id

[in] Ereignis-ID.

values[]

[out] Variable vom Typ MqlCalendarValue für die Übernahme der Ereigniswerte. Beispiele der Behandlung von Kalenderereignisse.

datetime_from

[in] Anfangszeitpunkt des Zeitraums der Ereignisse für die angegebene ID, während *datetime_from* < *datetime_to*.

datetime_to=0

[in] Endzeitpunkt des Zeitraums der Ereignisse für die angegebene ID. Wenn *datetime_to* nicht (oder auf 0) gesetzt wurde, werden alle Ereignisse ab dem angegebenen Zeitpunkt *datetime_from* aus der Kalenderdatenbank (inklusive der zukünftigen) zurückgegeben.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 4004 - ERR_NOT_ENOUGH_MEMORY (nicht genügend Speicherplatz für die Ausführung der Anforderung),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- 5400 - ERR_CALENDAR_MORE_DATA (Array ist zu klein, um die Beschreibungen aller Werte zu übernehmen, es wurden nur diejenigen übernommen, die eingetragen werden konnten).
- Fehler bei der Ausführung von ArrayResize()

Hinweis

Alle Funktionen, die mit dem Wirtschaftskalender arbeiten, nutzen die Zeit des Handelsservers (TimeTradeServer). Das bedeutet, dass die Zeit in der Struktur MqlCalendarValue und die Zeitangaben der Funktionen CalendarValueHistoryByEvent/CalendarValueHistory in der Zeitzone des Handelsservers eingestellt sind und nicht in der lokalen Zeitzone des Nutzers.

Die Struktur von MqlCalendarValue bietet Methoden zum Prüfen und Setzen von Werten der Felder actual_value, forecast_value, prev_value und revised_prev_value. Wenn kein Wert angegeben wird, speichert das Feld LONG_MIN (-9223372036854775808).

© 2000-2025, MetaQuotes Ltd.

Bitte beachten Sie, dass die in diesem Feld gespeicherten Werte mit einer Million multipliziert werden. Dies bedeutet, dass, wenn Sie Werte von MqlCalendarValue mithilfe der Funktionen CalendarValueById, CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent und CalendarValueLast abrufen, sollten Sie überprüfen, ob die Feldwerte gleich LONG_MIN sind; wenn in einem Feld ein Wert angegeben ist, sollten Sie den Wert durch 1.000.000 dividieren, um den Wert zu erhalten. Eine andere Methode zum Abrufen der Werte besteht darin, die Werte mit den Funktionen der Struktur MqlCalendarValue zu überprüfen und abzurufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Länder-Code der EU (ISO 3166-1 Alpha-2)
   string EU_code="EU";
//--- Abrufen der Ereignisse der EU
   MqlCalendarEvent events[];
   int events_count=CalendarEventByCountry(EU_code,events);
//--- Anzeige der Ereignisse der EU im Journal
   if(events_count>0)
     {
      PrintFormat("EU Ereignisse: %d",events_count);
      //--- Verkleinern der Ereignisliste, 10 Ereignisse reichen für die Analyse aus
      ArrayResize(events,10);
      ArrayPrint(events);
     }
//--- Anzeige, dass das Ereignis "ECB Interest Rate Decision" die ID event_id=99901000
   ulong event_id=events[6].id;        // die Ereignis-ID kann sich im Kalender änderr
   string event_name=events[6].name;   // Name des Ereignisses im Kalender
   PrintFormat("Abrufen der Werte für event_name=%s event_id=%d",event_name,event_id);
//--- Abrufen aller Werte des Ereignisses "ECB Interest Rate Decision"
   MqlCalendarValue values[];
//--- Festlegen der Zeitgrenzen des Intervalls für die Ereignisse
   datetime date_from=0;           // Abrufen aller Ereignisse ab dem ersten in der ve
   datetime date_to=D'01.01.2016'; // Übernehmen nur der Ereignisse, die nicht älter s
   if(CalendarValueHistoryByEvent(event_id,values,date_from,date_to))
     {
      PrintFormat("Erhaltene Werte für %s: %d",
                  event_name,ArraySize(values));
      //--- Verkleinern der Werteliste, 10 Werte reichen für die Analyse aus
      ArrayResize(values,10);
      ArrayPrint(values);
     }
   else
     {
      PrintFormat("Fehler! Fehler beim Abrufen von event_id=%d",event_id);
      PrintFormat("Fehlernummer: %d",GetLastError());
     }
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//---
/*
  Ergebnis:
  EU events: 56
           [id] [type] [sector] [frequency] [time_mode] [country_id] [unit] [importar
   [0] 999010001      0        5           0           0           999      0
   [1] 999010002      0        5           0           0           999      0
   [2] 999010003      0        5           0           0           999      0
   [3] 999010004      0        5           0           0           999      0
   [4] 999010005      0        5           0           0           999      0
   [5] 999010006      1        5           0           0           999      1
   [6] 999010007      1        5           0           0           999      1
   [7] 999010008      0        5           0           0           999      0
   [8] 999010009      1        5           0           0           999      2
   [9] 999010010      0        5           0           0           999      0
  Abrufen der Werte für event_name=ECB Interest Rate Decision, event_id=999010007
  Erhaltene Ereigniswerte für ECB Interest Rate Decision: 102
       [id] [event_id]              [time]            [period] [revision] [actual_valu
   [0] 2776  999010007 2007.03.08 11:45:00 1970.01.01 00:00:00          0        3750(
   [1] 2777  999010007 2007.05.10 11:45:00 1970.01.01 00:00:00          0        3750(
   [2] 2778  999010007 2007.06.06 11:45:00 1970.01.01 00:00:00          0        4000(
   [3] 2779  999010007 2007.07.05 11:45:00 1970.01.01 00:00:00          0        4000(
   [4] 2780  999010007 2007.08.02 11:45:00 1970.01.01 00:00:00          0        4000(
   [5] 2781  999010007 2007.09.06 11:45:00 1970.01.01 00:00:00          0        4000(
   [6] 2782  999010007 2007.10.04 11:45:00 1970.01.01 00:00:00          0        4000(
   [7] 2783  999010007 2007.11.08 12:45:00 1970.01.01 00:00:00          0        4000(
   [8] 2784  999010007 2007.12.06 12:45:00 1970.01.01 00:00:00          0        4000(
   [9] 2785  999010007 2008.01.10 12:45:00 1970.01.01 00:00:00          0        4000(
*/
```

Siehe auch

CalendarCountries, CalendarEventByCountry, CalendarValueHistory, CalendarEventById, CalendarValueById

© 2000-2025, MetaQuotes Ltd.

CalendarValueHistory

Abrufen des Arrays der Werte für alle Ereignisse in einem bestimmten Zeitbereich mit der Möglichkeit, nach Land und/oder Währung zu sortieren.

```
bool  CalendarValueHistory(
   MqlCalendarValue&  values[],              // Array der Wertbeschreibungen
   datetime           datetime_from,         // Anfang des Zeitraums
   datetime           datetime_to=0          // Ende des Zeitraums
   const string       country_code=NULL,     // Länder-Code (ISO 3166-1 alpha-2)
   const string       currency=NULL          // Währungs-Code des Landes
   );
```

Parameter

values[]

[out] Variable vom Typ MqlCalendarValue für die Übernahme der Ereigniswerte. Beispiele der Behandlung von Kalenderereignisse.

datetime_from

[in] Anfangszeitpunkt des Zeitraums der Ereignisse für die angegebene ID, während *datetime_from* < *datetime_to*.

datetime_to=0

[in] Endzeitpunkt des Zeitraums der Ereignisse für die angegebene ID. Wenn *datetime_to* nicht (oder auf 0) gesetzt wurde, werden alle Ereignisse ab dem angegebenen Zeitpunkt *datetime_from* aus der Kalenderdatenbank (inklusive der zukünftigen) zurückgegeben.

country_code=NULL

[in] Code-Name des Landes (ISO 3166-1 alpha-2)

currency=NULL

[in] Währungs-Code des Landes.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 4004 - ERR_NOT_ENOUGH_MEMORY (nicht genügend Speicherplatz für die Ausführung der Anforderung),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- 5400 - ERR_CALENDAR_MORE_DATA (Array ist zu klein, um die Beschreibungen aller Werte zu übernehmen, es wurden nur diejenigen übernommen, die eingetragen werden konnten).
- Fehler bei der Ausführung von ArrayResize()

Hinweis

Alle Funktionen, die mit dem Wirtschaftskalender arbeiten, nutzen die Zeit des Handelsservers (TimeTradeServer). Das bedeutet, dass die Zeit in der Struktur MqlCalendarValue und die

© 2000-2025, MetaQuotes Ltd.

Zeitangaben der Funktionen CalendarValueHistoryByEvent/CalendarValueHistory in der Zeitzone des Handelsservers eingestellt sind und nicht in der lokalen Zeitzone des Nutzers.

Wenn der Array *events[]* mit einer unveränderlichen Größe der Funktion übergeben wurde und die Größe für alle Ereignisse zu klein ist, wird der Fehler ERR_CALENDAR_MORE_DATA (5400) ausgeworfen.

Wenn *datetime_to* nicht (oder auf 0) gesetzt wurde, werden alle Ereignisse ab dem angegebenen Zeitpunkt *datetime_from* aus der Kalenderdatenbank (inklusive der zukünftigen) zurückgegeben.

Zu dem Länder-Code *country_code* und der *Währung* : NULL und "" sind äquivalent und bedeuten, es wird nichts heraus gefiltert.

Für *country_code*, dem Feld des *Codes* in der Struktur MqlCalendarCountry, sollten zum Beispiel "US", "RU" oder "EU" verwendet werden.

Für *currency*, dem Feld der *Währungen* in der Struktur MqlCalendarCountry, sollten zum Beispiel "USD", "RUB" oder "EUR" verwendet werden.

Die Filter werden über ein logisches 'UND' verbunden und werden verwendet, um nur die Werte auszuwählen, für die zugleich beide Bedingungen (Land und Währung) zutreffen.

Die Struktur von MqlCalendarValue bietet Methoden zum Prüfen und Setzen von Werten der Felder actual_value, forecast_value, prev_value und revised_prev_value. Wenn kein Wert angegeben wird, speichert das Feld LONG_MIN (-9223372036854775808).

Bitte beachten Sie, dass die in diesem Feld gespeicherten Werte mit einer Million multipliziert werden. Dies bedeutet, dass, wenn Sie Werte von MqlCalendarValue mithilfe der Funktionen CalendarValueById, CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent und CalendarValueLast abrufen, sollten Sie überprüfen, ob die Feldwerte gleich LONG_MIN sind; wenn in einem Feld ein Wert angegeben ist, sollten Sie den Wert durch 1.000.000 dividieren, um den Wert zu erhalten. Eine andere Methode zum Abrufen der Werte besteht darin, die Werte mit den Funktionen der Struktur MqlCalendarValue zu überprüfen und abzurufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Länder-Code der EU (ISO 3166-1 Alpha-2)
   string EU_code="EU";
//--- Abrufen aller Ereigniswerte der EU
   MqlCalendarValue values[];
//--- Festlegen der Zeitgrenzen des Intervalls für die Ereignisse
   datetime date_from=D'01.01.2018';  // alle Ereignisse ab 2018
   datetime date_to=0;                // 0 bedeutet, alle bekannten Ereignisse, einsch
//--- Abrufen der historischen Ereignisse seit dem Jahr 2018 für die EU
   if(CalendarValueHistory(values,date_from,date_to,EU_code))
     {
      PrintFormat("Erhaltene Ereigniswerte für country_code=%s: %d",
                  EU_code,ArraySize(values));
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Verkleinern der Arraygröße für die Ausgabe im Journal
      ArrayResize(values,10);
//--- Anzeigen der Ereigniswerte im Journal
      ArrayPrint(values);
     }
   else
     {
      PrintFormat("Fehler! Fehler beim Erhalt der Ereignisse für country_code=%s",EU_c
      PrintFormat("Fehlernummer: %d",GetLastError());
     }
//---
  }
/*
  Ergebnis:
  Erhaltene Ereigniswerte für country_code=EU: 1384
        [id] [event_id]           [time]               [period] [revision]   [actual_v
   [0] 54215  999500001 2018.01.02 09:00:00 2017.12.01 00:00:00          3       60600
   [1] 54221  999500002 2018.01.04 09:00:00 2017.12.01 00:00:00          3       56600
   [2] 54222  999500003 2018.01.04 09:00:00 2017.12.01 00:00:00          3       58100
   [3] 45123  999030005 2018.01.05 10:00:00 2017.11.01 00:00:00          0         600
   [4] 45124  999030006 2018.01.05 10:00:00 2017.11.01 00:00:00          0        2800
   [5] 45125  999030012 2018.01.05 10:00:00 2017.12.01 00:00:00          1         900
   [6] 45126  999030013 2018.01.05 10:00:00 2017.12.01 00:00:00          1        1400
   [7] 54953  999520001 2018.01.05 20:30:00 2018.01.02 00:00:00          0      127900
   [8] 22230  999040003 2018.01.08 10:00:00 2017.12.01 00:00:00          0        9100
   [9] 22231  999040004 2018.01.08 10:00:00 2017.12.01 00:00:00          0       18400
*/
```

Siehe auch

CalendarCountries, CalendarEventByCountry, CalendarValueHistoryByEvent, CalendarEventById, CalendarValueById

© 2000-2025, MetaQuotes Ltd.

CalendarValueLastByEvent

Abrufen des Arrays der Ereigniswerte nach seiner ID seit dem Status der Kalenderdatenbank nach einer angegebenen change_id.

```
int  CalendarValueLastByEvent(
   ulong                event_id,      // Ereignis-ID
   ulong&               change_id,     // Änderungs-ID der Kalenderdatenbank
   MqlCalendarValue&    values[]       // Array der Wertbeschreibungen
   );
```

Parameter

event_id

[in] Ereignis-ID.

change_id

[in][out] Änderungs-ID der Kalenderdatenbank.

values[]

[out] Variable vom Typ MqlCalendarValue für die Übernahme der Ereigniswerte. Beispiele der Behandlung von Kalenderereignisse.

Rückgabewert

Anzahl der erhaltenen Ereigniswerte. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 4004 - ERR_NOT_ENOUGH_MEMORY (nicht genügend Speicherplatz für die Ausführung der Anforderung),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- 5400 - ERR_CALENDAR_MORE_DATA (Array ist zu klein, um die Beschreibungen aller Werte zu übernehmen, es wurden nur diejenigen übernommen, die eingetragen werden konnten).
- Fehler bei der Ausführung von ArrayResize()

Hinweis

Alle Funktionen, die mit dem Wirtschaftskalender arbeiten, nutzen die Zeit des Handelsservers (TimeTradeServer). Das bedeutet, dass die Zeit in der Struktur MqlCalendarValue und die Zeitangaben der Funktionen CalendarValueHistoryByEvent/CalendarValueHistory in der Zeitzone des Handelsservers eingestellt sind und nicht in der lokalen Zeitzone des Nutzers.

Wenn der Array *events[]* mit einer unveränderlichen Größe der Funktion übergeben wurde und die Größe für alle Ereignisse zu klein ist, wird der Fehler ERR_CALENDAR_MORE_DATA (5400) ausgeworfen.

Wenn *change_id* = 0 der Funktion übergeben wird, wird der Variablen das aktuelle *change_id* der Kalenderdatenbank diesem Parameter zugewiesen; und die Funktion gibt 0 (Anzahl der gefundenen Nachrichten) zurück.

Die Funktion gibt das Array der verlangten Nachrichten und ein neues *change_id* zurück, das für nachfolgende Aufrufe der Funktion verwendet werden, um neuen Werte dieser Nachricht abzufragen.

© 2000-2025, MetaQuotes Ltd.

Damit ist es möglich die Werte bestimmter Nachrichten zu aktualisieren, in dem einfach diese Funktion mit der letztbekannten *change_id* aufgerufen wird.

Die Struktur von MqlCalendarValue bietet Methoden zum Prüfen und Setzen von Werten der Felder actual_value, forecast_value, prev_value und revised_prev_value. Wenn kein Wert angegeben wird, speichert das Feld LONG_MIN (-9223372036854775808).

Bitte beachten Sie, dass die in diesem Feld gespeicherten Werte mit einer Million multipliziert werden. Dies bedeutet, dass, wenn Sie Werte von MqlCalendarValue mithilfe der Funktionen CalendarValueById, CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent und CalendarValueLast abrufen, sollten Sie überprüfen, ob die Feldwerte gleich LONG_MIN sind; wenn in einem Feld ein Wert angegeben ist, sollten Sie den Wert durch 1.000.000 dividieren, um den Wert zu erhalten. Eine andere Methode zum Abrufen der Werte besteht darin, die Werte mit den Funktionen der Struktur MqlCalendarValue zu überprüfen und abzurufen.

Ein Beispiel-EA, der auf die Pressemeldung des US-Arbeitsmarkts (Nonfarm Payrolls) wartet:

```
#property description "Beispielsweise Verwendung der Funktion CalendarValueLastByEvent
#property description ", mit der auf die Pressemeldung des US-Arbeitsmarkts (Nonfarm P
#property description "Dafür holt er sich die aktuelle Änderungs-ID"
#property description " der Kalenderdatenbank. Dann wird mit dieser ID"
#property description " und der Verwendung des Timers nur auf neue Ereignisse gewartet
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Timer erstellen
   EventSetTimer(60);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Timer löschen
   EventKillTimer();
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
//| Timer Funktion                                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnTimer()
  {
//--- Änderungs-ID der Kalenderdatenbank
   static ulong calendar_change_id=0;
//--- Attribute beim ersten Start
   static bool first=true;
//--- Ereignis-ID
   static ulong event_id=0;
//--- Ereignisname
   static string event_name=NULL;
//--- Array der Ereigniswerte
   MqlCalendarValue values[];
//--- die Initialisierung - Abrufen der aktuellen calendar_change_id
   if(first)
     {
      MqlCalendarEvent events[];
      //--- Länder-Code für die USA (ISO 3166-1 Alpha-2)
      string USA_code="US";
      //--- Abrufen der Ereignisse für die USA
      int events_count=CalendarEventByCountry(USA_code,events);
      //--- Position des gesuchten Ereignisses im Array 'events'
      int event_pos=-1;
      //--- Anzeige der Ereignisse für die USA im Journal
      if(events_count>0)
        {
         PrintFormat("%s: USA -Ereignisse: %d",__FUNCTION__,events_count);
         for(int i=0;i<events_count;i++)
           {
            string event_name_low=events[i].name;
            //--- Ändern des Ereignisnamens in Kleinbuchstaben
            if(!StringToLower(event_name_low))
              {
               PrintFormat("StringToLower() erzeugte den Fehler %d",GetLastError());
               //--- Funktion vorzeitig beendet
               return;
              }
            //--- Suche nach dem Ereignis "Nonfarm Payrolls"
            if(StringFind(event_name_low,"nonfarm payrolls")!=-1)
              {
               //--- Ereignis gefunden, sichern der ID
               event_id=events[i].id;
               //--- Schreiben des Ereignisnamens von "Nonfarm Payrolls"
               event_name=events[i].name;
               //--- Sichern der Position des Ereignisses im Array 'events[]'
               event_pos=i;
               //--- Vergessen Sie nicht, dass der Kalender mehrere Ereignisse mit "nc
               PrintFormat("Event \"Nonfarm Payrolls\" found: event_id=%d  event_name=
               //--- Anzeigen aller Ereignisse durch das Auskommentieren von 'break' i
               break;
```

© 2000-2025, MetaQuotes Ltd.

```
            }
          }
        //--- Reduzieren der Liste durch das Löschen aller Ereignisse nach "Nonfarm P
        ArrayRemove(events,event_pos+1);
        //--- Es bleiben 9 Ereignisse vor "Nonfarm Payrolls" für eine leichtere Analy
        ArrayRemove(events,0,event_pos-9);
        ArrayPrint(events);
       }
     else
       {
        PrintFormat("%s: CalendarEventByCountry(%s) lieferte 0 Ereignisse, Fehlernumm
                    USA_code,__FUNCTION__,GetLastError());
        //--- Aufgabe mit einem Fehler beendet, neuer Versuch beim nächsten Aufruf de
        return;
       }

     //--- Abrufen der Änderungs-ID der Kalenderdatenbank für das angegebene Ereignis
     if(CalendarValueLastByEvent(event_id,calendar_change_id,values)>0)
       {
        //--- dieser Teil des Codes wird nicht beim Erststart ausgeführt, aber er sei
        PrintFormat("%s: Abrufen der aktuellen Änderungs-ID der Kalenderdatenbank: ch
                    __FUNCTION__,calendar_change_id);
        //--- Setzen des Flags und vor dem nächsten Ereignis des Timers zurückkehren
        first=false;
        return;
       }
     else
       {
        //--- es wurden keine Daten geliefert (das ist beim Erststart normal), ergo a
        int error_code=GetLastError();
        if(error_code==0)
          {
           PrintFormat("%s: Abrufen der aktuellen Änderungs-ID der Kalenderdatenbank:
                       __FUNCTION__,calendar_change_id);
           //--- Setzen des Flags und vor dem nächsten Ereignis des Timers zurückkehr
           first=false;
           //--- jetzt haben wir den Wert für calendar_change_id
           return;
          }
        else
          {
           //--- und das ist jetzt tatsächlich ein Fehler
           PrintFormat("%s: Fehler bei Erhalt der Werte für event_id=%d",__FUNCTION__
           PrintFormat("Fehlernummer: %d",error_code);
           //--- Aufgabe mit einem Fehler beendet, neuer Versuch beim nächsten Aufrut
           return;
          }
       }
    }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wir haben den letztbekannten Wert der Änderungs-ID der Kalenderdatenbank (change
   ulong old_change_id=calendar_change_id;
//--- Prüfen auf einen neuen Ereigniswert für "Nonfarm Payrolls"
   if(CalendarValueLastByEvent(event_id,calendar_change_id,values)>0)
     {
      PrintFormat("%s: Erhalten neue Ereignisse für \"%s\": %d",
                  __FUNCTION__,event_name,ArraySize(values));
      //--- Anzeigen der Daten des Arrays 'values' im Journal
      ArrayPrint(values);
      //--- Anzeigen der Werte der vorherigen und neuen Änderungs-IDs im Journal
      PrintFormat("%s: Vorheriges change_id=%d, neues change_id=%d",
                  __FUNCTION__,old_change_id,calendar_change_id);
/*
      Schreiben Sie hier Ihren Code, was beim Erscheinen von "Nonfarm Payrolls" gesche
      */
     }
//---
  }
/*
  Ergebnis:
   OnTimer: USA Ereignisse: 202
  Event "Nonfarm Payrolls" found: event_id=840030016  event_name=Nonfarm Payrolls
            [id] [type] [sector] [frequency] [time_mode] [country_id] [unit] [importar
   [0] 840030007      1        4           2           0          840      1
   [1] 840030008      1        4           2           0          840      1
   [2] 840030009      1        4           2           0          840      0
   [3] 840030010      1        4           2           0          840      0
   [4] 840030011      1        4           2           0          840      1
   [5] 840030012      1        4           2           0          840      1
   [6] 840030013      1        4           2           0          840      1
   [7] 840030014      1        4           2           0          840      1
   [8] 840030015      1        3           2           0          840      1
   [9] 840030016      1        3           2           0          840      4
   OnTimer: Erhalten wurde die aktuelle ID der Kalenderdatenbank: change_id=33986560

*/
```

Siehe auch

CalendarValueLast, CalendarValueHistory, CalendarValueHistoryByEvent, CalendarValueById

© 2000-2025, MetaQuotes Ltd.

CalendarValueLast

Abrufen des Arrays der Werte für alle Ereignisse mit der Möglichkeit, nach Land und/oder Währung ab dem Status der Kalenderdatenbank mit der angegebenen change_id zu sortieren.

```
int  CalendarValueLast(
   ulong&               change_id,             // Änderungs-ID der Kalenderdatenbank
   MqlCalendarValue&    values[],              // Array der Wertbeschreibungen
   const string         country_code=NULL,     // Code-Name des Landes (ISO 3166-1 alp
   const string         currency=NULL          // Währungs-Code des Landes
   );
```

Parameter

change_id

[in][out] Änderungs-ID der Kalenderdatenbank.

values[]

[out] Variable vom Typ MqlCalendarValue für die Übernahme der Ereigniswerte. Beispiele der Behandlung von Kalenderereignisse.

country_code=NULL

[in] Code-Name des Landes (ISO 3166-1 alpha-2)

currency=NULL

[in] Währungs-Code des Landes.

Rückgabewert

Anzahl der erhaltenen Ereigniswerte. Um Informationen über einen Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf. Mögliche Fehler:

- 4001 - ERR_INTERNAL_ERROR (allgemeiner Laufzeitfehler),
- 4004 - ERR_NOT_ENOUGH_MEMORY (nicht genügend Speicherplatz für die Ausführung der Anforderung),
- 5401 - ERR_CALENDAR_TIMEOUT (Zeitlimit für Anfragen überschritten),
- 5400 - ERR_CALENDAR_MORE_DATA (Array ist zu klein, um die Beschreibungen aller Werte zu übernehmen, es wurden nur diejenigen übernommen, die eingetragen werden konnten).
- Fehler bei der Ausführung von ArrayResize()

Hinweis

Alle Funktionen, die mit dem Wirtschaftskalender arbeiten, nutzen die Zeit des Handelsservers (TimeTradeServer). Das bedeutet, dass die Zeit in der Struktur MqlCalendarValue und die Zeitangaben der Funktionen CalendarValueHistoryByEvent/CalendarValueHistory in der Zeitzone des Handelsservers eingestellt sind und nicht in der lokalen Zeitzone des Nutzers.

Wenn der Array *events[]* mit einer unveränderlichen Größe der Funktion übergeben wurde und die Größe für alle Ereignisse zu klein ist, wird der Fehler ERR_CALENDAR_MORE_DATA (5400) ausgeworfen.

Wenn *change_id* = 0 der Funktion übergeben wird, wird *change_id* der Kalenderdatenbank diesem Parameter zugewiesen; und die Funktion gibt 0 zurück.

© 2000-2025, MetaQuotes Ltd.

Zu dem Länder-Code *country_code* und der *Währung* : NULL und "" sind äquivalent und bedeuten, es wird nichts heraus gefiltert.

Für *country_code*, dem Feld des *Codes* in der Struktur MqlCalendarCountry, sollten zum Beispiel "US", "RU" oder "EU" verwendet werden.

Für *currency*, dem Feld der *Währungen* in der Struktur MqlCalendarCountry, sollten zum Beispiel "USD", "RUB" oder "EUR" verwendet werden.

Die Filter werden über ein logisches 'UND' verbunden und werden verwendet, um nur die Werte auszuwählen, für die zugleich beide Bedingungen (Land und Währung) zutreffen.

Die Funktion gibt das Array der verlangten Nachrichten und ein neues *change_id* zurück, das für nachfolgende Aufrufe der Funktion verwendet werden, um neuen Werte dieser Nachricht abzufragen. Damit ist es möglich die Werte bestimmter Nachrichten zu aktualisieren, in dem einfach diese Funktion mit der letztbekannten *change_id* aufgerufen wird.

Die Struktur von MqlCalendarValue bietet Methoden zum Prüfen und Setzen von Werten der Felder actual_value, forecast_value, prev_value und revised_prev_value. Wenn kein Wert angegeben wird, speichert das Feld LONG_MIN (-9223372036854775808).

Bitte beachten Sie, dass die in diesem Feld gespeicherten Werte mit einer Million multipliziert werden. Dies bedeutet, dass, wenn Sie Werte von MqlCalendarValue mithilfe der Funktionen CalendarValueById, CalendarValueHistoryByEvent, CalendarValueHistory, CalendarValueLastByEvent und CalendarValueLast abrufen, sollten Sie überprüfen, ob die Feldwerte gleich LONG_MIN sind; wenn in einem Feld ein Wert angegeben ist, sollten Sie den Wert durch 1.000.000 dividieren, um den Wert zu erhalten. Eine andere Methode zum Abrufen der Werte besteht darin, die Werte mit den Funktionen der Struktur MqlCalendarValue zu überprüfen und abzurufen.

Hier das Beispiel eines EAs, der auf Ereignisse des Wirtschaftskalenders wartet:

```
#property description "Die beispielsweise Verwendung der Funktion CalendarValueLast"
#property description " für die Entwicklung eines EAs, der auf die Ereignisse des Wirt
#property description "Dafür holt er sich die aktuelle Änderungs-ID"
#property description " der Kalenderdatenbank. Dann wird mit dieser ID"
#property description " und der Verwendung des Timers nur auf neue Ereignisse gewartet
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Timer erstellen
   EventSetTimer(60);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Timer löschen
   EventKillTimer();
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
//| Timer Funktion                                                   |
//+------------------------------------------------------------------+
void OnTimer()
  {
//--- Änderungs-ID der Kalenderdatenbank
   static ulong calendar_change_id=0;
//--- Attribute beim ersten Start
   static bool first=true;
//--- Array der Ereigniswerte
   MqlCalendarValue values[];
//--- die Initialisierung - Abrufen der aktuellen calendar_change_id
   if(first)
     {
      //--- Abrufen der Änderungs-ID der Kalenderdatenbank
      if(CalendarValueLast(calendar_change_id,values)>0)
        {
         //--- dieser Teil des Codes wird nicht beim Erststart ausgeführt, aber er sei
         PrintFormat("%s: Abrufen der aktuellen Änderungs-ID der Kalenderdatenbank: ch
                     __FUNCTION__,calendar_change_id);
         //--- Setzen des Flags und vor dem nächsten Ereignis des Timers zurückkehren
         first=false;
         return;
        }
      else
        {
         //--- es wurden keine Daten geliefert (das ist beim Erststart normal), ergo a
         int error_code=GetLastError();
         if(error_code==0)
           {
            PrintFormat("%s: Abrufen der aktuellen Änderungs-ID der Kalenderdatenbank:
                        __FUNCTION__,calendar_change_id);
            //--- Setzen des Flags und vor dem nächsten Ereignis des Timers zurückkehr
            first=false;
            //--- jetzt haben wir den Wert für calendar_change_id
            return;
           }
         else
           {
            //--- und das ist jetzt tatsächlich ein Fehler
```

© 2000-2025, MetaQuotes Ltd.

```
            PrintFormat("%s: Fehler beim Abrufen des Ereignisses in CalendarValueLast.
                        __FUNCTION__,error_code);
            //--- Aufgabe mit einem Fehler beendet, Reinitialisieren beim nächsten Au
            return;
           }
        }
     }

//--- wir haben den letztbekannten Wert der Änderungs-ID der Kalenderdatenbank (change
   ulong old_change_id=calendar_change_id;
//--- Prüfen, ob es neue Kalendererereignisse gibt
   if(CalendarValueLast(calendar_change_id,values)>0)
     {
      PrintFormat("%s: Erhaltene neue Kalendererereignisse: %d",
                  __FUNCTION__,ArraySize(values));
      //--- Anzeigen der Daten des Arrays 'values' im Journal
      ArrayPrint(values);
      //--- Anzeigen der Werte der vorherigen und neuen Änderungs-IDs im Journal
      PrintFormat("%s: Vorheriges change_id=%d, neues change_id=%d",
                  __FUNCTION__,old_change_id,calendar_change_id);
      //--- Anzeige der neuen Ereignisse im Journal
      ArrayPrint(values);
      /*
     Schreiben Sie hier Ihren Code, was beim Erscheinen von Ereignissen geschehen soll
      */
     }
//---
  }
/*
  Beispiel für das Warten:
  OnTimer: Erhalten wurde die aktuelle ID der Kalenderdatenbank: change_id=33281792
  OnTimer: Erhalten wurden neue Ereignisse vom Kalender: 1
        [id] [event_id]              [time]            [period] [revision] [actual_val
   [0] 91040   76020013 2019.03.20 15:30:00 1970.01.01 00:00:00          0       -507
  OnTimer: Vorheriges change_id=33281792, neues change_id=33282048
        [id] [event_id]              [time]            [period] [revision] [actual_val
   [0] 91040   76020013 2019.03.20 15:30:00 1970.01.01 00:00:00          0       -507
  OnTimer: Erhalten wurden neue Ereignisse vom Kalender: 1
        [id] [event_id]              [time]            [period] [revision]       [actu
   [0] 91041   76020013 2019.03.27 15:30:00 1970.01.01 00:00:00          0 -922337203
  OnTimer: Vorheriges change_id=33282048, neues change_id=33282560
        [id] [event_id]              [time]            [period] [revision]       [actu
   [0] 91041   76020013 2019.03.27 15:30:00 1970.01.01 00:00:00          0 -922337203

*/
```

Siehe auch

© 2000-2025, MetaQuotes Ltd.

CalendarValueLast, CalendarValueHistory, CalendarValueHistoryByEvent, CalendarValueById

© 2000-2025, MetaQuotes Ltd.

Zugang zu Zeitreihen und Daten der Indikatoren

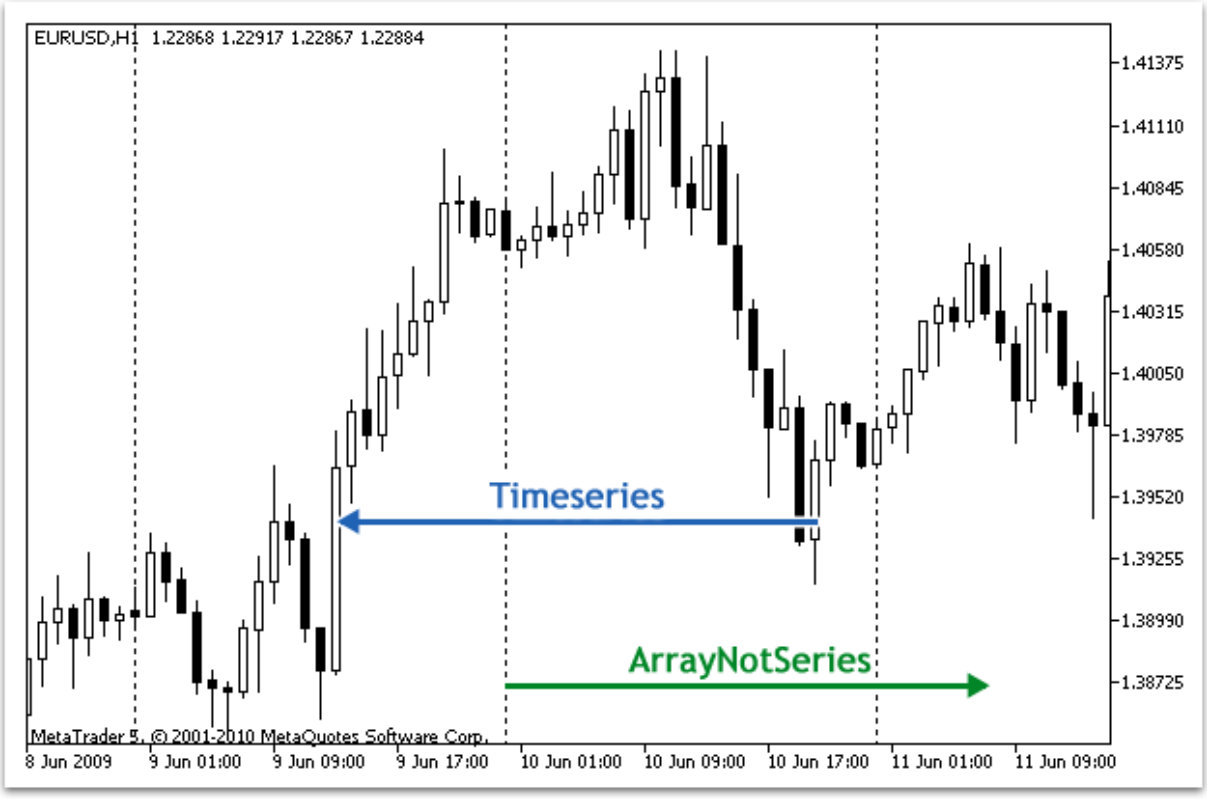
Funktionen für Arbeit mit Zeitreihen und Indikatoren. Zeitreihe unterscheidet sich vom Feld dadurch, dass Indizieren der Elemente einer Zeitreihe vom Ende des Feldes zum Anfang des Feldes durchgeführt wird (von den neuesten Daten zu den aeltesten). Für Kopieren der Werte der Zeitreihe ist es empfehlenswert nur dynamische Felder zu verwenden, denn die Kopierfunktionen verteilen die erforderliche Größe der Felder, die die Werte empfangen.

Es gibt eine **wichtige Ausnahme** von dieser Regel: wenn Kopieren der Zeitreihen und Indikatorwerte oft durchgeführt werden muss, ZB bei jedem Aufruf OnTick() in Experten oder bei jedem Aufruf OnCalculate() in Indikatoren, ist es besser statisch verteilte Felder zu verwenden, denn **Operationen der Speicherverteilung** für dynamische Felder erfordert **zusaetzliche Zeit** und das wird beim Testen oder Optimierung der Experte bedeutend sein.

Bei der Verwendung der Funktionen des Zugangs zu Zeitreihen und Werten des Indikators muss man Richtung des Indizierens beachten, das wird ausführlich im Abschnit Richtung des Indizierens in Feldern und Zeitreihen beschrieben.

Zugang zu Werten der Indikatoren und Zeitreihen wird unabhängig davon durchgeführt, ob die angeforderten Daten fertig sind (der sogenannte asynchroner Zugang). Das ist kritisch bedeutsam für Berechnung der Benutzerindikatoren, darum geben die Funktionen des Typs *Copy...()* beim Fehlen der angeforderten Daten sofort den Fehler zurück. Aber beim Zugang von Experten und Scripts versucht man mehrmals Daten zu erhalten und dabei mit einer Pause, die die Zeit für Ladung fehlender Zeitreihen oder für Berechnung der Indikatorwerte bereitzustellen is.

Im Abschnitt Datenzugang organisieren beschreibt man die Erhaltung, Aufbewahren und Anforderung der Preisdaten im Client-Terminal MetaTrader 5.



© 2000-2025, MetaQuotes Ltd.

Es ist historisch bedingt, dass Zugang zu Daten im Preisfeld vom Ende der Daten durchgeführt wurde. Physikalisch werden neue Daten an Ende des Feldes geschrieben, aber Index dieses Feldes ist immer 0. Index 0 im Feld-Zeitreihe bedeutet Daten der laufenden Bar, d. h. der Bar, die dem unbeendeten Zeitinterval auf dem Timeframe entspricht.

Timeframe ist Zeitperiode, während der die einzelne Preisbar gebildet wird; insgesamt werden 21 Standardtimeframes vorbestimmt.

Funktion	Massnahme
SeriesInfoInteger	Gibt die Information über den Zustand der historischen Daten zurück
Bars	Gibt die die entsprechende PeriodeAnzahl der Bars in der Geschichte für das entsprechende Symbol und die entsprechende Periode
BarsCalculated	Gibt die Anzahl der berechneten Daten im Anzeigerpuffer oder -1 beim Fehler (Daten sind noch nicht berechnet)
IndicatorCreate	Gibt handle des angegebenen technischen Anzeigers zurück, das vom Feld des Typs Parameter MqlParam erzeugt wurde
IndicatorParameters	Basierend auf das angegebenen Handle gibt Anzahl von Eingabeparametern des Indikators, sowie die Werte und Type der Parameter zurück
IndicatorRelease	Entfernt handle des Anzeigers und befreit den Berechnungsteol des Anzeigers, wenn niemand den verwendet
CopyBuffer	Bekommt Daten des angegebenen Puffers vom angegebenen Anzeiger im Feld
CopyRates	Bekommt historische Daten der Struktur Rates für das angegebene Symbol und Periode
CopySeries	Ruft die synchronisierten Zeitreihen ab aus der Struktur MqlRates für den angegebene Zeitrahmen des Symbols und der angegebenen Anzahl
CopyTime	Bekommt im Feld historische Daten über die Eroeffnungszeit der Bars für das angegebene Symbol und Periode
CopyOpen	Bekommt im Feld historische Daten über die Eroeffnungszeit der Bars für das angegebene Symbol und Periode
CopyHigh	Bekommt im Feld historische Daten über den maximalen Preis der Bars für das angegebene Symbol und Periode
CopyLow	Bekommt im Feld historische Daten über den minimalen Preis der Bars für das angegebene Symbol und Periode
CopyClose	Bekommt im Feld historische Daten über den Schlusspreis der Bars für das angegebene Symbol und Periode
CopyTickVolume	Bekommt im Feld historische Daten über Tickvolumen der Bars für das angegebene Symbol und Periode
CopyRealVolume	Bekommt im Feld historische Daten über Handelsvolumen der Bars für das angegebene Symbol und Periode

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
CopySpread	Bekommt im Feld historische Daten über Spreads für das angegebene Symbol und Periode
CopyTicks	Erhält für die aktuelle Arbeitssitzung angesammelten Ticks ins Array ticks_array
CopyTicksRange	Die Funktion erhält Ticks vom Typ MqlTick aus einem bestimmten Zeitbereich ins Array ticks_array
iBars	Gibt die Anzahl der Balken in der Historie nach dem entsprechenden Symbol und Zeitrahmen zurück
iBarShift	Gibt den Index des Balkens zurück, der der angegebenen Zeit entspricht
iClose	Gibt den Close-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iHigh	Gibt den High-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iHighest	Gibt den Index des größten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts zurück
iLow	Gibt den Low-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iLowest	Gibt den Index des kleinsten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts zurück
iOpen	Gibt den Open-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iTime	Gibt den Zeitpunkt der Eröffnung eines Balkens (der mithilfe des Parameters shift vorgegeben wurde) des entsprechenden Charts zurück
iTickVolume	Gibt den Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iRealVolume	Gibt den Wert des echten Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iVolume	Gibt den Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück
iSpread	Gibt den Wert des Spreads des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück

Abgesehen davon, dass durch die Funktion ArraySetAsSeries() der Zugang zu Elementen in Feldern wie in Zeitreihen vorgegeben werden kann, muss man sich daran erinnern, dass Elemente des Feldes in derselben Reihenfolge aufbewahren werden, verändert wird nur Richtung des Indizierens. Diese Tatsache kann am folgenden Beispiel demonstriert werden:

```
   datetime TimeAsSeries[];
//--- stellen wir Zugang zum Feld wie zur Timeserie ein
   ArraySetAsSeries(TimeAsSeries,true);
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
   int copied=CopyTime(NULL,0,0,10,TimeAsSeries);
   if(copied<=0)
     {
      Print("Nicht gelungen, die Eroeffnungszeit für die letzten 10 Bars zu kopieren")
      return;
     }
   Print("TimeCurrent = ",TimeCurrent());
   Print("ArraySize(Time) = ",ArraySize(TimeAsSeries));
   int size=ArraySize(TimeAsSeries);
   for(int i=0;i<size;i++)
     {
      Print("TimeAsSeries["+i+"] = ",TimeAsSeries[i]);
     }

   datetime ArrayNotSeries[];
   ArraySetAsSeries(ArrayNotSeries,false);
   ResetLastError();
   copied=CopyTime(NULL,0,0,10,ArrayNotSeries);
   if(copied<=0)
     {
      Print("Nicht gelungen, die Eroeffnungszeit für die letzten 10 Bars zu kopieren")
      return;
     }
   size=ArraySize(ArrayNotSeries);
   for(int i=size-1;i>=0;i--)
     {
      Print("ArrayNotSeries["+i+"] = ",ArrayNotSeries[i]);
     }
```

Im Ergebnis haben wir die Ausgabe, wie diese:

```
TimeCurrent = 2009.06.11 14:16:23
ArraySize(Time) = 10
TimeAsSeries[0] = 2009.06.11 14:00:00
TimeAsSeries[1] = 2009.06.11 13:00:00
TimeAsSeries[2] = 2009.06.11 12:00:00
TimeAsSeries[3] = 2009.06.11 11:00:00
TimeAsSeries[4] = 2009.06.11 10:00:00
TimeAsSeries[5] = 2009.06.11 09:00:00
TimeAsSeries[6] = 2009.06.11 08:00:00
TimeAsSeries[7] = 2009.06.11 07:00:00
TimeAsSeries[8] = 2009.06.11 06:00:00
TimeAsSeries[9] = 2009.06.11 05:00:00

ArrayNotSeries[9] = 2009.06.11 14:00:00
ArrayNotSeries[8] = 2009.06.11 13:00:00
ArrayNotSeries[7] = 2009.06.11 12:00:00
ArrayNotSeries[6] = 2009.06.11 11:00:00
```

© 2000-2025, MetaQuotes Ltd.

```
ArrayNotSeries[5] = 2009.06.11 10:00:00
ArrayNotSeries[4] = 2009.06.11 09:00:00
ArrayNotSeries[3] = 2009.06.11 08:00:00
ArrayNotSeries[2] = 2009.06.11 07:00:00
ArrayNotSeries[1] = 2009.06.11 06:00:00
ArrayNotSeries[0] = 2009.06.11 05:00:00
```

Wie es aus der Ausgabe ersichtlich ist, vermindert sich der Wert der Zeit für das Feld TimeAsSeries, d.h. wir bewegen uns von der Gegenwart zur Vergangenheit. Für das normale Feld ArrayNotSeries ist es umgekehrt - mit dem Anstieg des Indexes bewegen wir uns von der Vergangenheit zur Gegenwart.

Sehen Sie auch

ArrayIsDynamic, ArrayGetAsSeries, ArraySetAsSeries, ArrayIsSeries

© 2000-2025, MetaQuotes Ltd.

Richtung des Indizierens in Feldern, Puffern und Zeitreihen

Alle Felder und Indikatorpuffer haben als Vorgabe die Richtung des Indizierens von links nach rechts. Index des ersten Elementes ist immer der Null gleich. So befindet sich das erste Element eines Feldes oder eines Indikatorpuffers mit Index 0 als Vorgabe an der linken Grenzstelle, das letzte Element befindet sich an der rechten Grenzstelle.

Indikatorpuffer ist ein dynamisches Feld des Typs double, dessen Größe vom Client-Terminal verwaltet wird, damit er der Anzahl von Bars entspricht, die für Berechnung des Indikators verwendet werden. Das normale dynamische Feld des Typs double wird als Indikatorpuffer durch die Funktion SetIndexBuffer() vorgegeben. Für Indikatorpuffer braucht man nicht die Größe durch die Funktion ArrayResize() vorgeben, das ausführende System des Terminals wird das selbsstaendig machen.

Zeitreihen sind Felder mit dem umgekehrten Indizieren, d.h. das erste Element einer Zeitreihe befindet sich an der linken Grenzstelle, das letzte Element befindet sich an der rechten Grenzstelle. Da Zeitreihen dafür bestimmt sind, historische Preisdaten für Finanzinstrumente aufzubewahren und unbedingt zeitbezogene Information enthalten, kann man sagen, dass sich die neuesten Daten in der Zeitreihe an der rechten Grenzstelle und die aeltesten an der linken Grenzstelle befinden.

Darum hat Element mit Index 0 in der Zeitreihe Information über die letzte Quotation für Instrument. Wenn eine Zeitreihe Daten für für taegliches Timeframe darstellt, werden in der Null-Position Daten des laufenden nicht beendeten Tages geben, und in der Position mit Index 1 werden Daten von gestern aufbewahren.

Veränderung der Richtung des Indizierens

Funktion ArraySetAsSeries() erlaubt Zugangsverfahren zu Elementen eines dynamischen Feldes zu verändern, dabei wird die Ordnung der Datenaufbewahrung im Computerspeicher physisch nicht verändert. Die Funktion verändert Adressierungsart zu Feldelementen, darum wird der Inhalt des Feld-Rezipienten beim Kopieren eines Feldes in ein anderes Feld durch die Funktion ArrayCopy() nicht von der Richtung des Indizierens im Feld-Quelle abhängen.

Man darf nicht Richtung des Indizierens für statisch verteilte Felder verändern. Auch wenn ein Feld als Parameter in die Funktion übertragen wurde, werden die Versuche, Indizieren innerhalb dieser Funktion zu verändern, zum nichts führen.

für Indikatorpuffer ist es auch wie für normale Felder moeglich, Richtung des Indizierens umgekehrt wie in einer Zeitreihe einzustellen, d.h. Zugang zur Null-Position in Indikatorpuffer wird Zugang zum letzten Wert im entsprechenden Indikatorpuffer bedeuten und das wird dem Indikatorwert in der allerletzten Bar entsprechen. Dabei wird physisch Ordnung der Daten im Indikatorpuffer nicht verändern, wie es schon erwaehnt wurde.

Erhalten von Preisdaten in Indikatoren

Jeder Benutzerindikator muss unbedingt die Funktion OnCalculate() haben, der Preisdaten übertragen werden, die für Berechnung der Werte in Indikatorpuffern notwendig sind. Richtung des Indizierens in diesen übertragenen Feldern kann man durch die Funktion ArrayGetAsSeries() erfahren .

Die in die Funktion übertragenen Felder stellen Preisdaten dar, d.h. diese Felder haben das Zeichen der Zeitreihe und die Funktion ArrayIsSeries() gibt true bei der Pruefung dieser Felder zurück. Aber

© 2000-2025, MetaQuotes Ltd.

Richtung des Indizierens muss auf jeden Fall nur durch die Funktion ArrayGetAsSeries() geprueft werden.

Um nicht von den Default-Werten abhängig zu sein, muss man die Funktion ArraySetAsSeries() für die Felder aufrufen, mit denen es zu arbeiten soll und die notwendige Richtung des Indizierenz einstellen.

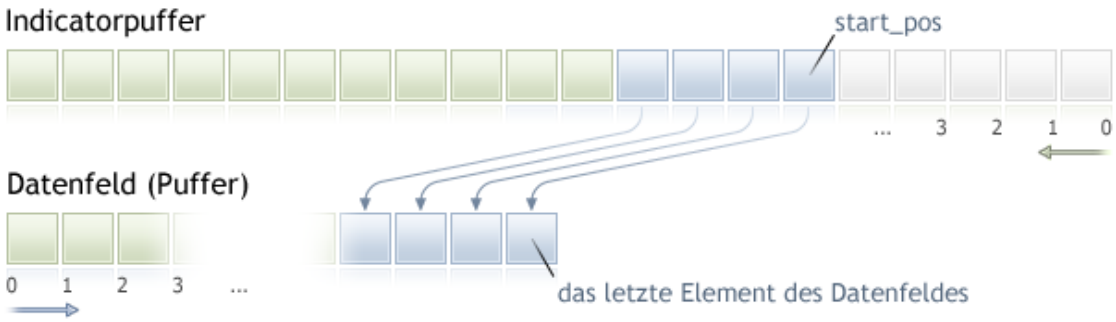
Erhalten von Preisdaten und Indikatorwerten

In Experten, Indikatoren und Scripts haben alle Felder als Default die Richtung des Indizierens von links nach rechts. Wenn notwendig können die Werte der Zeitreihen für jedes Symbol und Timeframe in jedem mql5-Programm angefordert werden aber auch Indikatorwerte, die in jedem Symbol und Timeframe berechnet werden können.

für Erhaltung dieser Daten sind die Funktionen Copy...() bestimmt:

- CopyBuffer - Kopieren der Werte des Indikatorpuffers ins Feld des Typs double;
- CopyRates - Kopieren der Preisgeschichte ins Feld der Strukturen MqlRates;
- CopyTime - Kopieren der Werte Time ins Feld des Typs datetime;
- CopyOpen - Kopieren der Werte Open ins Feld des Typs double;
- CopyHigh -Kopieren der Werte High ins Feld des Typs double;
- CopyLow - Kopieren der Werte Low ins Feld des Typs double;
- CopyClose - Kopieren der Werte Close ins Feld des Typs double;
- CopyTickVolume - Kopieren der Tickvolumen ins Feld des Typs long;
- CopyRealVolume - Kopieren der Boersenvolumen ins Feld des Typs long;
- CopySpread - Kopieren der Spreadgeschichte ins Feld des Typs int;

Alle diese Funktionen funktioniren gleich, darum reicht es, Mechanismuss der Datenerhaltung am Beispiel CopyBuffer() zu betrachten. Es wird angenommen, dass alle angeforderten Daten dieselbe Richtung des Indizierens haben, wie in der Zeitreihe, dabei wird angenommen, dass in der Position mit Index 0 (Null) Daten der unbeendeten Bar aufbewahren werden. Für Erhaltung des Zuganges zu diesen Daten muss man das notwendige Volumen ins Feld-Rezipient kopieren, ZB ins Feld *buffer*.



Beim Kopieren ist es notwendig, Startposition im Ausgangsfeld anzugeben, von der Daten ins Feld-Rezipient kopiert werden. Im Erfolgsfall wird ins Feld-Rezipient die angegebene Anzahl der Elementen vom Ausgangsfeld kopiert, in diesem Fall vom Indikatorpuffer. Dabei wird Kopieren immer so durchgeführt, wie es auf dem Bild gezeigt, unabhängig davon, welche Richtung des Indizierens im Feld-Rezipient angegeben wird.

© 2000-2025, MetaQuotes Ltd.

Wenn Preisdaten voraussichtlich in einer Schleife mit einer großen Anzahl von Iterationen behandelt werden, ist es ratsam, dass Sie die Tatsache der erzwungenen Beendigung des Programms mit der Funktion IsStopped() überprüfen:

```
int copied=CopyBuffer(ma_handle,// Handle von Indikator
                      0,        // Index der Indikator-Puffer
                      0,        // Startposition für das Kopieren
                      number,   // Anzahl der Werten für das Kopieren
                      Buffer    // Das Array, das die Werte erhält
                      );
if(copied<0) return;
int k=0;
while(k<copied && !IsStopped())
  {
   //--- Erhalten den Wert für den Index k
   double value=Buffer[k];
   // ...
   // Arbeit mit dem Wert value
   k++;
  }
```

Beispiel:

```
input int per=10; // Periode der Exponente
int ma_handle;    // handle des Indikators
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   ma_handle=iMA(_Symbol,0,per,0,MODE_EMA,PRICE_CLOSE);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//---
   double ema[10];
   int copied=CopyBuffer(ma_handle,// handle des Indikators
                         0,        // Index des Indikatorpuffers
                         0,        // Anfangsposition für Kopieren
                         10,       // Anzahl der Werte für Kopieren
                         ema       // Feld-Rezipient der Werte
                         );
   if(copied<0) return;
// .... weiterer Kode
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Sehen Sie auch

Datenzugang organisieren

© 2000-2025, MetaQuotes Ltd.

Datenzugriff organisieren

In diesem Abschnitt werden die Fragen behandelt, die mit Erfassung, Aufbewahren und Anforderung der Preisdaten (Timeserien) verbunden sind.

Datenerfassung vom Handelsserver

Bevor Preisdaten im Terminal MetaTrader 5 zugaenglich sind, muessen sie erfasst und verarbeitet werden. Für Datenerfassung ist der Anschluss an den Handelsserver MetaTrader 5 erforderlich. Daten werden vom Server erfasst auf Anfrage des Terminals in der Form der gepackten Blocks der Minutenbars.

Der Mechanismus der Serverreferenz hängt nicht davon ab, wie die Anfrage initiiert wurde - vom Benutzer bei der Navigation durch Chart oder vom Programm in der Sprache MQL5.

Aufbewahren der Zwischendaten

Die vom Server erfassten Daten werden verpackt und im spezialen Zwischenformat HCC aufbewahren. Daten für jedes Symbol werden in dem gesonderten Verzeichnis Terminalverzeichnis\bases\Name_Servers\history\Name_Symbols geschrieben . Z.B. Daten für Symbol EURUSD vom Handelsserver MetaQuotes-Demo werden sich im VerzeichnisKatalog_*Terminals*\bases\MetaQuotes-Demo\history\EURUSD\ befinden.

Daten werden in die Dateien mit der Endung .hcc geschrieben, jede Datei bewahrt Daten der Minutenbars für ein Jahr. ZB. due Datei 2009.hcc hat im Verzeichnis EURUSD Minutenbars für Symbol EURUSD für das Jahr 2009. Diese Daten werden für Vorbereitung der Preisdaten für alle Timeframes verwendet und sind nicht für den direkten Zugang bestimmt.

Erhaltung der Daten des notwendigen Timeframes aus den Zwischendaten

Dateien im Format HCC sind Quelle für Bauen der Preisdaten für angeforderte Timeframes im Format HC. Daten im Format HC sind Timeserien, die für den schnellen Zugang am meisten maximal sind. Sie werden nur auf Anfrage des Charts oder des mql5-Programms erzeugt, das Volumen ist nicht mehr als Parameterwert "Max bars in charts", und sie sind für weitere Verwendung in Dateien mit Endung hc aufbewahren.

für Ressourcenökonomie werden Daten des Timeframes gespeichert und aufbewahren im Operativspeicher wenn notwendig, beim langen Fehlen der Aufrufe werden sie aus dem Operativspeicher befreit und in der Datei gespeichert. Für jedes Timeframe werden Daten vorbereiten unabhängig davon, ob es schon fertige Daten für andere Timeframes gibt. Die Regeln des Zusammenstellens und Zugaenglichkeit der Daten sind für alle Timeframes gleich. D.h. Abgesehen davon, dass die Einheit der aufbewahrenen Daten im HCC Format Minutenbar ist, bedeutet das Vorhandensein der Daten im HCC Format nicht das Vorhandensein und Zugaenglichkeit der Daten des Timeframes M1 im HC Format in derselbenn Anzahl.

Erhalten neuer Daten vom Server ruft automatischen Update der verwendeten Preisdaten im HC Format für alle timeframes und Nachkalkulation aller Anzeiger, die die Daten explizit verwenden als Eingabedaten für Berechnung.

Parameter "Max bars in chart"

© 2000-2025, MetaQuotes Ltd.

Parameter "Max bars in charts" beschraenkt die Anzahl der Bars im Format HC, die für Charts, Anzeiger und mql5-Programme zugaenglich sind. Diese Einschraenkung gilt für Daten aller Timeframes und ist in erster Linie für Sparen der Ressourcen bestimmt.

Bei der Einstellung der grossen Werte dieses Parameters muss an sich daran erinnern, dass wenn die Geschichte der Preisdaten für kleine Timeframes tief ist, kann der Speicher für Aufbewahren der Zeitreihen und Puffer hunderte Megabytes sein und Einschraenkungen des Operativspeichers für Programm des Client-Terminals erreichen (2GB für 32-Bit MS Windows-Anwendungen).

Veränderung des Parameters "Max bars in charts" tritt in Kraft nach Wiederanlauf des Client-Terminals. Die Veränderung dieses Parameters selbst führt weder zum Anruf des Servers, um zusaetzliche Daten zu bekommen, noch zur Bildung zusaetzlicher Bars der Zeitreihen. Anforderung der zusaetzlichen Preisdaten und Erneuerung der Zeitreihen unter Beachtung der neuen Einschraenkung erfolgen entweder beim Scrolling des Charts zum Bereich mit den fehlenden Daten oder bei der Anforderung der fehlenden Daten vom mql5-Programm.

Das Volumen der vom Server angeforderten Daten entspricht der erforderlichen Anzahl der Bars des vorgegebenen Timeframes unter Beachtung vom Parameter "Max bars in charts". Einschraenkung, vorgegeben vom Parameter ist nicht strikt, und in einigen Faellen kann die Anzahl der zugaenglichen Daten für Timeframe bisschen groesser sein, als der laufende Wert des Parameters.

Zugaenglichkeit der Daten

Anwesenheit der Daten im HCC Format oder im fertigen für Verwendung HC Format bedeutet nicht, dass diese Daten für Darstellung auf dem Chart oder für Verwendung in mql5-Programmen vollstaendig zugaenglich sind.

Beim Zugang zu Preisdaten oder zu Indikatorwerten aus mql5-Programmen muss man behalten, dass ihrew Zugaenglichkeit für bestimmten Zeitpunket oder seit dem betimmten Zeitpunkt nicht garantiert wird. Das ist damit verbunden, dass zu Zwecken der Ressourcenökonomie wird in MetaTrader 5 nicht die ganze Kopie der erforderlichen Daten für mql5-Programm aufbewahren, sondern es wird ein direkter Zugang zur Datenbank des Terminals gegeben.

Preisgeschichte für alle Timeframes wird aus allgemeinen Daten des HCC Formats gebildet und jedes Update der Daten vom Server führt zum Update der Daten für alle Timeframes und Umbeechnung der Indikatoren. Deswegen kann der Datenzugang zurückgewiesen werden auch wenn diese Daten vor einem Moment zugaenglich waren.

Synchronisierung der Terminaldaten und der Serverdaten

Da ein mql5-Programm Daten für jedes Symbol und Timeframe aufrufen kann, ist es moeglich,dass Daten der erforderlichen Serie im Terminal noch nicht formiert sind oder Preisdaten mit dem Handelsserver noch nicht synchronisiert sind. In diesem Fall ist Wartezeit schwer vorherzusagen.

Algorithmen, die Wartezeit Zyklen verwenden sind keine beste Entscheidung. der einzige Ausschluss ist in diesem Fall Scripts, denn sie keine andere Algorithmenwahl haben wegen des Fehlens der Ereignisverarbitung. Für Benutzerindikatoren sind solche Algorithmen, wie auch andere Wartezeitzyklen strikt nicht empfohlen, denn sie führen zur Beendigung der Berechnung aller Indikatoren und einer anderen Verarbeitung der Preisdaten für das Symbol.

für Experten und Benutzerindikatoren ist es besser Ereignismodel der Verarbeitung zu verwenden. Wenn es bei der Verarbeitung des Ereignisses OnTick() oder OnCalculate() unmoeglich war, alle

© 2000-2025, MetaQuotes Ltd.

erforderlichen Daten der Angeforderten Zeitreihe zu bekommen, muss man Ereignisbearbeiter verlassen und vertrauen auf Datenzugang beim naechsten Aufruf des Bearbeiters.

Beispiel eines Scripts für Laden der Geschichte

Betrachten wir ein Beispiel des Scripts, das Anforderung erfuellt, Geschichte vom Handelsserver für das angegebene Symbol zu erhalten. Script ist dafür bestimmt, das erforderliche Instrument im Chart ablaufen zu lassen, Timeframe ist nicht von Bedeutung, denn, es wurde schon erwaehnt, alle Preisdaten kommen vom Handelsserver als verpackte Minutendaten, aus denen danach jede vorbestimmte Zeitreihe gebildet wird.

Schreiben wir alle Handlungen bezueglich Datenerhaltung als eine gesonderte Funktion CheckLoadHistory(symbol, timeframe, start_date):

```
int CheckLoadHistory(string symbol,ENUM_TIMEFRAMES period,datetime start_date)
  {
  }
```

Funktion ist CheckLoadHistory() ist als universelle Funktion gedacht, die aus jedem Programm (Expert, Script oder Indikator) aufgerufen werden kann und darum muss sie drei Eingabeparameter haben: Symbolname, Periode und Anfangsdatum, von dem wir Preisgeschichte brauchen.

Geben wir in den Kode der Funktion alle notwendigen Pruefungen ein, bevor wir die fehlende Geschichte fordern. Erstens, überzeugenwir uns, dass Symbolname und Periodenwert korrekt sind:

```
   if(symbol==NULL || symbol=="") symbol=Symbol();
   if(period==PERIOD_CURRENT)     period=Period();
```

Dann überzeugen wir uns davon, dass das angegebene Symbol im Fenster MarketWatch zugaenglich ist, d.h. die Geschichte des angegebenen Symbols auch beim Senden der Anforderung zum Handelsserver zugaenglich ist. Wenn solches Symbol in MarketWatch fehlt fuegen wir das zu durch die Funktion SymbolSelect().

```
   if(!SymbolInfoInteger(symbol,SYMBOL_SELECT))
     {
      if(GetLastError()==ERR_MARKET_UNKNOWN_SYMBOL) return(-1);
      SymbolSelect(symbol,true);
     }
```

Jetzt ist es notwendig, Anfangsdatum der zugaenglichen Geschichte für das angegebene Paar Symbol/Periode zu erhalten. Der Wert des Eingabeparameters startdate, der der Funktion CheckLoadHistory() übertragen wurde, kann in Interval der schon zugaenglichen Geschichte gelangen und dann ist keine Anforderung zum Handelsserver erforderlich. Für Erhaltung des ersten Datums für Symbol-Periode ist zum jetzigen Zeitpunkt die Funktion SeriesInfoInteger() mit dem Modifikator SERIES_FIRSTDATE bestimmt.

```
   SeriesInfoInteger(symbol,period,SERIES_FIRSTDATE,first_date);
   if(first_date>0 && first_date<=start_date) return(1);
```

Die naechste Pruefung ist die Pruefung des Programmtyps, aus dem die Funktion aufgerufen wird. Erinnern wir daran, dass Senden der Anforderung, Zeitreihe mit derselben Periode wie beim Indikator zu update ist gar nicht erwuenschenswert. Unerwuenschtheit der Datenanforderung für das selbe Symbol-Periode, wie beim Indikator ist dadurch bedingt, dass Update der historischen Daten in

© 2000-2025, MetaQuotes Ltd.

demselben Thread erfolgt, wo Indikator arbeitet. Darum ist clinch hoechstwarscheinlich. Für die Pruefung verwenden wir die Funktion MQL5InfoInteger() mit dem Modifikator MQL5_PROGRAM_TYPE.

```
   if(MQL5InfoInteger(MQL5_PROGRAM_TYPE)==PROGRAM_INDICATOR && Period()==period && Sy
      return(-4);
```

Wenn wir alle Pruefungen erfolgreich durchgelaufen haben, machen wir den letzten Versuch ohne Aufruf des Handelsservers. Zuerst erfahren wir über das Anfangsdatum, für das Minutendaten in HCC Format zugaenglich sind. Rufen wir diesen Wert durch die Funktion SeriesInfoInteger() mit dem Modiofikator SERIES_TERMINAL_FIRSTDATE auf und vergleichen ihn mitdem Parameterwert start_date.

```
   if(SeriesInfoInteger(symbol,PERIOD_M1,SERIES_TERMINAL_FIRSTDATE,first_date))
     {
      //--- there is loaded data to build timeseries
      if(first_date>0)
        {
         //--- force timeseries build
         CopyTime(symbol,period,first_date+PeriodSeconds(period),1,times);
         //--- check date
         if(SeriesInfoInteger(symbol,period,SERIES_FIRSTDATE,first_date))
            if(first_date>0 && first_date<=start_date) return(2);
        }
     }
```

Wenn nach aller Pruefungen Durchführungsthread im Koerpder der Funktion CheckLoadHistory() noch bleibt, dann bedeutet es, es ist notwendig, die fehlenden Preisdaten vom Handelsserver anzufordern. Vor allem erfahren wir über den Wert "Max bars in chart" durch die Funktion TerminalInfoInteger():

```
  int max_bars=TerminalInfoInteger(TERMINAL_MAXBARS);
```

Diese Funktion brauchen wir, um eAnforderung von extra Daten vorzubeugen. Dann klaeren wir das erste Datum in der Geschichte des Symbol sauf dem Handelsserver auf (unabhängig von der Periode) durch die schon bekannte Funktion SeriesInfoInteger() mit dem Modifikator SERIES_SERVER_FIRSTDATE.

```
   datetime first_server_date=0;
   while(!SeriesInfoInteger(symbol,PERIOD_M1,SERIES_SERVER_FIRSTDATE,first_server_date
      Sleep(5);
```

Da eine Anforderung eine asynchrone Operation ist, wird die Funktion mit der kleinen Verzoegerung 5 Milisekuden aufgerufen bis die Variable first_server_date den Wert erhaelt oder Zyklusdurchführung vom Benutzer unterbrochen wird (IsStopped(), in diesem Fall gibt den Wert true zurück). Geben wir den korrekten Wert des Anfangsdatums an, von dem wir Preisdaten vom Handelsserver anfordern.

```
   if(first_server_date>start_date) start_date=first_server_date;
   if(first_date>0 && first_date<first_server_date)
      Print("Warning: first server date ",first_server_date,
            " for ",symbol," does not match to first series date ",first_date);
```

Wenn das Anfangsdatum *first_server_date* auf dem Server niedriger als Anfangsdatum *first_date* für Symbol in HCC Format ist, wird im Journal die entsprechende Nachricht ausgegeben werden.

Jetzt sind wir bereit, Anforderung beim Handelsserver zu machen, um fehlende Preisdaten zu bekommen. Anforderung machen wir in Form eines Zyklus und fangen wir an, ihren Koerper auszufuellen:

© 2000-2025, MetaQuotes Ltd.

```
while(!IsStopped())
  {
   //1. warten auf Synchronisierung zwischen umgebildeten Zeitreihe und Zwischenges
   //2. laufende Anzahl der Bars in dieser Zeitreihe erhalten
   //   Wenn bars groesser ist, als Max_bars_in_chart, können wir verlassen, die An
   //3. Erhalten wir Anfangsdatum first_date in der umgebildeten Zeitreihe und verg
   //   start_date wenn first_date kleiner ist als start_date, können wir verlassen
   //4. Fordern wir vom Handelsserver einen neuen Teil der Geschichte - 100 Bars vo
   //   zugaenglichen Bar mit Nr bars
  }
```

Die ersten drei Punkte werden durch schon bekannte Methoden realisiert.

```
while(!IsStopped())
  {
   //--- 1.Warten darauf, bis Umbildung der Zeitreihe beendet wird
   while(!SeriesInfoInteger(symbol,period,SERIES_SYNCHRONIZED) && !IsStopped())
      Sleep(5);
   //--- 2.fordern wir an, wieviel Bars wir jetzt haben
   int bars=Bars(symbol,period);
   if(bars>0)
     {
      //--- Anzahl der Bars ist mehr, als man auf dem Chart zeigen kann, verlassen
      if(bars>=max_bars) return(-2);
      //--- 3. erfahren wir über das Anfangsdatum in der Zeitreihe
      if(SeriesInfoInteger(symbol,period,SERIES_FIRSTDATE,first_date))
         // Anfangsdatum ist früher als angefordert, Aufgabe ist erfuellt
         if(first_date>0 && first_date<=start_date) return(0);
     }
   //4. Fordern wir vom Handelsservereinen neuen Teil der Geschichte von
   //   der letzten zugaenglichen Bar mit der Nummer bars an - 100 Bars
  }
```

Es bleibt der letzter vierte Punkt - unmittelbare Anfirderung der Geschichte. Wir können nicht direkt Serveraufrufen, aber jede Copy-Funktion beim Fehlen der Geschichte in HCC Format initiiert automatisch das Senden dieser Anforderung vom Terminal zum Handelsserver. Da die Zeit des ersten Anfangsdatums in der Variable *first_date* als das einfachste und natuerlichste Kriterium um Grad der Erfuellung der Anforderung einzuschaetzen, wird es am leichtesten sein, die Funktion CopyTime() zu verwenden.

Beim Aufruf der Funktionen, durch die Kopieren jeder Daten von Zeitreihen erfolgt, muss man bemerken, dass Parameter *start* (Barnummer, mit der Kopieren der Preisdaten beginnt werden muss) immer innerhalb der zugaenglichen Geschichte des Terminlas sein muss. wenn wir nur 100 Bars haben, ist es sinnlos, 300 Bars zu kopieren versuchen, angefangen mit der Bar mit Index 500. Eine solche Anforderung wird als fehlerhafte angenommen werden und wird nicht verarbeitet, d.h. keine Geschichte vom Handelsserver wird geladen.

Eben darum werden wir 100 Bars kopieren, angefangen mit der Bar mit Index. Das garantiert sanfte Ladung der Geschichte vom Handelsserver, dabei wird mehr als angeforderte 100 Bars ausgeladen werden, Server gibt die Geschichte mit überschuss zurück.

```
int copied=CopyTime(symbol,period,bars,100,times);
```

Nach Kopieren muss man die Anzahl der kopiertenElemente analysieren, wenn der Versuch erfolglos geworden ist, wird der Wert der Variable *copied* gleich Null sein und der Wert des Counters fail_cnt wird um 1 steigen. Nach 100 erfolglosen Versuche wird die Funktionsarbeit unterbrochen.

© 2000-2025, MetaQuotes Ltd.

```
int fail_cnt=0;
...
   int copied=CopyTime(symbol,period,bars,100,times);
   if(copied>0)
     {
      //--- pruefen wir die Daten
      if(times[0]<=start_date)   return(0);  // der kopierte Wert ist kleiner, fertig
      if(bars+copied>=max_bars) return(-2); // Anzahl der Bars ist mehr, als auf dem C
                                            // gezeigt werden kann, fertig
      fail_cnt=0;
     }
   else
     {
      //--- nicht mehr als 100 erfolglose Versuche nacheinander
      fail_cnt++;
      if(fail_cnt>=100) return(-5);
      Sleep(10);
     }
```

So, ist in der Funktion nicht nur korrekte Verarbeitung der laufenden Situation zu jedem Zeitpunkt der Durchführung organisiert, sondern wird auch Beendigungskode zurückgegeben, den wir nach Aufruf der Funktion CheckLoadHistory() für die Erhaltung der zusaetzlichen Information verarbeiten können. ZB. auf dieser Weise:

```
   int res=CheckLoadHistory(InpLoadedSymbol,InpLoadedPeriod,InpStartDate);
   switch(res)
     {
      case -1 : Print("Unbekanntes Symbol",InpLoadedSymbol);
      case -2 : Print("Angeforderte Anzahl der Bars ist mehr, als es auf Chart gezeigt
      case -3 : Print("Durchführung wurde vom Benutzer unterbrochen");
      case -4 : Print("Indikator muss nicht eigene Namen laden");
      case -5 : Print("Laden ist nicht erfolgreich");
      case  0 : Print("alle Daten sind geladen");
      case  1 : Print("Anzahl der zugaenglichen Daten in der Zeitreihe ist genug");
      case  2 : Print("Zeitreihe ist aus vorhandenen Daten des Terminals gebildet");
      default : Print("Ergebnis der Durchführung ist nicht bestimmt");
     }
```

Der vollstaendige Funktionskode ist in Script angegeben, das richtiges Organisieren des Zuganges zu jeden Daten mit der Verarbeitung des Ergebnisses der Anforderung demonstriert.

Kode:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                TestLoadHistory.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.02"
#property script_show_inputs
//--- input parameters
input string          InpLoadedSymbol="NZDUSD";   // Symbol to be load
input ENUM_TIMEFRAMES InpLoadedPeriod=PERIOD_H1;  // Period to be load
input datetime        InpStartDate=D'2006.01.01'; // Start date
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   Print("Start load",InpLoadedSymbol+","+GetPeriodName(InpLoadedPeriod),"from",InpSta
//---
   int res=CheckLoadHistory(InpLoadedSymbol,InpLoadedPeriod,InpStartDate);
   switch(res)
     {
      case -1 : Print("Unknown symbol ",InpLoadedSymbol);              break;
      case -2 : Print("Requested bars more than max bars in chart"); break;
      case -3 : Print("Program was stopped");                         break;
      case -4 : Print("Indicator shouldn't load its own data");       break;
      case -5 : Print("Load failed");                                 break;
      case  0 : Print("Loaded OK");                                   break;
      case  1 : Print("Loaded previously");                           break;
      case  2 : Print("Loaded previously and built");                 break;
      default : Print("Unknown result");
     }
//---
   datetime first_date;
   SeriesInfoInteger(InpLoadedSymbol,InpLoadedPeriod,SERIES_FIRSTDATE,first_date);
   int bars=Bars(InpLoadedSymbol,InpLoadedPeriod);
   Print("First date",first_date,"-",bars,"bars");
//---
  }
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
int CheckLoadHistory(string symbol,ENUM_TIMEFRAMES period,datetime start_date)
  {
   datetime first_date=0;
   datetime times[100];
//--- check symbol & period
   if(symbol==NULL || symbol=="") symbol=Symbol();
   if(period==PERIOD_CURRENT)     period=Period();
//--- check if symbol is selected in the MarketWatch
   if(!SymbolInfoInteger(symbol,SYMBOL_SELECT))
     {
      if(GetLastError()==ERR_MARKET_UNKNOWN_SYMBOL) return(-1);
      SymbolSelect(symbol,true);
     }
//--- check if data is present
   SeriesInfoInteger(symbol,period,SERIES_FIRSTDATE,first_date);
   if(first_date>0 && first_date<=start_date) return(1);
//--- don't ask for load of its own data if it is an indicator
   if(MQL5InfoInteger(MQL5_PROGRAM_TYPE)==PROGRAM_INDICATOR && Period()==period && Syr
```

© 2000-2025, MetaQuotes Ltd.

```
      return(-4);
//--- second attempt
   if(SeriesInfoInteger(symbol,PERIOD_M1,SERIES_TERMINAL_FIRSTDATE,first_date))
     {
      //--- there is loaded data to build timeseries
      if(first_date>0)
        {
         //--- force timeseries build
         CopyTime(symbol,period,first_date+PeriodSeconds(period),1,times);
         //--- check date
         if(SeriesInfoInteger(symbol,period,SERIES_FIRSTDATE,first_date))
            if(first_date>0 && first_date<=start_date) return(2);
        }
     }
//--- max bars in chart from terminal options
   int max_bars=TerminalInfoInteger(TERMINAL_MAXBARS);
//--- load symbol history info
   datetime first_server_date=0;
   while(!SeriesInfoInteger(symbol,PERIOD_M1,SERIES_SERVER_FIRSTDATE,first_server_date
      Sleep(5);
//--- fix start date for loading
   if(first_server_date>start_date) start_date=first_server_date;
   if(first_date>0 && first_date<first_server_date)
      Print("Warning: first server date ",first_server_date,
```

© 2000-2025, MetaQuotes Ltd.

```
            " for ",symbol," does not match to first series date ",first_date);
//--- load data step by step
   int fail_cnt=0;
   while(!IsStopped())
     {
      //--- wait for timeseries build
      while(!SeriesInfoInteger(symbol,period,SERIES_SYNCHRONIZED) && !IsStopped())
         Sleep(5);
      //--- ask for built bars
      int bars=Bars(symbol,period);
      if(bars>0)
        {
         if(bars>=max_bars) return(-2);
         //--- ask for first date
         if(SeriesInfoInteger(symbol,period,SERIES_FIRSTDATE,first_date))
            if(first_date>0 && first_date<=start_date) return(0);
        }
      //--- copying of next part forces data loading
      int copied=CopyTime(symbol,period,bars,100,times);
      if(copied>0)
        {
         //--- check for data
         if(times[0]<=start_date)  return(0);
         if(bars+copied>=max_bars) return(-2);
         fail_cnt=0;
        }
      else
        {
         //--- no more than 100 failed attempts
         fail_cnt++;
         if(fail_cnt>=100) return(-5);
         Sleep(10);
        }
     }
//--- stopped
   return(-3);
  }
//+------------------------------------------------------------------+
//| gibt Zeilenwert der Periode zurück                               |
//+------------------------------------------------------------------+
string GetPeriodName(ENUM_TIMEFRAMES period)
  {
   if(period==PERIOD_CURRENT) period=Period();
//---
   switch(period)
     {
      case PERIOD_M1:  return("M1");
      case PERIOD_M2:  return("M2");
      case PERIOD_M3:  return("M3");
      case PERIOD_M4:  return("M4");
      case PERIOD_M5:  return("M5");
      case PERIOD_M6:  return("M6");
      case PERIOD_M10: return("M10");
      case PERIOD_M12: return("M12");
      case PERIOD_M15: return("M15");
      case PERIOD_M20: return("M20");
      case PERIOD_M30: return("M30");
      case PERIOD_H1:  return("H1");
      case PERIOD_H2:  return("H2");
      case PERIOD_H3:  return("H3");
      case PERIOD_H4:  return("H4");
```

© 2000-2025, MetaQuotes Ltd.

```
      case PERIOD_H6:  return("H6");
      case PERIOD_H8:  return("H8");
      case PERIOD_H12: return("H12");
      case PERIOD_D1:  return("Daily");
      case PERIOD_W1:  return("Weekly");
      case PERIOD_MN1: return("Monthly");
     }
//---
   return("unknown period");
  }
```

© 2000-2025, MetaQuotes Ltd.

SeriesInfoInteger

Gibt die Information über den Zustand der historischen Daten zurück. Es gibt 2 Varianten der Funktion.

Gibt den Wert der Eigenschaft direkt.

```
long  SeriesInfoInteger(
   string                      symbol_name,      // Symbolname
   ENUM_TIMEFRAMES             timeframe,        // Periode
   ENUM_SERIES_INFO_INTEGER    prop_id,          // Identifikator der Eigenschaft
   );
```

Gibt true oder false zurück abhängig von der erfolgreichen Durchführung der Funktion.

```
bool  SeriesInfoInteger(
   string                      symbol_name,      // Symbolname
   ENUM_TIMEFRAMES             timeframe,        // Periode
   ENUM_SERIES_INFO_INTEGER    prop_id,          // Identifikator der Eigenschaft
   long&                       long_var          // Variable für die Erhaltung der Info
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

prop_id

[in] Identifikator der angeforderten Eigenschaft, Enumerationswert ENUM_SERIES_INFO_INTEGER.

long_var

[out] Variable, in die der Wert der angeforderten Eigenschaft gesetzt wird.

Rückgabewert

Wert des Typs long für die erste Variante des Aufrufes.

für die zweite Variante des Aufrufs gibt true zurück, wenn diese Eigenschaft zugaenglich ist und der Wert in der Variable long_var aufbewahren wird, anderenfalls gibt false zurück. Für die zusaetzliche Information über den Fehler rufen Sie die Funktion GetLastError() auf.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//---
   Print("Anzahl der Bars für Symbol-Periode zur Zeit = ",
         SeriesInfoInteger(Symbol(),Period(),SERIES_BARS_COUNT));

   Print("Das erste Datum für Symbol-Periode zur Zeit = ",
         (datetime)SeriesInfoInteger(Symbol(),Period(),SERIES_FIRSTDATE));

   Print("Das erste Datum in der Geschichte für Symbol auf dem Server = ",
         (datetime)SeriesInfoInteger(Symbol(),Period(),SERIES_SERVER_FIRSTDATE));

   Print("Symboldaten werden synchronisiert = ",
         (bool)SeriesInfoInteger(Symbol(),Period(),SERIES_SYNCHRONIZED));
  }
```

© 2000-2025, MetaQuotes Ltd.

Bars

Gibt die Anzahl der Bars in der Historie für das angegebene Symbol und Periode. Es gibt 2 Varianten der Funktion.

Anzahl aller Bars in der Historie anfordern

```
int  Bars(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   );
```

Anzahl der Bars im angegebenen Intervall abrufen

```
int  Bars(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time         // Beendigungsdatum
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit,die dem letzten Element entspricht.

Rückgabewert

Wenn die Parameter start_time und stop_time angegeben werden, gibt die Funktion die Anzahl der Bars im Datenbereich zurück. Wenn diese Parameter nicht angegeben werden, gibt die Funktion die gesamte Anzahl der Bars zurück.

Hinweis

Wenn Daten für eine Zeitreihe mit angegebenen Parametern beim Aufruf der Funktion Bars() im Terminal noch nicht formiert sind oder Daten einer Zeitreihe beim Funktionsaufruf mit dem Handelsserver nicht synchronisiert sind, gibt die Funktion den Null-Wert zurück.

Beim Abruf der Anzahl der Bars im angegebenen Intervall werden nur die Bars berücksichtigt, deren Eröffnungszeit zu diesem Intervall gehört. Zum Beispiel, wenn der aktuelle Tag Samstag ist, gibt die Funktion beim Abruf der Anzahl der Wochenbars start_time=letzter_dienstag und stop_time=letzter_freitag Null aus, denn die Eröffnungszeit im Wochenzeitrahmen fällt immer auf Sonntag und keine Wochenbar liegt im angegebenen Bereich.

Ein Beispiel für den Abruf der Anzahl aller Bars in der Historie:

© 2000-2025, MetaQuotes Ltd.

```
int bars=Bars(_Symbol,_Period);
if(bars>0)
  {
   Print("Anzahl der Bars in der Terminalgeschichte für Symbol-Periode zum jetziger
  }
else  //keine zugaengliche Bars
  {
   //--- Daten des Symbols sollen mit Daten auf dem Server synchronisiert werden
   bool synchronized=false;
   //--- Counter des Zyklus
   int attempts=0;
   // versuchen wir 5 mal, auf Synchronisierung zu warten
   while(attempts<5)
     {
      if(SeriesInfoInteger(Symbol(),0,SERIES_SYNCHRONIZED))
        {
         //--- Synchronisierung gemacht, beenden
         synchronized=true;
         break;
        }
      //--- steigen wir den Counter
      attempts++;
      //--- warten wir 10 Millisekunden auf die naechste Iteration
      Sleep(10);
     }
   //--- Zyklus beenden nach der Synchronisierung
   if(synchronized)
     {
      Print("Anzahl der Bars in der Terminalgeschichte für Symbol-Periode zum Jetzi
      Print("Das erste Datum in der Terminalgeschichte für Symbol-Periode zum jetzi
            (datetime)SeriesInfoInteger(Symbol(),0,SERIES_FIRSTDATE));
      Print("Das erste Datum in der Geschichte für Symbol auf dem Server =",
            (datetime)SeriesInfoInteger(Symbol(),0,SERIES_SERVER_FIRSTDATE));
     }
   //--- Synchronisierung der Daten nicht passiert
   else
     {
      Print("Misslungen Anzahl der Bars für zu bekommen ",_Symbol);
     }
  }
```

Ein Beispiel für den Abruf der Anzahl der Bars im angegebenen Intervall:

```
int n;
datetime date1 = D'2016.09.02 23:55'; // Freitag
datetime date2 = D'2016.09.05 00:00'; // Montag
datetime date3 = D'2016.09.08 00:00'; // Donnerstag
//---
n=Bars(_Symbol,PERIOD_H1,D'2016.09.02 02:05',D'2016.09.02 10:55');
Print("Anzahl der Bars: ",n); // Gibt aus "Anzahl der Bars: 8", die 2-Stunden-Bar w
n=Bars(_Symbol,PERIOD_D1,date1,date2);
Print("Anzahl der Bars: ",n); // Gibt aus "Anzahl der Bars: 1", denn das Intervall
n=Bars(_Symbol,PERIOD_W1,date2,date3);
Print("Anzahl der Bars: ",n); // Gibt aus "Anzahl der Bars: 0", denn der angegebene
```

Sehen Sie auch

Ereignisbearbeiter

© 2000-2025, MetaQuotes Ltd.

BarsCalculated

Gibt die Anzahl der berechneten Daten für den angeforderten Indikator.

```
int  BarsCalculated(
   int       indicator_handle,     // handle des Indikators
   );
```

Parameter

indicator_handle

[in] Handle des Indikators, das von der entsprechenden Indikatorfunktion erhalten wurde.

Rückgabewert

Gibt die Anzahl der berechneten Daten im Indikatorpuffer oder -1 im Fall des Fehlers (Daten sind noch nicht berechnet).

Hinweis

Die Funktion ist in den Faellen sinnvoll, wenn die Daten des Indikators sofort nach seiner Erzeugung erhalten werden muessen (Handle des Indikators ist zugaenglich).

Beispiel:

```
void OnStart()
  {
   double Ups[];
//--- stellen wir Zeichen der Zeitreihe  für Felder
   ArraySetAsSeries(Ups,true);
//--- erzeugen wir handle des Indikators Fractals
   int FractalsHandle=iFractals(NULL,0);
//--- stellen wir Fehlerkode zurück
   ResetLastError();
//--- versuchen wir, Werte des Indikators zu kopieren
   int i,copied=CopyBuffer(FractalsHandle,0,0,1000,Ups);
   if(copied<=0)
     {
      Sleep(50);
      for(i=0;i<100;i++)
        {
         if(BarsCalculated(FractalsHandle)>0)
            break;
         Sleep(50);
        }
      copied=CopyBuffer(FractalsHandle,0,0,1000,Ups);
      if(copied<=0)
        {
         Print("Misslungen, obere Fractals zu kopieren. Error = ",GetLastError(),
         "i = ",i,"    copied = ",copied);
         return;
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      else
        Print("Gelungen, obere Fractals zu kopieren.",
        "i = ",i,"    copied = ",copied);
    }
  else Print("Misslungen, obere Fractals zu kopieren. ArraySize = ",ArraySize(Ups));
}
```

© 2000-2025, MetaQuotes Ltd.

IndicatorCreate

Gibt handle des angegebenen technischen Indikators, erzeugt auf der Grundlage des Feldes der Parameter des Typs MqlParam.

```
int  IndicatorCreate(
   string           symbol,                    // Symbolname
   ENUM_TIMEFRAMES  period,                    // Periode
   ENUM_INDICATOR   indicator_type,            // Typ des Indikators aus der E
   int              parameters_cnt=0,          // Anzahl der Parameter
   const MqlParam&  parameters_array[]=NULL,   // Feld der Parameter
   );
```

Parameter

symbol

[in] Symbolname auf deren Daten der Indikator berechnet werden wird. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe .

indicator_type

[in] Typ des Indikators, kann einen der Enumerationswerte ENUM_INDICATOR annehmen.

parameters_cnt

[in] Anzahl der Parameter, die im Feld parameters_array[] übertragen werden. Elemente des Feldes haben haben den Spezialtyp der Struktur MqlParam. Der Default-Wert ist Null - Parameter werden nicht übertragen. Wenn Nicht-Nullzahl der Parameter angegeben wird, ist der Parameter *parameters_array* obligatorisch. Man kann nicht mehr als 64 Parameter übertragen.

parameters_array[]=NULL

[in] Feld des Typs MqlParam, dessen Elemente Typ und Wert jedes Eingabeparameters des technischen Indikators enthalten.

Rückgabewert

Gibt handle des angegebenen technischen Indokators zurück, im Fall des Misserfolges gibt INVALID_HANDLE zurück.

Hinweis

Wenn handle des Indikators des Typs IND_CUSTOM erzeugt wird, muss das Feld *type* des ersten Elementes der Eingabeparameter *parameters_array* den Wert TYPE_STRING aus der Enumeration ENUM_DATATYPE haben, und das Feld *string_value* des ersten Elementes muss den Namen des Benutzerindikators enthalten. Benutzerindikator muss compiliert werden (Datei mit Verbreitung EX5) und sich im Verzeichnis MQL5/Indicators des Client-Terminals oder eingebettenem Verzeichnis befinden.

Die für Testenerforderliche Indikatoren werden automatisch aus Aufruf der Funktionen iCustom() bestimmt, wenn der entsprechende Parameter von der Konstantzeile vorgegeben ist. Für andere Fälle (Verwendung der Funktion IndicatorCreate() oder Verwendung einer nicht Konstantzeile im

© 2000-2025, MetaQuotes Ltd.

Parameter, der den Namen des Indikator vorgibt) ist diese Eigenschaft #property tester_indicator erforderlich :

```
#property tester_indicator "indicator_name.ex5"
```

Wenn im Benutzerindikator die erste Form des Aufrufes verwendet wird, kann bei der übertragung der Eingabeparameter vom letzten Parameter zusaetzlich angegeben werden, welche Daten für seine Beechnung verwendet werden. Wenn Parameter "Apply to" nicht explizit angegeben wird, wird als Default die Berechnung für Werte PRICE_CLOSE durchgefürt.

Beispiel:

```
void OnStart()
  {
   MqlParam params[];
   int      h_MA,h_MACD;
//--- create iMA("EURUSD",PERIOD_M15,8,0,MODE_EMA,PRICE_CLOSE);
   ArrayResize(params,4);
//--- set ma_period
   params[0].type         =TYPE_INT;
   params[0].integer_value=8;
//--- set ma_shift
   params[1].type         =TYPE_INT;
   params[1].integer_value=0;
//--- set ma_method
   params[2].type         =TYPE_INT;
   params[2].integer_value=MODE_EMA;
//--- set applied_price
   params[3].type         =TYPE_INT;
   params[3].integer_value=PRICE_CLOSE;
//--- create MA
   h_MA=IndicatorCreate("EURUSD",PERIOD_M15,IND_MA,4,params);
//--- create iMACD("EURUSD",PERIOD_M15,12,26,9,h_MA);
   ArrayResize(params,4);
//--- set fast ma_period
   params[0].type         =TYPE_INT;
   params[0].integer_value=12;
//--- set slow ma_period
   params[1].type         =TYPE_INT;
   params[1].integer_value=26;
//--- set smooth period for difference
   params[2].type         =TYPE_INT;
   params[2].integer_value=9;
//--- set indicator handle as applied_price
   params[3].type         =TYPE_INT;
   params[3].integer_value=h_MA;
//--- create MACD based on moving average
   h_MACD=IndicatorCreate("EURUSD",PERIOD_M15,IND_MACD,4,params);
//--- use indicators
//--- . . .
//--- release indicators (first h_MACD)
   IndicatorRelease(h_MACD);
   IndicatorRelease(h_MA);
  }
```

© 2000-2025, MetaQuotes Ltd.

IndicatorParameters

Basierend auf das angegebenen Handle gibt Anzahl von Eingabeparametern des Indikators, sowie die Werte und Type der Parameter zurück.

```
int  IndicatorParameters(
   int               indicator_handle,     // Handle des Indikators
   ENUM_INDICATOR&   indicator_type,       // Variable, um den Typ des Indikators zu e
   MqlParam&         parameters[]          // Array, um die Parameter zu erhalten
   );
```

Parameter

indicator_handle

[in] Handle auf den Indikator, für den Sie die Anzahl der Parameter, auf die er berechnet wurde, erhalten möchten.

indicator_type

[out] Variable vom Typ ENUM_INDICATOR, in die der Typ des Indikators geschrieben wird.

parameters[]

[out] Ein dynamisches Array für Werte vom Typ MqlParam, in das die Liste der Indikator-Parameter geschrieben wird. Die Funktion IndicatorParameters() gibt die Größe des Arrays zurück.

Rückgabewert

Anzahl der Eingabeparameter des Indikators mit dem angegebenen Handle, im Falle eines Fehlers gibt -1 zurück. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() an.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {

//--- Die Anzahl der Fenster auf dem Chart (es gibt immer mindestens ein Hauptfenster)
   int windows=(int)ChartGetInteger(0,CHART_WINDOWS_TOTAL);
//--- Für alle Fenster des Charts machen
   for(int w=0;w<windows;w++)
     {
      //--- Die Anzahl der Indikatoren in diesem Fenster/Unterfenster
      int total=ChartIndicatorsTotal(0,w);
      //--- Für alle Indikatoren im Fenster machen
      for(int i=0;i<total;i++)
        {
         //--- Kurzname des Indikators erhalten
         string name=ChartIndicatorName(0,w,i);
         //--- Handle des Indikators erhalten
         int handle=ChartIndicatorGet(0,w,name);
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- In Log schreiben
         PrintFormat("Window=%d,  indicator #%d,  handle=%d",w,i,handle);
         //---
         MqlParam parameters[];
         ENUM_INDICATOR indicator_type;
         int params=IndicatorParameters(handle,indicator_type,parameters);
         //--- Nachrichtenkopf
         string par_info="Short name "+name+", type "
                         +EnumToString(ENUM_INDICATOR(indicator_type))+"\r\n";
         //---
         for(int p=0;p<params;p++)
           {
            par_info+=StringFormat("parameter %d: type=%s, long_value=%d, double_value
                                   p,
                                   EnumToString((ENUM_DATATYPE)parameters[p].type),
                                   parameters[p].integer_value,
                                   parameters[p].double_value,
                                   parameters[p].string_value
                                   );
           }
         Print(par_info);
        }
      //--- Getan für alle Indikatoren im Fenster
     }
//---
  }
```

Sehen Sie auch

ChartIndicatorGet()

© 2000-2025, MetaQuotes Ltd.

IndicatorRelease

Entfernt handle des Indikators und setzt Berechnungsteil des Indikators frei, wenn es nicht von jemandem verwendet wird.

```
bool  IndicatorRelease(
   int       indicator_handle     // handle des Indikators
   );
```

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Funktion erlaubt handle des Indikators zu entfernen, wenn es nicht brauchbar ist und spart den Speicher damit. Handle wird sofort entfernt, Berechnungsteil des Indikators wird nach einiger Zeit durchgeführt (wenn er nicht mehr aufgerufen wird).

Im Strategietester wird die Funktion IndicatorRelease() nicht ausgeführt.

Beispiel:

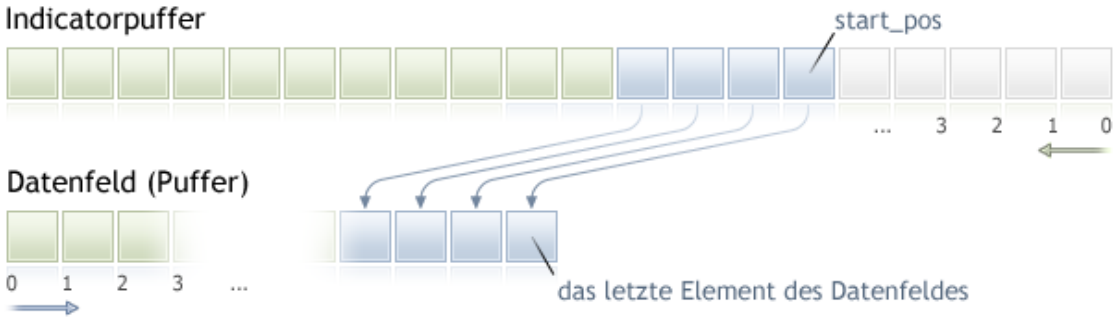
© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                        Test_IndicatorRelease.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- input parameters
input int                 MA_Period=15;
input int                 MA_shift=0;
input ENUM_MA_METHOD      MA_smooth=MODE_SMA;
input ENUM_APPLIED_PRICE price=PRICE_CLOSE;
//--- wir werden handle des Indikators aufbewahren
int MA_handle;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- erzeugen wir handle des Indikators
   MA_handle=iMA(Symbol(),0,MA_Period,MA_shift,MA_smooth,price);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- wenn der Wert der globalen Variable noch nicht festgelegt wird
   if(GlobalVariableCheck("MA_value")==0)
     {
      //--- dynamisches Feld für Erhaltung der Werte des Indikators
      double v[];
      //--- erhalten wir Werte des Indikators in zwei letzten Bars
      if(CopyBuffer(MA_handle,0,0,2,v)==2 && v[1]!=EMPTY_VALUE)
        {
         //--- speichern wir Wert der globalen Variable in der vorletzten Bar
         if(GlobalVariableSet("MA_value",v[1]))
           {
            //--- befreien wir handle des Indikators
            if(!IndicatorRelease(MA_handle))
               Print("IndicatorRelease() failed. Error ",GetLastError());
           }
        }
     }
//---
  }
```

© 2000-2025, MetaQuotes Ltd.

CopyBuffer

Bekommt Daten des angegebenen Puffers in der angegebenen Zahl. Es gibt 3 Varianten der Funktion.



Abzählen der Elemente der kopierten Daten (Indikatorpuffer mit dem Index buffer_num) von der Startposition wird von der Gegenwart zur Vergangenheit durchgeführt, d. h. die Startposition, 0. bedeutet die laufende Bar (Indikatorwert für die laufende Bar).

Als Feld-Rezipient buffer[] ist es wünschenswert, ein dynamisches Feld zu verwenden, denn die Funktion CopyBuffer() versucht, die Größe des Feld-Rezipienten für Größe der kopierten Daten zu verteilen. Wenn als Feld-Rezipient buffer[] ein Indikatorpuffer auftritt (Feld, das für Aufbewahren der Indikatorwerte durch die Funktion SetIndexBufer() verteilt wurde), ist partielles Kopieren zugelassen. Beispiel kann man im Benutzerindikator Awesome_Oscillator.mq5 in der Standardlieferung des Terminals sehen.

Wenn es erforderlich ist, Indikatorwerte in ein anderes Feld (nicht Indikatorpuffer)partiell zu kopieren, muss man dazu ein Zwischenfeld verwenden, in das die erforderliche Anzahl kopiert wird. Und von diesen Zwischenfeld wird die erforderliche Anzahl der Werte in die notwendigen Stellen des Feld-Rezipienten elementenweise kopiert.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyBuffer(
   int       indicator_handle,     // handle des Indikators
   int       buffer_num,           // Nummer des Puffers des Indikators
   int       start_pos,            // Anfangsposition
   int       count,                // Anzahl für Kopieren
   double    buffer[]              // Feld, wohin Daten kopiert werden
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyBuffer(
   int       indicator_handle,     // handle des Indikators
```

© 2000-2025, MetaQuotes Ltd.

```
   int       buffer_num,          // Nummer des Puffers des Indikators
   datetime  start_time,          // Anfangsdatum
   int       count,               // Anzahl für Kopieren
   double    buffer[]             // Feld, wohin Daten kopiert werden
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

```
int  CopyBuffer(
   int       indicator_handle,    // handle des Indikators
   int       buffer_num,          // Nummer des Puffers des Indikators
   datetime  start_time,          // Anfangsdatum
   datetime  stop_time,           // Beendigungsdatum
   double    buffer[]             // Feld, wohin Daten kopiert werden
   );
```

Parameter

indicator_handle

[in] Handle des Indikators, das durch die Indikatorfunktion zurückgegeben wird.

buffer_num

[in] Nummer des Puffers des Indikators.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

buffer[]

[out] Feld des Typs double.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                TestCopyBuffer3.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot MA
#property indicator_label1  "MA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input bool                AsSeries=true;
input int                 period=15;
input ENUM_MA_METHOD      smootMode=MODE_EMA;
input ENUM_APPLIED_PRICE  price=PRICE_CLOSE;
input int                 shift=0;
//--- indicator buffers
double                    MABuffer[];
int                       ma_handle;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,MABuffer,INDICATOR_DATA);
   Print("Prameter AsSeries =",AsSeries);
   Print("Indikatorpuffer nach SetIndexBuffer() ist eine Zeitreihe = ",
         ArrayGetAsSeries(MABuffer));
//--- set short indicator name
   IndicatorSetString(INDICATOR_SHORTNAME,"MA("+period+")"+AsSeries);
//--- set AsSeries(dependes from input parameter)
   ArraySetAsSeries(MABuffer,AsSeries);
   Print("Indikatorpuffer nach ArraySetAsSeries(MABuffer,true); ist eine Zeitreihe = "
         ArrayGetAsSeries(MABuffer));
//---
   ma_handle=iMA(Symbol(),0,period,shift,smootMode,price);
   return(INIT_SUCCEEDED);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- check if all data calculated
   if(BarsCalculated(ma_handle)<rates_total) return(0);
//--- we can copy not all data
   int to_copy;
   if(prev_calculated>rates_total || prev_calculated<=0) to_copy=rates_total;
   else
     {
      to_copy=rates_total-prev_calculated;
      //--- last value is always copied
      to_copy++;
    ;}
//--- try to copy
   if(CopyBuffer(ma_handle,0,0,to_copy,MABuffer)<=0) return(0);
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Im oben angeführten Beispiel ist die Ausfüllung des Indikatorpuffers durch Werte eines anderen Indikatorpuffers vom Indikator in demselben Symbol/Periode dargestellt.

Sehen Sie ein umfassenderes Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

Sehen Sie auch

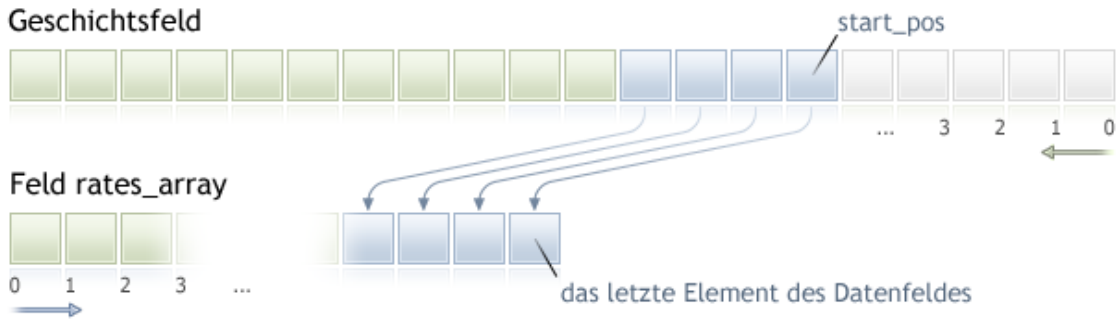
© 2000-2025, MetaQuotes Ltd.

Eigenschaften der Benutzerindikatoren, SetIndexBuffer

© 2000-2025, MetaQuotes Ltd.

CopyRates

Bekommt im Feld rates_array historische Daten der Struktur MqlRates des angegebenen Symbol-Periode in der angegebenen Menge. Abzählen der Elemente von der Startposition wird von der Gegenwart zur Vergangenheit durchgeführt, d. h. Startposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist als ein Feld enthalten kann, versucht man, ein Feld so zu verteilen, dass angeforderte Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyRates(
   string           symbol_name,       // Symbolname
   ENUM_TIMEFRAMES  timeframe,         // Periode
   int              start_pos,         // Anfangsposition
   int              count,             // Anzahl für Kopieren
   MqlRates         rates_array[]      // Feld, wohin Daten kopiert werden
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyRates(
   string           symbol_name,       // Symbolname
   ENUM_TIMEFRAMES  timeframe,         // Periode
   datetime         start_time,        // Anfangsdatum
   int              count,             // Anzahl für Kopieren
   MqlRates         rates_array[]      // Feld, wohin Daten kopiert werden
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

```
int  CopyRates(
```

© 2000-2025, MetaQuotes Ltd.

```
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   MqlRates         rates_array[]     // Feld, wohin Daten kopiert werden
   );
```

Parameter

symbol_name

[in] symbol.

timeframe

[in] Periode.

start_time

[in] Barzeit, die dem ersten Element entspricht.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

rates_array[]

[out] Feld des Typs MqlRates.

Rückgabewert

Anzahl der kopierten Elemente oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der

© 2000-2025, MetaQuotes Ltd.

Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

Beispiel:

```
void OnStart()
  {
//---
   MqlRates rates[];
   ArraySetAsSeries(rates,true);
   int copied=CopyRates(Symbol(),0,0,100,rates);
   if(copied>0)
     {
      Print("Anzahl der kopierten Bars: "+copied);
      string format="open = %G, high = %G, low = %G, close = %G, volume = %d";
      string out;
      int size=fmin(copied,10);
      for(int i=0;i<size;i++)
        {
         out=i+":"+TimeToString(rates[i].time);
         out=out+" "+StringFormat(format,
                                  rates[i].open,
                                  rates[i].high,
                                  rates[i].low,
                                  rates[i].close,
                                  rates[i].tick_volume);
         Print(out);
        }
     }
   else Print("Nicht gelungen, historische Daten nach Symbol zu erhalten ",Symbol());
  }
```

Sehen Sie ein umfassenderes Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart

© 2000-2025, MetaQuotes Ltd.

anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

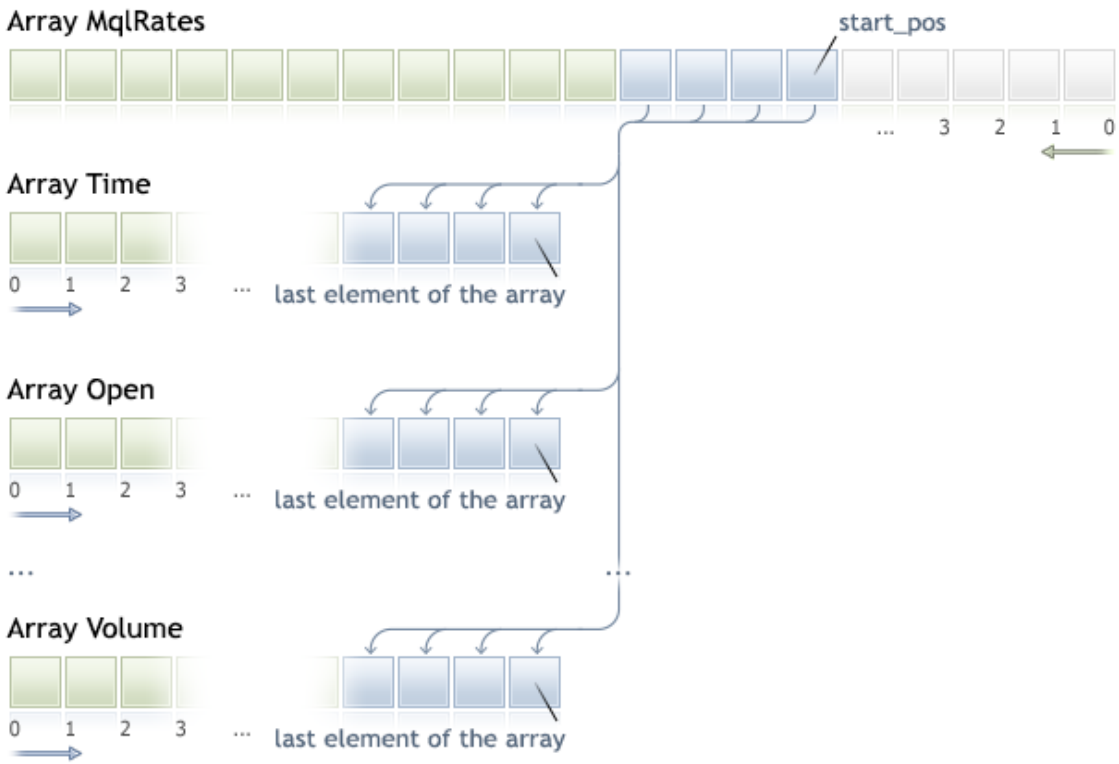
Sehen Sie auch

Strukturen und Klassen, TimeToString, StringFormat

© 2000-2025, MetaQuotes Ltd.

CopySeries

Ruft die synchronisierten Zeitreihen ab aus der Struktur MqlRates für den angegebene Zeitrahmen des Symbols und der angegebenen Anzahl.Die Daten werden dem angegebenen Satz von Arrays zugewiesen. Die Elemente werden von der Gegenwart in die Vergangenheit gereiht und das heißt, dass die Startposition 0 gleich der aktuellen Bar ist.



Wenn die zu kopierende Datenmenge nicht bekannt ist, wird empfohlen, einen dynamischen Array für die empfangenden Arrays zu verwenden, da, wenn die Datenmenge die Größe des Array übersteigt, dies den Versuch verursachen wird, das Array neu zu organisieren, um alle angeforderten Daten unterzubringen.

Wenn Sie eine vorher festgelegte Datenmenge kopieren müssen, ist es empfehlenswert, einen statisch deklarierten Array zu verwenden, um eine unnötige Neuzuweisung von Speicher zu vermeiden.

Die Eigenschaft des empfangenden Arrays — as_series=true oder as_series=false — wird ignoriert: beim Kopieren wird das älteste Zeitreihenelement an den Anfang des dem Array zugewiesenen physischen Speichers kopiert.

```
int  CopySeries(
   string           symbol_name,       // Symbolname
   ENUM_TIMEFRAMES  timeframe,         // Zeitrahmen
   int              start_pos,         // Startposition
   int              count,             // Anzahl zu kopieren
   ulong            rates_mask,        // Kombination der Flags, um die angeforderten
   void&            array1[],          // Array zur Aufnahme der Daten der ersten Zeit
   void&            array2[]           // Array zur Aufnahme der Daten der zweiten Zei
```

© 2000-2025, MetaQuotes Ltd.

```
   ...
   );
```

Parameter

symbol_name

[in] Symbolname

timeframe

[in] Zeitrahmen.

start_pos

[in] Index des ersten zu kopierenden Elements.

count

[in] Anzahl der zu kopierenden Elemente.

rates_mask

[in] Die Kombination der Flags aus der Enumeration ENUM_COPY_RATES.

array1, array2,...

[out] Arrays der entsprechenden Typen für den Empfang der Zeitreihen aus der Struktur MqlRates. Die Reihenfolge der an die Funktion übergebenen Arrays muss mit der Reihenfolge der Felder in der MqlRates-Struktur übereinstimmen.

Rückgabewert

Die Anzahl der kopierten Elemente oder -1, wenn ein Fehler aufgetreten ist.

Hinweis

Wenn das gesamte Intervall der angeforderten Daten außerhalb der auf dem Server verfügbaren Daten liegt, gibt die Funktion -1 zurück. Wenn die angeforderten Daten über TERMINAL_MAXBARS (die maximale Anzahl von Balken im Chart) hinausgehen, gibt die Funktion ebenfalls -1 zurück.

Wenn Daten von einem Indikator angefordert werden, gibt die Funktion sofort -1 zurück, wenn die angeforderten Zeitreihen noch nicht aufgebaut sind oder erst vom Server heruntergeladen werden müssen. Die Funktion initiiert jedoch das Herunterladen/Konstruieren der Daten selbst.

Wenn Daten von einem Expert Advisor oder einem Skript angefordert werden, wird der Download vom Server initiiert, wenn das Terminal die entsprechenden Daten nicht lokal hat, oder die Konstruktion der notwendigen Zeitreihen beginnt, wenn die Daten aus der lokalen Historie konstruiert werden können, aber noch nicht bereit sind. Die Funktion gibt die Datenmenge zurück, die zum Zeitpunkt der Zeitüberschreitung bereitgestellt werden konnte, der Download der Historie wird jedoch fortgesetzt, und die Funktion gibt bei der nächsten ähnlichen Anfrage weitere Daten zurück.

Unterschiede zwischen CopySeries und CopyRates

Die Funktion CopySeries ermöglicht es, während eines Aufrufs nur die erforderlichen Zeitreihen in verschiedenen angegebenen Arrays zu erhalten, wobei alle Zeitreihendaten synchronisiert werden. Das bedeutet, dass alle Werte in den resultierenden Arrays bei einem bestimmten Index N zum selben Balken auf dem angegebenen Symbol/Zeitrahmen-Paar gehören werden. Daher muss der

© 2000-2025, MetaQuotes Ltd.

Programmierer keine Sorge tragen, um alle empfangenen Zeitreihen mit der Öffnungszeit des Balkens zu synchronisieren.

Im Gegensatz zu CopyRates, das den kompletten Satz von Zeitreihen als MqlRates-Array zurückgibt, erlaubt die CopySeries-Funktion dem Programmierer, nur die benötigten Zeitreihen in separaten Arrays zu erhalten. Dies kann durch die Angabe einer Kombination von Flags geschehen, um den Typ der Zeitreihe auszuwählen. Die Reihenfolge der an die Funktion übergebenen Arrays muss mit der Reihenfolge der Felder in der MqlRates Struktur übereinstimmen:

```
struct MqlRates
  {
   datetime time;         // Startzeit des Intervalls
   double   open;         // Eröffnungspreis
   double   high;         // Hoch des Balkens
   double   low;          // Tief des Balkens
   double   close;        // Schlusskurs
   long     tick_volume;  // Tick-Volumen
   int      spread;       // Spread
   long     real_volume;  // Handelsplatzvolumen
  }
```

Wenn Sie also die Zeitreihenwerte von *time*, *close* und *real_volume* für die letzten 100 Balken des aktuellen Symbol/Zeitrahmens benötigen, sollten Sie den folgenden Aufruf verwenden:

```
datetime time[];
double   close[];
long     volume[];
CopySeries(NULL,0,0,100,COPY_RATES_TIME|COPY_RATES_CLOSE|COPY_RATES_VOLUME_REAL,time,c
```

Beachte die Reihenfolge der Arrays "*time*, *close*, *volume*" — sie muss mit der Reihenfolge der Felder in der Struktur MqlRates übereinstimmen. Die Reihenfolge der Werte in der *rates_mask* spielt keine Rolle. Die Maske könnte auch folgendermaßen aussehen:

```
COPY_RATES_VOLUME_REAL|COPY_RATES_TIME|COPY_RATES_CLOSE
```

Beispiel:

```
//--- Eingabeparameter
input datetime InpDateFrom=D'2022.01.01 00:00:00';
input datetime InpDateTo  =D'2023.01.01 00:00:00';
input uint     InpCount   =20;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Arrays zum Abrufen von Zeitreihen aus der MqlRates-Preisstruktur
   double   open[];
   double   close[];
   float    closef[];
```

© 2000-2025, MetaQuotes Ltd.

```
   datetime time1[], time2[];
//--- Abfrage der Schlusskurse in einem Double-Array
   ResetLastError();
   int res1=CopySeries(NULL, PERIOD_CURRENT, 0, InpCount,
                       COPY_RATES_TIME|COPY_RATES_CLOSE, time1, close);
   PrintFormat("1. CopySeries  liefert %d values. Fehlercode=%d", res1, GetLastError()
   ArrayPrint(close);

//--- jetzt auch die Eröffnungspreise abrufen; Float-Array für Schlusskurse verwenden
   ResetLastError();
   int res2=CopySeries(NULL, PERIOD_CURRENT, 0, InpCount,
                       COPY_RATES_TIME|COPY_RATES_CLOSE|COPY_RATES_OPEN, time2, open,
   PrintFormat("2. CopySeries  liefert %d values. Fehlernummer=%d", res2, GetLastErro
   ArrayPrint(closef);
//--- Vergleich der erhaltenen Daten
   if((res1==res2) && (time1[0]==time2[0]))
     {
      Print("  | Time             |    Open      | Close double | Close float |");
      for(int i=0; i<10; i++)
        {
         PrintFormat("%d | %s |   %.5f    |   %.5f    |   %.5f   |",
                     i, TimeToString(time1[i]), open[i], close[i], closef[i]);
        }
     }
//--- Ergebnis
      1. CopySeries  liefert 20 Werte. Fehlernummer=0
      [ 0] 1.06722 1.06733 1.06653 1.06520 1.06573 1.06649 1.06694 1.06675 1.06684 1.(
      [10] 1.06514 1.06557 1.06456 1.06481 1.06414 1.06394 1.06364 1.06386 1.06239 1.(
      2. CopySeries  liefert 20 Werte. Fehlernummer=0
      [ 0] 1.06722 1.06733 1.06653 1.06520 1.06573 1.06649 1.06694 1.06675 1.06684 1.(
      [10] 1.06514 1.06557 1.06456 1.06481 1.06414 1.06394 1.06364 1.06386 1.06239 1.(
        | Time             |    Open      | Close double | Close float |
      0 | 2023.03.01 17:00 |   1.06660    |   1.06722    |   1.06722   |
      1 | 2023.03.01 18:00 |   1.06722    |   1.06733    |   1.06733   |
      2 | 2023.03.01 19:00 |   1.06734    |   1.06653    |   1.06653   |
      3 | 2023.03.01 20:00 |   1.06654    |   1.06520    |   1.06520   |
      4 | 2023.03.01 21:00 |   1.06520    |   1.06573    |   1.06573   |
      5 | 2023.03.01 22:00 |   1.06572    |   1.06649    |   1.06649   |
      6 | 2023.03.01 23:00 |   1.06649    |   1.06694    |   1.06694   |
      7 | 2023.03.02 00:00 |   1.06683    |   1.06675    |   1.06675   |
      8 | 2023.03.02 01:00 |   1.06675    |   1.06684    |   1.06684   |
      9 | 2023.03.02 02:00 |   1.06687    |   1.06604    |   1.06604   |
//---
  }
```

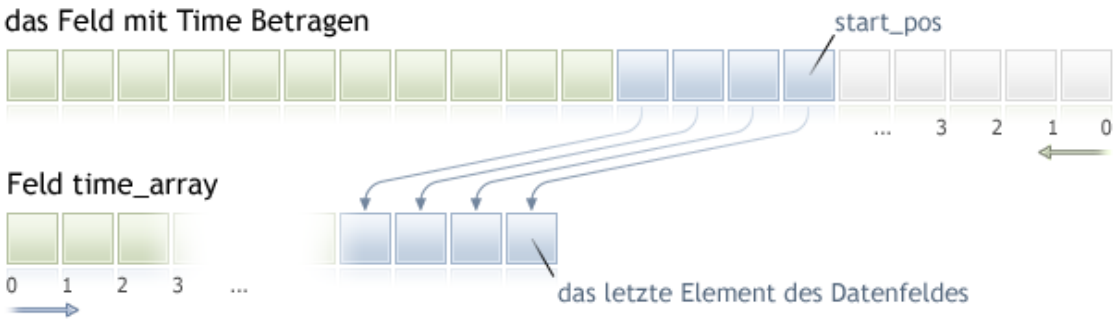
Siehe auch

Strukturen und Klassen, CopyRates

© 2000-2025, MetaQuotes Ltd.

CopyTime

Funktion erhaelt im Feld time_array historische Daten der Eroeffnungszeit der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist als ein Feld enthalten kann, versucht man, ein Feld so zu verteilen, dass angeforderte Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden. .

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyTime(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   int               start_pos,        // Anfangsposition
   int               count,            // Anzahl für Kopieren
   datetime          time_array[]      // Feld für Kopieren der Eroeffnungszeit
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyTime(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   datetime          start_time,       // Anfangsdatum
   int               count,            // Anzahl für Kopieren
   datetime          time_array[]      // Feld für Kopieren der Eroeffnungszeit
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopyTime(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   datetime         time_array[]      // Feld für Kopieren der Eroeffnungszeit
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

time_array[]

[out] Feld des Typs datetime.

Rückgabewert

Anzahl der kopierten Elemente des Feldes -1beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

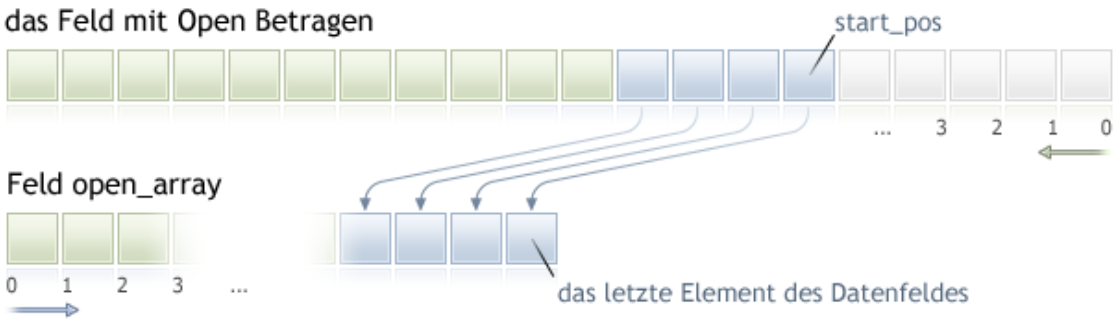
Sehen Sie das Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopyOpen

Funktion erhaelt im Feld open_array historische Daten der Eroeffnungszeit der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyOpen(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   int               start_pos,        // Anfangsposition
   int               count,            // Anzahl für Kopieren
   double            open_array[]      // Feld für Kopieren der Eroeffnungspreise
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyOpen(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   datetime          start_time,       // Anfansdatum
   int               count,            // Anzahl für Kopieren
   double            open_array[]      // Feld für Kopieren der Eroeffnungspreise
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopyOpen(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   double           open_array[]      // Feld für Kopieren der Eroeffnungspreise
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Zeit der Bar, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

open_array[]

[out] Feld des Typs double.

Rückgabewert

Anzahl der kopierten Elemente oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

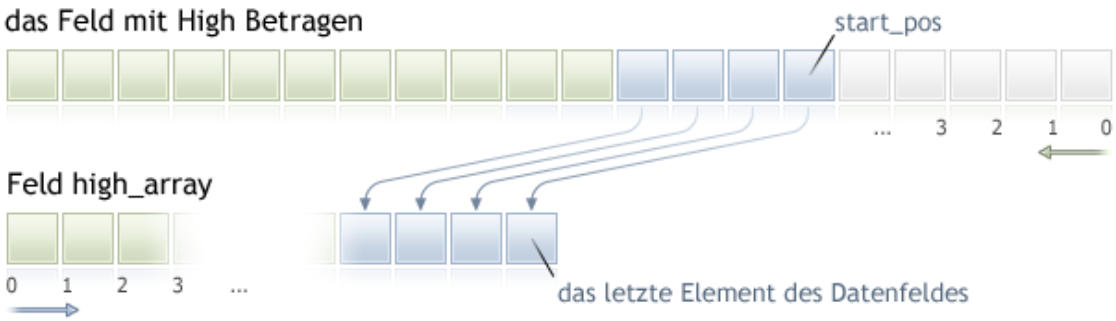
Sehen Sie das Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopyHigh

Funktion erhaelt im Feld high_array historische Daten der Eroeffnungszeit der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyHigh(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   int               start_pos,        // Anfangsdatum
   int               count,            // Anzahl für Kopieren
   double            high_array[]      // Feld für Kopieren der maximalen Preise
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyHigh(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   datetime          start_time,       // Anfangsdatum
   int               count,            // Anzahl für Kopieren
   double            high_array[]      // Feld für Kopieren der maximalen Preise
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopyHigh(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   double           high_array[]      // Feld für Kopieren der maximalen Preise
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

high_array[]

[out] Feld des Typs double.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

Beispiel:

```
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Beispiel der Ausgabe der Werte High[i] und Low[i]"
#property description "für Bars, die zufaellig gewaehlt wurden"

double High[],Low[];
//+------------------------------------------------------------------+
//| Bekommen wir Low für die angegebene Nummer der Bar               |
//+------------------------------------------------------------------+
double iLow(string symbol,ENUM_TIMEFRAMES timeframe,int index)
  {
   double low=0;
   ArraySetAsSeries(Low,true);
   int copied=CopyLow(symbol,timeframe,0,Bars(symbol,timeframe),Low);
   if(copied>0 && index<copied) low=Low[index];
   return(low);
  }
//+------------------------------------------------------------------+
//| Bekommen wir High für die angegebene Nummer der Bar              |
//+------------------------------------------------------------------+
double iHigh(string symbol,ENUM_TIMEFRAMES timeframe,int index)
  {
   double high=0;
   ArraySetAsSeries(High,true);
   int copied=CopyHigh(symbol,timeframe,0,Bars(symbol,timeframe),High);
   if(copied>0 && index<copied) high=High[index];
   return(high);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- geben wir für jeden Tick die Werte High und Low für Bar mit Index aus,
//--- der einer Ticksekunde gleich ist
   datetime t=TimeCurrent();
   int sec=t%60;
   printf("High[%d] = %G  Low[%d] = %G",
          sec,iHigh(Symbol(),0,sec),
          sec,iLow(Symbol(),0,sec));
  }
```

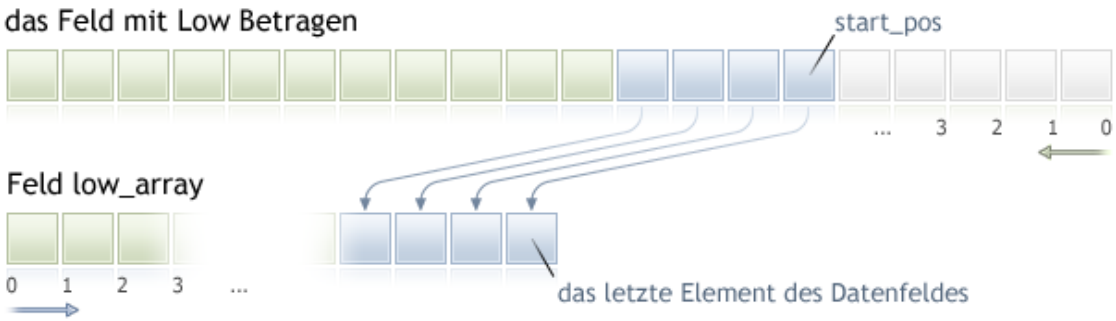
Sehen Sie ein umfassenderes Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopyLow

Funktion erhaelt im Feld low_array historische Daten der minimalen Preisen der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar. .



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Feld-Rezipient hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyLow(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   int               start_pos,        // Anfangsposition
   int               count,            // Anzahl für Kopieren
   double            low_array[]       // Feld für Kopieren der minimalen Preise
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyLow(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   datetime          start_time,       // Anfangsdatum
   int               count,            // Anzahl für Kopieren
   double            low_array[]       // Feld für Kopieren der minimalen Preise
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopyLow(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   double           low_array[]       // Feld für Kopieren der minimalen Preise
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

low_array[]

[out] Feld des Typs double.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

Sehen Sie das Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

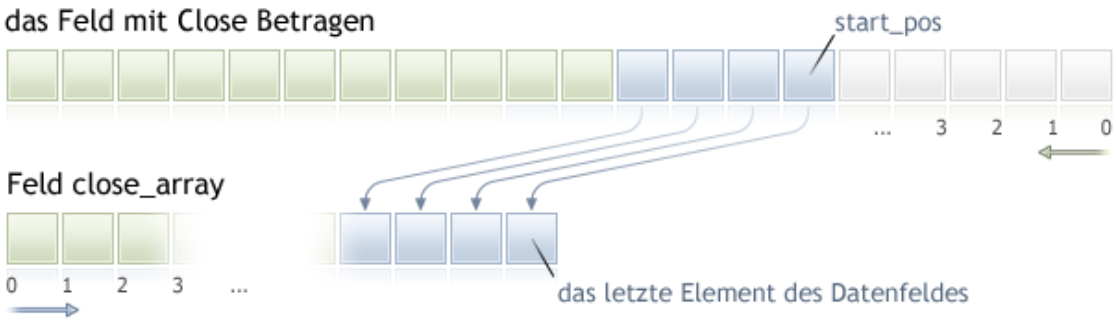
Sehen Sie auch

CopyHigh

© 2000-2025, MetaQuotes Ltd.

CopyClose

Funktion erhaelt im Feld close_array historische Daten der minimalen Preisen der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld, zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Feld-Rezipient hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion..

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyClose(
   string           symbol_name,       // Symbolname
   ENUM_TIMEFRAMES  timeframe,         // Periode
   int              start_pos,         // Startposition
   int              count,             // Anzahl für Kopieren
   double           close_array[]      // Feld für Kopieren der Schlusspreise
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyClose(
   string           symbol_name,       // Symbolname
   ENUM_TIMEFRAMES  timeframe,         // Periode
   datetime         start_time,        // Anfangsdatum
   int              count,             // Anzahl für Kopieren
   double           close_array[]      // Feld für Kopieren der Schlusspreise
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopyClose(
   string           symbol_name,       // Symbolname
   ENUM_TIMEFRAMES  timeframe,         // Periode
   datetime         start_time,        // Anfangsdatum
   datetime         stop_time,         // Beendigungsdatum
   double           close_array[]      // Feld für Kopieren der Schlusspreise
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

close_array[]

[out] Feld des Typs double.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

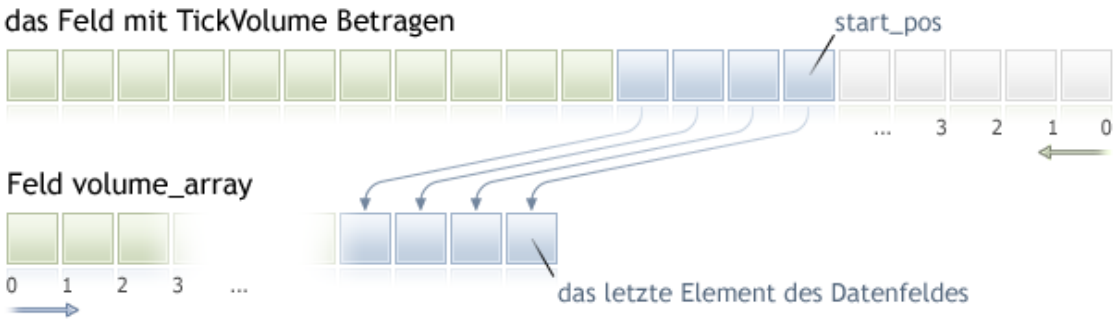
Sehen Sie das Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopyTickVolume

Funktion erhaelt im Feld volume_array historische Daten der minimalen Preisen der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch vetrteilten Puffer, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Feld-Rezipient hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyTickVolume(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   int              start_pos,        // Anfangsposition
   int              count,            // Anzahl für Kopieren
   long             volume_array[]    // Feld für Kopieren der Tickvolumen
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyTickVolume(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   int              count,            // Anzahl für Kopieren
   long             volume_array[]    // Feld für Kopieren der Tickvolumen
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

```
int  CopyTickVolume(
```

© 2000-2025, MetaQuotes Ltd.

```
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfnagsdatum
   datetime         stop_time,        // Beendigungsdatum
   long             volume_array[]    // Feld für Kopieren der Tickvolumen
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entpricht.

volume_array[]

[out] Feld des Typs long.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der

© 2000-2025, MetaQuotes Ltd.

Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

Beispiel:

```
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot TickVolume
#property indicator_label1  "TickVolume"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  C'143,188,139'
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input int      bars=3000;
//--- indicator buffers
double         TickVolumeBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,TickVolumeBuffer,INDICATOR_DATA);
   IndicatorSetInteger(INDICATOR_DIGITS,0);
//---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
```

© 2000-2025, MetaQuotes Ltd.

```
                  const double &low[],
                  const double &close[],
                  const long &tick_volume[],
                  const long &volume[],
                  const int &spread[])
  {
//---
   if(prev_calculated==0)
     {
      long timeseries[];
      ArraySetAsSeries(timeseries,true);
      int prices=CopyTickVolume(Symbol(),0,0,bars,timeseries);
      for(int i=0;i<rates_total-prices;i++) TickVolumeBuffer[i]=0.0;
      for(int i=0;i<prices;i++) TickVolumeBuffer[rates_total-1-i]=timeseries[prices-1-
      Print("Wir haben erhalten die folgende Anzahl der historischen Werte TickVolume:
     }
   else
     {
      long timeseries[];
      int prices=CopyTickVolume(Symbol(),0,0,1,timeseries);
      TickVolumeBuffer[rates_total-1]=timeseries[0];
    ;}
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

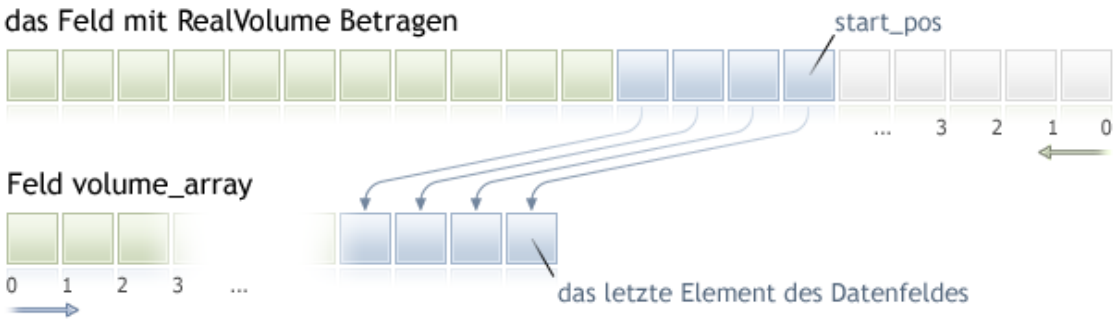
Sehen Sie ein umfassenderes Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopyRealVolume

Funktion erhaelt im Feld volume_array historische Daten der minimalen Preisen der Bars für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Beim Kopieren der im voraus unbekannten Anzahl der Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Umverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopyRealVolume(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   int              start_pos,        // Anfangsposition
   int              count,            // Anzahl für Kopieren
   long             volume_array[]    // Feld für Kopieren der Volumen
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopyRealVolume(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   int              count,            // Anzahl für Kopieren
   long             volume_array[]    // Feld für Kopieren der Volumen
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopyRealVolume(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   long             volume_array[]    // Feld für Kopieren der Volumen
   );
```

Parameter

symbol_name

[in] Symbol.

timeframe

[in] Periode.

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

volume_array[]

[out] Feld des Typs long.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

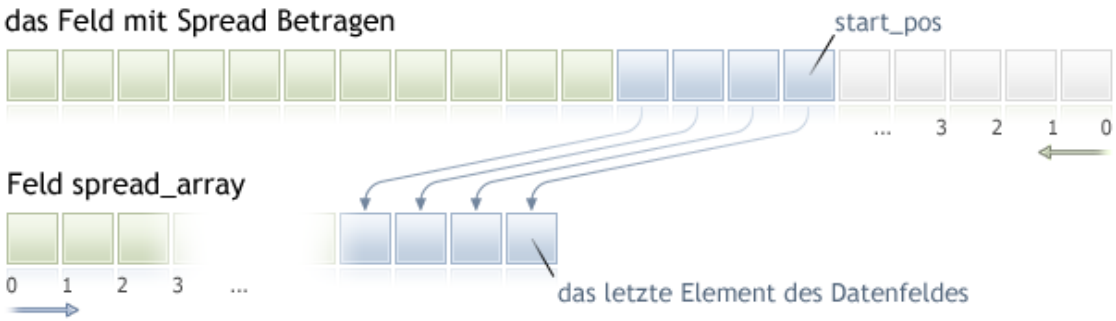
Sehen Sie das Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopySpread

Funktion bekommt im Feld spread_array historiche Daten der Spreads für das angegebene Paar Symbol-Periode in der angegebenen Menge. Es muss bemerkt werden, dass Abzählen der Elemente von der Startposition von der Gegenwart zur Vergangenheit durchgeführt wird, d.h. die Anfangsposition, 0, bedeutet die laufende Bar.



Bei Kopieren der im voraus unbekannten Anzahl von Daten ist es empfehlenswert, als Feld-Rezipient ein dynamisches Feld zu verwenden, denn wenn die Anzahl der Daten weniger (oder mehr) ist, als ein Feld enthalten kann, versucht man das Feld so zu verteilen, dass die angeforderten Daten vollstaendig hineinpassen.

Wenn man die im voraus bekannte Anzahl der Daten kopieren muss, ist es besser dafür einen statisch verteilten Puffer zu verwenden, um unnoetige Neuverteilung des Speichers zu vermeiden.

Es ist egal, welche Eigenschaft das Empfangsfeld hat - as_series=true oder as_series=false, Daten werden so kopiert, dass das aelteste Element am Anfang des physischen Speichers sein wird, der für das Feld verteilt wurde. Es gibt 3 Varianten der Funktion.

Aufruf nach Anfangsposition und Anzahl der angeforderten Elemente

```
int  CopySpread(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   int               start_pos,        // Anfangsposition
   int               count,            // Zahl für Kopieren
   int               spread_array[]    // Feld für Kopieren der Spreads
   );
```

Aufruf nach Anfangsdatum und Anzahl der angeforderten Elemente

```
int  CopySpread(
   string            symbol_name,      // Symbolname
   ENUM_TIMEFRAMES   timeframe,        // Periode
   datetime          start_time,       // Anfangsdatum
   int               count,            // Anzahl für Kopieren
   int               spread_array[]    // Feld für Kopieren der Spreads
   );
```

Aufruf nach Anfangs- und Beendigungsdatum des angeforderten Zeitintervals

© 2000-2025, MetaQuotes Ltd.

```
int  CopySpread(
   string           symbol_name,      // Symbolname
   ENUM_TIMEFRAMES  timeframe,        // Periode
   datetime         start_time,       // Anfangsdatum
   datetime         stop_time,        // Beendigungsdatum
   int              spread_array[]    // Feld für Kopieren der Spreads
   );
```

Parameter

symbol_name

[in] Symbol .

timeframe

[in] Periode

start_pos

[in] Nummer des ersten kopierten Elementes.

count

[in] Anzahl der kopierten Elemente.

start_time

[in] Barzeit, die dem ersten Element entspricht.

stop_time

[in] Barzeit, die dem letzten Element entspricht.

spread_array[]

[out] Feld des Typs int.

Rückgabewert

Anzahl der kopierten Elemente des Feldes oder -1 beim Fehler.

Hinweis

Wenn das Interval der angeforderten Daten ausser den zugänglichen Daten auf dem Server ist, kehrt die Funktion -1 zurück. Falls die Daten ausserhalb TERMINAL_MAXBARS angefordert werden (maximale Anzahl der Bars auf dem Chart) gibt die Funktion auch -1 zurück.

Wenn die angeforderten Timeserien noch nicht gebildet sind oder müssen vom Server geladen werden, kehrt die Funktion sofort bei der Anforderung der Daten aus dem Indikator -1 zurück, aber dabei wird das Prozess der Ladung/Bildung initialisiert.

Bei der Anforderung der Daten aus dem Experten oder Script wird die Ladung vom Server initialisiert, wenn dieses Terminal keine lokale Daten hat oder Bildung der notwendigen Timeserie fängt an, wenn Daten aus der lokalen Geschichte gebildet werden kann, aber sie sind noch nicht fertig. Funktion kehrt diese Anzahl der Daten, die zum Augenblick des Timeoutsablaufs fertig sind, aber Ladung der Geschichte wird fortsetzen und bei der weiteren analogischen Anforderung kehrt die Funktion schon mehr Daten zurück.

Bei der Anforderung der Daten nach Anfangsdatum und Anzahl der angeforderten Elemente kehren nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei

© 2000-2025, MetaQuotes Ltd.

wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time), ist immer gleich oder kleiner als das angegebene Datum.

Bei der Anforderung der Daten im vorgegebenen Datenbereich kehren nur die Daten zurück, die in das angeforderte Interval kommen, dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar, für die der Wert zurückgegeben wird (Volumen, Spread, Wert im Indikatorpuffer, Preis Open, High, Low, Close oder Zeit der Öffnung Time) befindet sich immer im angeforderten Interval.

So wenn der laufende Wochentag Samstag ist, kehrt die Funktion beim Versuch, Daten in dem wöchentlichen Timeframe mit Andeutung *start_time=Letzter_Dienstag* und *stop_time=Letzter_Freitag* zu kopieren, 0 zurück, denn die Eröffnungszeit im wöchentlichen Timeframe ist immer der Sonntag, aber keine angegebene Bar gerät in den angegebenen Bereich.

Wenn man muss, den ersten Wert zu erhalten, der der laufenden nicht beendeten Bar entspricht, kann man die erste Aufrufform mit der Andeutung *start_pos*=0 und *count*=1 verwenden.

Beispiel:

```
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Spread
#property indicator_label1  "Spread"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input int      bars=3000;
//--- indicator buffers
double         SpreadBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,SpreadBuffer,INDICATOR_DATA);
   IndicatorSetInteger(INDICATOR_DIGITS,0);
//---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
```

© 2000-2025, MetaQuotes Ltd.

```
                  const double &high[],
                  const double &low[],
                  const double &close[],
                  const long &tick_volume[],
                  const long &volume[],
                  const int &spread[])
  {
//---
   if(prev_calculated==0)
     {
      int spread_int[];
      ArraySetAsSeries(spread_int,true);
      int spreads=CopySpread(Symbol(),0,0,bars,spread_int);
      Print("Wir haben die folgende Anzahl der historischen Werte des Spreads bekommen
      for (int i=0;i<spreads;i++)
       {
       SpreadBuffer[rates_total-1-i]=spread_int[i];
       if(i<=30) Print("spread["+i+"] = ",spread_int[i]);
       }
     }
   else
     {
      double Ask,Bid;
      Ask=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
      Bid=SymbolInfoDouble(Symbol(),SYMBOL_BID);
      Comment("Ask = ",Ask,"  Bid = ",Bid);
      SpreadBuffer[rates_total-1]=(Ask-Bid)/Point();
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Sehen Sie ein umfassenderes Beispiel von Anfrage der historischen Daten in Methode der Objektbindung. Der Skript zeigt, wie Daten von Indikator iFractals an den letztem 1000 Balken zu erhalten und wie dann die letzten 10 Fraktale nach oben und 10 Fraktale nach unten auf dem Chart anzuzeigen. Sie können eine ähnliche Technik für alle Indikatoren, die fehlende Werte haben und typischerweise unter Verwendung der folgenden Stil hergestellt werden, verwenden:

- DRAW_SECTION,
- DRAW_ARROW,
- DRAW_ZIGZAG,
- DRAW_COLOR_SECTION,
- DRAW_COLOR_ARROW,
- DRAW_COLOR_ZIGZAG.

© 2000-2025, MetaQuotes Ltd.

CopyTicks

Die Funktion nimmt Ticks im Dateiformat MqlTick in ticks_array entgegen, die Indizierung erfolgt von der Vergangenheit in die Gegenwart, d.h. der Tick mit dem Index 0 ist der älteste im Array. Für die Analyse eines Ticks muss das Feld *flags* überprüft werden, welches Änderungen in diesem Tick anzeigt.

```
int  CopyTicks(
   string           symbol_name,           // Symbolname
   MqlTick&         ticks_array[],         // Array für das Empfangen von Ticks
   uint             flags=COPY_TICKS_ALL,  // Flagge, die den Typ von Ticks bestimmt
   ulong            from=0,                // das Datum, von dem an Ticks abgerufen we
   uint             count=0                // Anzahl der Ticks, die entgegengenommen v
   );
```

Parameter

symbol_name

[in] Symbol.

ticks_array

[out] Array vom Typ MqlTick für das Entgegennehmen von Ticks.

flags

[in] Flagge, die den Typ der angeforderten Ticks bestimmt. COPY_TICKS_INFO - Ticks, die durch Änderungen von Bid bzw. Ask verursacht wurden, COPY_TICKS_TRADE - Ticks mit Änderungen von Last und Volume, COPY_TICKS_ALL - alle Ticks. Bei jedem Anforderungstyp werden die Werte des vorherigen Ticks in die übrigen Felder der MqlTick Struktur geschrieben.

from

[in] Das Datum, von dem an Ticks abgerufen werden. Das Datum wird vom 01.01.1970 in Millisekunden angegeben. Wenn der Parameter *from*=0, dann werden die letzten *count* Ticks übermittelt.

count

[in] Anzahl angeforderter Ticks. Wenn die Parameter from und count nicht angegeben sind, dann werden alle vorhandenen Ticks im Array ticks_array[] gespeichert, aber nicht mehr als 2000.

Rückgabewert

Anzahl der kopierten Ticks oder -1 im Fehlerfall.

Hinweis

Die CopyTicks() Funktion erlaubt es, alle eingegangenen Ticks abzurufen und zu analysieren. Der erste Aufruf von CopyTicks() initiiert die Synchronisierung der Datenbank von Ticks, welche für jedes Symbol auf der Festplatte gespeichert sind. Wenn die Ticks in der lokalen Datenbank nicht ausreichend sind, dann werden fehlende Ticks vom Handelsserver heruntergeladen. Dabei werden die Ticks vom Datum ***from*** an, angegeben in CopyTicks(), bis zum aktuellen Moment synchronisiert. Danach werden alle Ticks für dieses Symbol in die Datenbank eingehen und diese in dem aktuellen synchronisierten Zustand aufrechterhalten.

© 2000-2025, MetaQuotes Ltd.

Wenn die Parameter *from* und *count* nicht angegeben sind, werden alle vorhandenen Ticks im Array *ticks_array[]* gespeichert, aber nicht mehr als 2000. Der *flags* Parameter erlaubt es, den Typ der benötigten Ticks zu setzen.

COPY_TICKS_INFO – es werden die Ticks übermittelt, in welchen es Änderungen des Bid bzw. Ask Preises gibt. Dabei werden auch andere Felder ausgefüllt, z.B. wenn sich nur der Bid-Preis geändert hat, werden in den *ask* und *volume* Feldern die letzten bekannten Werte gespeichert. Um genau zu erfahren, was sich geändert hat, muss das Feld *flags*analysiert werden, das den Wert TICK_FLAG_BID bzw. TICK_FLAG_ASK haben wird. Wenn die Bid und Ask Preise des Ticks gleich Null sind, und die Flaggen dabei anzeigen, dass sich die Preisdaten geändert haben (*flags*=TICK_FLAG_BID|TICK_FLAG_ASK), weist das darauf hin, dass die Markttiefe leer ist. Mit anderen Worten, sind in diesem Moment weder Kauf- noch Verkaufsanfragen vorhanden.

COPY_TICKS_TRADE – es werden die Ticks übermittelt, in welchen es Änderungen von Last Price und des Volumens gibt. Aber dabei werden auch andere Felder ausgefüllt, d.h. in den Bid und Ask Feldern werden die letzten bekannten Werte gespeichert. Um genau zu erfahren, was sich geändert hat, muss das Feld *flags* analysiert werden, das den Wert TICK_FLAG_LAST und TICK_FLAG_VOLUME haben wird.

COPY_TICKS_ALL – alle Ticks werden übermittelt, in welchen es irgendeine Änderungen gibt. In den Feldern ohne Anderungen werden die letzten bekannten Werte gespeichert.

Der Aufruf von CopyTicks() mit der Flagge COPY_TICKS_ALL gibt gleichzeitig alle Ticks aus dem angefragten Intervall aus, während der Aufruf in anderen Modi einige Zeit für die Vorbereitung und Auswahl von Ticks, deswegen ist die Geschwindigkeit der Ausführung nicht besonders schneller.

Beim Abruf von Ticks (egal ob COPY_TICKS_INFO oder COPY_TICKS_TRADE) beinhaltet jeder Tick vollständige Preisinformationen im Moment des Ticks (*bid*, *ask*, *last* und *volume*). Das trägt zur Einfachheit der Analyse der Handelsumgebung im Moment jeden Ticks bei, damit man nicht jedes Mal eine tiefe Tick-Historie abrufen und Werte nach anderen Feldern in dieser Historie suchen muss.

In Indikatoren liefert die CopyTicks() Funktion das Ergebnis umgehend: Beim Aufruf aus dem Indikator gibt CopyTick() alle zugänglichen Ticks zurück und startet die Synchronisierung der Tick-Datenbank, wenn es nicht ausreichend Daten gibt. Alle Indikatoren arbeiten in einem gemeinsamen Thread auf einem Symbol, deswegen darf der Indikator nicht auf das Ende der Synchronisierung warten. Nach dem die Synchronisierung abgeschlossen ist, liefert CopyTicks() beim nächsten Aufruf alle angeforderten Ticks. Die OnCalculate() Funktion wird nach dem Eingehen jeden Ticks in den Indikatoren aufgerufen.

In Experten und Skripts kann die CopyTicks() Funktion bis zu 45 Sekunden auf das Ergebnis warten: Im Gegensatz zum Indikator arbeitet jeder Expert Advisor und Skript in einem eigenen Thread, deswegen kann er bis zu 45 Sekunden auf das Ende der Synchronisierung warten. Wenn die Ticks innerhalb dieser Zeit nicht im benötigten Volumen synchronisiert werden, liefert CopyTicks() nur vorhandene Ticks nach Time Out, dabei wird die Synchronisierung fortgesetzt. Die OnTick() Funktion dient nicht als Jeder-Tick-Handler in Expert Advisors, sie informiert den Expert Advisor über Änderungen auf dem Markt. Es kann sich um Paket-Änderungen handeln: mehrere Ticks können gleichzeitig ins Terminal eingehen, aber die OnTick() Funktion wird nur einmal aufgerufen, um den Expert Advisor über den letzten Zustand des Markts zu informieren.

Die Geschwindigkeit: das Terminal speichert die letzten 4096 Ticks für jeden Symbol im Cache für schnellen Zugriff (für Symbole mit der Markttiefe – 65536 Ticks), diese Daten werden am schnellsten abgerufen. Beim Abruf von Ticks der aktuellen Handelssitzung außerhalb des Cache greift CopyTicks() auf die Ticks im Speicher des Terminals zu. Das nimmt mehr Zeit in Anspruch. Am

© 2000-2025, MetaQuotes Ltd.

langsamsten werden die Ticks für andere Tage abgerufen, denn in diesem Fall werden die Daten von der Festplatte gelesen.

Beispiel:

```
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property script_show_inputs
//--- Requesting 100 million ticks to be sure we receive the entire tick history
input int      getticks=100000000; // The number of required ticks
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   int     attempts=0;     // Count of attempts
   bool    success=false;  // The flag of a successful copying of ticks
   MqlTick tick_array[];   // Tick receiving array
   MqlTick lasttick;       // To receive last tick data
   SymbolInfoTick(_Symbol,lasttick);
//--- Make 3 attempts to receive ticks
   while(attempts<3)
     {
      //--- Measuring start time before receiving the ticks
      uint start=GetTickCount();
//--- Requesting the tick history since 1970.01.01 00:00.001 (parameter from=1 ms)
      int received=CopyTicks(_Symbol,tick_array,COPY_TICKS_ALL,1,getticks);
      if(received!=-1)
        {
         //--- Showing information about the number of ticks and spent time
         PrintFormat("%s: received %d ticks in %d ms",_Symbol,received,GetTickCount()-
         //--- If the tick history is synchronized, the error code is equal to zero
         if(GetLastError()==0)
           {
            success=true;
            break;
           }
         else
            PrintFormat("%s: Ticks are not synchronized yet, %d ticks received for %d
            _Symbol,received,GetTickCount()-start,_LastError);
        }
      //--- Counting attempts
      attempts++;
      //--- A one-second pause to wait for the end of synchronization of the tick data
      Sleep(1000);
     }
//--- Receiving the requested ticks from the beginning of the tick history failed in t
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!success)
     {
      PrintFormat("Error! Failed to receive %d ticks of %s in three attempts",getticks
      return;
     }
   int ticks=ArraySize(tick_array);
//--- Showing the time of the first tick in the array
   datetime firstticktime=tick_array[ticks-1].time;
   PrintFormat("Last tick time = %s.%03I64u",
               TimeToString(firstticktime,TIME_DATE|TIME_MINUTES|TIME_SECONDS),tick_ar
//--- Zeit des letzten Ticks im Array ausgeben
   datetime lastticktime=tick_array[0].time;
   PrintFormat("First tick time = %s.%03I64u",
               TimeToString(lastticktime,TIME_DATE|TIME_MINUTES|TIME_SECONDS),tick_ar

//---
   MqlDateTime today;
   datetime current_time=TimeCurrent();
   TimeToStruct(current_time,today);
   PrintFormat("current_time=%s",TimeToString(current_time));
   today.hour=0;
   today.min=0;
   today.sec=0;
   datetime startday=StructToTime(today);
   datetime endday=startday+24*60*60;
   if((ticks=CopyTicksRange(_Symbol,tick_array,COPY_TICKS_ALL,startday*1000,endday*100
     {
      PrintFormat("CopyTicksRange(%s,tick_array,COPY_TICKS_ALL,%s,%s) failed, error %d
                  _Symbol,TimeToString(startday),TimeToString(endday),GetLastError());
      return;
     }
   ticks=MathMax(100,ticks);tday+24*60*60;
//--- Showing the first 100 ticks of the last day
   int counter=0;
   for(int i=0;i<ticks;i++)
     {
      datetime time=tick_array[i].time;
      if((time>=startday) && (time<endday) && counter<100)
        {
         counter++;
         PrintFormat("%d. %s",counter,GetTickDescription(tick_array[i]));
        }
     }
//--- Showing the first 100 deals of the last day
   counter=0;
   for(int i=0;i<ticks;i++)
     {
      datetime time=tick_array[i].time;
      if((time>=startday) && (time<endday) && counter<100)
```

© 2000-2025, MetaQuotes Ltd.

```
         {
          if(((tick_array[i].flags&TICK_FLAG_BUY)==TICK_FLAG_BUY) || ((tick_array[i].fl
            {
             counter++;
             PrintFormat("%d. %s",counter,GetTickDescription(tick_array[i]));
            }
         }
      }
   }
//+------------------------------------------------------------------+
//| Returns the string description of a tick                         |
//+------------------------------------------------------------------+
string GetTickDescription(MqlTick &tick)
   {
    string desc=StringFormat("%s.%03d ",
                             TimeToString(tick.time),tick.time_msc%1000);
//--- Checking flags
    bool buy_tick=((tick.flags&TICK_FLAG_BUY)==TICK_FLAG_BUY);
    bool sell_tick=((tick.flags&TICK_FLAG_SELL)==TICK_FLAG_SELL);
    bool ask_tick=((tick.flags&TICK_FLAG_ASK)==TICK_FLAG_ASK);
    bool bid_tick=((tick.flags&TICK_FLAG_BID)==TICK_FLAG_BID);
    bool last_tick=((tick.flags&TICK_FLAG_LAST)==TICK_FLAG_LAST);
    bool volume_tick=((tick.flags&TICK_FLAG_VOLUME)==TICK_FLAG_VOLUME);
//--- Checking trading flags in a tick first
    if(buy_tick || sell_tick)
      {
       //--- Forming an output for the trading tick
       desc=desc+(buy_tick?StringFormat("Buy Tick: Last=%G Volume=%d ",tick.last,tick.v
       desc=desc+(sell_tick?StringFormat("Sell Tick: Last=%G Volume=%d ",tick.last,tick
       desc=desc+(ask_tick?StringFormat("Ask=%G ",tick.ask):"");
       desc=desc+(bid_tick?StringFormat("Bid=%G ",tick.ask):"");
       desc=desc+"(Trade tick)";
      }
    else
      {
       //--- Form a different output for an info tick
       desc=desc+(ask_tick?StringFormat("Ask=%G ",tick.ask):"");
       desc=desc+(bid_tick?StringFormat("Bid=%G ",tick.ask):"");
       desc=desc+(last_tick?StringFormat("Last=%G ",tick.last):"");
       desc=desc+(volume_tick?StringFormat("Volume=%d ",tick.volume):"");
       desc=desc+"(Info tick)";
      }
//--- Returning tick description
    return desc;
   }
//+------------------------------------------------------------------+
/* Example of the output
Si-12.16: received 11048387 ticks in 4937 ms
Last tick time = 2016.09.26 18:32:59.775
```

© 2000-2025, MetaQuotes Ltd.

```
First tick time = 2015.06.18 09:45:01.000
1.  2016.09.26 09:45.249 Ask=65370 Bid=65370 (Info tick)
2.  2016.09.26 09:47.420 Ask=65370 Bid=65370 (Info tick)
3.  2016.09.26 09:50.893 Ask=65370 Bid=65370 (Info tick)
4.  2016.09.26 09:51.827 Ask=65370 Bid=65370 (Info tick)
5.  2016.09.26 09:53.810 Ask=65370 Bid=65370 (Info tick)
6.  2016.09.26 09:54.491 Ask=65370 Bid=65370 (Info tick)
7.  2016.09.26 09:55.913 Ask=65370 Bid=65370 (Info tick)
8.  2016.09.26 09:59.350 Ask=65370 Bid=65370 (Info tick)
9.  2016.09.26 09:59.678 Bid=65370 (Info tick)
10. 2016.09.26 10:00.000 Sell Tick: Last=65367 Volume=3 (Trade tick)
11. 2016.09.26 10:00.000 Sell Tick: Last=65335 Volume=45 (Trade tick)
12. 2016.09.26 10:00.000 Sell Tick: Last=65334 Volume=95 (Trade tick)
13. 2016.09.26 10:00.191 Sell Tick: Last=65319 Volume=1 (Trade tick)
14. 2016.09.26 10:00.191 Sell Tick: Last=65317 Volume=1 (Trade tick)
15. 2016.09.26 10:00.191 Sell Tick: Last=65316 Volume=1 (Trade tick)
16. 2016.09.26 10:00.191 Sell Tick: Last=65316 Volume=10 (Trade tick)
17. 2016.09.26 10:00.191 Sell Tick: Last=65315 Volume=5 (Trade tick)
18. 2016.09.26 10:00.191 Sell Tick: Last=65313 Volume=3 (Trade tick)
19. 2016.09.26 10:00.191 Sell Tick: Last=65307 Volume=25 (Trade tick)
20. 2016.09.26 10:00.191 Sell Tick: Last=65304 Volume=1 (Trade tick)
21. 2016.09.26 10:00.191 Sell Tick: Last=65301 Volume=1 (Trade tick)
22. 2016.09.26 10:00.191 Sell Tick: Last=65301 Volume=10 (Trade tick)
23. 2016.09.26 10:00.191 Sell Tick: Last=65300 Volume=5 (Trade tick)
24. 2016.09.26 10:00.191 Sell Tick: Last=65300 Volume=1 (Trade tick)
25. 2016.09.26 10:00.191 Sell Tick: Last=65300 Volume=6 (Trade tick)
26. 2016.09.26 10:00.191 Sell Tick: Last=65299 Volume=1 (Trade tick)
27. 2016.09.26 10:00.191 Bid=65370 (Info tick)
28. 2016.09.26 10:00.232 Ask=65297 (Info tick)
29. 2016.09.26 10:00.276 Sell Tick: Last=65291 Volume=31 (Trade tick)
30. 2016.09.26 10:00.276 Sell Tick: Last=65290 Volume=1 (Trade tick)
*/
```

Sieh auch

SymbolInfoTick, Struktur für das Empfangen aktueller Preise, OnTick()

© 2000-2025, MetaQuotes Ltd.

CopyTicksRange

Die Funktion erhält Ticks vom Typ MqlTick aus einem bestimmten Zeitbereich ins Array ticks_array. Dabei erfolgt die Indizierung der Ticks von der Vergangenheit in die Gegenwart, d.h. das älteste Tick ist das Tick mit dem Index 0. Für die Analyse eines Ticks muss das Feld *flags* geprüft werden, in welchem Änderungen angezeigt werden.

```
int  CopyTicksRange(
   const string     symbol_name,           // Symbolname
   MqlTick&         ticks_array[],         // Array für das Erhalten von Ticks
   uint             flags=COPY_TICKS_ALL,  // Flag, das den Typ der Ticks definiert
   ulong            from_msc=0,            // Datum, von welchem die Ticks abgerufen w
   ulong            to_msc=0               // Datum, bis welchem die Ticks abgerufen w
   );
```

Parameter

symbol_name

[in] Ein Symbol.

ticks_array

[out] Ein statisches oder dynamisches Array MqlTick für das Erhalten von Ticks. Wenn das statische Array nicht alle abgerufenen Ticks aus dem angegebenen Bereich speichern kann, werden so viele Ticks aufgenommen, wie viele ins Array hineinpassen. Dabei generiert die Funktion den Fehler ERR_HISTORY_SMALL_BUFFER (4407).

flags

[in] Ein Flag, das den Typ der Ticks definiert. COPY_TICKS_INFO - Ticks mit Änderungen von Bid und/oder Ask, COPY_TICKS_TRADE - Ticks mit Änderungen von Last und Volume, COPY_TICKS_ALL - alle Ticks. Bei jedem Anfragetyp werden Werte des vorherigen Ticks in die Felder der MqlTick Struktur gespeichert.

from_msc

[in] Das Datum, von welchem Ticks abgerufen werden. Das Datum wird in Millisekunden vom 01.01.1970 angegeben. Wenn der Parameter *from_msc* nicht festgelegt wurde, werden alle Ticks vom Anfang der Historie zurückgegeben. Es werden Ticks mit einer Zeit >= from_msc zurückgegeben.

to_msc

[in] Das Datum, bis welchem Ticks abgerufen werden. Das Datum wird in Millisekunden vom 01.01.1970 angegeben. Es werden Ticks mit einer Zeit <= to_msc ins Array geschrieben. Wenn der Parameter *to_msc* nicht festgelegt wurde, werden alle Ticks bis zum Ende der Historie zurückgegeben.

Rückgabewert

Anzahl von kopierten Ticks oder -1 im Fehlerfall. GetLastError() kann folgende Fehler ausgeben:

- ERR_HISTORY_TIMEOUT - die Synchronisierungszeit ist abgelaufen, die Funktion gab alle vorhandenen Ticks zurück.
- ERR_HISTORY_SMALL_BUFFER - der statische Puffer ist zu klein, die Funktion gab so viele Ticks zurück, wie viele in das Array hineinpassten.

© 2000-2025, MetaQuotes Ltd.

- ERR_NOT_ENOUGH_MEMORY – nicht ausreichend Speicherplatz für das Erhalten der Historie aus dem angegebenen Bereich in das dynamische Array. Es ist nicht gelungen, Speicher für das Array zu reservieren.

Hinweis

Die CopyTicksRange() Funktion dient zum Abruf von Ticks aus einem strikt angegebenen Bereich, z.B. Tick für einen konkreten Tag der Historie, während CopyTicks() nur das Anfangsdatum angibt, z.B. - alle Ticks vom Anfang des Monats bis heute erhalten.

Siehe auch

SymbolInfoTick, Struktur für das Erhalten aktueller Preise, OnTick, CopyTicks

© 2000-2025, MetaQuotes Ltd.

iBars

Gibt die Anzahl der Balken in der Historie nach dem entsprechenden Symbol und Zeitrahmen zurück.

```
int  iBars(
   const string           symbol,          // Symbol
   ENUM_TIMEFRAMES        timeframe        // Zeitrahmen
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

Rückgabewert

Anzahl der Balken in der Historie nach dem entsprechenden Symbol und Zeitrahmen, aber nicht mehr als im Parameter "Max. bars in chart" in den Einstellungen der Plattform.

Beispiel:

```
  Print("Bar count on the 'EURUSD,H1' is ",iBars("EURUSD",PERIOD_H1));
```

Siehe auch

Bars

© 2000-2025, MetaQuotes Ltd.

iBarShift

Suche nach Balken nach Zeit. Die Funktion gibt den Index des Balkens zurück, der der angegebenen Zeit entspricht.

```
int  iBarShift(
   const string        symbol,          // Symbol
   ENUM_TIMEFRAMES     timeframe,       // Zeitrahmen
   datetime            time,            // Zeit
   bool                exact=false      // Modus
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. PERIOD_CURRENT bedeutet den Zeitrahmen des aktuellen Charts.

time

[in] Zeit für die Suche.

exact=false

[in] Rückgabewert, wenn kein Balken entsprechend der angegebenen Zeit gefunden wurde. Wenn *exact=false* gibt *iBarShift* den Index des nächsten Balkens zurück, dessen Eröffnungszeit kleiner als die angegebene Zeit ist (*time_open<time*). Wenn solcher Balken nicht gefunden wurde (keine Historie vor dem angegebenen Zeitpunkt), gibt die Funktion -1 zurück. Wenn *exact=true*, wird es nicht nach dem nächsten Balken gesucht, die Funktion iBarShift gibt sofort -1 zurück.

Rückgabewert

Index des Balkens, der der angegebenen Zeit entspricht. Wenn kein Balken für die angegebene Zeit vorhanden ist (eine Kurslücke in der Historie), gibt die Funktion -1 oder den Index des nächsten Balkens (je nach dem Parameter *exact*) zurück.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- das Datum fällt auf einen Sonntag
   datetime time=D'2002.04.25 12:00';
   string symbol="GBPUSD";
   ENUM_TIMEFRAMES tf=PERIOD_H1;
   bool exact=false;
//--- wenn es keinen Balken für die angegebene Zeit gibt, gibt iBarShift den Index des
   int bar_index=iBarShift(symbol,tf,time,exact);
//--- überprüfen wir den Fehlercode nach dem Aufruf von iBarShift()
```

© 2000-2025, MetaQuotes Ltd.

```
   int error=GetLastError();
   if(error!=0)
    {
      PrintFormat("iBarShift(): GetLastError=%d - das abgefragte Datum %s "+
                  "für %s %s in der verfügbaren Historie nicht gefunden",
                  error,TimeToString(time),symbol,EnumToString(tf));
      return;
     }
//--- die Funktion iBarShift() wurde erfolgreich ausgeführt, Ergebnisse für exact=fals
   PrintFormat("1. %s %s %s(%s): bar index is %d (exact=%s)",
               symbol,EnumToString(tf),TimeToString(time),
               DayOfWeek(time),bar_index,string(exact));
   datetime bar_time=iTime(symbol,tf,bar_index);
   PrintFormat("Time of bar #%d is %s (%s)",
               bar_index,TimeToString(bar_time),DayOfWeek(bar_time));
//--- Abfrage des Index des Balkens mit der angegebenen Zeit, wenn es keinen Balken gi
   exact=true;
   bar_index=iBarShift(symbol,tf,time,exact);
//--- die Funktion iBarShift() wurde erfolgreich ausgeführt, Ergebnisse für exact=true
   PrintFormat("2. %s %s %s (%s):bar index is %d (exact=%s)",
               symbol,EnumToString(tf),TimeToString(time)
               ,DayOfWeek(time),bar_index,string(exact));
  }
//+------------------------------------------------------------------+
//| Gibt den Wochentag zurück                                        |
//+------------------------------------------------------------------+
string DayOfWeek(const datetime time)
 {
   MqlDateTime dt;
   string day="";
   TimeToStruct(time,dt);
   switch(dt.day_of_week)
    {
      case 0: day=EnumToString(SUNDAY);
      break;
      case 1: day=EnumToString(MONDAY);
      break;
      case 2: day=EnumToString(TUESDAY);
      break;
      case 3: day=EnumToString(WEDNESDAY);
      break;
      case 4: day=EnumToString(THURSDAY);
      break;
      case 5: day=EnumToString(FRIDAY);
      break;
      default:day=EnumToString(SATURDAY);
      break;
     }
+//---+
```

© 2000-2025, MetaQuotes Ltd.

```
   return day;
  }
//+------------------------------------------------------------------+
/* Das Ergebnis
   1. GBPUSD PERIOD_H1 2018.06.10 12:00(SUNDAY): bar index is 64 (exact=false)
   Time of bar #64 is 2018.06.08 23:00 (FRIDAY)
   2. GBPUSD PERIOD_H1 2018.06.10 12:00 (SUNDAY):bar index is -1 (exact=true)
*/
```

© 2000-2025, MetaQuotes Ltd.

iClose

Gibt den Close-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
double  iClose(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Close-Preises des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyClose, CopyRates

© 2000-2025, MetaQuotes Ltd.

iHigh

Gibt den High-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
double  iHigh(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des High-Preises des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyHigh, CopyRates

© 2000-2025, MetaQuotes Ltd.

iHighest

Gibt den Index des größten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts zurück.

```
int  iHighest(
   const string        symbol,              // Symbol
   ENUM_TIMEFRAMES     timeframe,           // Zeitrahmen
   ENUM_SERIESMODE     type,                // Identifikator der Zeitreihe
   int                 count=WHOLE_ARRAY,   // Anzahl der Elemente
   int                 start=0              // Index
  );
```

Parameter

symbol

[in] Symbol, auf welchem es gesucht wird. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

type

[in] Identifikator der Zeitreihe, in welcher es gesucht wird. Einer der Werte von ENUM_SERIESMODE.

count=WHOLE_ARRAY

[in] Anzahl der Elemente der Zeitreihe (vom aktuellen Balken, Index aufsteigend), unter welchen es gesucht werden muss.

start=0

[in] Index (Verschiebung relativ zum aktuellen Balken) des anfänglichen Balkens, mit welchem die Suche beginnt. Negative Werte werden ignoriert und durch Null ersetzt.

Rückgabewert

Index des höchsten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts oder -1 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Beispiel:

```
   double val;
//--- Berechnung des maximalen Close-Balkens auf 20 aufeinanderfolgenden Balken
//--- vom Index 4 bis zum Index 23 einschließlich auf dem aktuellen Chart
   int val_index=iHighest(NULL,0,MODE_CLOSE,20,4);
   if(val_index!=-1)
      val=High[val_index];
   else
      PrintFormat("Fehler beim Aufruf von iHighest(). Fehlercode=%d",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

iLow

Gibt den Low-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
double  iLow(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Low-Preises des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyLow, CopyRates

© 2000-2025, MetaQuotes Ltd.

iLowest

Gibt den Index des kleinsten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts zurück.

```
int  iLowest(
   const string        symbol,              // Symbol
   ENUM_TIMEFRAMES     timeframe,           // Zeitrahmen
   ENUM_SERIESMODE     type,                // Identifikator der Zeitreihe
   int                 count=WHOLE_ARRAY,   // Anzahl der Elemente
   int                 start=0              // Index
  );
```

Parameter

symbol

[in] Symbol, auf welchem es gesucht wird. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

type

[in] Identifikator der Zeitreihe, in welcher es gesucht wird. Einer der Werte von ENUM_SERIESMODE.

count=WHOLE_ARRAY

[in] Anzahl der Elemente der Zeitreihe (vom aktuellen Balken, Index aufsteigend), unter welchen es gesucht werden muss.

start=0

[in] Index des (Verschiebung relativ zum aktuellen Balken) anfänglichen Balken, mit welchem die Suche nach dem kleinsten Wert beginnt. Negative Werte werden ignoriert und durch Null ersetzt.

Rückgabewert

Index des kleinsten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts oder -1 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Beispiel:

```
   double val;
//--- Suche nach dem Balken mit dem minimalen Wert des echten Volumens auf 15 aufeinan
//--- vom Index 10 bis 24 einschließlich auf dem aktuellen Chart
   int val_index=iLowest(NULL,0,MODE_REAL_VOLUME,15,10);
   if(val_index!=-1)
      val=Low[val_index];
   else
      PrintFormat("Fehler beim Aufruf von iLowest(). Fehlercode=%d",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

iOpen

Gibt den Open-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
double  iOpen(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Eröffnungspreises des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyOpen, CopyRates

© 2000-2025, MetaQuotes Ltd.

iTime

Gibt den Zeitpunkt der Eröffnung eines Balkens (der mithilfe des Parameters shift vorgegeben wurde) des entsprechenden Charts zurück.

```
datetime  iTime(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Open-Preises (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- das Datum fällt auf einen Sonntag
   datetime time=D'2018.06.10 12:00';
   string symbol="GBPUSD";
   ENUM_TIMEFRAMES tf=PERIOD_H1;
   bool exact=false;
//--- es gibt keinen Bar für die angegebene Zeit, daher gibt iBarShift den Index des r
```

© 2000-2025, MetaQuotes Ltd.

```
   int bar_index=iBarShift(symbol,tf,time,exact);
   PrintFormat("1. %s %s %s(%s): bar index is %d (exact=%s)",
               symbol,EnumToString(tf),TimeToString(time),DayOfWeek(time),bar_index,st
   datetime bar_time=iTime(symbol,tf,bar_index);
   PrintFormat("Time of bar #%d is %s (%s)",
               bar_index,TimeToString(bar_time),DayOfWeek(bar_time));
//PrintFormat(iTime(symbol,tf,bar_index));
//--- Abfrage des Index des Bars mit der angegebenen Zeit, aber es gibt keinen Balken
   exact=true;
   bar_index=iBarShift(symbol,tf,time,exact);
   PrintFormat("2. %s %s %s (%s):bar index is %d (exact=%s)",
               symbol,EnumToString(tf),TimeToString(time),DayOfWeek(time),bar_index,st
  }
//+------------------------------------------------------------------+
//| Gibt den Wochentag zurück                                        |
//+------------------------------------------------------------------+
string DayOfWeek(const datetime time)
 {
   MqlDateTime dt;
   string day="";
   TimeToStruct(time,dt);
   switch(dt.day_of_week)
    {
      case 0: day=EnumToString(SUNDAY);
      break;
      case 1: day=EnumToString(MONDAY);
      break;
      case 2: day=EnumToString(TUESDAY);
      break;
      case 3: day=EnumToString(WEDNESDAY);
      break;
      case 4: day=EnumToString(THURSDAY);
      break;
      case 5: day=EnumToString(FRIDAY);
      break;
      default:day=EnumToString(SATURDAY);
      break;
     }
+//---+
   return day;
  }
/* Das Ergebnis:
   1. GBPUSD PERIOD_H1 2018.06.10 12:00(SUNDAY): bar index is 64 (exact=false)
   Time of bar #64 is 2018.06.08 23:00 (FRIDAY)
   2. GBPUSD PERIOD_H1 2018.06.10 12:00 (SUNDAY):bar index is -1 (exact=true)
*/
```

Siehe auch

© 2000-2025, MetaQuotes Ltd.

CopyTime, CopyRates

© 2000-2025, MetaQuotes Ltd.

iTickVolume

Gibt den Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
long  iTickVolume(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyTickVolume, CopyRates

© 2000-2025, MetaQuotes Ltd.

iRealVolume

Gibt den Wert des echten Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
long  iRealVolume(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des echten Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyRealVolume, CopyRates

© 2000-2025, MetaQuotes Ltd.

iVolume

Gibt den Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
long  iVolume(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopyTickVolume, CopyRates

© 2000-2025, MetaQuotes Ltd.

iSpread

Gibt den Wert des Spreads des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück.

```
long  iSpread(
   const string         symbol,          // Symbol
   ENUM_TIMEFRAMES      timeframe,       // Zeitrahmen
   int                  shift            // Verschiebung
   );
```

Parameter

symbol

[in] Name des Finanzinstruments. NULL bedeutet das aktuelle Symbol.

timeframe

[in] Zeitrahmen. Der Wert kann einer der Werte der Aufzählung ENUM_TIMEFRAMES sein. 0 bedeutet den Zeitrahmen des aktuellen Charts.

shift

[in] Index des abgefragten Wertes aus der Zeitreihe (Rückwärtsverschiebung um die angegebenen Anzahl der Balken relativ zum aktuellen Balken).

Rückgabewert

Wert des Spreads für den Balken (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts oder 0 im Fehlerfall. Um zusätzliche Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion gibt immer aktuelle Daten zurück, dafür fragt sie die Zeitreihe nach dem angegebenen Symbol/Zeitrahmen bei jedem Aufruf ab. Wenn keine Daten beim ersten Aufruf der Funktion vorhanden sind, kann die Vorbereitung des Ergebnisses der Ausführung einige Zeit in Anspruch nehmen.

Die Funktion speichert die Ergebnisse der vorherigen Aufrufe nicht, es gibt keinen lokalen Cache für eine schnelle Rückgabe des Wertes.

Beispiel:

```
input int shift=0;
//+------------------------------------------------------------------+
//| Function-event handler "tick"                                    |
//+------------------------------------------------------------------+
void OnTick()
 {
   datetime time  = iTime(Symbol(),Period(),shift);
   double   open  = iOpen(Symbol(),Period(),shift);
   double   high  = iHigh(Symbol(),Period(),shift);
   double   low   = iLow(Symbol(),Period(),shift);
   double   close = iClose(NULL,PERIOD_CURRENT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
   long     volume= iVolume(Symbol(),0,shift);
   int      bars  = iBars(NULL,0);

   Comment(Symbol(),",",EnumToString(Period()),"\n",
           "Time: "  ,TimeToString(time,TIME_DATE|TIME_SECONDS),"\n",
           "Open: "  ,DoubleToString(open,Digits()),"\n",
           "High: "  ,DoubleToString(high,Digits()),"\n",
           "Low: "   ,DoubleToString(low,Digits()),"\n",
           "Close: " ,DoubleToString(close,Digits()),"\n",
           "Volume: ",IntegerToString(volume),"\n",
           "Bars: "  ,IntegerToString(bars),"\n"
           );
  }
```

Siehe auch

CopySpread, CopyRates

© 2000-2025, MetaQuotes Ltd.

Benutzerdefinierte Symbole

Funktionen für die Erstellung und Bearbeitung von Eigenschaften benutzerdefinierter Symbole.

Wenn das Terminal sich mit einem konkreten Handelsserver verbindet, bekommt der Nutzer die Möglichkeit, mit Zeitreihen der Finanzinstrumenten zu arbeiten, die der Broker bietet. Die verfügbaren Symbole werden als Liste in Market Watch angezeigt, eine Gruppe von Funktionen erlaubt es, Informationen über Eigenschaften des Symbols, Handelssitzungen und Aktualisierungen der Markttiefe erhalten.

Die in diesem Abschnitt dargestellte Gruppe der Funktionen erlaubt es, benutzerdefinierte Symbole zu erstellen. Dafür kann man die vorhandenen Symbole des Handelsservers, Text-Dateien oder externe Quellen verwenden.

Funktion	Aktion
CustomSymbolCreate	Erstellt ein benutzerdefiniertes Symbol mit dem angegebenen Namen in der angegebenen Gruppe
CustomSymbolDelete	Löscht ein benutzerdefinierte Symbol mit dem angegebenen Namen
CustomSymbolSetInteger	Setzt den Wert einer Eigenschaft vom ganzzahligen Typ für das benutzerdefinierte Symbol
CustomSymbolSetDouble	Setzt den Wert einer Eigenschaft vom reellen Typ für das benutzerdefinierte Symbol
CustomSymbolSetString	Setzt den Wert einer Eigenschaft vom String-Typ für das benutzerdefinierte Symbol
CustomSymbolSetMarginRate	Setzt den Koeffizienten der Margin je nach Typ und Richtung einer Order für das benutzerdefinierte Symbol
CustomSymbolSetSessionQuote	Setzt den Anfang und das Ende der angegebenen Notierungssitzung für das angegebene Symbol und Wochentag
CustomSymbolSetSessionTrade	Setzt den Anfang und das Ende der angegebenen Notierungssitzungitzung für das angegebene Symbol und Wochentag
CustomRatesDelete	Löscht alle Balken im angegebenen Zeitintervall aus der Preishistorie des benutzerdefinierten Symbols
CustomRatesReplace	Ersetzt die komplette Preishistorie des benutzerdefinierten Symbols im angegebenen Zeitintervall durch die Daten aus dem Array vom Typ MqlRates
CustomRatesUpdate	Fügt der Historie des benutzerdefinierten Symbols fehlende Balken hinzu und ersetzt die vorhandenen Balken durch die Daten aus dem Array vom Typ MqlRates
CustomTicksAdd	Fügt Daten aus einem Array vom Typ MqlTick in die Preishistorie eines benutzerdefinierten Symbols hinzu. Das

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
	benutzerdefinierte Symbol muss im Fenster MarketWatch (Marktübersicht) ausgewählt werden
CustomTicksDelete	Löscht alle Balken im angegebenen Zeitintervall aus der Preishistorie des benutzerdefinierten Symbols
CustomTicksReplace	Ersetzt die komplette Preishistorie des benutzerdefinierten Symbols im angegebenen Zeitintervall durch die Daten aus dem Array vom Typ MqlTick
CustomBookAdd	Überträgt den Status der Markttiefe (Depth of Market) für ein nutzerdefiniertes Symbol

© 2000-2025, MetaQuotes Ltd.

CustomSymbolCreate

Erstellt ein nutzerdefiniertes Symbol mit dem angegebenen Namen in der angegebenen Gruppe.

```
bool  CustomSymbolCreate(
   const string      symbol_name,          // Name des nutzerdefinierten Symbols
   const string      symbol_path="",       // Name der Gruppe in der das Symbole erstell
   const string      symbol_origin=NULL    // Name des Basissymbols für die Erstellung d
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols. Es sollte keine Gruppen oder Untergruppen enthalten, in denen sich das Symbol befindet.

symbol_path=""

[in] Der Gruppenname, in der sich das Symbol befindet.

symbol_origin=NULL

[in] Name des Basissymbols, dessen Eigenschaften für nutzerdefinierte Symbol kopiert werden. Nach dem Erstellen des nutzerdefinierten Symbols kann jede Eigenschaft auf das Benötigte mittels der entsprechenden Funktionen geändert werden.

Rückgabewert

true - bei Erfolg, sonst - false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Alle nutzerdefinierten Symbole werden im speziellen Nutzerbereich erstellt. Wenn der Gruppenname nicht angegeben wurde (der Parameter *symbol_path* der Funktion CustomSymbolCreate ist entweder eine leere Zeichenkette oder enthält den Wert NULL), wird das nutzerdefinierte Symbol im Stammordner des Nutzerbereichs erstellt. Das ist analog zum Dateisystem, das Gruppen und Untergruppen in Form von Ordnern und Unterordnern anzeigt.

Die Namen der Symbole und Gruppen dürfen nur lateinische Buchstaben ohne Interpunktion, Leer- oder Sonderzeichen enthalten (mit Ausnahme von ".", "_", "&" und "#"). Es wird nicht empfohlen folgende Zeichen zu verwenden <, >, :, ", /, |, ?, *.

Der Name des nutzerdefinierten Symbols sollte eindeutig sein, unabhängig von der Gruppe, in der es erstellt wurde. Wenn bereits ein Symbol gleichen Namens existiert, gibt die Funktion CustomSymbolCreate() 'false' zurück, und der nachfolgende Aufruf GetLastError() wirft den Fehler 5300 (ERR_NOT_CUSTOM_SYMBOL) oder 5304 (ERR_CUSTOM_SYMBOL_EXIST) aus.

Die Länge des Symbolnamens sollte 31 Zeichen nicht überschreiten. Andernfalls gibt CustomSymbolCreate() 'false' zurück und der Fehler 5302 - ERR_CUSTOM_SYMBOL_NAME_LONG wird gesetzt.

Der Parameter *symbol_path* kann auf zwei Arten bestimmt werden:

- nur einen Gruppennamen ohne Namen des nutzerdefinierten Symbols - "CFD\\Metals". Es ist am besten, diese Option zu verwenden, um Fehler zu vermeiden.

© 2000-2025, MetaQuotes Ltd.

- oder <Gruppe> Name + Gruppenseparator "\\"+<nutzerdefinierter Symbolname>, zum Beispiel - "CFD\\Metalle\\Platin". In diesem Fall sollte der Gruppenname mit dem genauen Namen des nutzerdefinierten Symbols enden. Stimmen sie nicht überein, wird das nutzerdefinierte Symbol dennoch erstellt, jedoch nicht in der beabsichtigten Gruppe. Zum Beispiel, wenn *symbol_path*="CFD\\Metals**P**latinum" und *symbol_name*="**p**latinum" (Fehler der Groß-/Kleinschreibung), dann wird in der Gruppe "Custom\CFD\Metals\Platinum" ein nutzerdefiniertes Symbol namens "platinum" erstellt. Die Funktion SymbolInfoGetString("platinum",SYMBOL_PATH) gibt den Wert "Custom\CFD\Metals\Platinum\Platinum\platinum" zurück.

Beachten Sie, das die Eigenschaft SYMBOL_PATH den Pfad mit dem Symbolnamen am Ende zurückgibt. Daher kann er nicht ohne Änderungen kopiert werden, wenn Sie ein nutzerdefiniertes Symbol in genau der gleichen Gruppe erstellen möchten. In diesem Fall ist es notwendig, den Symbolnamen zu kürzen, um nicht das oben beschriebene Ergebnis zu erhalten.

Wenn ein nicht vorhandenes Symbol als Parameter *symbol_origin* verwendet wird, wird ein 'leeres', nutzerdefinierte Symbol erstellt, als ob der Parameter *symbol_origin* nicht gesetzt wäre. Der Fehler 4301 - ERR_MARKET_UNKNOWN_SYMBOL wird in diesem Fall ausgeworfen.

Die Länge des Parameters *symbol_path* sollte, unter Berücksichtigung von "Custom\\", den Gruppenseparatoren "\\" und dem Symbolnamen, 127 Zeichen nicht überschreiten, wenn der Name am Ende angegeben ist.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                              CustomSymbolCreate.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"  // Name des nutzerdefinierten Symbols
#define   CUSTOM_SYMBOL_PATH     "Forex"        // Name der Gruppe, in der das Symbol
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()       // Name des Symbols, das als Basis des

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird im I
   if(!CustomSymbolCreate(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL_ORIGIN
     {
      Print("CustomSymbolCreate() failed. Error ", GetLastError());
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die Existenz des erstellten Symbols prüfen und die Gruppe abrufen, in der es ers
   bool   custom= false;
   bool   exist = SymbolExist(CUSTOM_SYMBOL_NAME, custom);
   string path  = SymbolInfoString(CUSTOM_SYMBOL_NAME, SYMBOL_PATH);

//--- das Ergebnis der Symbolerstellung und den Namen der Gruppe (angegeben und aus de
   PrintFormat("Custom symbol '%s' created\n"+
               "Symbol '%s' is exist: %s\n"+
               "Symbol '%s' is custom: %s\n"+
               "Path specified in the settings: '%s'\n"+
               "Path returned from the 'SYMBOL_PATH' property: '%s'",
               CUSTOM_SYMBOL_NAME,
               CUSTOM_SYMBOL_NAME, (string)exist,
               CUSTOM_SYMBOL_NAME, (string)custom,
               CUSTOM_SYMBOL_PATH,
               path);

//--- zwei Sekunden warten und das erstellte Symbol mit der resultierenden Nachricht s
   Sleep(2000);
   ResetLastError();
   bool deleted = CustomSymbolDelete(CUSTOM_SYMBOL_NAME);
   Print(deleted ? StringFormat("Custom symbol '%s' removed", CUSTOM_SYMBOL_NAME) : St
   /*
   Ergebnis:
   Custom symbol 'EURUSD.C' created
   Symbol 'EURUSD.C' is exist: true
   Symbol 'EURUSD.C' is custom: true
   Path specified in the settings: 'Forex'
   Path returned from the 'SYMBOL_PATH' property: 'Custom\Forex\EURUSD.C'
   Custom symbol 'EURUSD.C' removed
   */
  }
```

Siehe auch

SymbolName, SymbolSelect, CustomSymbolDelete

© 2000-2025, MetaQuotes Ltd.

CustomSymbolDelete

Löscht das nutzerdefinierte Symbol mit dem angegebenen Namen.

```
bool  CustomSymbolDelete(
   const string     symbol_name          // Name des nutzerdefinierten Symbols
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols. Der Name darf nicht mit dem Namen eines bereits vorhandenen Symbols übereinstimmen.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Ein nutzerdefiniertes Symbol, das im Fenster Marktübersicht (Market Watch) angezeigt wird, oder dessen Chart offen ist, kann nicht gelöscht werden.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                          CustomSymbolDelete.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"  // Name des nutzerdefinierten Symbols
#define   CUSTOM_SYMBOL_PATH     "Forex"        // Name der Gruppe, in der das Symbol
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()       // Name des Symbols, das als Basis des

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird im I
   if(!CustomSymbolCreate(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL_ORIGIN
     {
      Print("CustomSymbolCreate() failed. Error ", GetLastError());
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Überprüfung der Existenz des erstellten Symbols und Ausdruck des Ergebnisses im
   bool custom= false;
   bool exist = SymbolExist(CUSTOM_SYMBOL_NAME, custom);
   PrintFormat("Custom symbol '%s' exists: %s", CUSTOM_SYMBOL_NAME, (string)exist);

//--- zwei Sekunden warten und das erstellte Symbol mit der resultierenden Nachricht
   Sleep(2000);
   ResetLastError();
   bool deleted = CustomSymbolDelete(CUSTOM_SYMBOL_NAME);
   Print(deleted ? StringFormat("Custom symbol '%s' removed", CUSTOM_SYMBOL_NAME) : St

//--- Überprüfung der Existenz des erstellten Symbols und Ausdruck des Ergebnisses im
   exist = SymbolExist(CUSTOM_SYMBOL_NAME, custom);
   PrintFormat("Custom symbol '%s' exists: %s", CUSTOM_SYMBOL_NAME, (string)exist);
   /*
   Ergebnis:
   Custom symbol 'EURUSD.C' exists: true
   Custom symbol 'EURUSD.C' removed
   Custom symbol 'EURUSD.C' exists: false
   */
  }
```

Siehe auch

SymbolName, SymbolSelect, CustomSymbolCreate

© 2000-2025, MetaQuotes Ltd.

CustomSymbolSetInteger

Setzt den Wert einer ganzzahligen Eigenschaft für das nutzerdefinierte Symbol.

```
bool  CustomSymbolSetInteger(
   const string               symbol_name,      // Symbolname
   ENUM_SYMBOL_INFO_INTEGER   property_id,      // Identifikator der Eigenschaft
   long                       property_value    // Wert der Eigenschaft
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols.

property_id

[in] Identifikator der Eigenschaft des Symbols. Der Wert kann einer der Werte der Enumeration ENUM_SYMBOL_INFO_INTEGER sein.

property_value

[in] Variable vom Typ long, die den Wert der Eigenschaft enthält.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Die Minuten- und Tick-Historie des nutzerdefinierten Symbols wird komplett gelöscht, wenn man eine der folgenden Eigenschaften in der Spezifikation des Symbols ändert:

- SYMBOL_CHART_MODE - Preistyp, der für das Zeichnen von Balken verwendet wird (Bid oder Last)
- SYMBOL_DIGITS - Zahl der Nachkommastellen für die Anzeige des Preises

Nach dem Löschen der Historie des nutzerdefinierten Symbols versucht das Terminal eine neue Historie unter Verwendung der aktualisierten Eigenschaften zu erstellen. Das Gleiche geschieht bei manueller Änderung der Eigenschaften eines nutzerdefinierten Symbols.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                      CustomSymbolSetInteger.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"  // Name des nutzerdefinierten Symbols
#define   CUSTOM_SYMBOL_PATH     "Forex"        // Name der Gruppe, in der das Symbol
```

© 2000-2025, MetaQuotes Ltd.

```
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()       // Name des Symbols, das als Basis des

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- die Eigenschaften des Symbols abrufen und im Log drucken, auf dem das nutzerdefi
//--- (Handelsmodus, Stop-Order-Installationsebene und Mindestabstand der Handelsopera
   ENUM_SYMBOL_TRADE_EXECUTION origin_exe_mode = (ENUM_SYMBOL_TRADE_EXECUTION)SymbolIn
   int origin_stops_level = (int)SymbolInfoInteger(CUSTOM_SYMBOL_ORIGIN, SYMBOL_TRADE_
   int origin_freeze_level= (int)SymbolInfoInteger(CUSTOM_SYMBOL_ORIGIN, SYMBOL_TRADE_

   PrintFormat("The '%s' symbol from which the custom '%s' was created:\n"+
               "  Deal execution mode: %s\n  Stops Level: %d\n  Freeze Level: %d",
               CUSTOM_SYMBOL_ORIGIN, CUSTOM_SYMBOL_NAME,
               StringSubstr(EnumToString(origin_exe_mode), 23), origin_stops_level, or

//--- andere Werte für die nutzerdefinierten Symboleigenschaften festlegen
   ResetLastError();
   bool res=true;
   res &=CustomSymbolSetInteger(CUSTOM_SYMBOL_NAME, SYMBOL_TRADE_EXEMODE, SYMBOL_TRADE
   res &=CustomSymbolSetInteger(CUSTOM_SYMBOL_NAME, SYMBOL_TRADE_STOPS_LEVEL, 10);
   res &=CustomSymbolSetInteger(CUSTOM_SYMBOL_NAME, SYMBOL_TRADE_FREEZE_LEVEL, 3);

//--- wenn beim Setzen einer der Eigenschaften ein Fehler aufgetreten ist, wird eine e
   if(!res)
      Print("CustomSymbolSetInteger() failed. Error ", GetLastError());

//--- Abrufen und Drucken der geänderten nutzerdefinierten Symboleigenschaften im Log
//--- (Handelsmodus, Stop-Order-Installationsebene und Mindestabstand der Handelsopera
   ENUM_SYMBOL_TRADE_EXECUTION custom_exe_mode = (ENUM_SYMBOL_TRADE_EXECUTION)SymbolIn
   int custom_stops_level = (int)SymbolInfoInteger(CUSTOM_SYMBOL_NAME, SYMBOL_TRADE_ST
   int custom_freeze_level= (int)SymbolInfoInteger(CUSTOM_SYMBOL_NAME, SYMBOL_TRADE_FR

   PrintFormat("Custom symbol '%s' based on '%s':\n"+
               "  Deal execution mode: %s\n  Stops Level: %d\n  Freeze Level: %d",
               CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_ORIGIN,
               StringSubstr(EnumToString(custom_exe_mode), 23), custom_stops_level, cu

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
```

© 2000-2025, MetaQuotes Ltd.

```
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- durch das Drücken der Entf-Taste wird das erstellte nutzerdefinierte Symbo
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   The 'EURUSD' symbol from which the custom 'EURUSD.C' was created:
     Deal execution mode: INSTANT
     Stops Level: 0
     Freeze Level: 0
   Custom symbol 'EURUSD.C' based on 'EURUSD':
     Deal execution mode: MARKET
     Stops Level: 10
     Freeze Level: 3
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
```

Siehe auch

SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

CustomSymbolSetDouble

Setzt den Wert einer Eigenschaft vom reellen Typ für das nutzerdefinierte Symbol.

```
bool  CustomSymbolSetDouble(
   const string              symbol_name,      // Symbolname
   ENUM_SYMBOL_INFO_DOUBLE   property_id,      // Identifikator der Eigenschaft
   double                    property_value    // Wert der Eigenschaft
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols.

property_id

[in] Identifikator der Eigenschaft des Symbols. Der Wert kann einer der Werte der Enumeration ENUM_SYMBOL_INFO_DOUBLE sein.

property_value

[in] Variable vom Typ double, die den Wert der Eigenschaft beinhaltet.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Die Minuten- und Tick-Historie des nutzerdefinierten Symbols wird komplett gelöscht, wenn man eine der folgenden Eigenschaften in der Spezifikation des Symbols ändert:

- SYMBOL_POINT - Wert eines Punktes
- SYMBOL_TRADE_TICK_SIZE - Wert eines Ticks, der die minimal zulässige Preisveränderung festlegt
- SYMBOL_TRADE_TICK_VALUE - Wert der Preisveränderung von einem Tick für eine profitable Position

Nach dem Löschen der Historie des nutzerdefinierten Symbols versucht das Terminal eine neue Historie unter Verwendung der aktualisierten Eigenschaften zu erstellen. Das Gleiche geschieht bei manueller Änderung der Eigenschaften eines nutzerdefinierten Symbols.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                        CustomSymbolSetDouble.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd.   |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"  // Name des nutzerdefinierten Symbols
#define   CUSTOM_SYMBOL_PATH     "Forex"        // Name der Gruppe, in der das Symbol
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()       // Name des Symbols, das als Basis des

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- die Eigenschaften des Symbols abrufen und im Log drucken, auf dem das nutzerdefi
//--- (Mindestvolumen, Höchstvolumen, Mindestvolumenänderungsschritt für die Ausführun
   double origin_vol_min = SymbolInfoDouble(CUSTOM_SYMBOL_ORIGIN, SYMBOL_VOLUME_MIN);
   double origin_vol_max = SymbolInfoDouble(CUSTOM_SYMBOL_ORIGIN, SYMBOL_VOLUME_MAX);
   double origin_vol_step= SymbolInfoDouble(CUSTOM_SYMBOL_ORIGIN, SYMBOL_VOLUME_STEP);

   PrintFormat("The '%s' symbol from which the custom '%s' was created:\n"+
               "  Volume Min: %.2f\n  Volume Max: %.2f\n  Volume Step: %.2f",
               CUSTOM_SYMBOL_ORIGIN, CUSTOM_SYMBOL_NAME,
               origin_vol_min, origin_vol_max, origin_vol_step);

//--- andere Werte für die nutzerdefinierten Symboleigenschaften festlegen
   ResetLastError();
   bool res=true;
   res &=CustomSymbolSetDouble(CUSTOM_SYMBOL_NAME, SYMBOL_VOLUME_MIN, 0.1);
   res &=CustomSymbolSetDouble(CUSTOM_SYMBOL_NAME, SYMBOL_VOLUME_MAX, 1000);
   res &=CustomSymbolSetDouble(CUSTOM_SYMBOL_NAME, SYMBOL_VOLUME_STEP, 0.1);

//--- wenn beim Setzen einer der Eigenschaften ein Fehler aufgetreten ist, wird eine e
   if(!res)
      Print("CustomSymbolSetDouble() failed. Error ", GetLastError());

//--- Abrufen und Drucken der geänderten nutzerdefinierten Symboleigenschaften im Log
//--- (Mindestvolumen, Höchstvolumen, Mindestvolumenänderungsschritt für die Ausführun
   double custom_vol_min = SymbolInfoDouble(CUSTOM_SYMBOL_NAME, SYMBOL_VOLUME_MIN);
   double custom_vol_max = SymbolInfoDouble(CUSTOM_SYMBOL_NAME, SYMBOL_VOLUME_MAX);
   double custom_vol_step= SymbolInfoDouble(CUSTOM_SYMBOL_NAME, SYMBOL_VOLUME_STEP);

   PrintFormat("Custom symbol '%s' based on '%s':\n"+
               "  Volume Min: %.2f\n  Volume Max: %.2f\n  Volume Step: %.2f",
               CUSTOM_SYMBOL_ORIGIN, CUSTOM_SYMBOL_NAME,
               custom_vol_min, custom_vol_max, custom_vol_step);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex

//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- durch das Drücken der Entf-Taste wird das erstellte nutzerdefinierte Symbo
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   The 'EURUSD' symbol from which the custom 'EURUSD.C' was created:
     Volume Min: 0.01
     Volume Max: 500.00
     Volume Step: 0.01
   Custom symbol 'EURUSD' based on 'EURUSD.C':
     Volume Min: 0.10
     Volume Max: 1000.00
     Volume Step: 0.10
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
```

Siehe auch

SymbolInfoDouble

© 2000-2025, MetaQuotes Ltd.

CustomSymbolSetString

Setzt den Wert einer Eigenschaft des Typs string für das nutzerdefinierte Symbol.

```
bool  CustomSymbolSetString(
   const string              symbol_name,      // Symbolname
   ENUM_SYMBOL_INFO_STRING   property_id,      // Identifikator der Eigenschaft
   string                    property_value    // Wert der Eigenschaft
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols.

property_id

[in] Identifikator der Eigenschaft des Symbols. Der Wert kann einer der Werte der Enumeration ENUM_SYMBOL_INFO_STRING sein.

property_value

[in] Variable vom Typ string, die den Wert der Eigenschaft beinhaltet.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Die Minuten- und Tick-Historie des nutzerdefinierten Symbols wird komplett gelöscht, wenn man die Eigenschaft SYMBOL_FORMULA in der Spezifikation des Symbols ändert, die die Formel für die Berechnung des Preises des nutzerdefinierten Symbols setzt. Nach dem Löschen der Historie des nutzerdefinierten Symbols versucht das Terminal eine neue Historie anhand der neuen Formel zu erstellen. Das Gleiche geschieht bei manueller Änderung der Formel des nutzerdefinierten Symbols.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                        CustomSymbolSetString.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd.   |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"  // Name des nutzerdefinierten Symbols
#define   CUSTOM_SYMBOL_PATH     "Forex"        // Name der Gruppe, in der das Symbol
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()       // Name des Symbols, das als Basis des

//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOI

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- die Eigenschaften des Symbols abrufen und im Log drucken, auf dem das nutzerdefi
//--- (Mindestvolumen, Höchstvolumen, Mindestvolumenänderungsschritt für die Ausführur
   string origin_basis    = SymbolInfoString(CUSTOM_SYMBOL_ORIGIN, SYMBOL_BASIS);
   string origin_category = SymbolInfoString(CUSTOM_SYMBOL_ORIGIN, SYMBOL_CATEGORY);
   string origin_formula  = SymbolInfoString(CUSTOM_SYMBOL_ORIGIN, SYMBOL_FORMULA);

   PrintFormat("The '%s' symbol from which the custom '%s' was created:\n"+
               "  Basis: %s\n  Category: %s\n  Formula: %s",
               CUSTOM_SYMBOL_ORIGIN, CUSTOM_SYMBOL_NAME,
               origin_basis, origin_category, origin_formula);

//--- andere Werte für die nutzerdefinierten Symboleigenschaften festlegen
   ResetLastError();
   bool res=true;
   res &=CustomSymbolSetString(CUSTOM_SYMBOL_NAME, SYMBOL_BASIS, CUSTOM_SYMBOL_ORIGIN)
   res &=CustomSymbolSetString(CUSTOM_SYMBOL_NAME, SYMBOL_CATEGORY, "FX");
   res &=CustomSymbolSetString(CUSTOM_SYMBOL_NAME, SYMBOL_FORMULA, ("1.0 / "+CUSTOM_SY

//--- wenn beim Setzen einer der Eigenschaften ein Fehler aufgetreten ist, wird eine e
   if(!res)
      Print("CustomSymbolSetString() failed. Error ", GetLastError());

//--- Abrufen und Drucken der geänderten nutzerdefinierten Symboleigenschaften im Log
//--- (Mindestvolumen, Höchstvolumen, Mindestvolumenänderungsschritt für die Ausführur
   string custom_basis    = SymbolInfoString(CUSTOM_SYMBOL_NAME, SYMBOL_BASIS);
   string custom_category = SymbolInfoString(CUSTOM_SYMBOL_NAME, SYMBOL_CATEGORY);
   string custom_formula  = SymbolInfoString(CUSTOM_SYMBOL_NAME, SYMBOL_FORMULA);

   PrintFormat("Custom symbol '%s' based on '%s':\n"+
               "  Basis: %s\n  Category: %s\n  Formula: %s",
               CUSTOM_SYMBOL_ORIGIN, CUSTOM_SYMBOL_NAME,
               custom_basis, custom_category, custom_formula);

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex

//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Sleep(16);
       //--- durch das Drücken der Entf-Taste wird das erstellte nutzerdefinierte Symbo
       if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
         {
          if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
             PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
          break;
         }
      }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   The 'EURUSD' symbol from which the custom 'EURUSD.C' was created:
     Basis:
     Category:
     Formula:
   Custom symbol 'EURUSD' based on 'EURUSD.C':
     Basis: EURUSD
     Category: FX
     Formula: 1.0 / EURUSD
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
```

Siehe auch

SymbolInfoString

© 2000-2025, MetaQuotes Ltd.

CustomSymbolSetMarginRate

Setzt die Koeffizienten der Margin je nach Typ und Richtung einer Order für das nutzerdefinierte Symbol.

```
bool  CustomSymbolSetMarginRate(
   const string       symbol_name,              // Symbolname
   ENUM_ORDER_TYPE    order_type,               // Auftragstyp
   double             initial_margin_rate,      // Koeffizient der Initial Margin
   double             maintenance_margin_rate   // Koeffizient der Maintenance Margin
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols.

order_type

[in] Auftragstyp.

initial_margin_rate

[in] Variable vom Typ double mit dem Wert des Koeffizienten der Initial Margin. Initial Margin - der Betrag einer Sicherheitsleistung für die Ausführung eines Handels mit dem Volumen von einem Lot in der entsprechenden Richtung. Wenn wir den Koeffizienten mit der Initial Margin multiplizieren, erhalten wir den Betrag, der bei der Platzierung einer Order des angegebenen Typs auf dem Konto reserviert wird.

maintenance_margin_rate

[in] Variable vom Typ double mit dem Wert des Koeffizienten der Maintenance Margin. Maintenance Margin - der Mindestbetrag für den Erhalt einer offenen Position mit dem Volumen von einem Lot in die entsprechende Richtung. Wenn wir den Koeffizienten mit der Maintenance Margin multiplizieren, erhalten wir den Betrag, der nach der Auslösung einer Order des angegebenen Typs reserviert wird.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                    CustomSymbolSetMarginRate.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd.   |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"  // Name des nutzerdefinierten Symbols
#define   CUSTOM_SYMBOL_PATH     "Forex"        // Name der Gruppe, in der das Symbol
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()       // Name des Symbols, das als Basis des

#define   INITIAL_MARGIN_RATE       1.5         // Rate der Initial Margin
#define   MAINTENANCE_MARGIN_RATE   1.5         // Rate der Maintenance Margin

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- die Eigenschaften des Symbols abrufen und im Log drucken, auf dem das nutzerdefi
//--- (Initial Margin und Maintenance Margin für Kauf- und Verkaufsaufträge)
   double initial_margin_rate_buy=0;
   double maintenance_margin_rate_buy=0;
   double initial_margin_rate_sell=0;
   double maintenance_margin_rate_sell=0;

   if(!GetSymbolMarginRate(CUSTOM_SYMBOL_ORIGIN, ORDER_TYPE_BUY, initial_margin_rate_b
      return;
   if(!GetSymbolMarginRate(CUSTOM_SYMBOL_ORIGIN, ORDER_TYPE_SELL,initial_margin_rate_s
      return;

   PrintFormat("The '%s' symbol from which the custom '%s' was created:\n"+
               "  Buy order initial margin rate: %f\n  Buy order maintenance margin ra
               "  Sell order initial margin rate: %f\n  Sell order maintenance margin
               CUSTOM_SYMBOL_ORIGIN, CUSTOM_SYMBOL_NAME,
               initial_margin_rate_buy, maintenance_margin_rate_buy, initial_margin_ra

//--- andere Werte für die nutzerdefinierten Symboleigenschaften festlegen
   ResetLastError();
   bool res=true;
   res &=CustomSymbolSetMarginRate(CUSTOM_SYMBOL_NAME, ORDER_TYPE_BUY, INITIAL_MARGIN_
   res &=CustomSymbolSetMarginRate(CUSTOM_SYMBOL_NAME, ORDER_TYPE_SELL,INITIAL_MARGIN_

//--- wenn beim Setzen einer der Eigenschaften ein Fehler aufgetreten ist, wird eine e
   if(!res)
      Print("CustomSymbolSetMarginRate() failed. Error ", GetLastError());

//--- Abrufen und Drucken der geänderten nutzerdefinierten Symboleigenschaften im Log
//--- (Initial Margin und Maintenance Margin für Kauf- und Verkaufsaufträge)
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!GetSymbolMarginRate(CUSTOM_SYMBOL_NAME, ORDER_TYPE_BUY, initial_margin_rate_buy
      return;
   if(!GetSymbolMarginRate(CUSTOM_SYMBOL_NAME, ORDER_TYPE_SELL,initial_margin_rate_sel
      return;

   PrintFormat("Custom symbol '%s' based on '%s':\n"+
               "  Buy order initial margin rate: %f\n  Buy order maintenance margin ra
               "  Sell order initial margin rate: %f\n  Sell order maintenance margin
               CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_ORIGIN,
               initial_margin_rate_buy, maintenance_margin_rate_buy, initial_margin_ra

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex

//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- durch das Drücken der Entf-Taste wird das erstellte nutzerdefinierte Symbo
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   The 'EURUSD' symbol from which the custom 'EURUSD.C' was created:
     Buy order initial margin rate: 1.000000
     Buy order maintenance margin rate: 0.000000
     Sell order initial margin rate: 1.000000
     Sell order maintenance margin rate: 0.000000
   Custom symbol 'EURUSD.C' based on 'EURUSD':
     Buy order initial margin rate: 1.500000
     Buy order maintenance margin rate: 1.500000
     Sell order initial margin rate: 1.500000
     Sell order maintenance margin rate: 1.500000
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastErro
      return(false);
     }
//--- Erfolg
   return(true);
  }
//+------------------------------------------------------------------+
//| Return margin ratios                                             |
//+------------------------------------------------------------------+
bool GetSymbolMarginRate(const string symbol, const ENUM_ORDER_TYPE order_type, double
  {
   ResetLastError();
   if(!SymbolInfoMarginRate(symbol, order_type, initial_margin_rate, maintenance_margi
     {
      PrintFormat("%s: SymbolInfoMarginRate(%s, %s) failed. Error %d",__FUNCTION__, sy
      return false;
     }
   return true;
  }
```

© 2000-2025, MetaQuotes Ltd.

Siehe auch

SymbolInfoMarginRate

© 2000-2025, MetaQuotes Ltd.

CustomSymbolSetSessionQuote

Setzt den Anfang und das Ende der angegebenen Kurssitzung für das angegebene Symbol und Wochentag.

```
bool  CustomSymbolSetSessionQuote(
   const string      symbol_name,          // Symbolname
   ENUM_DAY_OF_WEEK  day_of_week,          // Wochentag
   uint              session_index,        // Nummer der Sitzung
   datetime          from,                 // Anfangszeitpunkt der Sitzung
   datetime          to                    // Endzeitpunkt
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols.

ENUM_DAY_OF_WEEK

[in] Wochentag, ein Wert aus der Enumeration ENUM_DAY_OF_WEEK.

uint

[in] Ordnungsnummer der Sitzung, für welche man den Anfang und das Ende setzen muss. Die Indexierung der Sitzungen beginnt mit 0.

from

[in] Anfangszeitpunkt der Sitzung in Sekunden von 00 Uhr 00 Minuten, der Wert des Datums in der Variablen wird ignoriert.

to

[in] Endzeitpunkt der Sitzung in Sekunden von 00 Stunden 00 Minuten, der Wert des Datums in der Variablen wird ignoriert.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Wenn eine Sitzung mit dem angegebenen *session_index* bereits existiert, wird die Funktion einfach den Anfang und das Ende der Sitzung bearbeiten.

Wenn für die Sitzung den Parametern für Anfangs- und Endzeitpunkt Null übergeben wurden, mit anderen Worten *from=0* und *to=0*, wird die entsprechende Sitzung mit dem Index *session_index* gelöscht, und die Nummerierung wird nach unten verschoben.

Die Sitzungen können nur der Reihe nach hinzugefügt, d.h. die Sitzung mit dem Index *session_index=1* kann nur dann hinzugefügt werden, wenn bereits eine Sitzung mit dem Index 0 existiert. Wenn diese Regel nicht entsprochen wird, wird keine neue Sitzung erstellt, und die Funktion gibt false zurück.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                  CustomSymbolSetSessionQuote.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"           // Name des nutzerdefinierter
#define   CUSTOM_SYMBOL_PATH     "Forex"                 // Name der Gruppe, für die c
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()                // Name des Symbols, das die

#define   SESSION_0_FROM         D'1970.01.01 00:15:00'  // Startzeit der Sitzung 0
#define   SESSION_0_TO           D'1970.01.01 11:59:00'  // Endzeit der Sitzung 0
#define   SESSION_1_FROM         D'1970.01.01 12:15:00'  // Startzeit der Sitzung 1
#define   SESSION_1_TO           D'1970.01.01 23:59:00'  // Endzeit der Sitzung 1

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOI

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Kopfzeile mit Basissymbol und Sitzungsindex drucken und
//--- in einer Schleife nach Wochentag von Montag bis Freitag die Start- und Endzeiter
   for(int session=0; session<2; session++)
     {
      PrintFormat("Quote session %d of '%s' symbol from which the custom '%s' was crea
      for(int day_of_week=MONDAY; day_of_week<SATURDAY; day_of_week++)
         SymbolInfoSessionQuotePrint(CUSTOM_SYMBOL_ORIGIN, (ENUM_DAY_OF_WEEK)day_of_we
     }

//--- in einer Schleife mit zwei Sitzungen
   bool res=true;
   for(int session=0; session<2; session++)
     {
      datetime from = SESSION_0_FROM;
      datetime to   = SESSION_0_TO;
      if(session>0)
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         from = SESSION_1_FROM;
         to   = SESSION_1_TO;
        }
      //--- Setzen der Kurssitzungszeit für ein nutzerdefiniertes Symbol für jeden Tag
      ResetLastError();
      for(int day_of_week=MONDAY; day_of_week<SATURDAY; day_of_week++)
         res &=CustomSymbolSetSessionQuote(CUSTOM_SYMBOL_NAME, (ENUM_DAY_OF_WEEK)day_o
     }

//--- wenn beim Festlegen einer der Sitzungen ein Fehler aufgetreten ist, wird eine er
   if(!res)
      Print("CustomSymbolSetSessionQuote() failed. Error ", GetLastError());

//--- Ausdrucken der Kopfzeilen mit dem nutzerdefinierten Symbol und dem Sitzungsindex
//--- in einer Schleife nach Wochentag von Montag bis Freitag die Start- und Endzeiten
   for(int session=0; session<2; session++)
     {
      PrintFormat("Quote session %d of custom symbol '%s' based on '%s'", session, CUS
      for(int day_of_week=MONDAY; day_of_week<SATURDAY; day_of_week++)
         SymbolInfoSessionQuotePrint(CUSTOM_SYMBOL_NAME, (ENUM_DAY_OF_WEEK)day_of_week
     }

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- durch das Drücken der Entf-Taste wird das erstellte nutzerdefinierte Symbo
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Quote session 0 of 'EURUSD' symbol from which the custom 'EURUSD.C' was created
   - Monday     00:15 - 23:55
   - Tuesday    00:15 - 23:55
   - Wednesday  00:15 - 23:55
   - Thursday   00:15 - 23:55
   - Friday     00:15 - 23:55
   Quote session 1 of 'EURUSD' symbol from which the custom 'EURUSD.C' was created
   - Monday     Session not set
   - Tuesday    Session not set
```

© 2000-2025, MetaQuotes Ltd.

```
   - Wednesday  Session not set
   - Thursday   Session not set
   - Friday     Session not set
   Quote session 0 of custom symbol 'EURUSD.C' based on 'EURUSD'
   - Monday     00:15 - 11:59
   - Tuesday    00:15 - 11:59
   - Wednesday  00:15 - 11:59
   - Thursday   00:15 - 11:59
   - Friday     00:15 - 11:59
   Quote session 1 of custom symbol 'EURUSD.C' based on 'EURUSD'
   - Monday     12:15 - 23:59
   - Tuesday    12:15 - 23:59
   - Wednesday  12:15 - 23:59
   - Thursday   12:15 - 23:59
   - Friday     12:15 - 23:59
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
//+------------------------------------------------------------------+
//| Sende Anfang und Ende der angegebenen Kurssitzung                |
//| des angegebenen Symbols und Wochentags ins Log drucken           |
//+------------------------------------------------------------------+
void SymbolInfoSessionQuotePrint(const string symbol, const ENUM_DAY_OF_WEEK day_of_we
  {
//--- Deklarieren der Variablen, um den Beginn und das Ende der Kurssitzung aufzuzeich
   datetime date_from;  // Startzeit der Sitzung
   datetime date_to;    // Endzeit der Sitzung

//--- den Namen des Wochentags aus dem Enumerationswert erstellen
   string week_day=EnumToString(day_of_week);
   if(week_day.Lower())
      week_day.SetChar(0, ushort(week_day.GetChar(0)-32));

//--- Abrufen der Daten aus der Kurssitzung nach Symbol und Wochentag
   if(!SymbolInfoSessionQuote(symbol, day_of_week, session_index, date_from, date_to))
     {
      int err=GetLastError();
      string message=(err==4307 ? StringFormat("- %-10s Session not set", week_day) :
                      StringFormat("SymbolInfoSessionQuote(%s, %s, session %d) failed.
      Print(message);
      return;
     }

//--- Daten für die angegebene Kurssitzung an das Log senden
   PrintFormat("- %-10s %s - %s", week_day, TimeToString(date_from, TIME_MINUTES), Tir
  }
```

Siehe auch

SymbolInfoSessionQuote, Information über das Symbol, TimeToStruct, Datumstruktur

© 2000-2025, MetaQuotes Ltd.

CustomSymbolSetSessionTrade

Setzt den Anfang und das Ende der angegebenen Handelssitzung für das angegebene Symbol und Wochentag.

```
bool  CustomSymbolSetSessionTrade(
   const string      symbol_name,          // Symbolname
   ENUM_DAY_OF_WEEK  day_of_week,          // Wochentag
   uint              session_index,        // Nummer der Sitzung
   datetime          from,                 // Anfangszeitpunkt der Sitzung
   datetime          to                    // Endzeitpunkt
   );
```

Parameter

symbol_name

[in] Name des nutzerdefinierten Symbols.

ENUM_DAY_OF_WEEK

[in] Wochentag, ein Wert aus der Enumeration ENUM_DAY_OF_WEEK.

uint

[in] Ordnungsnummer der Sitzung, für welche man den Anfang und das Ende setzen muss. Die Indexierung der Sitzungen beginnt mit 0.

from

[in] Anfangszeitpunkt der Sitzung in Sekunden von 00 Uhr 00 Minuten, der Wert des Datums in der Variablen wird ignoriert.

to

[in] Endzeitpunkt der Sitzung in Sekunden von 00 Stunden 00 Minuten, der Wert des Datums in der Variablen wird ignoriert.

Rückgabewert

true - bei Erfolg, andernfalls false. Um Fehlerdetails zu erhalten, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Wenn eine Sitzung mit dem angegebenen *session_index* bereits existiert, wird die Funktion einfach den Anfang und das Ende der Sitzung bearbeiten.

Wenn für die Sitzung den Parametern für Anfangs- und Endzeitpunkt Null übergeben wurden, mit anderen Worten *from=0* und *to=0*, wird die entsprechende Sitzung mit dem Index *session_index* gelöscht, und die Nummerierung wird nach unten verschoben.

Die Sitzungen können nur der Reihe nach hinzugefügt, d.h. die Sitzung mit dem Index *session_index=1* kann nur dann hinzugefügt werden, wenn bereits eine Sitzung mit dem Index 0 existiert. Wenn diese Regel nicht entsprochen wird, wird keine neue Sitzung erstellt, und die Funktion gibt false zurück.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                  CustomSymbolSetSessionTrade.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"           // Name des nutzerdefinierter
#define   CUSTOM_SYMBOL_PATH     "Forex"                 // Name der Gruppe, für die d
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()                // Name des Symbols, das die

#define   SESSION_0_FROM         D'1970.01.01 00:15:00'  // Startzeit der Sitzung 0
#define   SESSION_0_TO           D'1970.01.01 11:59:00'  // Endzeit der Sitzung 0
#define   SESSION_1_FROM         D'1970.01.01 12:15:00'  // Startzeit der Sitzung 1
#define   SESSION_1_TO           D'1970.01.01 23:59:00'  // Endzeit der Sitzung 1

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Kopfzeile mit Basissymbol und Sitzungsindex drucken und
//--- in einer Schleife nach Wochentagen von Mo. bis Fr. die Start- und Endzeiten jede
   for(int session=0; session<2; session++)
     {
      PrintFormat("Trade session %d of the '%s' symbol from which the custom '%s' was
      for(int day_of_week=MONDAY; day_of_week<SATURDAY; day_of_week++)
         SymbolInfoSessionQuotePrint(CUSTOM_SYMBOL_ORIGIN, (ENUM_DAY_OF_WEEK)day_of_we
     }

//--- in einer Schleife mit zwei Sitzungen
   bool res=true;
   for(int session=0; session<2; session++)
     {
      datetime from = SESSION_0_FROM;
      datetime to   = SESSION_0_TO;
      if(session>0)
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         from = SESSION_1_FROM;
         to   = SESSION_1_TO;
        }
      //--- Setzen der Kurssitzungszeit für ein nutzerdefiniertes Symbol für jeden Tag
      ResetLastError();
      for(int day_of_week=MONDAY; day_of_week<SATURDAY; day_of_week++)
         res &=CustomSymbolSetSessionQuote(CUSTOM_SYMBOL_NAME, (ENUM_DAY_OF_WEEK)day_o
     }

//--- wenn beim Festlegen einer der Sitzungen ein Fehler aufgetreten ist, wird eine er
   if(!res)
      Print("CustomSymbolSetSessionTrade() failed. Error ", GetLastError());

//--- Ausdrucken der Kopfzeilen mit dem nutzerdefinierten Symbol und dem Sitzungsindex
//--- in einer Schleife nach Wochentagen von Mo. bis Fr. die Start- und Endzeiten jede
   for(int session=0; session<2; session++)
     {
      PrintFormat("Trade session %d of a custom symbol '%s' based on '%s'", session, C
      for(int day_of_week=MONDAY; day_of_week<SATURDAY; day_of_week++)
         SymbolInfoSessionQuotePrint(CUSTOM_SYMBOL_NAME, (ENUM_DAY_OF_WEEK)day_of_week
     }

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- durch das Drücken der Entf-Taste wird das erstellte nutzerdefinierte Symbo
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Trade session 0 of the 'EURUSD' symbol from which the custom 'EURUSD.C' was created
   - Monday     00:15 - 23:55
   - Tuesday    00:15 - 23:55
   - Wednesday  00:15 - 23:55
   - Thursday   00:15 - 23:55
   - Friday     00:15 - 23:55
   Trade session 1 of the 'EURUSD' symbol from which the custom 'EURUSD.C' was created
   - Monday     Session not set
   - Tuesday    Session not set
```

© 2000-2025, MetaQuotes Ltd.

```
   - Wednesday  Session not set
   - Thursday   Session not set
   - Friday     Session not set
   Trade session 0 of a custom symbol 'EURUSD.C' based on 'EURUSD'
   - Monday     00:15 - 11:59
   - Tuesday    00:15 - 11:59
   - Wednesday  00:15 - 11:59
   - Thursday   00:15 - 11:59
   - Friday     00:15 - 11:59
   Trade session 1 of a custom symbol 'EURUSD.C' based on 'EURUSD'
   - Monday     12:15 - 23:59
   - Tuesday    12:15 - 23:59
   - Wednesday  12:15 - 23:59
   - Thursday   12:15 - 23:59
   - Friday     12:15 - 23:59
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
//+------------------------------------------------------------------+
//| Sende Anfang und Ende der angegebenen Kurssitzung                |
//| des angegebenen Symbols und Wochentags ins Log drucken           |
//+------------------------------------------------------------------+
void SymbolInfoSessionTradePrint(const string symbol, const ENUM_DAY_OF_WEEK day_of_we
  {
//--- Deklarieren der Variablen, um den Beginn und das Ende der Kurssitzung aufzuzeich
   datetime date_from;  // Startzeit der Sitzung
   datetime date_to;    // Endzeit der Sitzung

//--- den Namen des Wochentags aus dem Enumerationswert erstellen
   string week_day=EnumToString(day_of_week);
   if(week_day.Lower())
      week_day.SetChar(0, ushort(week_day.GetChar(0)-32));

//--- Abrufen der Daten aus der Kurssitzung nach Symbol und Wochentag
   if(!SymbolInfoSessionTrade(symbol, day_of_week, session_index, date_from, date_to))
     {
      int err=GetLastError();
      string message=(err==4307 ? StringFormat("- %-10s Session not set", week_day) :
                      StringFormat("SymbolInfoSessionTrade(%s, %s, session %d) failed.
      Print(message);
      return;
     }

//--- Daten für die angegebene Handelssitzung an das Log senden
   PrintFormat("- %-10s %s - %s", week_day, TimeToString(date_from, TIME_MINUTES), Tir
  }
```

Siehe auch

SymbolInfoSessionTrade, Information über das Symbol, TimeToStruct, Datumstruktur

© 2000-2025, MetaQuotes Ltd.

CustomRatesDelete

Löscht alle Balken aus der Preishistorie eines nutzerdefinierten Symbols im angegebenen Zeitintervall.

```
int  CustomRatesDelete(
   const string      symbol,        // Symbolname
   datetime          from,          // von
   datetime          to             // bis
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols.

from

[in] Zeit des ersten Balkens in der Preishistorie aus dem angegebenen Zeitintervall, der gelöscht werden muss.

to

[in] Zeit des letzten Balkens in der Preishistorie aus dem angegebenen Zeitintervall, der gelöscht werden muss.

Rückgabewert

Anzahl der gelöschten Balken oder -1 im Fehlerfall.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                              CustomRatesDelete.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                           https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"     // Name des nutzerdefinierten Symbo
#define   CUSTOM_SYMBOL_PATH     "Forex"           // Name der Gruppe, in der das Symb
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()          // Name des Symbols, das als Basis

#define   DATARATES_COUNT        4                 // Anzahl der Balken, die an das Lo

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
```

© 2000-2025, MetaQuotes Ltd.

```
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Anzahl der Balken des Standardsymbols abrufen
   int bars=Bars(CUSTOM_SYMBOL_ORIGIN, PERIOD_M1);

//--- Abrufen der Daten aller Balken des Standardsymbols des Zeitrahmens 1 Minute in d
   MqlRates rates[]={};
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_ORIGIN, PERIOD_M1, 0, bars, rates)!=bars)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_OR
      return;
     }

//--- Setzen die kopierten Daten auf den Minuten-Historie des nutzerdefinierten Symbol
   ResetLastError();
   if(CustomRatesUpdate(CUSTOM_SYMBOL_NAME, rates)<0)
     {
      PrintFormat("CustomRatesUpdate(%s) failed. Error %d", CUSTOM_SYMBOL_NAME, GetLas
      return;
     }

//--- nach dem Aktualisieren der historischen Daten die Anzahl der nutzerdefinierten B
   bars=Bars(CUSTOM_SYMBOL_NAME, PERIOD_M1);

//--- die Daten aller Balken des nutzerdefinierten Symbols des Zeitrahmens 1 Minute in
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_NAME, PERIOD_M1, 0, bars, rates)!=bars)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_NA
      return;
     }

//--- die letzten DATARATES_COUNT Balken der M1-Historie des nutzerdefinierten Symbols
   int digits=(int)SymbolInfoInteger(CUSTOM_SYMBOL_NAME, SYMBOL_DIGITS);
   PrintFormat("Last %d bars of the custom symbol's minute history:", DATARATES_COUNT)
   ArrayPrint(rates, digits, NULL, bars-DATARATES_COUNT, DATARATES_COUNT);

//--- Löschen der beiden vorletzten Datenbalken der Minutenhistorie des nutzerdefinier
   datetime time_from= rates[bars-3].time;
   datetime time_to  = rates[bars-2].time;
   ResetLastError();
   int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, time_from, time_to);
   if(deleted<0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("CustomRatesDelete(%s) failed. Error %d", CUSTOM_SYMBOL_NAME, GetLas
      return;
     }

//--- nach dem Löschen von zwei Balken der historischen Daten wird die Anzahl der Bal
   bars=Bars(CUSTOM_SYMBOL_NAME, PERIOD_M1);

//--- die Daten aller verbleibenden Balken des Minutenzeitrahmens des nutzerdefinierte
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_NAME, PERIOD_M1, 0, bars, rates)!=bars)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_NA
      return;
     }

//--- die letzten DATARATES_COUNT Balken der aktualisierten M1-Historie des nutzerdefi
   PrintFormat("\nLast %d bars after applying CustomRatesDelete() with %d deleted bars
   ArrayPrint(rates, digits, NULL, bars-DATARATES_COUNT, DATARATES_COUNT);

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- beim Drücken von Entf das erstellte nutzerdefinierte Symbol und seine Date
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         //--- Balkendaten löschen
         int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history bars of the custom symbol '%s' were successfully d

         //--- Tickdaten löschen
         deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history ticks of the custom symbol '%s' were successfully

         //--- Symbol löschen
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Last 4 bars of the custom symbol's minute history:
```

© 2000-2025, MetaQuotes Ltd.

```
                    [time]   [open]   [high]    [low] [close] [tick_volume] [spread] [rea
   [0] 2024.06.18 20:53:00 1.07341 1.07347 1.07336 1.07343             38         0
   [1] 2024.06.18 20:54:00 1.07344 1.07354 1.07344 1.07353             21         0
   [2] 2024.06.18 20:55:00 1.07353 1.07362 1.07351 1.07356             32         0
   [3] 2024.06.18 20:56:00 1.07356 1.07358 1.07352 1.07354             24         0

   Last 4 bars after applying CustomRatesDelete() with 2 deleted bars:
                    [time]   [open]   [high]    [low] [close] [tick_volume] [spread] [rea
   [0] 2024.06.18 20:51:00 1.07357 1.07358 1.07347 1.07349             25         0
   [1] 2024.06.18 20:52:00 1.07349 1.07350 1.07336 1.07341             31         0
   [2] 2024.06.18 20:53:00 1.07341 1.07347 1.07336 1.07343             38         0
   [3] 2024.06.18 20:56:00 1.07356 1.07358 1.07352 1.07354             24         0
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
```

Siehe auch

CustomRatesReplace, CustomRatesUpdate, CopyRates

© 2000-2025, MetaQuotes Ltd.

CustomRatesReplace

Ersetzt die komplette Preishistorie eines nutzerdefinierten Symbols im angegebenen Zeitintervall durch die Daten eines Arrays vom Typ MqlRates.

```
int  CustomRatesReplace(
   const string     symbol,              // Symbolname
   datetime         from,                // von
   datetime         to,                  // bis
   const MqlRates&  rates[],             // Array mit den Daten, die dem nutzerdefinier
   uint             count=WHOLE_ARRAY    // Anzahl der zu verwendenden Elemente des Ar
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols.

from

[in] Zeit des ersten Balkens in der Preishistorie aus dem angegebenen Zeitintervall, das aktualisiert werden muss.

to

[in] Zeit des letzten Balkens in der Preishistorie aus dem angegebenen Zeitintervall, das aktualisiert werden muss.

rates[]

[in] Array der historischen Daten vom Typ MqlRates für den Zeitrahmen M1.

count=WHOLE_ARRAY

[in] Anzahl der zu verwendenden Elemente des Arrays *rates[]*, die für das Ersetzen verwendet werden. WHOLE_ARRAY bedeutet, es werden alle Elemente des Arrays *rates[]* für das Ersetzen verwendet.

Rückgabewert

Anzahl der aktualisierten Balken oder -1 im Fehlerfall.

Hinweis

Wenn ein Balken aus dem Array *rates[]* das angegebene Intervall überschreitet, wird er ignoriert. Wenn ein solcher Balken in der Preishistorie bereits vorhanden ist und innerhalb des angegebenen Intervalls liegt, wird er ersetzt. Alle anderen Balken der Preishistorie außerhalb des angegebenen Intervalls bleiben unverändert. Die Daten im Array *rates[]* müssen den OHLC-Kursen entsprechen und die Eröffnungszeit der Balken muss dem M1-Zeitrahmen entsprechen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                            CustomRatesReplace.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"     // Name des nutzerdefinierten Symbo
#define   CUSTOM_SYMBOL_PATH     "Forex"           // Name der Gruppe, in der das Symb
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()          // Name des Symbols, das als Basis

#define   DATARATES_COUNT        4                 // Anzahl der Balken, die an das Lo

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Anzahl der Balken des Standardsymbols abrufen
   int bars=Bars(CUSTOM_SYMBOL_ORIGIN, PERIOD_M1);

//--- Abrufen der Daten aller Balken des Standardsymbols des Zeitrahmens 1 Minute in d
   MqlRates rates[]={};
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_ORIGIN, PERIOD_M1, 0, bars, rates)!=bars)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_OR
      return;
     }

//--- Setzen die kopierten Daten auf den Minuten-Historie des nutzerdefinierten Symbol
   ResetLastError();
   if(CustomRatesUpdate(CUSTOM_SYMBOL_NAME, rates)<0)
     {
      PrintFormat("CustomRatesUpdate(%s) failed. Error %d", CUSTOM_SYMBOL_NAME, GetLas
      return;
     }

//--- nach dem Aktualisieren der historischen Daten die Anzahl der nutzerdefinierten B
   bars=Bars(CUSTOM_SYMBOL_NAME, PERIOD_M1);

//--- die Daten aller Balken des nutzerdefinierten Symbols des Zeitrahmens 1 Minute in
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CopyRates(CUSTOM_SYMBOL_NAME, PERIOD_M1, 0, bars, rates)!=bars)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_NA
      return;
     }

//--- die letzten DATARATES_COUNT Balken der M1-Historie des nutzerdefinierten Symbols
   int digits=(int)SymbolInfoInteger(CUSTOM_SYMBOL_NAME, SYMBOL_DIGITS);
   PrintFormat("Last %d bars of the custom symbol's minute history:", DATARATES_COUNT)
   ArrayPrint(rates, digits, NULL, bars-DATARATES_COUNT, DATARATES_COUNT);

//--- Änderung der beiden vorletzten Datenbalken der M1-Historie des nutzerdefinierten
   datetime time_from= rates[bars-3].time;
   datetime time_to  = rates[bars-2].time;

//--- alle Preise der beiden vorletzten Balken gleich den Eröffnungspreisen dieser Bal
   rates[bars-3].high=rates[bars-3].open;
   rates[bars-3].low=rates[bars-3].open;
   rates[bars-3].close=rates[bars-3].open;

   rates[bars-2].high=rates[bars-2].open;
   rates[bars-2].low=rates[bars-2].open;
   rates[bars-2].close=rates[bars-2].open;

//--- Ersetzen der vorhandene Balken durch Daten aus dem geänderten Array „rates".
   ResetLastError();
   int replaced=CustomRatesUpdate(CUSTOM_SYMBOL_NAME, rates);
   if(replaced<0)
     {
      PrintFormat("CustomRatesUpdate(%s) failed. Error %d", CUSTOM_SYMBOL_NAME, GetLas
      return;
     }

//--- nach dem Ändern von zwei Balken der historischen Daten wird die Anzahl der Balke
   bars=Bars(CUSTOM_SYMBOL_NAME, PERIOD_M1);

//--- die Daten aller Balken des Minutenzeitrahmens des nutzerdefinierten Symbols erne
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_NAME, PERIOD_M1, 0, bars, rates)!=bars)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_NA
      return;
     }

//--- die letzten DATARATES_COUNT Balken der aktualisierten M1-Historie des nutzerdefi
   PrintFormat("\nLast %d bars after applying CustomRatesUpdate() with %d replaced bar
   ArrayPrint(rates, digits, NULL, bars-DATARATES_COUNT, DATARATES_COUNT);

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
```

© 2000-2025, MetaQuotes Ltd.

```
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- beim Drücken von Entf das erstellte nutzerdefinierte Symbol und seine Date
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         //--- Balkendaten löschen
         int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history bars of the custom symbol '%s' were successfully d

         //--- Tickdaten löschen
         deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history ticks of the custom symbol '%s' were successfully

         //--- Symbol löschen
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Last 4 bars of the custom symbol's minute history:
                    [time]  [open]  [high]   [low] [close] [tick_volume] [spread] [rea
   [0] 2024.07.29 13:37:00 1.08394 1.08396 1.08388 1.08390            16        1
   [1] 2024.07.29 13:38:00 1.08389 1.08400 1.08389 1.08398            35        1
   [2] 2024.07.29 13:39:00 1.08398 1.08410 1.08394 1.08410            29        1
   [3] 2024.07.29 13:40:00 1.08409 1.08414 1.08408 1.08414            14        1

   Last 4 bars after applying CustomRatesUpdate() with 250820 replaced bars:
                    [time]  [open]  [high]   [low] [close] [tick_volume] [spread] [rea
   [0] 2024.07.29 13:37:00 1.08394 1.08396 1.08388 1.08390            16        1
   [1] 2024.07.29 13:38:00 1.08389 1.08389 1.08389 1.08389            35        1
   [2] 2024.07.29 13:39:00 1.08398 1.08398 1.08398 1.08398            29        1
   [3] 2024.07.29 13:40:00 1.08409 1.08414 1.08408 1.08414            14        1
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
```

© 2000-2025, MetaQuotes Ltd.

```
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
```

Siehe auch

CustomRatesDelete, CustomRatesUpdate, CopyRates

© 2000-2025, MetaQuotes Ltd.

CustomRatesUpdate

Fügt der Historie eines nutzerdefinierten Symbols fehlende Balken hinzu und ersetzt die vorhandenen Balken durch die Daten aus dem Array vom Typ MqlRates.

```
int  CustomRatesUpdate(
   const string     symbol,             // Name des nutzerdefinierten Symbols
   const MqlRates&  rates[],            // Array mit den Daten, die an das nutzerdefir
   uint             count=WHOLE_ARRAY   // Anzahl der zu verwendenden Elemente des Ar
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols.

rates[]

[in] Array der historischen Daten vom Typ MqlRates für den Zeitrahmen M1.

count=WHOLE_ARRAY

[in] Anzahl der zu verwendenden Elemente des Arrays *rates[]*, die für das Aktualisieren verwendet werden. WHOLE_ARRAY bedeutet, es werden alle Elemente des Arrays *rates[]* zum Aktualisieren verwendet.

Rückgabewert

Anzahl der aktualisierten Balken oder -1 im Fehlerfall.

Hinweis

Wenn der Balken aus dem Array *rates[]* in der aktuellen Historie des nutzerdefinierten Symbols fehlt, wird es hinzugefügt. Wenn ein solcher Balken bereits vorhanden ist, wird er ersetzt. Alle anderen Balken der Preishistorie bleiben unverändert. Die Daten im Array *rates[]* müssen den OHLC-Kursen entsprechen und die Eröffnungszeit der Balken muss dem M1-Zeitrahmen entsprechen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                              CustomRatesUpdate.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                           https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"      // Name des nutzerdefinierten Symbo
#define   CUSTOM_SYMBOL_PATH     "Forex"            // Name der Gruppe, in der das Symb
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()           // Name des Symbols, das als Basis

#define   DATARATES_COUNT        4                  // Anzahl der Balken, die an das Lo
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Anzahl der Balken des Standardsymbols abrufen und im Log drucken
   int bars_origin=Bars(CUSTOM_SYMBOL_ORIGIN, PERIOD_M1);
   PrintFormat("The symbol '%s' from which the custom '%s' was created has %d bars of

//--- Anzahl der nutzerdefinierten Symbolbalken abrufen und im Log ausdrucken
   int bars_custom=Bars(CUSTOM_SYMBOL_NAME, PERIOD_M1);
   PrintFormat("Custom symbol '%s' created from symbol '%s' has %d bars of minute hist

//--- Abrufen der Daten aller Balken des Standardsymbols des Zeitrahmens 1 Minute in d
   MqlRates rates[]={};
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_ORIGIN, PERIOD_M1, 0, bars_origin, rates)!=bars_origin)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_OR
      return;
     }

//--- Setzen die kopierten Daten auf den Minuten-Historie des nutzerdefinierten Symbol
   ResetLastError();
   int updated=CustomRatesUpdate(CUSTOM_SYMBOL_NAME, rates);
   if(updated<0)
     {
      PrintFormat("CustomRatesUpdate(%s) failed. Error %d", CUSTOM_SYMBOL_NAME, GetLas
      return;
     }

//--- Anzahl der Balken des nutzerdefinierten Symbols nach dem Hinzufügen des Verlaufs
   bars_custom=Bars(CUSTOM_SYMBOL_NAME, PERIOD_M1);
   PrintFormat("\nAfter CustomRatesUpdate(), the custom symbol '%s' has %d bars of min

//--- die Daten aller Balken des nutzerdefinierten Symbols des Zeitrahmens 1 Minute in
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_NAME, PERIOD_M1, 0, bars_custom, rates)!=bars_custom)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_NA
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
     }

//--- die letzten vier Balken der M1-Historie des nutzerdefinierten Symbols im Log dru
   int digits=(int)SymbolInfoInteger(CUSTOM_SYMBOL_NAME, SYMBOL_DIGITS);
   PrintFormat("Last %d bars of the custom symbol's minute history:", DATARATES_COUNT)
   ArrayPrint(rates, digits, NULL, bars_custom-DATARATES_COUNT, DATARATES_COUNT);

//--- Ersetzen der Daten im MqlRates-Array durch den Umkehrwert der Daten von SymbolNa
   for(int i=0; i<bars_custom; i++)
     {
      rates[i].open  =(rates[i].open !=0  ? 1.0 / rates[i].open  : rates[i].open);
      rates[i].high  =(rates[i].high !=0  ? 1.0 / rates[i].high  : rates[i].high);
      rates[i].low   =(rates[i].low  !=0  ? 1.0 / rates[i].low   : rates[i].low);
      rates[i].close =(rates[i].close!=0  ? 1.0 / rates[i].close : rates[i].close);
     }

//--- Zuweisen der geänderten Daten der Minutenhistorie des nutzerdefinierten Symbols
   ResetLastError();
   updated=CustomRatesUpdate(CUSTOM_SYMBOL_NAME, rates);
   if(updated<0)
     {
      PrintFormat("CustomRatesUpdate(%s) failed. Error %d", CUSTOM_SYMBOL_NAME, GetLas
      return;
     }

//--- Daten aller Balken des Minuten-Zeitrahmens des nutzerdefinierten Symbols erneut
   ResetLastError();
   if(CopyRates(CUSTOM_SYMBOL_NAME, PERIOD_M1, 0, bars_custom, rates)!=bars_custom)
     {
      PrintFormat("CopyRates(%s, PERIOD_M1, 0, %d) failed. Error %d", CUSTOM_SYMBOL_NA
      return;
     }

//--- die letzten vier Balken des aktualisierten Minutenhistorie des nutzerdefinierter
   Print("\nLast %d bars after changing the custom symbol calculation formula:", DATAR
   ArrayPrint(rates, digits, NULL, bars_custom-DATARATES_COUNT, DATARATES_COUNT);

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- beim Drücken von Entf das erstellte nutzerdefinierte Symbol und seine Date
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         //--- Balkendaten löschen
         int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
```

© 2000-2025, MetaQuotes Ltd.

```
            PrintFormat("%d history bars of the custom symbol '%s' were successfully c

         //--- Tickdaten löschen
         deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history ticks of the custom symbol '%s' were successfully

         //--- Symbol löschen
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   The symbol 'EURUSD' from which the custom 'EURUSD.C' was created has 250488 bars of
   Custom symbol 'EURUSD.C' created from symbol 'EURUSD' has 0 bars of minute history

   After CustomRatesUpdate(), the custom symbol 'EURUSD.C' has 250488 bars of minute h
   Last 4 bars of the custom symbol's minute history:
                     [time]  [open]  [high]   [low] [close] [tick_volume] [spread] [rea
   [0] 2024.06.18 11:14:00 1.07235 1.07239 1.07232 1.07239            24        0
   [1] 2024.06.18 11:15:00 1.07238 1.07239 1.07232 1.07235            44        0
   [2] 2024.06.18 11:16:00 1.07234 1.07238 1.07227 1.07234            37        0
   [3] 2024.06.18 11:17:00 1.07234 1.07234 1.07217 1.07225            41        0

   Last 4 bars after changing the custom symbol calculation formula:
                     [time]  [open]  [high]   [low] [close] [tick_volume] [spread] [rea
   [0] 2024.06.18 11:14:00 0.93253 0.93250 0.93256 0.93250            24        0
   [1] 2024.06.18 11:15:00 0.93251 0.93250 0.93256 0.93253            44        0
   [2] 2024.06.18 11:16:00 0.93254 0.93251 0.93260 0.93254            37        0
   [3] 2024.06.18 11:17:00 0.93254 0.93254 0.93269 0.93262            41        0
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
```

Siehe auch

CustomRatesReplace, CustomRatesDelete, CopyRates

© 2000-2025, MetaQuotes Ltd.

CustomTicksAdd

Fügt Daten aus einem Array vom Typ MqlTick in die Preishistorie eines nutzerdefinierten Symbols hinzu. Das nutzerdefinierte Symbol muss im Fenster Marktübersicht ausgewählt werden.

```
int  CustomTicksAdd(
   const string     symbol,              // Symbolname
   const MqlTick&   ticks[],             // Array mit Tickdaten, die man auf das nutzer
   uint             count=WHOLE_ARRAY    // Anzahl der zu verwendenden Elemente des Ar
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols.

ticks[]

[in] Das Array der Tickdaten vom Typ MqlTick, die nach Zeit aufsteigend geordnet wurden, das heißt, ticks[k].time_msc <= ticks[n].time_msc, wenn k<n.

count=WHOLE_ARRAY

[in] Anzahl der zu verwendenden Elemente des Arrays *ticks[]*, die hinzugefügt werden. WHOLE_ARRAY bedeutet, es werden alle Elemente des Arrays *ticks[]* hinzugefügt.

Rückgabewert

Die Anzahl der hinzugefügten Ticks oder -1 im Fehlerfall.

Hinweis

Die Funktion CustomTicksAdd arbeitet nur für nutzerdefinierte Symbole im Fenster der Marktübersicht. Wenn das Symbol in der Marktübersicht nicht ausgewählt wurde, muss man für das Einfügen von Ticks CustomTicksReplace verwenden.

Die Funktion CustomTicksAdd erlaubt es, die Ticks so zu übertragen, als ob sie vom Server des Brokers eingehen würden. Die Daten werden nicht direkt in die Datenbank von Ticks geschrieben, sondern ins Fenster Marktübersicht gesendet. Das Terminal speichert dann die Ticks aus diesem Fenster in seiner Datenbank. Wenn das Datenvolumen, das während eines Aufrufs übertragen wird, zu groß ist, ändert die Funktion ihr Verhalten, um Ressourcen zu sparen. Wenn über 256 Ticks übertragen werden, werden die Daten in zwei Teile geteilt. Das erste Teil (der größere) wird direkt in die Datenbank von Ticks geschrieben (wie CustomTicksReplace das tut). Das zweite Teil, das aus den letzten 128 Ticks besteht, wird in das Fenster Marktübersicht geleitet, und danach vom Terminal in der Datenbank gespeichert.

Die Struktur MqlTick hat zwei Zeitfelder - time (Tickzeit in Sekunden) und time_msc (Tickzeit in Millisekunden) - seit dem 01. Januar 1970. Die Verarbeitung dieser Felder erfolgt in den hinzugefügten Ticks nach folgenden Regeln in der angegebenen Reihenfolge:

1. wenn ticks[k].time_msc!=0, dann verwenden wir diesen Wert für die Ausfüllung des Feldes ticks[k].time, d.h. für den Tick wird die Zeit ticks[k].time=ticks[k].time_msc/1000 (ganzzahlige Division) gesetzt.
2. wenn ticks[k].time_msc==0 und ticks[k].time!=0, dann erhalten wir die Zeit in Millisekunden durch die Multiplikation mit 1000, d.h. ticks[k].time_msc=ticks[k].time*1000

© 2000-2025, MetaQuotes Ltd.

3. wenn ticks[k].time_msc==0 und ticks[k].time==0, wird die aktuelle Zeit des Handelsservers im Moment des Aufrufs der Funktion CustomTicksAdd bis auf Millisekunden in diese Felder geschrieben.

Wenn der Wert der Felder ticks[k].bid, ticks[k].ask, ticks[k].last oder ticks[k].volume größer als Null ist, wird eine Kombination der entsprechenden Flags in das Feld ticks[k].flags geschrieben:

- TICK_FLAG_BID - der Tick hat den Bid-Preis geändert
- TICK_FLAG_ASK - der Tick hat den Ask-Preis geändert
- TICK_FLAG_LAST - der Tick hat den Last-Preis geändert
- TICK_FLAG_VOLUME - der Tick hat das Volumen geändert

Wenn der Wert eines Feldes kleiner oder gleich Null ist, wird das entsprechende Flag nicht ins Feld ticks[k].flags geschrieben.

Die Flags TICK_FLAG_BUY und TICK_FLAG_SELL werden der Historie des nutzerdefinierten Symbols nicht hinzugefügt.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                CustomTicksAdd.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd.   |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"     // Name des nutzerdefinierten Symbo
#define   CUSTOM_SYMBOL_PATH     "Forex"           // Name der Gruppe, in der das Symb
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()          // Name des Symbols, das als Basis

#define   DATATICKS_TO_COPY      UINT_MAX          // Anzahl der kopierten Ticks
#defineDATATICKS_TO_PRINT     20                // Anzahl der an das Log gesendeten Ti

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Abrufen der Tick-Daten des Standardsymbols in einem MqlTick-Array
   MqlTick array[]={};
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!GetTicksToArray(CUSTOM_SYMBOL_ORIGIN, DATATICKS_TO_COPY, array))
      return;

//--- Ausdruck der Zeit des ersten und des letzten empfangenen Ticks des Standardsymbo
   int total=(int)array.Size();
   PrintFormat("First tick time: %s.%03u, Last tick time: %s.%03u",
               TimeToString(array[0].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
               TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des Standardsymbols in das Log
   PrintFormat("\nThe last %d ticks for the standard symbol '%s':", DATATICKS_TO_PRINT
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
     }

//--- ein nutzerdefiniertes Symbol zum Fenster der Marktübersicht hinzufügen
   ResetLastError();
   if(!SymbolSelect(CUSTOM_SYMBOL_NAME, true))
     {
      Print("SymbolSelect() failed. Error ", GetLastError());
      return;
     }

//--- Hinzufügen der Daten des Tick-Arrays zur Preishistorie des nutzerdefinierten Sym
   Print("...");
   uint start=GetTickCount();
   PrintFormat("Start of adding %u ticks to the history of the custom symbol '%s'", ar
   int added=CustomTicksAdd(CUSTOM_SYMBOL_NAME, array);
   PrintFormat("Added %u ticks to the history of the custom symbol '%s' in %u ms", add

//--- Abrufen der Tick-Daten des nutzerdefinierten Symbols mit dem MqlTick-Array
   Print("...");
   if(!GetTicksToArray(CUSTOM_SYMBOL_NAME, array.Size(), array))
      return;

//--- Ausdruck der Zeit des ersten und letzten empfangenen Ticks des nutzerdefinierter
   total=(int)array.Size();
   PrintFormat("First tick time: %s.%03u, Last tick time: %s.%03u",
               TimeToString(array[0].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
               TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des nutzerdefinierten Symbols in d
   PrintFormat("\nThe last %d ticks for the custom symbol '%s':", DATATICKS_TO_PRINT,
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
```

© 2000-2025, MetaQuotes Ltd.

```
         continue;
       PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
      }

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- beim Drücken von Entf das erstellte nutzerdefinierte Symbol und seine Date
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         //--- Balkendaten löschen
         int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history bars of the custom symbol '%s' were successfully c

         //--- Tickdaten löschen
         deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history ticks of the custom symbol '%s' were successfully

         //--- Symbol löschen
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Requested 4294967295 ticks to download tick history for the symbol 'EURUSD'
   The tick history for the 'EURUSD' symbol is received in the amount of 351183943 tic
   First tick time: 2011.12.19 00:00:08.000, Last tick time: 2024.06.20 21:18:12.010

   The last 20 ticks for the standard symbol 'EURUSD':
     351183923th Tick: 2024.06.20 21:17:46.380 Bid=1.07124 (Info tick)
     351183924th Tick: 2024.06.20 21:17:47.779 Ask=1.07125 Bid=1.07125 (Info tick)
     351183925th Tick: 2024.06.20 21:17:48.584 Ask=1.07124 Bid=1.07124 (Info tick)
     351183926th Tick: 2024.06.20 21:17:49.481 Ask=1.07125 (Info tick)
     351183927th Tick: 2024.06.20 21:17:49.985 Ask=1.07122 Bid=1.07122 (Info tick)
     351183928th Tick: 2024.06.20 21:17:50.482 Ask=1.07124 Bid=1.07124 (Info tick)
     351183929th Tick: 2024.06.20 21:17:51.584 Ask=1.07123 Bid=1.07123 (Info tick)
     351183930th Tick: 2024.06.20 21:17:52.786 Ask=1.07124 Bid=1.07124 (Info tick)
     351183931th Tick: 2024.06.20 21:17:53.487 Ask=1.07125 Bid=1.07125 (Info tick)
     351183932th Tick: 2024.06.20 21:17:53.989 Ask=1.07126 Bid=1.07126 (Info tick)
     351183933th Tick: 2024.06.20 21:17:55.789 Ask=1.07125 Bid=1.07125 (Info tick)
```

© 2000-2025, MetaQuotes Ltd.

```
     351183934th Tick: 2024.06.20 21:17:58.495 Ask=1.07126 Bid=1.07126 (Info tick)
     351183935th Tick: 2024.06.20 21:18:00.102 Bid=1.07126 (Info tick)
     351183936th Tick: 2024.06.20 21:18:00.698 Ask=1.07129 Bid=1.07129 (Info tick)
     351183937th Tick: 2024.06.20 21:18:03.699 Bid=1.07129 (Info tick)
     351183938th Tick: 2024.06.20 21:18:04.699 Ask=1.07128 Bid=1.07128 (Info tick)
     351183939th Tick: 2024.06.20 21:18:05.901 Ask=1.07129 Bid=1.07129 (Info tick)
     351183940th Tick: 2024.06.20 21:18:07.606 Ask=1.07128 Bid=1.07128 (Info tick)
     351183941th Tick: 2024.06.20 21:18:11.512 Ask=1.07127 Bid=1.07127 (Info tick)
     351183942th Tick: 2024.06.20 21:18:12.010 Ask=1.07126 Bid=1.07126 (Info tick)
   ...
   Start of adding 351183943 ticks to the history of the custom symbol 'EURUSD.C'
   Added 351183943 ticks to the history of the custom symbol 'EURUSD.C' in 269890 ms
   ...
   Requested 351183943 ticks to download tick history for the symbol 'EURUSD.C'
   The tick history for the 'EURUSD.C' symbol is received in the amount of 351183943 t
   First tick time: 2011.12.19 00:00:08.000, Last tick time: 2024.06.20 21:18:12.010

   The last 20 ticks for the custom symbol 'EURUSD.C':
     351183923th Tick: 2024.06.20 21:17:46.380 Ask=1.07124 Bid=1.07124 (Info tick)
     351183924th Tick: 2024.06.20 21:17:47.779 Ask=1.07125 Bid=1.07125 (Info tick)
     351183925th Tick: 2024.06.20 21:17:48.584 Ask=1.07124 Bid=1.07124 (Info tick)
     351183926th Tick: 2024.06.20 21:17:49.481 Ask=1.07125 Bid=1.07125 (Info tick)
     351183927th Tick: 2024.06.20 21:17:49.985 Ask=1.07122 Bid=1.07122 (Info tick)
     351183928th Tick: 2024.06.20 21:17:50.482 Ask=1.07124 Bid=1.07124 (Info tick)
     351183929th Tick: 2024.06.20 21:17:51.584 Ask=1.07123 Bid=1.07123 (Info tick)
     351183930th Tick: 2024.06.20 21:17:52.786 Ask=1.07124 Bid=1.07124 (Info tick)
     351183931th Tick: 2024.06.20 21:17:53.487 Ask=1.07125 Bid=1.07125 (Info tick)
     351183932th Tick: 2024.06.20 21:17:53.989 Ask=1.07126 Bid=1.07126 (Info tick)
     351183933th Tick: 2024.06.20 21:17:55.789 Ask=1.07125 Bid=1.07125 (Info tick)
     351183934th Tick: 2024.06.20 21:17:58.495 Ask=1.07126 Bid=1.07126 (Info tick)
     351183935th Tick: 2024.06.20 21:18:00.102 Ask=1.07126 Bid=1.07126 (Info tick)
     351183936th Tick: 2024.06.20 21:18:00.698 Ask=1.07129 Bid=1.07129 (Info tick)
     351183937th Tick: 2024.06.20 21:18:03.699 Ask=1.07129 Bid=1.07129 (Info tick)
     351183938th Tick: 2024.06.20 21:18:04.699 Ask=1.07128 Bid=1.07128 (Info tick)
     351183939th Tick: 2024.06.20 21:18:05.901 Ask=1.07129 Bid=1.07129 (Info tick)
     351183940th Tick: 2024.06.20 21:18:07.606 Ask=1.07128 Bid=1.07128 (Info tick)
     351183941th Tick: 2024.06.20 21:18:11.512 Ask=1.07127 Bid=1.07127 (Info tick)
     351183942th Tick: 2024.06.20 21:18:12.010 Ask=1.07126 Bid=1.07126 (Info tick)
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
//+------------------------------------------------------------------+
//| Abrufen der angegebenen Anzahl von Ticks im Array                |
//+------------------------------------------------------------------+
bool GetTicksToArray(const string symbol, const uint count, MqlTick &array[])
  {
//--- Benachrichtigung über den Beginn des Ladens historischer Daten
   PrintFormat("Requested %u ticks to get tick history for the symbol '%s'", count, sy

//--- 3 Versuche, Ticks abzurufen
   int attempts=0;
   while(attempts<3)
     {
      //--- Startzeit vor Empfang der Ticks messen
      uint start=GetTickCount();
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Anforderung der Tick-Historie seit 1970.01.01 00:00.001 (Parameter von=1 r
      int received=CopyTicks(symbol, array, COPY_TICKS_ALL, 1, count);
      if(received!=-1)
        {
         //--- Informationen über die Anzahl der Ticks und die verwendete Zeit anzeige
         PrintFormat("The tick history for the '%s' symbol is received in the amount o

         //--- Wenn die Tick-Historie synchronisiert ist, ist der Fehlercode gleich Nu
         if(GetLastError()==0)
            return(true);

         PrintFormat("%s: Ticks are not synchronized yet, %d ticks received for %d ms.
                     symbol, received, GetTickCount()-start, GetLastError());
        }
      //--- Versuche zählen
      attempts++;
      //--- eine Pause von einer Sekunde, um auf das Ende der Synchronisierung der Tic
      Sleep(1000);
     }
//--- Konnten die Ticks nach 3 Versuchen nicht kopiert werden:
   return(false);
  }
//+------------------------------------------------------------------+
//| gibt die String-Beschreibung eines Ticks zurück                  |
//+------------------------------------------------------------------+
string GetTickDescription(MqlTick &tick)
  {
   string desc=StringFormat("%s.%03u ", TimeToString(tick.time, TIME_DATE|TIME_MINUTES

//--- Prüfen der Flags
   bool buy_tick   = ((tick.flags &TICK_FLAG_BUY)    == TICK_FLAG_BUY);
   bool sell_tick  = ((tick.flags &TICK_FLAG_SELL)   == TICK_FLAG_SELL);
   bool ask_tick   = ((tick.flags &TICK_FLAG_ASK)    == TICK_FLAG_ASK);
   bool bid_tick   = ((tick.flags &TICK_FLAG_BID)    == TICK_FLAG_BID);
   bool last_tick  = ((tick.flags &TICK_FLAG_LAST)   == TICK_FLAG_LAST);
   bool volume_tick= ((tick.flags &TICK_FLAG_VOLUME)== TICK_FLAG_VOLUME);

//--- zuerst den Tick auf die Handelsflags prüfen (es gibt keine für CustomTicksAdd())
   if(buy_tick || sell_tick)
     {
      //--- eine Ausgabe für einen Handelstick bilden
      desc += (buy_tick ? StringFormat("Buy Tick: Last=%G Volume=%d ", tick.last, tick
      desc += (sell_tick? StringFormat("Sell Tick: Last=%G Volume=%d ",tick.last, tick
      desc += (ask_tick ? StringFormat("Ask=%G ", tick.ask) : "");
      desc += (bid_tick ? StringFormat("Bid=%G ", tick.ask) : "");
      desc += "(Trade tick)";
     }
   else
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- die Ausgabe für einen Info-Tick etwas anders gestalten
      desc += (ask_tick   ? StringFormat("Ask=%G ",  tick.ask)    : "");
      desc += (bid_tick   ? StringFormat("Bid=%G ",  tick.ask)    : "");
      desc += (last_tick  ? StringFormat("Last=%G ", tick.last)   : "");
      desc += (volume_tick? StringFormat("Volume=%d ",tick.volume): "");
      desc += "(Info tick)";
     }
//--- Tick-Beschreibung zurückgeben
   return(desc);
  }
```

Siehe auch

CustomRatesDelete, CustomRatesUpdate, CustomTicksReplace, CopyTicks, CopyTicksRange

© 2000-2025, MetaQuotes Ltd.

CustomTicksDelete

Löscht alle Ticks im angegebenen Zeitintervall aus der Preishistorie des nutzerdefinierten Symbols.

```
int  CustomTicksDelete(
   const string     symbol,           // Symbolname
   long             from_msc,         // von welchem Tag
   long             to_msc            // bis zu welchem Tag
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols.

from_msc

[in] Zeit des ersten Ticks in der Preishistorie aus dem angegebenen Intervall, der gelöscht werden muss. Zeit in Millisekunden ab dem 01.01.1970.

to_msc

[in] Zeit des letzten Ticks in der Preishistorie aus dem angegebenen Intervall, der gelöscht werden muss. Zeit in Millisekunden ab dem 01.01.1970.

Rückgabewert

Anzahl der gelöschten Balken oder -1 im Fehlerfall.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                            CustomTicksDelete.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd.   |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"     // Name des nutzerdefinierten Symbo
#define   CUSTOM_SYMBOL_PATH     "Forex"           // Name der Gruppe, in der das Symb
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()          // Name des Symbols, das als Basis

#define   DATATICKS_TO_COPY      UINT_MAX          // Anzahl der kopierten Ticks
#define   DATATICKS_TO_DELETE    10                // Anzahl der gelöschten Ticks
#defineDATATICKS_TO_PRINT     20                // Anzahl der an das Log gesendeten Ti

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Abrufen der Tick-Daten des Standardsymbols in einem MqlTick-Array
   MqlTick array[]={};
   if(!GetTicksToArray(CUSTOM_SYMBOL_ORIGIN, DATATICKS_TO_COPY, array))
      return;

//--- Ausdruck der Zeit des ersten und des letzten empfangenen Ticks des Standardsymbo
   int total=(int)array.Size();
   PrintFormat("First tick time: %s.%03u, Last tick time: %s.%03u",
               TimeToString(array[0].time,TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
               TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des Standardsymbols in das Log
   PrintFormat("\nThe last %d ticks for the standard symbol '%s':", DATATICKS_TO_PRINT
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
     }

//--- ein nutzerdefiniertes Symbol zum Fenster der Marktübersicht hinzufügen
   ResetLastError();
   if(!SymbolSelect(CUSTOM_SYMBOL_NAME, true))
     {
      Print("SymbolSelect() failed. Error ", GetLastError());
      return;
     }

//--- Hinzufügen der Daten des Tick-Arrays zur Preishistorie des nutzerdefinierten Sym
   Print("...");
   uint start=GetTickCount();
   PrintFormat("Start of adding %u ticks to the history of the custom symbol '%s'", ar
   int added=CustomTicksAdd(CUSTOM_SYMBOL_NAME, array);
   PrintFormat("Added %u ticks to the history of the custom symbol '%s' in %u ms", add

//--- Abrufen der neu hinzugefügten Tick-Daten des nutzerdefinierten Symbols in das Mo
   Print("...");
   if(!GetTicksToArray(CUSTOM_SYMBOL_NAME, array.Size(), array))
      return;

//--- Ausdruck der Zeit des ersten und letzten empfangenen Ticks des nutzerdefinierter
```

© 2000-2025, MetaQuotes Ltd.

```
   total=(int)array.Size();
   PrintFormat("First tick time: %s.%03u, Last tick time: %s.%03u",
               TimeToString(array[0].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
               TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des nutzerdefinierten Symbols in c
   PrintFormat("\nThe last %d ticks for the custom symbol '%s':", DATATICKS_TO_PRINT,
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
     }

//--- Abrufen der Tick-Zeit in Millisekunden, ab der wir den Bereich der Ticks aus der
   long time_from=array[total-DATATICKS_TO_DELETE-1].time_msc;

//--- Löschen von DATATICKS_TO_DELETE, den Bereich der letzten Ticks des nutzerdefinie
   Print("...");
   start=GetTickCount();
   PrintFormat("Start deleting %u ticks in the history of the custom symbol '%s'", DAT
   int deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, time_from, array[total-2].time_ms
   PrintFormat("Deleted %u ticks in the history of the custom symbol '%s' in %u ms", c

//--- die neu geänderten nutzerdefinierten Symbol-Tick-Daten dem MqlTick-Array zuweise
   Print("...");
   if(!GetTicksToArray(CUSTOM_SYMBOL_NAME, array.Size(), array))
      return;

//--- Ausdruck der Zeit des ersten und letzten Ticks eines nutzerdefinierten Symbols c
   total=(int)array.Size();
   PrintFormat("Time of the first tick from the changed history: %s.%03u, Time of the
               TimeToString(array[0].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
               TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des nutzerdefinierten Symbols in c
   PrintFormat("\nThe last %d ticks of custom symbol '%s' with modified history:", DAT
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
     }

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Sleep(16);
      //--- beim Drücken von Entf das erstellte nutzerdefinierte Symbol und seine Date
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         //--- Balkendaten löschen
         int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history bars of the custom symbol '%s' were successfully d

         //--- Tickdaten löschen
         deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history ticks of the custom symbol '%s' were successfully

         //--- Symbol löschen
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Requested 4294967295 ticks to get tick history for the symbol 'EURUSD'
   The tick history for the 'EURUSD' symbol is received in the amount of 351199027 tic
   First tick time: 2011.12.19 00:00:08.000, Last tick time: 2024.06.21 10:10:40.392

   The last 20 ticks for the standard symbol 'EURUSD':
     351199007th Tick: 2024.06.21 10:10:23.045 Bid=1.07032 (Info tick)
     351199008th Tick: 2024.06.21 10:10:24.045 Ask=1.07031 Bid=1.07031 (Info tick)
     351199009th Tick: 2024.06.21 10:10:24.545 Ask=1.07032 (Info tick)
     351199010th Tick: 2024.06.21 10:10:25.146 Bid=1.07032 (Info tick)
     351199011th Tick: 2024.06.21 10:10:25.649 Ask=1.07037 Bid=1.07037 (Info tick)
     351199012th Tick: 2024.06.21 10:10:27.050 Ask=1.07036 Bid=1.07036 (Info tick)
     351199013th Tick: 2024.06.21 10:10:28.153 Ask=1.07039 Bid=1.07039 (Info tick)
     351199014th Tick: 2024.06.21 10:10:29.157 Ask=1.07037 Bid=1.07037 (Info tick)
     351199015th Tick: 2024.06.21 10:10:29.658 Ask=1.07036 Bid=1.07036 (Info tick)
     351199016th Tick: 2024.06.21 10:10:30.258 Bid=1.07036 (Info tick)
     351199017th Tick: 2024.06.21 10:10:30.872 Ask=1.07035 Bid=1.07035 (Info tick)
     351199018th Tick: 2024.06.21 10:10:31.358 Ask=1.07036 (Info tick)
     351199019th Tick: 2024.06.21 10:10:31.859 Ask=1.07037 Bid=1.07037 (Info tick)
     351199020th Tick: 2024.06.21 10:10:32.377 Ask=1.07039 Bid=1.07039 (Info tick)
     351199021th Tick: 2024.06.21 10:10:32.962 Ask=1.0704 Bid=1.0704 (Info tick)
     351199022th Tick: 2024.06.21 10:10:33.961 Ask=1.07039 Bid=1.07039 (Info tick)
     351199023th Tick: 2024.06.21 10:10:34.667 Ask=1.0704 (Info tick)
     351199024th Tick: 2024.06.21 10:10:35.170 Bid=1.0704 (Info tick)
     351199025th Tick: 2024.06.21 10:10:38.266 Ask=1.07041 Bid=1.07041 (Info tick)
     351199026th Tick: 2024.06.21 10:10:40.392 Ask=1.07042 Bid=1.07042 (Info tick)
```

© 2000-2025, MetaQuotes Ltd.

```
...
Start of adding 351199027 ticks to the history of the custom symbol 'EURUSD.C'
Added 351199027 ticks to the history of the custom symbol 'EURUSD.C' in 261594 ms
...
Requested 351199027 ticks to get tick history for the symbol 'EURUSD.C'
The tick history for the 'EURUSD.C' symbol is received in the amount of 351199027 t
First tick time: 2011.12.19 00:00:08.000, Last tick time: 2024.06.21 10:10:40.392

The last 20 ticks for the custom symbol 'EURUSD.C':
  351199007th Tick: 2024.06.21 10:10:23.045 Ask=1.07032 Bid=1.07032 (Info tick)
  351199008th Tick: 2024.06.21 10:10:24.045 Ask=1.07031 Bid=1.07031 (Info tick)
  351199009th Tick: 2024.06.21 10:10:24.545 Ask=1.07032 Bid=1.07032 (Info tick)
  351199010th Tick: 2024.06.21 10:10:25.146 Ask=1.07032 Bid=1.07032 (Info tick)
  351199011th Tick: 2024.06.21 10:10:25.649 Ask=1.07037 Bid=1.07037 (Info tick)
  351199012th Tick: 2024.06.21 10:10:27.050 Ask=1.07036 Bid=1.07036 (Info tick)
  351199013th Tick: 2024.06.21 10:10:28.153 Ask=1.07039 Bid=1.07039 (Info tick)
  351199014th Tick: 2024.06.21 10:10:29.157 Ask=1.07037 Bid=1.07037 (Info tick)
  351199015th Tick: 2024.06.21 10:10:29.658 Ask=1.07036 Bid=1.07036 (Info tick)
  351199016th Tick: 2024.06.21 10:10:30.258 Ask=1.07036 Bid=1.07036 (Info tick)
  351199017th Tick: 2024.06.21 10:10:30.872 Ask=1.07035 Bid=1.07035 (Info tick)
  351199018th Tick: 2024.06.21 10:10:31.358 Ask=1.07036 Bid=1.07036 (Info tick)
  351199019th Tick: 2024.06.21 10:10:31.859 Ask=1.07037 Bid=1.07037 (Info tick)
  351199020th Tick: 2024.06.21 10:10:32.377 Ask=1.07039 Bid=1.07039 (Info tick)
  351199021th Tick: 2024.06.21 10:10:32.962 Ask=1.0704 Bid=1.0704 (Info tick)
  351199022th Tick: 2024.06.21 10:10:33.961 Ask=1.07039 Bid=1.07039 (Info tick)
  351199023th Tick: 2024.06.21 10:10:34.667 Ask=1.0704 Bid=1.0704 (Info tick)
  351199024th Tick: 2024.06.21 10:10:35.170 Ask=1.0704 Bid=1.0704 (Info tick)
  351199025th Tick: 2024.06.21 10:10:38.266 Ask=1.07041 Bid=1.07041 (Info tick)
  351199026th Tick: 2024.06.21 10:10:40.392 Ask=1.07042 Bid=1.07042 (Info tick)
...
Start deleting 10 ticks in the history of the custom symbol 'EURUSD.C'
Deleted 10 ticks in the history of the custom symbol 'EURUSD.C' in 188 ms
...
Requested 351199027 ticks to get tick history for the symbol 'EURUSD.C'
The tick history for the 'EURUSD.C' symbol is received in the amount of 351199017 t
Time of the first tick from the changed history: 2011.12.19 00:00:08.000, Time of t

The last 20 ticks of custom symbol 'EURUSD.C' with modified history:
  351198997th Tick: 2024.06.21 10:10:14.935 Ask=1.07036 Bid=1.07036 (Info tick)
  351198998th Tick: 2024.06.21 10:10:15.533 Ask=1.07035 Bid=1.07035 (Info tick)
  351198999th Tick: 2024.06.21 10:10:17.736 Ask=1.07036 Bid=1.07036 (Info tick)
  351199000th Tick: 2024.06.21 10:10:18.540 Ask=1.07037 Bid=1.07037 (Info tick)
  351199001th Tick: 2024.06.21 10:10:19.046 Ask=1.07038 Bid=1.07038 (Info tick)
  351199002th Tick: 2024.06.21 10:10:19.542 Ask=1.07036 Bid=1.07036 (Info tick)
  351199003th Tick: 2024.06.21 10:10:20.041 Ask=1.07035 Bid=1.07035 (Info tick)
  351199004th Tick: 2024.06.21 10:10:21.041 Ask=1.07035 Bid=1.07035 (Info tick)
  351199005th Tick: 2024.06.21 10:10:21.544 Ask=1.07032 Bid=1.07032 (Info tick)
  351199006th Tick: 2024.06.21 10:10:22.344 Ask=1.07032 Bid=1.07032 (Info tick)
  351199007th Tick: 2024.06.21 10:10:23.045 Ask=1.07032 Bid=1.07032 (Info tick)
```

© 2000-2025, MetaQuotes Ltd.

```
     351199008th Tick: 2024.06.21 10:10:24.045 Ask=1.07031 Bid=1.07031 (Info tick)
     351199009th Tick: 2024.06.21 10:10:24.545 Ask=1.07032 Bid=1.07032 (Info tick)
     351199010th Tick: 2024.06.21 10:10:25.146 Ask=1.07032 Bid=1.07032 (Info tick)
     351199011th Tick: 2024.06.21 10:10:25.649 Ask=1.07037 Bid=1.07037 (Info tick)
     351199012th Tick: 2024.06.21 10:10:27.050 Ask=1.07036 Bid=1.07036 (Info tick)
     351199013th Tick: 2024.06.21 10:10:28.153 Ask=1.07039 Bid=1.07039 (Info tick)
     351199014th Tick: 2024.06.21 10:10:29.157 Ask=1.07037 Bid=1.07037 (Info tick)
     351199015th Tick: 2024.06.21 10:10:29.658 Ask=1.07036 Bid=1.07036 (Info tick)
     351199016th Tick: 2024.06.21 10:10:40.392 Ask=1.07042 Bid=1.07042 (Info tick)
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastErro
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
     }
//--- Erfolg
   return(true);
  }
//+------------------------------------------------------------------+
//| Abrufen der angegebenen Anzahl von Ticks im Array                |
//+------------------------------------------------------------------+
bool GetTicksToArray(const string symbol, const uint count, MqlTick &array[])
  {
//--- Benachrichtigung über den Beginn des Ladens historischer Daten
   PrintFormat("Requested %u ticks to get tick history for the symbol '%s'", count, sy

//--- 3 Versuche, Ticks abzurufen
   int attempts=0;
   while(attempts<3)
     {
      //--- Startzeit vor Empfang der Ticks messen
      uint start=GetTickCount();

      //--- Anforderung der Tick-Historie seit 1970.01.01 00:00.001 (Parameter von=1 r
      int received=CopyTicks(symbol, array, COPY_TICKS_ALL, 1, count);
      if(received!=-1)
        {
         //--- Informationen über die Anzahl der Ticks und die verwendete Zeit anzeige
         PrintFormat("The tick history for the '%s' symbol is received in the amount o

         //--- Wenn die Tick-Historie synchronisiert ist, ist der Fehlercode gleich Nu
         if(GetLastError()==0)
            return(true);

         PrintFormat("%s: Ticks are not synchronized yet, %d ticks received for %d ms.
                     symbol, received, GetTickCount()-start, GetLastError());
        }
      //--- Versuche zählen
      attempts++;
      //--- eine Pause von einer Sekunde, um auf das Ende der Synchronisierung der Tic
      Sleep(1000);
     }
//--- Konnten die Ticks nach 3 Versuchen nicht kopiert werden:
   return(false);
  }
//+------------------------------------------------------------------+
//| gibt die String-Beschreibung eines Ticks zurück                  |
//+------------------------------------------------------------------+
string GetTickDescription(MqlTick &tick)
  {
   string desc=StringFormat("%s.%03u ", TimeToString(tick.time, TIME_DATE|TIME_MINUTES
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Prüfen der Flags
   bool buy_tick   = ((tick.flags &TICK_FLAG_BUY)    == TICK_FLAG_BUY);
   bool sell_tick  = ((tick.flags &TICK_FLAG_SELL)   == TICK_FLAG_SELL);
   bool ask_tick   = ((tick.flags &TICK_FLAG_ASK)    == TICK_FLAG_ASK);
   bool bid_tick   = ((tick.flags &TICK_FLAG_BID)    == TICK_FLAG_BID);
   bool last_tick  = ((tick.flags &TICK_FLAG_LAST)   == TICK_FLAG_LAST);
   bool volume_tick= ((tick.flags &TICK_FLAG_VOLUME)== TICK_FLAG_VOLUME);

//--- zuerst den Tick auf die Handelsflags prüfen (es gibt keine für CustomTicksAdd())
   if(buy_tick || sell_tick)
     {
      //--- eine Ausgabe für einen Handelstick bilden
      desc += (buy_tick ? StringFormat("Buy Tick: Last=%G Volume=%d ", tick.last, tick
      desc += (sell_tick? StringFormat("Sell Tick: Last=%G Volume=%d ",tick.last, tick
      desc += (ask_tick ? StringFormat("Ask=%G ", tick.ask) : "");
      desc += (bid_tick ? StringFormat("Bid=%G ", tick.ask) : "");
      desc += "(Trade tick)";
     }
   else
     {
      //--- die Ausgabe für einen Info-Tick etwas anders gestalten
      desc += (ask_tick   ? StringFormat("Ask=%G ",  tick.ask)    : "");
      desc += (bid_tick   ? StringFormat("Bid=%G ",  tick.ask)    : "");
      desc += (last_tick  ? StringFormat("Last=%G ", tick.last)   : "");
      desc += (volume_tick? StringFormat("Volume=%d ",tick.volume): "");
      desc += "(Info tick)";
     }
//--- Tick-Beschreibung zurückgeben
   return(desc);
  }
```

Siehe auch

CustomRatesDelete, CustomRatesUpdate, CustomTicksReplace, CopyTicks, CopyTicksRange

© 2000-2025, MetaQuotes Ltd.

CustomTicksReplace

Ersetzt die komplette Preishistorie eines nutzerdefinierten Symbols im angegebenen Zeitintervall durch die Daten aus einem Array vom Typ MqlTick.

```
int  CustomTicksReplace(
   const string     symbol,            // Symbolname
   long             from_msc,          // von welchem Tag
   long             to_msc,            // bis zu welchem Tag
   const MqlTick&   ticks[],           // Array mit den Tickdaten, die dem nutzerdefin
   uint             count=WHOLE_ARRAY  // Anzahl der zu verwendenden Elemente des Arra
   );
```

Parameter

symbol

[in] Name des nutzerdefinierten Symbols.

from_msc

[in] Zeit des ersten Ticks in der Preishistorie aus dem angegebenen Intervall, der gelöscht werden muss. Zeit in Millisekunden ab dem 01.01.1970.

to_msc

[in] Zeit des letzten Ticks in der Preishistorie aus dem angegebenen Intervall, der gelöscht werden muss. Zeit in Millisekunden ab dem 01.01.1970.

ticks[]

[in] Array mit den Tickdaten vom Typ MqlTick, die nach Zeit aufsteigend geordnet wurden.

count=WHOLE_ARRAY

[in] Anzahl der zu verwendenden Elemente des Arrays *ticks[]*, die für das Ersetzen im angegebenen Zeitintervall verwendet werden. WHOLE_ARRAY bedeutet, es werden alle Elemente des Arrays *ticks[]* hinzugefügt.

Rückgabewert

Anzahl der aktualisierten Balken oder -1 im Fehlerfall.

Hinweis

Da gleichzeitig mehrere Ticks im Kurs-Feed eine und dieselbe Zeit bis auf Millisekunden haben können (die genaue Zeit eines Ticks wird im Feld*time_msc* der Struktur MqlTick gespeichert), sortiert die Funktion CustomTicksReplace die Elemente des Array *ticks[]* nicht automatisch nach Zeit. Aus diesem Grund muss das Tick-Array zuerst nach Zeit aufsteigend geordnet werden.

Die Ersetzung von Ticks erfolgt konsequent von Tag zu Tag bis zu dem in *to_msc* angegebenen Zeitpunkt oder bis zum Auftreten eines Fehlers. Zuerst wird der erste Tag aus dem angegebenen Intervall verarbeitet, dann der nächste und so weiter. Sobald es festgestellt wird, dass die Zeit ein eines Ticks nicht der aufsteigenden (nicht absteigenden) Folge entspricht, wird der Vorgang sofort am aktuellen Tag abgebrochen. Dabei werden die Ticks der vorherigen Tage erfolgreich ersetzt, und der aktuelle Tag (im Moment des Eintreffens des falschen Ticks) sowie alle restlichen Tage im angegebenen Intervall bleiben unverändert.

© 2000-2025, MetaQuotes Ltd.

Wenn im Array *ticks[]* Tickdaten für einen Tag fehlen (gilt auch für jedes Intervall mit beliebiger Länge), entsteht nach der Anwendung von Tick-Daten mit `ticks[]` eine Lücke, die den fehlenden Tagen entspricht. Mit anderen Worten ist der Aufruf von CustomTicksReplace mit gelöschten Ticks für ein konkretes Intervall gleich dem Löschen eines Teils der Tick-Historie, als ob CustomTicksDelete mit dem Intervall "Lücke" aufgerufen wird.

Wenn es keine Daten für den angegebenen Zeitraum in der Datenbank von Ticks gibt, fügt CustomTicksReplace der Datenbank Ticks aus dem Array ticks[] hinzu.

Die Funktion CustomTicksReplace arbeitet direkt mit der Datenbank von Ticks.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                          CustomTicksReplace.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#define   CUSTOM_SYMBOL_NAME     Symbol()+".C"     // Name des nutzerdefinierten Symbo
#define   CUSTOM_SYMBOL_PATH     "Forex"           // Name der Gruppe, in der das Symb
#define   CUSTOM_SYMBOL_ORIGIN   Symbol()          // Name des Symbols, das als Basis

#define   DATATICKS_TO_COPY      UINT_MAX          // Anzahl der kopierten Ticks
#defineDATATICKS_TO_PRINT     20                // Anzahl der an das Log gesendeten Ti

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen des Fehlercodes beim Erstellen eines nutzerdefinierten Symbols
   int create=CreateCustomSymbol(CUSTOM_SYMBOL_NAME, CUSTOM_SYMBOL_PATH, CUSTOM_SYMBOL

//--- wenn der Fehlercode nicht 0 (erfolgreiche Symbolerstellung) und nicht 5304 (Symb
   if(create!=0 && create!=5304)
      return;

//--- Abrufen der Tick-Daten des Standardsymbols in einem MqlTick-Array
   MqlTick array[]={};
   if(!GetTicksToArray(CUSTOM_SYMBOL_ORIGIN, DATATICKS_TO_COPY, array))
      return;

//--- Ausdruck der Zeit des ersten und des letzten empfangenen Ticks des Standardsymbo
   int total=(int)array.Size();
   PrintFormat("First tick time: %s.%03u, Last tick time: %s.%03u",
```

© 2000-2025, MetaQuotes Ltd.

```
                TimeToString(array[0].time,TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
                TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des Standardsymbols in das Log
   PrintFormat("\nThe last %d ticks for the standard symbol '%s':", DATATICKS_TO_PRINT
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
     }

//--- ein nutzerdefiniertes Symbol zum Fenster der Marktübersicht hinzufügen
   ResetLastError();
   if(!SymbolSelect(CUSTOM_SYMBOL_NAME, true))
     {
      Print("SymbolSelect() failed. Error ", GetLastError());
      return;
     }

//--- Hinzufügen der Daten des Tick-Arrays zur Preishistorie des nutzerdefinierten Sym
   Print("...");
   uint start=GetTickCount();
   PrintFormat("Start of adding %u ticks to the history of the custom symbol '%s'", ar
   int added=CustomTicksAdd(CUSTOM_SYMBOL_NAME, array);
   PrintFormat("Added %u ticks to the history of the custom symbol '%s' in %u ms", add

//--- Abrufen der neu hinzugefügten Tick-Daten des nutzerdefinierten Symbols in das Mo
   Print("...");
   if(!GetTicksToArray(CUSTOM_SYMBOL_NAME, array.Size(), array))
      return;

//--- Ausdruck der Zeit des ersten und letzten empfangenen Ticks des nutzerdefinierten
   total=(int)array.Size();
   PrintFormat("First tick time: %s.%03u, Last tick time: %s.%03u",
                TimeToString(array[0].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
                TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT letzten Ticks des nutzerdefinierten Symbols in d
   PrintFormat("\nThe last %d ticks for the custom symbol '%s':", DATATICKS_TO_PRINT,
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Tick: %s", i, GetTickDescription(array[i]));
     }

//--- jetzt werden die Ask- und Bid-Tickwerte im Array mit der Gleichung Ask(Symbol) =
   for(int i=0; i<total; i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       array[i].ask = (array[i].ask !=0 ? 1.0 / array[i].ask : array[i].ask);
       array[i].bid = (array[i].bid !=0 ? 1.0 / array[i].bid : array[i].bid);
      }
   Print("\nNow the ticks are changed");

//--- Ersetzen der Tick-Historie des nutzerdefinierten Symbols durch Daten aus dem geä
   Print("...");
   start=GetTickCount();
   PrintFormat("Start replacing %u changed ticks in the history of the custom symbol '
   int replaced=CustomTicksReplace(CUSTOM_SYMBOL_NAME, array[0].time_msc, array[total-
   PrintFormat("Replaced %u ticks in the history of the custom symbol '%s' in %u ms",

//--- Abrufen der neu ersetzten, nutzerdefinierten Tick-Daten des Symbols in das MqlTi
   Print("...");
   if(!GetTicksToArray(CUSTOM_SYMBOL_NAME, array.Size(), array))
      return;

//--- Ausdruck der Zeit des ersten und letzten empfangenen, geänderten Ticks des nutze
   total=(int)array.Size();
   PrintFormat("First changed tick time: %s.%03u, Last changed tick time: %s.%03u",
               TimeToString(array[0].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS), array
               TimeToString(array[total-1].time, TIME_DATE|TIME_MINUTES|TIME_SECONDS),

//--- Ausdruck von DATATICKS_TO_PRINT, der zuletzt geänderte Ticks des nutzerdefiniert
   PrintFormat("\nThe last %d changed ticks for the custom symbol '%s':", DATATICKS_TO
   for(int i=total-DATATICKS_TO_PRINT; i<total; i++)
     {
      if(i<0)
         continue;
      PrintFormat("  %dth Changed tick: %s", i, GetTickDescription(array[i]));
     }

//--- einen Hinweis zu den Tasten zur Beendigung des Skripts im Kommentar des Charts a
   Comment(StringFormat("Press 'Esc' to exit or 'Del' to delete the '%s' symbol and ex
//--- warten, bis die Tasten „Esc" oder „Entf" gedrückt werden, um die Endlosschleife
   while(!IsStopped() && TerminalInfoInteger(TERMINAL_KEYSTATE_ESCAPE)==0)
     {
      Sleep(16);
      //--- beim Drücken von Entf das erstellte nutzerdefinierte Symbol und seine Date
      if(TerminalInfoInteger(TERMINAL_KEYSTATE_DELETE)<0)
        {
         //--- Balkendaten löschen
         int deleted=CustomRatesDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
         if(deleted>0)
            PrintFormat("%d history bars of the custom symbol '%s' were successfully d

         //--- Tickdaten löschen
         deleted=CustomTicksDelete(CUSTOM_SYMBOL_NAME, 0, LONG_MAX);
```

© 2000-2025, MetaQuotes Ltd.

```
         if(deleted>0)
            PrintFormat("%d history ticks of the custom symbol '%s' were successfully

         //--- Symbol löschen
         if(DeleteCustomSymbol(CUSTOM_SYMBOL_NAME))
            PrintFormat("Custom symbol '%s' deleted successfully", CUSTOM_SYMBOL_NAME)
         break;
        }
     }
//--- Chart vor dem Ende löschen
   Comment("");
   /*
   Ergebnis:
   Requested 4294967295 ticks to get tick history for the symbol 'EURUSD'
   The tick history for the 'EURUSD' symbol is received in the amount of 351195822 tic
   First tick time: 2011.12.19 00:00:08.000, Last tick time: 2024.06.21 08:39:03.113

   The last 20 ticks for the standard symbol 'EURUSD':
     351195802th Tick: 2024.06.21 08:38:10.076 Ask=1.07194 (Info tick)
     351195803th Tick: 2024.06.21 08:38:13.162 Ask=1.07195 (Info tick)
     351195804th Tick: 2024.06.21 08:38:13.872 Bid=1.07195 (Info tick)
     351195805th Tick: 2024.06.21 08:38:14.866 Ask=1.07194 Bid=1.07194 (Info tick)
     351195806th Tick: 2024.06.21 08:38:17.374 Bid=1.07194 (Info tick)
     351195807th Tick: 2024.06.21 08:38:18.883 Bid=1.07194 (Info tick)
     351195808th Tick: 2024.06.21 08:38:19.771 Bid=1.07194 (Info tick)
     351195809th Tick: 2024.06.21 08:38:20.873 Ask=1.07195 Bid=1.07195 (Info tick)
     351195810th Tick: 2024.06.21 08:38:22.278 Ask=1.07196 Bid=1.07196 (Info tick)
     351195811th Tick: 2024.06.21 08:38:22.775 Bid=1.07196 (Info tick)
     351195812th Tick: 2024.06.21 08:38:23.477 Bid=1.07196 (Info tick)
     351195813th Tick: 2024.06.21 08:38:38.194 Ask=1.07197 (Info tick)
     351195814th Tick: 2024.06.21 08:38:38.789 Ask=1.07196 (Info tick)
     351195815th Tick: 2024.06.21 08:38:39.290 Ask=1.07197 (Info tick)
     351195816th Tick: 2024.06.21 08:38:43.695 Ask=1.07196 (Info tick)
     351195817th Tick: 2024.06.21 08:38:52.203 Ask=1.07195 Bid=1.07195 (Info tick)
     351195818th Tick: 2024.06.21 08:38:55.105 Ask=1.07196 Bid=1.07196 (Info tick)
     351195819th Tick: 2024.06.21 08:38:57.607 Ask=1.07195 Bid=1.07195 (Info tick)
     351195820th Tick: 2024.06.21 08:39:00.512 Ask=1.07196 Bid=1.07196 (Info tick)
     351195821th Tick: 2024.06.21 08:39:03.113 Ask=1.07195 Bid=1.07195 (Info tick)
   ...
   Start of adding 351195822 ticks to the history of the custom symbol 'EURUSD.C'
   Added 351195822 ticks to the history of the custom symbol 'EURUSD.C' in 349407 ms
   ...
   Requested 351195822 ticks to get tick history for the symbol 'EURUSD.C'
   The tick history for the 'EURUSD.C' symbol is received in the amount of 351195822 t
   First tick time: 2011.12.19 00:00:08.000, Last tick time: 2024.06.21 08:39:03.113

   The last 20 ticks for the custom symbol 'EURUSD.C':
     351195802th Tick: 2024.06.21 08:38:10.076 Ask=1.07194 Bid=1.07194 (Info tick)
     351195803th Tick: 2024.06.21 08:38:13.162 Ask=1.07195 Bid=1.07195 (Info tick)
```

© 2000-2025, MetaQuotes Ltd.

```
   351195804th Tick: 2024.06.21 08:38:13.872 Ask=1.07195 Bid=1.07195 (Info tick)
   351195805th Tick: 2024.06.21 08:38:14.866 Ask=1.07194 Bid=1.07194 (Info tick)
   351195806th Tick: 2024.06.21 08:38:17.374 Ask=1.07194 Bid=1.07194 (Info tick)
   351195807th Tick: 2024.06.21 08:38:18.883 Ask=1.07194 Bid=1.07194 (Info tick)
   351195808th Tick: 2024.06.21 08:38:19.771 Ask=1.07194 Bid=1.07194 (Info tick)
   351195809th Tick: 2024.06.21 08:38:20.873 Ask=1.07195 Bid=1.07195 (Info tick)
   351195810th Tick: 2024.06.21 08:38:22.278 Ask=1.07196 Bid=1.07196 (Info tick)
   351195811th Tick: 2024.06.21 08:38:22.775 Ask=1.07196 Bid=1.07196 (Info tick)
   351195812th Tick: 2024.06.21 08:38:23.477 Ask=1.07196 Bid=1.07196 (Info tick)
   351195813th Tick: 2024.06.21 08:38:38.194 Ask=1.07197 Bid=1.07197 (Info tick)
   351195814th Tick: 2024.06.21 08:38:38.789 Ask=1.07196 Bid=1.07196 (Info tick)
   351195815th Tick: 2024.06.21 08:38:39.290 Ask=1.07197 Bid=1.07197 (Info tick)
   351195816th Tick: 2024.06.21 08:38:43.695 Ask=1.07196 Bid=1.07196 (Info tick)
   351195817th Tick: 2024.06.21 08:38:52.203 Ask=1.07195 Bid=1.07195 (Info tick)
   351195818th Tick: 2024.06.21 08:38:55.105 Ask=1.07196 Bid=1.07196 (Info tick)
   351195819th Tick: 2024.06.21 08:38:57.607 Ask=1.07195 Bid=1.07195 (Info tick)
   351195820th Tick: 2024.06.21 08:39:00.512 Ask=1.07196 Bid=1.07196 (Info tick)
   351195821th Tick: 2024.06.21 08:39:03.113 Ask=1.07195 Bid=1.07195 (Info tick)

Now the ticks are changed
...
Start replacing 351195822 changed ticks in the history of the custom symbol 'EURUSI
Replaced 351195822 ticks in the history of the custom symbol 'EURUSD.C' in 452266 r
...
Requested 351195822 ticks to get tick history for the symbol 'EURUSD.C'
The tick history for the 'EURUSD.C' symbol is received in the amount of 351195822 t
First changed tick time: 2011.12.19 00:00:08.000, Last changed tick time: 2024.06.2

The last 20 changed ticks for the custom symbol 'EURUSD.C':
   351195802th Changed tick: 2024.06.21 08:38:10.076 Ask=0.93289 Bid=0.93289 (Info t
   351195803th Changed tick: 2024.06.21 08:38:13.162 Ask=0.93288 Bid=0.93288 (Info t
   351195804th Changed tick: 2024.06.21 08:38:13.872 Ask=0.93288 Bid=0.93288 (Info t
   351195805th Changed tick: 2024.06.21 08:38:14.866 Ask=0.93289 Bid=0.93289 (Info t
   351195806th Changed tick: 2024.06.21 08:38:17.374 Ask=0.93289 Bid=0.93289 (Info t
   351195807th Changed tick: 2024.06.21 08:38:18.883 Ask=0.93289 Bid=0.93289 (Info t
   351195808th Changed tick: 2024.06.21 08:38:19.771 Ask=0.93289 Bid=0.93289 (Info t
   351195809th Changed tick: 2024.06.21 08:38:20.873 Ask=0.93288 Bid=0.93288 (Info t
   351195810th Changed tick: 2024.06.21 08:38:22.278 Ask=0.93287 Bid=0.93287 (Info t
   351195811th Changed tick: 2024.06.21 08:38:22.775 Ask=0.93287 Bid=0.93287 (Info t
   351195812th Changed tick: 2024.06.21 08:38:23.477 Ask=0.93287 Bid=0.93287 (Info t
   351195813th Changed tick: 2024.06.21 08:38:38.194 Ask=0.93286 Bid=0.93286 (Info t
   351195814th Changed tick: 2024.06.21 08:38:38.789 Ask=0.93287 Bid=0.93287 (Info t
   351195815th Changed tick: 2024.06.21 08:38:39.290 Ask=0.93286 Bid=0.93286 (Info t
   351195816th Changed tick: 2024.06.21 08:38:43.695 Ask=0.93287 Bid=0.93287 (Info t
   351195817th Changed tick: 2024.06.21 08:38:52.203 Ask=0.93288 Bid=0.93288 (Info t
   351195818th Changed tick: 2024.06.21 08:38:55.105 Ask=0.93287 Bid=0.93287 (Info t
   351195819th Changed tick: 2024.06.21 08:38:57.607 Ask=0.93288 Bid=0.93288 (Info t
   351195820th Changed tick: 2024.06.21 08:39:00.512 Ask=0.93287 Bid=0.93287 (Info t
   351195821th Changed tick: 2024.06.21 08:39:03.113 Ask=0.93288 Bid=0.93288 (Info t
```

© 2000-2025, MetaQuotes Ltd.

```
   */
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol erstellen, Fehlercode zurückgeben       |
//+------------------------------------------------------------------+
int CreateCustomSymbol(const string symbol_name, const string symbol_path, const strin
  {
//--- Definition des Namens eines Symbols, auf dem ein nutzerdefiniertes Symbol basier
   string origin=(symbol_origin==NULL ? Symbol() : symbol_origin);

//--- Wenn das Erstellen eines nutzerdefinierten Symbols fehlgeschlagen ist und nicht
   ResetLastError();
   int error=0;
   if(!CustomSymbolCreate(symbol_name, symbol_path, origin))
     {
      error=GetLastError();
      if(error!=5304)
         PrintFormat("CustomSymbolCreate(%s, %s, %s) failed. Error %d", symbol_name, s
     }
//--- Erfolg
   return(error);
  }
//+------------------------------------------------------------------+
//| Nutzerdefiniertes Symbol entfernen                               |
//+------------------------------------------------------------------+
bool DeleteCustomSymbol(const string symbol_name)
  {
//--- das Symbol aus dem Fenster der Marktübersicht ausblenden
   ResetLastError();
   if(!SymbolSelect(symbol_name, false))
     {
      PrintFormat("SymbolSelect(%s, false) failed. Error %d", GetLastError());
      return(false);
     }

//--- Wenn das Löschen eines nutzerdefinierten Symbols fehlgeschlagen ist, wird das in
   ResetLastError();
   if(!CustomSymbolDelete(symbol_name))
     {
      PrintFormat("CustomSymbolDelete(%s) failed. Error %d", symbol_name, GetLastError
      return(false);
     }
//--- Erfolg
   return(true);
  }
//+------------------------------------------------------------------+
//| Abrufen der angegebenen Anzahl von Ticks im Array                |
//+------------------------------------------------------------------+
bool GetTicksToArray(const string symbol, const uint count, MqlTick &array[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Benachrichtigung über den Beginn des Ladens historischer Daten
   PrintFormat("Requested %u ticks to get tick history for the symbol '%s'", count, sy

//--- 3 Versuche, Ticks abzurufen
   int attempts=0;
   while(attempts<3)
     {
      //--- Startzeit vor Empfang der Ticks messen
      uint start=GetTickCount();

      //--- Anforderung der Tick-Historie seit 1970.01.01 00:00.001 (Parameter von=1 r
      int received=CopyTicks(symbol, array, COPY_TICKS_ALL, 1, count);
      if(received!=-1)
        {
         //--- Informationen über die Anzahl der Ticks und die verwendete Zeit anzeige
         PrintFormat("The tick history for the '%s' symbol is received in the amount o

         //--- Wenn die Tick-Historie synchronisiert ist, ist der Fehlercode gleich Nu
         if(GetLastError()==0)
            return(true);

         PrintFormat("%s: Ticks are not synchronized yet, %d ticks received for %d ms.
                     symbol, received, GetTickCount()-start, GetLastError());
        }
      //--- Versuche zählen
      attempts++;
      //--- eine Pause von einer Sekunde, um auf das Ende der Synchronisierung der Tic
      Sleep(1000);
     }
//--- Konnten die Ticks nach 3 Versuchen nicht kopiert werden:
   return(false);
  }
//+------------------------------------------------------------------+
//| gibt die String-Beschreibung eines Ticks zurück                  |
//+------------------------------------------------------------------+
string GetTickDescription(MqlTick &tick)
  {
   string desc=StringFormat("%s.%03u ", TimeToString(tick.time, TIME_DATE|TIME_MINUTES

//--- Prüfen der Flags
   bool buy_tick   = ((tick.flags &TICK_FLAG_BUY)    == TICK_FLAG_BUY);
   bool sell_tick  = ((tick.flags &TICK_FLAG_SELL)   == TICK_FLAG_SELL);
   bool ask_tick   = ((tick.flags &TICK_FLAG_ASK)    == TICK_FLAG_ASK);
   bool bid_tick   = ((tick.flags &TICK_FLAG_BID)    == TICK_FLAG_BID);
   bool last_tick  = ((tick.flags &TICK_FLAG_LAST)   == TICK_FLAG_LAST);
   bool volume_tick= ((tick.flags &TICK_FLAG_VOLUME)== TICK_FLAG_VOLUME);

//--- zuerst den Tick auf die Handelsflags prüfen (es gibt keine für CustomTicksAdd())
```

© 2000-2025, MetaQuotes Ltd.

```
   if(buy_tick || sell_tick)
     {
      //--- eine Ausgabe für einen Handelstick bilden
      desc += (buy_tick ? StringFormat("Buy Tick: Last=%G Volume=%d ", tick.last, tick
      desc += (sell_tick? StringFormat("Sell Tick: Last=%G Volume=%d ",tick.last, tick
      desc += (ask_tick ? StringFormat("Ask=%G ", tick.ask) : "");
      desc += (bid_tick ? StringFormat("Bid=%G ", tick.ask) : "");
      desc += "(Trade tick)";
     }
   else
     {
      //--- die Ausgabe für einen Info-Tick etwas anders gestalten
      desc += (ask_tick   ? StringFormat("Ask=%G ",  tick.ask)    : "");
      desc += (bid_tick   ? StringFormat("Bid=%G ",  tick.ask)    : "");
      desc += (last_tick  ? StringFormat("Last=%G ", tick.last)   : "");
      desc += (volume_tick? StringFormat("Volume=%d ",tick.volume): "");
      desc += "(Info tick)";
     }
//--- Tick-Beschreibung zurückgeben
   return(desc);
  }
```

Siehe auch

CustomRatesDelete, CustomRatesUpdate, CustomTicksDelete, CopyTicks, CopyTicksRange

© 2000-2025, MetaQuotes Ltd.

CustomBookAdd

Überträgt den Status der Markttiefe (Depth of Market) für ein nutzerdefiniertes Symbol. Die Funktion erlaubt die Übertragung der Markttiefe als ob die Preise vom Server des Brokers kommen.

```
bool  CustomBookAdd(
   const string        symbol,            // Symbolname
   const MqlBookInfo&  books[]            // Array der Beschreibungen der Elemente der
   uint                count=WHOLE_ARRAY  // Anzahl der verwendeten Elemente
   );
```

Parameter

symbol

[in] Name des benutzerdefinierten Symbols.

books[]

[in] Der Array mit den Daten vom Typ MqlBookInfo, die den Status der Markttiefe komplett beschreiben — alle Kauf- und Verkaufsanfragen. Der übergebene Status der Markttiefe ersetzt vollständig den vorherigen.

count=WHOLE_ARRAY

[in] Die Anzahl der Elemente im Array 'books', die der Funktion zu übergeben ist. Standardmäßig wird ganze Array verwendet.

Rückgabewert

true - Erfolge, sonst - false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() auf.

Hinweis

Die Funktion CustomBookAdd arbeitet nur mit benutzerdefinierten Symbolen, für die die Markttiefe geöffnet wird — über die Plattformoberfläche oder die Funktion MarketBookAdd.

Wenn Sie die Markttiefe öffnen, werden die Geld- und Briefkurse des Symbols nicht aktualisiert. Sie sollten die Änderung der Bestens-Preise kontrollieren und die Ticks mit CustomTicksAdd. übergeben.

Die Funktion prüft die Richtigkeit der übermittelten Daten: für jedes Element müssen Art, Preis und Volumen angegeben werden. Außerdem dürfen MqlBookInfo.volume und MqlBookInfo.volume_real weder null oder negativ sein; sind beide Volumina negativ, wird dies als Fehler gewertet. Sie können eines der Volumen oder beide angeben: Das angegebene oder positive Volumen wird verwendet:

```
volume=-1 && Volumen_real=2 — volume_real=2 wird verwendet,
a Volumen=3 && volume_real=0 — volume=3 wird verwendet.
```

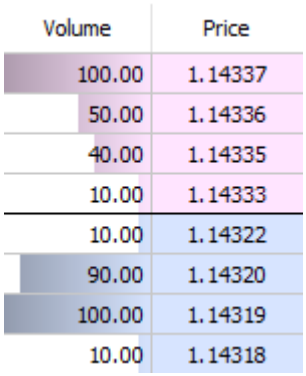
Die Reihenfolge der Elemente von MqlBookInfo im Array 'books' spielt keine Rolle. Beim Speichern der Daten sortiert das Terminal diese selbstständig nach dem Preis.

Beim Speichern von Daten wird der Parameter "Book depth" (SYMBOL_TICKS_BOOKDEPTH) des nutzerdefinierten Empfängersymbols überprüft. Wenn die Anzahl der Verkaufsanfragen diesen Wert in der übergebenen Markttiefe überschreitet, werden die überzähligen Level verworfen. Das Gleiche gilt für Kaufanfragen.

© 2000-2025, MetaQuotes Ltd.

Das Beispiel einer Zuweisung zum Array 'books':

Status der Markttiefe	Ausfüllen von books[]

	books[0].type=BOOK_TYPE_SELL;
	books[0].price=1.14337;
	books[0].volume=100;
	books[1].type=BOOK_TYPE_SELL;
	books[1].price=1.14330;
	books[1].volume=50;
	books[2].type=BOOK_TYPE_SELL;
	books[2].price=1.14335;
	books[2].volume=40;
	books[3].type=BOOK_TYPE_SELL;
	books[3].price=1.14333;
	books[3].volume=10;
	books[4].type=BOOK_TYPE_BUY;
	books[4].price=1.14322;
	books[4].volume=10;
	books[5].type=BOOK_TYPE_BUY;
	books[5].price=1.14320;
	books[5].volume=90;
	books[6].type=BOOK_TYPE_BUY;
	books[6].price=1.14319;
	books[6].volume=100;
	books[7].type=BOOK_TYPE_BUY;
	books[7].price=1.14318;
	books[7].volume=10;

Beispiel:

```
//+------------------------------------------------------------------+
//| Expert-Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- aktivieren der Markttiefe für ein Symbol, für das wir Daten erhalten
   MarketBookAdd(Symbol());
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert Deinitialisierungsfunktion                                |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
  }
//+------------------------------------------------------------------+
//| Tickdaten - Funktion                                             |
//+------------------------------------------------------------------+
void OnTick(void)
  {
   MqlTick ticks[];
   ArrayResize(ticks,1);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Kopieren der aktuellen Preise des allgemeinen Symbols für das nutzerdefinierte
   if(SymbolInfoTick(Symbol(),ticks[0]))
     {
      string symbol_name=Symbol()+".SYN";
      CustomTicksAdd(symbol_name,ticks);
     }
  }
//+------------------------------------------------------------------+
//| Book - Funktion                                                  |
//+------------------------------------------------------------------+
voidOnBookEvent(const string &book_symbol)
  {
//--- Kopieren des Status' der Markttiefe vom allgemeinen Symbol für das nutzerdefinie
   if(book_symbol==Symbol())
     {
      MqlBookInfo book_array[];
      if(MarketBookGet(Symbol(),book_array))
        {
         string symbol_name=Symbol()+".SYN";
         CustomBookAdd(symbol_name,book_array);
        }
     }
  }
//+------------------------------------------------------------------+
```

Siehe auch

MarketBookAdd, CustomTicksAdd, OnBookEvent

© 2000-2025, MetaQuotes Ltd.

Operationen mit Charts

Funktionen zum Setzen der Eigenschaften eines Charts (ChartSetInteger, ChartSetDouble, ChartSetString) sind asynchron und werden für das Senden von Aktualisierungsbefehlen an das Chart verwendet. Werden diese Funktionen erfolgreich ausgeführt, wird der Befehl zur Warteschlange der Ereignisse des Charts hinzugefügt. Die Änderungen der Eigenschaften werden in der Reihenfolge der Verarbeitung der Ereignisse des Charts implementiert.

Aus diesem Grund sollte man keine sofortige Aktualisierung des Charts nach dem Aufruf asynchroner Funktionen erwarten. Für eine erzwungene Aktualisierung der Ansicht und der Eigenschaften des Charts verwenden Sie die Funktion ChartRedraw().

Funktion	Massnahme
ChartApplyTemplate	Wendet ein Template auf das angegebene Chart aus der angegebenen Datei an
ChartSaveTemplate	Speichert aktuelle Chart-Einstellungen in einem Template mit dem angegebenen Namen
ChartWindowFind	Gibt die Nummer des Unterfensters zurück, in welchem der Indikator gezeichnet wird
ChartTimePriceToXY	Konvertiert die Koordinaten eines Charts von der Darstellung Zeit/Geld in X- und Y-Koordinaten
ChartXYToTimePrice	Konvertiert die X- und Y-Koordinaten eines Charts in Zeit und Preis
ChartOpen	Öffnet ein neues Chart mit dem angegebenen Symbol und Periode
ChartClose	Schließt das angegebene Chart
ChartFirst	Gibt den Identifikator des ersten Charts des Terminals zurück
ChartNext	Gibt den Identifikator des Charts zurück, der auf das angegebene Chart folgt
ChartSymbol	Gibt den Symbolnamen des angegebenen Charts zurück
ChartPeriod	Gibt den Wert der Periode des angegebenen Charts zurück
ChartRedraw	Erzwingt das Neuzeichnen des angegebenen Charts
ChartSetDouble	Setzt den double Wert der entsprechenden Eigenschaft des angegebenen Charts
ChartSetInteger	Setzt den integer Wert (datetime, int, color, bool oder char) der entsprechenden Eigenschaft des angegebenen Charts
ChartSetString	Setzt den string Wert der entsprechenden Eigenschaft des angegebenen Charts
ChartGetDouble	Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück
ChartGetInteger	Gibt den integer Wert der entsprechenden Eigenschaft des angegebenen Charts zurück

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
ChartGetString	Gibt den string Wert der entsprechenden Eigenschaft des angegebenen Charts zurück
ChartNavigate	Verschiebt den angegebenen Chart um die angegebene Anzahl der Balken hinsichtlich der angegebenen Position des Charts
ChartID	Gibt den Indikator des aktuellen Charts zurück
ChartIndicatorAdd	Fügt den Indikator mit dem angegebenen Handle zum angegebenen Chartfenster hinzu
ChartIndicatorDelete	Löscht den Indikator mit dem angegebenen Namen vom angegebenen Chartfenster
ChartIndicatorGet	Gibt das Handle des Indikators mit dem angegebenen Kurznamen im angegebenen Chartfenster zurück
ChartIndicatorName	Gibt den Kurznamen des Indikators nach der Nummer in der Liste der Indikatoren im angegebenen Chartfenster zurück
ChartIndicatorsTotal	Gibt die Anzahl aller Indikatoren im angegebenen Chartfenster zurück
ChartWindowOnDropped	Gibt die Nummer des Unterfensters zurück, in das der Expert Advisor, Skript, Objekt oder Indikator platziert worden sind
ChartPriceOnDropped	Gibt die Preiskoordinate des Punktes zurück, in den das Script oder der Expert Advisor platziert worden sind
ChartTimeOnDropped	Gibt die Preiskoordinate des Punktes zurück, in den der Expert Advisor oder das Script platziert worden sind
ChartXOnDropped	Gibt die X-Koordinate des Punktes zurück, in den der Expert Advisor oder das Script platziert worden sind
ChartYOnDropped	Gibt die Y-Koordinate des Punktes zurück, in den der Expert Advisor oder das Script platziert worden sind
ChartSetSymbolPeriod	Ändert die Werte des Symbols und der Periode des angegebenen Charts
ChartScreenShot	Macht einen Screenshot des angegebenen Charts in Format GIF, PNG oder BMP je nach angegebener Erweiterung

© 2000-2025, MetaQuotes Ltd.

ChartApplyTemplate

Wendet zum Chart die angegebene Schablone zu. Der Befehl wird in die Warteschlange der Chartnachrichten gestellt und wird nach der Prozessirrung aller vorherigen Befehle abgearbeitet werden.

```
bool  ChartApplyTemplate(
   long          chart_id,     // Identifikator des Charts
   const string  filename      // Dateiname mit der Schablone
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

filename

[in] Dateiname mit dem Symbol.

Rückgabewert

Gibt true zurück, wenn der Befehl in die Warteschlange des Charts gestellt wurde, andernfalls gibt false zurück. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Wenn durch diese Funktion eine neue Schablone in den Chart geladen wird, zu der die Schablone angehängt ist, wird Expert ausgeladen und seine Arbeit nicht fortsetzen kann.

Aus Sicherheitsgründen kann die Erlaubnis zum Handeln unter Anwendung einer Vorlage begrenzt werden:

Die Erlaubnis zum Handeln kann nicht beim Start des Expert Advisors mithilfe der Vorlage anhand der ChartApplyTemplate() Funktion erweitert werden.

Wenn das mql5-Programm, das die ChartApplyTemplate() Funktion aufruft, über keine Erlaubnis zum Handeln verfügt, wird der Expert Advisor, der über die Vorlage gestartet wurde, auch keine Rechte auf Handel haben unabhängig von den Einstellungen der Vorlage.

Wenn das mql5-Programm, das die ChartApplyTemplate() Funktion aufruft, die Erlaubnis zum Handeln hat, und in den Einstellungen der Vorlage gibt es die Erlaubnis nicht, wird der Expert Advisor, der mit der Vorlage gestartet wurde, auch keine Erlaubnis zum Handel haben.

Verwendung von Vorlagen

Mit Hilfe der Mittel der MQL5-Sprache können Sie viele Eigenschaften des Charts definieren, einschließlich der Farben mit der ChartSetInteger()-Funktion:

- Hintergrundfarbe des Charts;
- Farbe der Achsen, Skalen und OHLC-Reihen;
- Grid-Farbe;
- Farbe der Volumen und Position offen Ebenen;

© 2000-2025, MetaQuotes Ltd.

- Farbe des up-Bars, Schatten und Rand eines bullish Candlestick;
- Farbe des unten-Bars, Schatten und Rand eines bullish Candlestick;
- Farbe der Chartslinien und Doji Candlesticks;
- Farbe der bullishen Candlestick Körper;
- Farbe der bearishen Candlestick Körper;
- Farbe der Bid-Preis Linie;
- Farbe der Ask-Preis Linie;
- Farbe der Linie der letzten Transaktion (Last);
- Farbe der Stop-Order Ebenen (Stop Loss und Take Profit).

Außerdem kann es mehrere grafische Objekte und Indikatoren in einem Chart werden. Sie können ein Chart mit allen notwendigen Indikatoren einmal abstimmen und speichern es als Schablone. Eine solche Schablone kann auf jeden Chart aufgebracht werden.

Die Funktion ChartApplyTemplate() ist für die Verwendung einer zuvor gespeicherten Schablone bestimmt, und es kann in jeder MQL5 Programm verwendet werden. Der Pfad zu der Datei, in der die Schablone liegt, wird als zweiter Parameter zu ChartApplyTemplate() übergeben. Die Schablone-Datei wird nach folgenden Regeln gesucht:

- wenn der Backslash "\" Separator (wie geschrieben "\\") wird am Anfang des Pfads platziert, wird die Schablone relativ zum Pfad _Terminaldaten_Verzeichnis\MQL5 gesucht,
- wenn es kein Backslash gibt, wird die Schablone relativ zur ausführbaren EX5-Datei, in denen ChartApplyTemplate() aufgerufen wird, gesucht;
- wenn die Schablone nicht in den ersten beiden Varianten gefunden ist, wird die Suche in dem Ordner Terminal_Verzeichnis\Profiles\Templates\ durchgeführt.

Hier Terminal_Verzeichnis ist der Ordner, aus dem MetaTrader 5 Client Terminal läuft, und Terminal_Verzeichnis ist der Ordner, in denen editierbare Dateien gespeichert sind, sein Lage ist auf dem Betriebssystem, den Benutzernamen und Computer-Security-Einstellungen abhängig. Normalerweise werden sie verschiedenen Ordnern, sondern in einigen Fällen können sie zusammenfallen.

Der Speicherort der Ordner Terminaldaten_Verzeichnis und Terminal_Verzeichnis kann mit der Funktion TerminalInfoString() erfährt werden.

```
//--- Verzeichnis, aus dem das Terminal gestartet war
   string terminal_path=TerminalInfoString(TERMINAL_PATH);
   Print("Terminal_Verzeichnis:",terminal_path);
//--- Terminaldaten-Verzeichnis, in dem der MQL5-Ordner mit EAs und Indikatoren liegt
   string terminal_data_path=TerminalInfoString(TERMINAL_DATA_PATH);
   Print("Terminaldaten_Verzeichnis:",terminal_data_path);
```

Beispiele:

```
//--- Suche nach einer Schablone in Terminaldaten_Verzeichnis\MQL5\
ChartApplyTemplate(0,"\\first_template.tpl"))
//--- Suche nach einer Schablone in EX5-Datei_Verzeichnis\, dann in Terminaldaten_Ver
ChartApplyTemplate(0,"second_template.tpl"))
//--- Suche in EX5-Datei_Verzeichnis\My_templates\, dann in Terminaldaten_Verzeichnis
ChartApplyTemplate(0,"My_templates\\third_template.tpl"))
```

Schablonen sind nicht Ressourcen, können sie nicht in eine ausführbare EX5-Datei einbezogen werden.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ein Beispiel für die Anwendung der Vorlage, die im Verzeichnis \MQL5\Files gespe
   if(FileIsExist("my_template.tpl"))
     {
      Print("Die Vorlage my_template.tpl wurde im Verzeichnis \Files' gefunden");
      //--- die gefundene Vorlage anwenden
      if(ChartApplyTemplate(0,"\\Files\\my_template.tpl"))
        {
         Print("Die Vorlage 'my_template.tpl' erfolgreich angewandt");
         //--- den Chart neu zeichnen
         ChartRedraw();
        }
      else
         Print("Die Vorlage 'my_template.tpl' konnte nicht angewandt werden, Fehler ",
     }
   else
     {
      Print("Die Datei 'my_template.tpl' konnte im Ordner nicht gefunden werden "
            +TerminalInfoString(TERMINAL_PATH)+"\\MQL5\\Files");
     }
  }
```

Sehen Sie auch

Ressourcen

© 2000-2025, MetaQuotes Ltd.

ChartSaveTemplate

Speichert die aktuelle Chart-Einstellungen in der Schablone mit dem angegebenen Namen.

```
bool  ChartSaveTemplate(
   long          chart_id,     // Chart ID
   const string  filename      // Dateiname für die Schablone
   );
```

Parameter

chart_id

[in] Die ID des Charts. 0 bedeutet den aktuellen Chart.

filename

[in] Dateiname um die Schablone zu speichern. Die Erweiterung ".tpl" wird an den Dateinamen automatisch hinzugefügt, so müssen Sie nicht es angeben. Die Schablone ist im Ordner **Terminal_Ordner**\Profiles\Templates\ gespeichert und kann auch für manuelle Anwendung im Terminal verwendet werden. Wenn eine Schablone mit diesem Namen bereits existiert, wird ihr Inhalt überschrieben werden.

Rückgabewert

Bei der erfolgreichen Anwendung der Schablone gibt die Funktion true zurück, anderenfalls gibt sie false zurück. Um die Information über Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

In einer Schablone können Sie die Einstellungen des Charts mit allen darauf aufgesetzten Indikatoren und Grafiken speichern, damit sie auf ein anderer Chart zu anwenden.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                      Test_ChartSaveTemplate.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property script_show_inputs
//--- input parameters
input string               symbol="GBPUSD";  // Symbol des neuen Charts
input ENUM_TIMEFRAMES      period=PERIOD_H3; // Timeframe des neuen Charts
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Zuerst Indikatoren auf den Chart aufsetzen
```

© 2000-2025, MetaQuotes Ltd.

```
   int handle;
//--- Den Indikator zu nutzen vorbereiten
   if(!PrepareZigzag(NULL,0,handle)) return; // Es hat nicht geklappt
//--- Den Indikator auf den aktuellen Chart aber einem separaten Fenster aufsetzen
   if(!ChartIndicatorAdd(0,1,handle))
     {
      PrintFormat("Fehler beim Aufsetzung von Indikator mit Handle=%d auf Chart %s/%s.
                  _Symbol,
                  EnumToString(_Period),
                  handle,
                  GetLastError());
      //--- Das Programm vorzeitig beenden
      return;
     }
//--- Den Chart aktualisieren, im den Indikator zu sehen
   ChartRedraw();
//--- Hier finden wir zwei letzten Zick-Zack-Frakturen
   double two_values[];
   datetime two_times[];
   if(!GetLastTwoFractures(two_values,two_times,handle))
     {
      PrintFormat("Fehler beim Such der zwei letzten Zick-Zack-Frakturen im Indikator
      //--- Das Programm vorzeitig beenden
      return;
     }
//--- Jetzt Standardabweichung Kanal aufsetzen
   string channel="StdDeviation Channel";
   if(!ObjectCreate(0,channel,OBJ_STDDEVCHANNEL,0,two_times[1],0))
     {
      PrintFormat("Objekt %s konnte nicht erstellt werden. Fehlercode %d",
                  EnumToString(OBJ_STDDEVCHANNEL),GetLastError());
      return;
     }
   else
     {
      //--- Der Kanal erstellt wurde, definieren wir den zweiten Punkt
      ObjectSetInteger(0,channel,OBJPROP_TIME,1,two_times[0]);
      //--- Tooltip für Kanal definieren
      ObjectSetString(0,channel,OBJPROP_TOOLTIP,"Demo from MQL5 Help");
      //--- Chart aktualisieren
      ChartRedraw();
     }
//--- Dies in einer Schablone speichern
   ChartSaveTemplate(0,"StdDevChannelOnZigzag");
//--- Ein neuer Chart öffnen und die Schablone darauf aufsetzen
   long new_chart=ChartOpen(symbol,period);
   //--- Tooltips für graphische Objekten entscheiden
   ChartSetInteger(new_chart,CHART_SHOW_OBJECT_DESCR,true);
   if(new_chart!=0)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- Die neue Schablone auf den Chart aufsetzen
       ChartApplyTemplate(new_chart,"StdDevChannelOnZigzag");
      }
    Sleep(10000);
   }
//+------------------------------------------------------------------+
//| Erzeugt Handle auf Zick-Zack, sorgt für Bereitschaft seinen Daten|
//+------------------------------------------------------------------+
bool PrepareZigzag(string sym,ENUM_TIMEFRAMES tf,int &h)
   {
    ResetLastError();
//--- Indikator Zigzag muss in Terminal_Angaben_Ordner\MQL5\Examples sein
    h=iCustom(sym,tf,"Examples\\Zigzag");
    if(h==INVALID_HANDLE)
      {
       PrintFormat("%s: Handle von Zigzag konnte nicht erstellt werden. Fehlercode %d",
                   __FUNCTION__,GetLastError());
       return false;
      }
//--- Bei der Erstellung eines Indokatorhandles, benötigt es Zeit um Werte zu berechne
    int k=0; // Die Anzahl der Versuche für Warten der Berechnung des Indikators
//--- Auf die Berechnung in Zyklus warten, eine Pause von 50 Millisekunden machen, wer
    while(BarsCalculated(h)<=0)
      {
       k++;
       //--- Die Anzahl der Versuche anzeigen
       PrintFormat("%s: k=%d",__FUNCTION__,k);
       //--- 50 Millisekunden warten, während der Indikator berechnet wird
       Sleep(50);
       //--- Wenn mehr als 100 Versuche gemacht worden, dann ist etwas falsch
       if(k>100)
         {
          //--- Problem melden
          PrintFormat("Indikator konnte nicht in %d Versuche berechnet werden!");
          //--- Problem melden
          return false;
         }
      }
//--- Alles ist fertig, der Indikator ist erstellt und die Werde sind berechnet
    return true;
 ;}
//+------------------------------------------------------------------+
//| Sucht für die letzten 2 Zick-Zack-Frakturen und setzt in Arrays  |
//+------------------------------------------------------------------+
bool GetLastTwoFractures(double &get_values[],datetime &get_times[],int handle)
   {
    double values[];          // Array für die Werte der Zick-Zack
    datetime times[];         // Array, um Zeit zu bekommen
```

© 2000-2025, MetaQuotes Ltd.

```
   int size=100;            // Größe der Arrays
   ResetLastError();
//--- Letzten 100 Werte kopieren
   int copied=CopyBuffer(handle,0,0,size,values);
//--- Die Anzahl der kopierten Werte überprüfen
   if(copied<100)
     {
      PrintFormat("%s: %d Werte des Indikators mit Handle =%d konnte nicht kopiert wer
                  __FUNCTION__,size,handle,GetLastError());
      return false;
     }
//--- Reihenfolge der Zugriff auf das Array als in einer Zeitreihe definieren
   ArraySetAsSeries(values,true);
//--- Die Nummer der Bars mit Frakturen hier schreiben
   int positions[];
//--- Dimensionen der Arrays definieren
   ArrayResize(get_values,3); ArrayResize(get_times,3); ArrayResize(positions,3);
//--- Zähler
   int i=0,k=0;
//--- Start der Suche nach Frakturen
   while(i<100)
     {
      double v=values[i];
      //--- NULL-Werte interessieren uns hier nicht
      if(v!=0.0)
        {
         //--- Merken wir uns die Nummer des Bars
         positions[k]=i;
         //--- Merken wir uns den Wert des Zick-Zacks auf die Fraktur
         get_values[k]=values[i];
         PrintFormat("%s: Zigzag[%d]=%G",__FUNCTION__,i,values[i]);
         //--- Erhöhung des Zählers
         k++;
         //--- Wenn zwei Frakturen sind gefunden, brechen den Zyklus
         if(k>2) break;
        }
      i++;
     }
//--- Reihenfolge der Zugriff auf das Array als in einer Zeitreihe definieren
   ArraySetAsSeries(times,true);   ArraySetAsSeries(get_times,true);
   if(CopyTime(_Symbol,_Period,0,size,times)<=0)
     {
      PrintFormat("%s: %d Werte aus CopyTime() konnten nicht kopiert werden. Fehlercod
                  __FUNCTION__,size,GetLastError());
      return false;
     }
//--- Finden die Öffnungzeit von Bars, an denen die 2 Frakturen sind
   get_times[0]=times[positions[1]];// Der vorletzte Wert ist als erster Fraktur gesch
   get_times[1]=times[positions[2]];// Dritter von Ende Wert ist als zweiter Fraktur g
```

© 2000-2025, MetaQuotes Ltd.

```
   PrintFormat("%s: Erst=%s,  Zweit=%s",__FUNCTION__,TimeToString(get_times[1]),TimeTo
//--- Fertig
   return true;
  }
```

Sehen Sie auch

ChartApplyTemplate(), Ressourcen

© 2000-2025, MetaQuotes Ltd.

ChartWindowFind

Gibt die Nummer des Subfensters zurück, in dem sich Indikator befindet. Es gibt zwei Varianten der Funktion.

1. Funktion sucht auf dem angegebenen Chart das Subfenster mit dem angegebenen "kurzen Namen" des Indikatoren (kurzer Name wird im oberen linke Teil des Subfensters ausgegeben) und im Erfolgsfall gibt die Nummer des Subfensters zurück.

```
int  ChartWindowFind(
   long     chart_id,                  // Identifikator des Charts
   string   indicator_shortname        // kurzer Name des Indikatoren, s. INDICATOR_SH
```

2. Funktion muss vom Benutzerindikator aufgerufen werden und sie gibt die Nummer des Subfensters, wo Indikator arbeitet, zurück.

```
int  ChartWindowFind();
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

indicator_shortname

[in] Kurzer Name des Anzeigers.

Rückgabewert

Nummer des Subfensters im Erfolgsfall. Null bedeutet Hauptfenster des Charts. Im Fall des Misserfolges gibt -1 zurück.

Hinweis

Wenn die zweite Variante der Funktion (ohne Parameter) aus Script oder Experten aufgerufen wird, gibt sie -1 zurück.

Mann sollte nicht Indikators Kurzname mit dem Dateinamen der beim Erstellen des Indikators mit den Funktionen iCustom() und IndicatorCreate() gezeigt wird verwechseln. Wenn der Kurzname des Indikators nicht explizit festgelegt wird, dann bei der Kompilation wird den Namen der Quelldatei des Indikators angegeben.

Sie sollen richtig einen kurzen Namen des Indikators formen, der in der Eigenschaft INDICATOR_SHORTNAME mit Hilfe der FunktionIndicatorSetString() geschrieben wird. Wir empfehlen, dass der Kurzname die Werte der Eingabeparameter des Indikators enthält, da der Indikator, der aus dem Chart gelöscht wird, in der Funktion ChartIndicatorDelete() beim Kurzname identifiziert wird.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property script_show_inputs
//--- input parameters
input string   shortName="MACD(12,26,9)";
//+------------------------------------------------------------------+
//| Gibt Nummer des Charts mit dem angegebenen Indikator zurück      |
//+------------------------------------------------------------------+
int GetIndicatorSubWindowNumber(long chartID=0,string short_name="")
  {
   int window=-1;
//---
   if((ENUM_PROGRAM_TYPE)MQL5InfoInteger(MQL5_PROGRAM_TYPE)==PROGRAM_INDICATOR)
     {
      //--- Funktion wird vom Indikator aufgerufen, der Name ist nicht erforderlich.
      window=ChartWindowFind();
     }
   else
     {
      //--- Funktion wird vom Experten oder Script aufgerufen
      window=ChartWindowFind(0,short_name);
      if(window==-1) Print(__FUNCTION__+"() : Error = ",GetLastError());
     }
//---
   return(window);
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//---
   int window=GetIndicatorSubWindowNumber(0,shortName);
   if(window!=-1)
      Print("Anzeiger "+shortName+" befindet sich im Fenster #"+(string)window);
   else
      Print("Anzeiger "+shortName+" ist nicht gefunden. window = "+(string)window);
  }
```

Sehen Sie auch

ObjectCreate(), ObjectFind()

© 2000-2025, MetaQuotes Ltd.

ChartTimePriceToXY

Konvertiert die Koordinaten des Charts aus der Darstellung von Zeit/Preis in die X- und Y- Koordinaten.

```
bool  ChartTimePriceToXY(
   long           chart_id,      // ID des Charts
   int            sub_window,    // Nummer des Unterfensters
   datetime       time,          // Zeit auf dem Chart
   double         price,         // Preis auf dem Chart
   int&           x,             // X-Koordinate für die Zeit auf dem Chart
   int&           y              // Y-Koordinate für den Preis auf dem Chart
   );
```

Parameter

chart_id

[in] ID des Charts. 0 bedeutet den aktuellen Chart.

sub_window

[in] Nummer des Unterfensters. 0 bedeutet das Hauptfenster des Charts.

time

[in] Der Wert der Zeit auf dem Chart, für den ein Wert in Pixels auf der X-Achse erhalten wird. Der Ursprung ist in der oberen linken Ecke das Hauptfenster des Charts.

price

[in] Der Wert des Preises auf dem Chart, für den ein Wert in Pixels auf der Y-Achse erhalten wird. Der Ursprung ist in der oberen linken Ecke das Hauptfenster des Charts.

x

[out] Die Variable, in die Konvertierung von Zeit in X-Koordinate erhalten wird.

y

[out] Die Variable, in die Konvertierung von Preis in Y-Koordinate erhalten wird.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Beispiel:

```
#define   BAR_NUMBER    0  // Nummer des Balkens dessen Preis und die Zeit wir erhalte

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Daten eines einzelnen Balkens durch den Index BAR_NUMBER kopieren
   MqlRates rates[]={};
   if(CopyRates(_Symbol, _Period, BAR_NUMBER, 1, rates)!=1)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       PrintFormat("CopyRates() failed for bar %d. Error %d", BAR_NUMBER, GetLastError
       return;
      }

//--- Konvertierung des erhaltenen Preises und der Zeit in Chart-Pixel-Koordinaten
   int x=0, y=0;
   ResetLastError();
   if(!ChartTimePriceToXY(ChartID(), 0, rates[0].time, rates[0].close, x, y))
     {
      Print("ChartTimePriceToXY() failed. Error ", GetLastError());
      return;
     }

//--- das erhaltene Ergebnis in das Journal drucken
   PrintFormat("For bar[%d] with opening time %s and price %.*f, the chart coordinates

   /*
   Ergebnis:
   For bar[0] with opening time 2024.08.09 15:06 and price 1.27378, the chart coordina
   */
  }
```

Siehe auch

ChartXYToTimePrice()

© 2000-2025, MetaQuotes Ltd.

ChartXYToTimePrice

Konvertiert die X- und Y-Koordinate des Charts in Zeit und Preis.

```
bool  ChartXYToTimePrice(
   long            chart_id,      // ID des Charts
   int             x,             // X-Koordinate auf dem Chart
   int             y,             // Y-Koordinate auf dem Chart
   int&            sub_window,    // Nummer des Unterfensters
   datetime&       time,          // Zeit auf dem Chart
   double&         price          // Preis auf dem Chart
   );
```

Parameter

chart_id

[in] ID des Charts. 0 bedeutet den aktuellen Chart.

x

[in] X-Koordinate.

y

[in] Y-Koordinate.

sub_window

[out] Die Variable, in die die Nummer des Chartunterfenster geschrieben wird. 0 bedeutet das Hauptfenster des Charts.

time

[out] Der Wert der Zeit auf dem Chart, für den ein Wert in Pixels auf der X-Achse erhalten wird. Der Ursprung ist in der oberen linken Ecke das Hauptfenster des Charts.

price

[out] Der Wert des Preises auf dem Chart, für den ein Wert in Pixels auf der Y-Achse erhalten wird. Der Ursprung ist in der oberen linken Ecke das Hauptfenster des Charts.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
//--- Zeigen die Parameter des Ereignisses auf dem Chart
   Comment(__FUNCTION__,": id=",id," lparam=",lparam," dparam=",dparam," sparam=",spar
//--- wenn es ein Ereignis von Mausklick auf dem Chart ist
   if(id==CHARTEVENT_CLICK)
     {
      //--- Vorbereitung der Variablen
      int      x     =(int)lparam;
      int      y     =(int)dparam;
      datetime dt    =0;
      double   price =0;
      int      window=0;
      //--- Konvertieren wir X und Y in Zeit/Preis
      if(ChartXYToTimePrice(0,x,y,window,dt,price))
        {
         PrintFormat("Window=%d X=%d  Y=%d  =>  Time=%s  Price=%G",window,x,y,TimeToSt
         //--- inverse Umwandlung: (X,Y) => (Time,Price)
         if(ChartTimePriceToXY(0,window,dt,price,x,y))
            PrintFormat("Time=%s  Price=%G  =>  X=%d  Y=%d",TimeToString(dt),price,x,y
         else
            Print("ChartTimePriceToXY return error code: ",GetLastError());
         //--- delete lines
         ObjectDelete(0,"V Line");
         ObjectDelete(0,"H Line");
         //--- create horizontal and vertical lines of the crosshair
         ObjectCreate(0,"H Line",OBJ_HLINE,window,dt,price);
         ObjectCreate(0,"V Line",OBJ_VLINE,window,dt,price);
         ChartRedraw(0);
        }
      else
         Print("ChartXYToTimePrice return error code: ",GetLastError());
      Print("+------------------------------------------------------------------+");
     }
  }
```

Sehen Sie auch

ChartTimePriceToXY()

© 2000-2025, MetaQuotes Ltd.

ChartOpen

Eröffnet einen neuen Chart mit dem angegebenen Symbol und Zeitrahmen.

```
long  ChartOpen(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Zeitrahmen
   );
```

Parameter

symbol

[in] Symbol des Charts. NULL bedeutet das Symbol des aktuellen Charts (auf dem der Experte läuft).

period

[in] Zeitrahmen des Charts (Timeframe). Kann einen der Werte der Enumeration ENUM_TIMEFRAMES annehmen. 0 bedeutet der Zeitrahmen des aktuellen Charts.

Rückgabewert

Bei der erfolgreichen Eröffnung des Charts gibt die Funktion Identifikator des Charts zurück. Anderenfalls gibt sie 0 zurück.

Hinweis

Die maximal mögliche Zahl der gleichzeitig geöffneten Charts im Terminal kann nicht mehr als der Wert von CHARTS_MAX sein.

Beispiel:

```
#define CHART_SYMBOL  NULL
#define CHART_PERIOD  PERIOD_CURRENT

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ein neues Chartsymbol und einen neuen Zeitrahmen festlegen
   string symbol=CHART_SYMBOL;
   if(symbol==NULL)
      symbol=Symbol();
   ENUM_TIMEFRAMES timeframe = (CHART_PERIOD==PERIOD_CURRENT ? Period() : CHART_PERIOI

//--- ein neues Chart mit dem angegebenen Symbol und Zeitrahmen öffnen
   long chart_id=ChartOpen(symbol, timeframe);
   if(chart_id==0)
     {
      Print("ChartOpen() failed. Error ", GetLastError());
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Parameter des geöffneten Charts im Journal ausdrucken
   PrintFormat("A new chart of the %s symbol has been opened with a period of %s and C
               symbol, StringSubstr(EnumToString(timeframe), 7), chart_id);
   /*
   Ergebnis:
   A new chart of the GBPUSD symbol has been opened with a period of M1 and ChartID 13
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartFirst

Gibt den Identifikator des ersten Charts des Client-Terminals zurück.

```
long  ChartFirst();
```

Rückgabewert

Identifikator des Charts.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln der ID des allerersten geöffneten Charts im Client-Terminal
   long            chart_id= ChartFirst();

//--- zusätzlich das Chart-Symbol und den Zeitrahmen unter Verwendung der erhaltenen I
   string          symbol  = ChartSymbol(chart_id);
   ENUM_TIMEFRAMES period  = ChartPeriod(chart_id);

//--- Anzeige der Beschreibung des ersten Charts des Client-Terminals im Journal
   PrintFormat("ID of the first chart of the client terminal: %I64d, chart symbol: %s,
   /*
   Ergebnis:
   ID of the first chart of the client terminal: 133246248352168440, chart symbol: EUI
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartNext

Funktion gibt Identifikator des Charts zurück, der nach dem angegebenen folgt.

```
long  ChartNext(
   long  chart_id      // Identifikator des Charts
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet nicht den laufenden Chart. 0 bedeutet "identifikator des ersten Charts zurückgeben".

Rückgabewert

Identifikator des Charts. Wenn die Liste der Charts zu Ende ist, gibt die Funktion -1 zurück.

Beispiel:

```
//--- Variablen für Identifikatoren der Charts
   long currChart,prevChart=ChartFirst();
   int i=0,limit=100;
   Print("ChartFirst = ",ChartSymbol(prevChart)," ID = ",prevChart);
   while(i<limit)// sie haben bestimmt mehr als 100 geoeffnete Dateien
     {
      currChart=ChartNext(prevChart); // bekommen einen neuen Chart mittels des vorhe
      if(currChart<0) break;          // haben das Ende der Chartliste erreicht
      Print(i,ChartSymbol(currChart)," ID =",currChart);
      prevChart=currChart;//speichern wir Identifikator des laufenden Charts für Chart
      i++;// nicht vergessen, den Counter zu steigern
     }
```

© 2000-2025, MetaQuotes Ltd.

ChartClose

Schliesst den angegebenen Chart.

```
bool  ChartClose(
   long  chart_id=0      // Identifikator des Charts
   );
```

Parameter

chart_id=0

[in] Identifikator des Charts. 0 bedeutet denen Chart.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ein neues Chart mit demselben Symbol und derselben Zeitrahmen wie das aktuelle I
   long chart_id=ChartOpen(_Symbol, _Period);
   if(chart_id==0)
     {
      Print("ChartOpen() failed. Error ", GetLastError());
      return;
     }

//--- Parameter des geöffneten Charts im Journal ausdrucken
   PrintFormat("A new chart of the %s symbol has been opened with a period of %s and C
               _Symbol, StringSubstr(EnumToString(_Period), 7), chart_id);

//--- zwei Sekunden warten und das neu geöffnete Chart schließen
   PrintFormat("Waiting 3 seconds before closing a newly opened chart with ID %I64d ..
   Sleep(3000);
   ResetLastError();
   if(!ChartClose(chart_id))
     {
      Print("ChartClose() failed. Error ", GetLastError());
      return;
     }

//--- Meldung über das erfolgreiche Schließen des Charts im Journal ausdrucken
   PrintFormat("The chart with ID %I64d was successfully closed", chart_id);
   /*
   Ergebnis:
   A new chart of the GBPUSD symbol has been opened with a period of M1 and ChartID 13
```

© 2000-2025, MetaQuotes Ltd.

```
   Waiting 3 seconds before closing a newly opened chart with ID 133346697706632016 ..
   The chart with ID 133346697706632016 was successfully closed
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartSymbol

Gibt den Symbolnamen des angegebenen Charts zurück.

```
string  ChartSymbol(
   long  chart_id=0      // Identifikator des Charts
   );
```

Parameter

chart_id=0

[in] Identifikator des Charts. 0 bedeutet denen Chart.

Rückgabewert

Wenn der Chart nicht existiert, wird eine leere Zeichenkette zurückgegeben.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- das aktuelle Chartsymbol ermitteln und den erhaltenen Wert im Journal ausdrucken
   string chart_symbol = ChartSymbol();
   Print("Current chart symbol: ", chart_symbol);

//--- übernimmt die vorhandene (in diesem Fall aktuelle) Chart-ID
   long chart_id=ChartID();
   chart_symbol=ChartSymbol(chart_id);
   PrintFormat("Chart symbol with ID %I64d: %s", chart_id, chart_symbol);

//--- eine zufällige Chart-ID setzen, wenn ein Symbol empfangen wird
   chart_symbol = ChartSymbol(1234567890);
   if(chart_symbol=="")
      Print("The chart with ID 1234567890 does not exist");
   else
      Print("Chart symbol with ID 1234567890: ", chart_symbol);
   /*
   Ergebnis:
   Current chart symbol: GBPUSD
   Chart symbol with ID 132966427583395104: GBPUSD
   The chart with ID 1234567890 does not exist
   */
  }
```

Siehe auch

ChartSetSymbolPeriod

© 2000-2025, MetaQuotes Ltd.

ChartPeriod

Gibt den Zeitrahmen des angegebenen Charts zurück.

```
ENUM_TIMEFRAMES  ChartPeriod(
   long  chart_id=0      // Identifikator des Charts
   );
```

Parameter

chart_id=0

[in] Identifikator des Charts. 0 bedeutet denen Chart.

Rückgabewert

Wert des Typs ENUM_TIMEFRAMES. Wenn der Chart nicht existiert, wird 0 zurückgegeben.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Ermitteln des aktuellen Zeitrahmen des Charts und Ausdruck des erhaltenen Wertes
   ENUM_TIMEFRAMES period=ChartPeriod();
   Print("Current chart period: ", EnumToString(period));

//--- übernimmt die vorhandene (in diesem Fall aktuelle) Chart-ID
   long chart_id=ChartID();
   period=ChartPeriod(chart_id);
   PrintFormat("Chart period with ID %I64d: %s", chart_id, EnumToString(period));

//--- eine zufällige Chart-ID festlegen
   period=ChartPeriod(1234567890);
   if(period==0)
      Print("The chart with ID 1234567890 does not exist");
   else
      Print("Chart period with ID 1234567890: ", EnumToString(period));
   /*
   Ergebnis:
   Current chart period: PERIOD_M15
   Chart period with ID 133510090240498505: PERIOD_M15
   The chart with ID 1234567890 does not exist
   */
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartRedraw

Ruft das zwangsweise Neuzeichnen des angegebenen Chart auf.

```
void  ChartRedraw(
   long  chart_id=0      // Identifikator des Charts
```

Parameter

chart_id=0

[in] Identifikator des Charts. 0 bedeutet denen Chart.

Hinweis

Gewöhnlich wird nach der Veränderung der Objekteigenschaften verwendet.

Beispiel:

```
#define   WIDTH      50    // Breite des Rechteckes in Balken

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- aktuelle Chart-ID abrufen und grafischen Objektnamen festlegen
   long   chart_id = ChartID();
   string obj_name = MQLInfoString(MQL_PROGRAM_NAME)+"_RectLabel";

//--- zwei rechteckige Zeitkoordinaten abrufen
   datetime time1  = iTime(_Symbol, _Period, WIDTH);
   datetime time2  = iTime(_Symbol, _Period, 0);
   if(time1==0 || time2==0)
     {
      Print("Error getting time ", GetLastError());
      return;
     }

//--- Holen der Höchst- und Tiefstpreise im Bereich der Rechteckbreite
   double price1 = HighestHigh(_Symbol, _Period, 0, WIDTH);
   double price2 = LowestLow(_Symbol, _Period, 0, WIDTH);
   if(price1==EMPTY_VALUE || price2==EMPTY_VALUE)
      return;

//--- Erstellen eines rechteckigen Objekts
   Print("Create a wheat-colored rectangle");
   if(!ObjectCreate(chart_id, obj_name, OBJ_RECTANGLE, 0, time1, price1, time2, price2
     {
      Print("ObjectCreate() failed. Error ", GetLastError());
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- füllen des Rechtecks mit der Originalfarbe
   ObjectSetInteger(chart_id, obj_name, OBJPROP_FILL, true);
   ObjectSetInteger(chart_id, obj_name, OBJPROP_BACK, true);
   ObjectSetInteger(chart_id, obj_name, OBJPROP_COLOR, clrWheat);
   ChartRedraw();

//--- Warten von einer Sekunde für das Füllen des Rechtecks mit der Farbe DodgerBlue u
   Sleep(1000);
   Print("Change color to DodgerBlue");
   ObjectSetInteger(chart_id, obj_name, OBJPROP_COLOR, clrDodgerBlue);
   ChartRedraw();

//--- Warten von einer Sekunde für das Füllen des Rechtecks mit der Farbe „LimeGreen"
   Sleep(1000);
   Print("Change color to LimeGreen");
   ObjectSetInteger(chart_id, obj_name, OBJPROP_COLOR, clrLimeGreen);
   ChartRedraw();

//--- Warten von einer Sekunde für das Füllen des Rechtecks mit der Farbe „OrangeRed"
   Sleep(1000);
   Print("Change color to OrangeRed");
   ObjectSetInteger(chart_id, obj_name, OBJPROP_COLOR, clrOrangeRed);
   ChartRedraw();

//--- Warten von einer Sekunde für das Füllen des Rechtecks mit der Farbe „Wheat" und
   Sleep(1000);
   Print("Reset color to original");
   ObjectSetInteger(chart_id, obj_name, OBJPROP_COLOR, clrWheat);
   ChartRedraw();

//--- Entfernen des Objekts aus dem Chart nach drei Sekunden
   Sleep(3000);
   Print("Delete the rectangle");
   ObjectDelete(chart_id, obj_name);
  }
//+------------------------------------------------------------------+
//| Liefert das höchste Hoch im angegebenen Bereich der Balken       |
//+------------------------------------------------------------------+
double HighestHigh(const string symbol, const ENUM_TIMEFRAMES timeframe, const uint st
  {
   ResetLastError();
   int index=iHighest(symbol, timeframe, MODE_HIGH, count, start);
   if(index==-1)
     {
      PrintFormat("%s: iHighest() failed. Error %d",__FUNCTION__, GetLastError());
      return(EMPTY_VALUE);
     }
   GetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
   double price=iHigh(symbol, timeframe, index);
   if(price==0)
     {
      PrintFormat("%s: iHigh() failed. Error %d",__FUNCTION__, GetLastError());
      return(EMPTY_VALUE);
     }
   return(price);
  }
//+------------------------------------------------------------------+
//| Rückgabe des tiefsten Tiefs im angegebenen Bereich der Balken    |
//+------------------------------------------------------------------+
double LowestLow(const string symbol, const ENUM_TIMEFRAMES timeframe, const uint sta
  {
   ResetLastError();
   int index=iLowest(symbol, timeframe, MODE_LOW, count, start);
   if(index==-1)
     {
      PrintFormat("%s: iLowest() failed. Error %d",__FUNCTION__, GetLastError());
      return(EMPTY_VALUE);
     }
   GetLastError();
   double price=iLow(symbol, timeframe, index);
   if(price==0)
     {
      PrintFormat("%s: iLow() failed. Error %d",__FUNCTION__, GetLastError());
      return(EMPTY_VALUE);
     }
   return(price);
  }
```

Siehe auch

Graphische Objekte

© 2000-2025, MetaQuotes Ltd.

ChartSetDouble

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts vor. Eigenschaft des Charts muss des Typs double sein. Der Befehl wird in die Warteschlange der Chartnachrichten gestellt und wird nach der Durchführung aller vorherigen Befehle abgearbeitet werden.

```
bool  ChartSetDouble(
   long                          chart_id,      // Identifikator des Charts
   ENUM_CHART_PROPERTY_DOUBLE    prop_id,       // Identifikator der Eigenschaft
   double                        value          // Wert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet denen Chart.

prop_id

[in] Identifikator der Eigenschaft des Charts. Wert kann einer der Enumerationswerte ENUM_CHART_PROPERTY_DOUBLE sein (ausser read-only Eigenschaften).

value

[in] Wert der Eigenschaft.

Rückgabewert

Gibt true zurück, wenn der Befehl in die Warteschlange des Charts gestellt wurde, andernfalls false. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Diese Funktion ist asynchron, das bedeutet, dass die Funktion nicht auf die Ausführung eines Befehls wartet, der zur Warteschlange des angegebenen Charts erfolgreich hinzugefügt wurde, sondern direkt die Kontrolle zurückgibt. Die Änderung der Eigenschaft wird erst nach der Verarbeitung des Befehls in der Warteschlange des Charts implementiert. Für eine sofortige Ausführung von Befehlen in der Warteschlange ist die Funktion ChartRedraw aufzurufen.

Wenn man gleich mehrere Eigenschaften des Charts ändern muss, muss man die entsprechenden Funktionen (ChartSetString, ChartSetDouble, ChartSetString) in einem Code-Block ausführen und dann ChartRedraw einmal aufrufen.

Für die Überprüfung des Ergebnisses der Ausführung kann man eine Funktion verwenden, die die angegebene Eigenschaft des Charts abruft ChartGetInteger, ChartGetDouble, ChartSetString). Dabei muss man beachten, dass diese Funktionen synchron sind und auf das Ergebnis der Ausführung warten.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- aktuelle Chart-ID abrufen
```

© 2000-2025, MetaQuotes Ltd.

```
   long chart_id = ChartID();

//--- Speichern des anfänglichen Abstands des Balkens Null vom rechten Rand in Prozent
   double shift = ChartGetDouble(chart_id, CHART_SHIFT_SIZE);
   bool   scroll= ChartGetInteger(chart_id,CHART_AUTOSCROLL);

//--- Rücksetzen des Auto-Scroll-Chart-Flags
   ChartSetInteger(chart_id, CHART_AUTOSCROLL, true);
   PrintFormat("Initial chart shift size: %.2f", shift);

//--- in der Schleife von 2,0 % bis 50,0 % mit einer Schrittweite von 0,5 %
   for(int i=20; i<=500; i+=5)
     {
      //--- Setzen der Verschiebungsgröße auf i / 10 und Ausdruck des angegebenen Wert
      ChartSetDouble(chart_id, CHART_SHIFT_SIZE, i/10.0);
      PrintFormat("Set chart shift size to %.1f%%", i/10.0);

      //--- den Balken Null des Charts auf der angegebenen Verschiebungsdistanz halter
      ChartNavigate(chart_id, CHART_END, 0);

      //--- wart ein Weilchen
      Sleep(16);
     }

//--- Einstellen der anfängliche Chartverschiebung und den automatischen Bildlauf
   Print("Set the chart shift size to the initial value of ", shift);
   ChartSetDouble(ChartID(), CHART_SHIFT_SIZE, shift);
   ChartSetInteger(chart_id, CHART_AUTOSCROLL, scroll);
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartSetInteger

Legt den Wert der entsprechenden Eigenschaft des angegebenen Charts fest. Die Eigenschaft des Charts muss den Typ datetime, int, color, bool oder char haben. Der Befehl wird zur Warteschlange der Nachrichten des Charts hinzugefügt und erst nach der Verarbeitung aller vorherigen Befehle verarbeitet.

```
bool  ChartSetInteger(
   long                          chart_id,      // Identifikator des Charts
   ENUM_CHART_PROPERTY_INTEGER   prop_id,       // Identifikator der Eigenschaft
   long                          value          // Wert
   );
```

Legt den Wert der entsprechenden Eigenschaft des angegebenen Unterfensters fest

```
bool  ChartSetInteger(
   long                          chart_id,      // Identifikator des Charts
   ENUM_CHART_PROPERTY_INTEGER   prop_id,       // Identifikator der Eigenschaft
   int                           sub_window,    // Nummer des Unterfensters
   long                          value          // Wert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den aktuellen Chart.

prop_id

[in] Identifikator der Eigenschaften des Charts. Der Wert kann einer der Werte der Aufzählung ENUM_CHART_PROPERTY_INTEGER (außer read-only Eigenschaften) sein.

sub_window

[in] Nummer des Unterfensters des Charts. Für die erste Variante ist der standardmäßige Wert gleich 0 (Hauptfenster des Charts). Die meisten Eigenschaften erfordern die Nummer des Unterfensters nicht.

value

[in] Wert der Eigenschaft.

Rückgabewert

Gibt true zurück, wenn der Befehl zur Warteschlange des Charts hinzugefügt wurde, andernfalls false. Um zusätzliche Information über den Fehler zu bekommen, rufen Sie die Funktion GetLastError() auf.

Hinweis

Diese Funktion ist asynchron, das bedeutet, dass die Funktion nicht auf die Ausführung eines Befehls wartet, der zur Warteschlange des angegebenen Charts erfolgreich hinzugefügt wurde, sondern direkt die Kontrolle zurückgibt. Die Änderung der Eigenschaft wird erst nach der Verarbeitung des Befehls in der Warteschlange des Charts implementiert. Für eine sofortige Ausführung von Befehlen in der Warteschlange ist die Funktion ChartRedraw aufzurufen.

© 2000-2025, MetaQuotes Ltd.

Wenn man gleich mehrere Eigenschaften des Charts ändern muss, muss man die entsprechenden Funktionen (ChartSetString, ChartSetDouble, ChartSetString) in einem Code-Block ausführen und dann ChartRedraw einmal aufrufen.

Für die Überprüfung des Ergebnisses der Ausführung kann man eine Funktion verwenden, die die angegebene Eigenschaft des Charts abruft ChartGetInteger, ChartGetDouble, ChartSetString). Dabei muss man beachten, dass diese Funktionen synchron sind und auf das Ergebnis der Ausführung warten.

Beispiel:

```
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
void OnInit()
 {
//--- Aktivierung der Nachrichten über Bewegungen der Maus im Fensterchart
   ChartSetInteger(0,CHART_EVENT_MOUSE_MOVE,1);
//--- erzwungene Aktualisierung der Charteigenschaften garantiert die Bereitschaft zu
   ChartRedraw();
  }
//+------------------------------------------------------------------+
//| MouseState                                                       |
//+------------------------------------------------------------------+
string MouseState(uint state)
 {
   string res;
   res+="\nML: "   +(((state& 1)== 1)?"DN":"UP");   // mouse left
   res+="\nMR: "   +(((state& 2)== 2)?"DN":"UP");   // mouse right
   res+="\nMM: "   +(((state&16)==16)?"DN":"UP");   // mouse middle
   res+="\nMX: "   +(((state&32)==32)?"DN":"UP");   // mouse first X key
   res+="\nMY: "   +(((state&64)==64)?"DN":"UP");   // mouse second X key
   res+="\nSHIFT: "+(((state& 4)== 4)?"DN":"UP");   // shift key
   res+="\nStrg: " +(((state& 8)== 8)?"DN":"UP");   // control key
   return(res);
  }
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,const long &lparam,const double &dparam,const string &s
 {
   if(id==CHARTEVENT_MOUSE_MOVE)
      Comment("POINT: ",(int)lparam,",",(int)dparam,"\n",MouseState((uint)sparam));
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartSetString

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts vor. Eigenschaft des Charts muss des Typs string sein. Der Befehl wird in die Warteschlange der Chartnachrichten gestellt und wird nach der Prozessirrung aller vorherigen Befehle abgearbeitet werden.

```
bool  ChartSetString(
   long                          chart_id,     // Identifikator des Charts
   ENUM_CHART_PROPERTY_STRING    prop_id,      // Identifikator der Eigenschaft
   string                        str_value     // Wert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

prop_id

[in] Identifikator der Eigenschaften des Charts. Wert kann einer der Enumerationswerte ENUM_CHART_PROPERTY_STRING (ausser read-only Eigenschaften) sein.

str_value

[in] Zeile für Einstellung der Eigenschaft. Laenge der Zeile kann nicht mehr als 2045 Symbole sein (überschüssige Symbole werden entfernt werden).

Rückgabewert

Gibt true zurück, wenn der Befehl in die Warteschlange des Charts gestellt wurde, andernfalls gibt false zurück. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Funktion ChartSetString kann für Ausgabe der Kommentare auf den Chart statt der Funktion Comment verwendet werden.

Diese Funktion ist asynchron, das bedeutet, dass die Funktion nicht auf die Ausführung eines Befehls wartet, der zur Warteschlange des angegebenen Charts erfolgreich hinzugefügt wurde, sondern direkt die Kontrolle zurückgibt. Die Änderung der Eigenschaft wird erst nach der Verarbeitung des Befehls in der Warteschlange des Charts implementiert. Für eine sofortige Ausführung von Befehlen in der Warteschlange ist die Funktion ChartRedraw aufzurufen.

Wenn man gleich mehrere Eigenschaften des Charts ändern muss, muss man die entsprechenden Funktionen (ChartSetString, ChartSetDouble, ChartSetString) in einem Code-Block ausführen und dann ChartRedraw einmal aufrufen.

Für die Überprüfung des Ergebnisses der Ausführung kann man eine Funktion verwenden, die die angegebene Eigenschaft des Charts abruft ChartGetInteger, ChartGetDouble, ChartSetString). Dabei muss man beachten, dass diese Funktionen synchron sind und auf das Ergebnis der Ausführung warten.

Beispiel:

```
void OnTick()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//---
   double Ask,Bid;
   int Spread;
   Ask=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
   Bid=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   Spread=SymbolInfoInteger(Symbol(),SYMBOL_SPREAD);
   string comment=StringFormat("Выводим цены:\nAsk = %G\nBid = %G\nSpread = %d",
                               Ask,Bid,Spread);
   ChartSetString(0,CHART_COMMENT,comment);
  }
```

Sehen Sie auch

Comment, ChartGetString

© 2000-2025, MetaQuotes Ltd.

ChartGetDouble

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück. Die Eigenschaft des Charts muss vom Typ double sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft direkt zurück.

```
double  ChartGetDouble(
   long                          chart_id,         // Identifikator des Charts
   ENUM_CHART_PROPERTY_DOUBLE    prop_id,          // Identifikator der Eigenschaft
   int                           sub_window=0      // Nummer des Unterfensters wenn erfo
   );
```

2. Gibt true oder false zurück je nach dem, ob eine Funktion erfolgreich ausgeführt wurde. Wenn erfolgreich wird der Wert der Eigenschaft in eine Zielvariable platziert, die als Referenz durch den letzten Parameter übergeben wird.

```
bool  ChartGetDouble(
   long                          chart_id,         // Identifikator des Charts
   ENUM_CHART_PROPERTY_DOUBLE    prop_id,          // Identifikator der Eigenschaft
   int                           sub_window,       // Nummer des Unterfensters
   double&                       double_var        // Zielvariable für den Wert der Eige
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den aktuellen Chart.

prop_id

[in] Identifikator der Eigenschaft des Charts. Der Wert kann einer der Werte der Aufzählung ENUM_CHART_PROPERTY_DOUBLE sein.

sub_window

[in] Nummer des Unterfensters des Charts. Für die erste Variante ist der standardmäßige Wert gleich 0 (Hauptfenster des Charts). Die meisten Eigenschaften erfordern die Nummer des Unterfensters nicht.

double_var

[out] Die Variable vom Typs double für die gewünschte Eigenschaft.

Rückgabewert

Der Wert vom Typ double.

Für die zweite Variante des Aufrufs wird true zurückgegeben, wenn diese Eigenschaft unterstützt wird und wenn der Wert in die Variable double_var platziert wurde, anderenfalls false. Um zusätzliche Information über den Fehler zu bekommen, rufen Sie die Funktion GetLastError() auf.

Hinweis

Diese Funktion ist synchron, das bedeutet, dass sie erst nach der Ausführung aller Befehle ausgeführt wird, die vor dem Aufruf der Funktion zur Warteschlange des Charts hinzugefügt wurden.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
void OnStart()
 {
   double priceMin=ChartGetDouble(0,CHART_PRICE_MIN,0);
   double priceMax=ChartGetDouble(0,CHART_PRICE_MAX,0);
   Print("CHART_PRICE_MIN = ",priceMin);
   Print("CHART_PRICE_MAX = ",priceMax);
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartGetInteger

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück. Die Eigenschaft des Charts muss der Typen datetime, int oder bool sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft direkt zurück.

```
long  ChartGetInteger(
   long                          chart_id,         // Identifikator des Charts
   ENUM_CHART_PROPERTY_INTEGER   prop_id,          // Identifikator der Eigenschaft
   int                           sub_window=0      // Nummer des Subfensters wenn erfo
   );
```

2. Gibt true oder false zurück, abhängig von der Erfolgreichkeit der Funktion. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gestellt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  ChartGetInteger(
   long                          chart_id,         // Identifikator des Charts
   ENUM_CHART_PROPERTY_INTEGER   prop_id,          // Identifikator der Eigenschaft
   int                           sub_window,       // Nummer des Subfensters
   long&                         long_var          // hier erhalten wir den Wert der E
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

prop_id

[in] Identifikator der Eigenschaft des Charts. Wert kann einer der Enumerationswerte ENUM_CHART_PROPERTY_INTEGER sein.

sub_window

[in] Nummer des Subfenster des Charts. Für den ersten Fall der Default-Wert ist 0 (Hauptfenster des Charts). Die meisten Eigenschaften ist die Nummer des Subfensters nicht erforderlich.

long_var

[out] Variable des Typs long, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs long.

für die zweite Variante des Aufrufes gibt true zurück, wenn die Eigenschaft zugaenglich ist und der Wert in der Variable long_var aufbewahren wird, anderenfalls gibt false zurück. Für die zusaetzliche Information über das Fehler, rufen Sie die Funktion GetLastError() auf.

Hinweis

Diese Funktion ist synchron, das bedeutet, dass sie erst nach der Ausführung aller Befehle ausgeführt wird, die vor dem Aufruf der Funktion zur Warteschlange des Charts hinzugefügt wurden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   int height=ChartGetInteger(0,CHART_HEIGHT_IN_PIXELS,0);
   int width=ChartGetInteger(0,CHART_WIDTH_IN_PIXELS,0);
   Print("CHART_HEIGHT_IN_PIXELS = ",height," pixels");
   Print("CHART_WIDTH_IN_PIXELS = ",width," pixels");
  }
```

© 2000-2025, MetaQuotes Ltd.

ChartGetString

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück. Die Eigenschaft des Charts muss des Typs string sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft direkt zurück.

```
string  ChartGetString(
   long                        chart_id,        // Identifikator des Charts
   ENUM_CHART_PROPERTY_STRING  prop_id          // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig von der erfolgreichen Durchführung der Funktion. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gestellt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  ChartGetString(
   long                        chart_id,        // Identifikator des Charts
   ENUM_CHART_PROPERTY_STRING  prop_id,         // Identifikator der Eigenschaft
   string&                     string_var       // hier erhalten wir den Wert der Eig
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

prop_id

[in] Identifikator der Eigenschaft des Charts. Wert kann einer der Enumerationswerte ENUM_CHART_PROPERTY_STRING sein.

string_var

[out] Variable des Typs string, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs string.

für die zweite Variante des Ausrufes gibt true zurück, wenn diese Eigenschaft zugaenglich wird und der Wert in der Variable string_var aufbewahren wird, anderenfalls gibt false zurück. Für die Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Funktion ChartGetString kann für Einlesen der Kommentare verwendet werden, die durch die FunktionenComment oder ChartSetString ausgegeben werden.

Diese Funktion ist synchron, das bedeutet, dass sie erst nach der Ausführung aller Befehle ausgeführt wird, die vor dem Aufruf der Funktion zur Warteschlange des Charts hinzugefügt wurden.

Beispiel:

```
void OnStart()
```

© 2000-2025, MetaQuotes Ltd.

```
{
 ChartSetString(0,CHART_COMMENT,"Test comment.\nSecond line.\nThird!");
 ChartRedraw();
 Sleep(1000);
 string comm=ChartGetString(0,CHART_COMMENT);
 Print(comm);
}
```

Sehen Sie auch

Comment, ChartSetString

© 2000-2025, MetaQuotes Ltd.

ChartNavigate

Verschiebt den Chart um die angegebene Anzahl der Balken in Bezug auf die angegebene Chartposition.

```
bool  ChartNavigate(
   long                  chart_id,      // Identifikator des Charts
   ENUM_CHART_POSITION   position,      // Position
   int                   shift=0        // Verschiebungswert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

position

[in] Chartposition, in Bezug auf die die Verschiebung durchgeführt wird. Wert kann einer der Enumerationswerte ENUM_CHART_POSITION sein.

shift=0

[in] Anzahl der Bars, um die der Chart verschoben werden muss. Positive Verschiebung ist Verschiebung nach rechts (zum Chartende), negative Verschiebung ist Verschiebung nach links (zum Chartanfang). Null-Verschiebung ist dann sinnvoll, wenn Navigierung zum Anfang oder zum Ende des Charts durchgeführt wird.

Rückgabewert

Gibt true im Efolgsfall zurück, anderenfalls false.

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- handle des aktuellen Charts erhalten
   long handle=ChartID();
   string comm="";
   if(handle>0) // wenn dies gelingt, zusätzlich einrichten
     {
      //--- Autoscroll deaktivieren
      ChartSetInteger(handle,CHART_AUTOSCROLL,false);
      //--- eine Verschiebung von der rechten Chart-Grenze setzen
      ChartSetInteger(handle,CHART_SHIFT,true);
      //--- Candlesticks ziehen
      ChartSetInteger(handle,CHART_MODE,CHART_CANDLES);
      //--- Anzeige-Modus für Tick-Volumen einstellen
      ChartSetInteger(handle,CHART_SHOW_VOLUMES,CHART_VOLUME_TICK);

      //--- einen Text vorzubereiten, um in Comment() anzuzeigen
```

© 2000-2025, MetaQuotes Ltd.

```
   comm="10 Balken nach rechts von Anfang der Geschichte scrollen";
   //--- Kommentar anzeigen
   Comment(comm);
   //--- 10 Balken nach rechts von Anfang der Geschichte scrollen
   ChartNavigate(handle,CHART_BEGIN,10);
   //--- erhalten wir die Nummer des ersten Balken, der auf dem Chart sichtbar ist
   long first_bar=ChartGetInteger(0,CHART_FIRST_VISIBLE_BAR,0);
   //--- Zeilenvorschubzeichen hinzufügen
   comm=comm+"\r\n";
   //--- Kommentar hinzufügen
   comm=comm+"Der erste Balken auf dem Chart hat Nummer "+IntegerToString(first_bar
   //--- Kommentar anzeigen
   Comment(comm);
   //--- 5 Sekunden warten, um zu sehen, wie das Chart bewegt
   Sleep(5000);

   //--- Kommentar hinzufügen
   comm=comm+"\r\n"+"10 Balken nach links von der rechten Chart-Grenze scrollen";
   Comment(comm);
   //--- 10 Balken nach links von der rechten Chart-Grenze scrollen
   ChartNavigate(handle,CHART_END,-10);
   //--- erhalten wir die Nummer des ersten Balken, der auf dem Chart sichtbar ist
   first_bar=ChartGetInteger(0,CHART_FIRST_VISIBLE_BAR,0);
   comm=comm+"\r\n";
   comm=comm+"Der erste Balken auf dem Chart hat Nummer "+IntegerToString(first_bar
   Comment(comm);
   //--- 5 Sekunden warten, um zu sehen, wie das Chart bewegt
   Sleep(5000);

   //--- neuen Block von Chart-Scrollen
   comm=comm+"\r\n"+"300 Balken nach rechts von Anfang der Geschichte scrollen";
   Comment(comm);
   //--- 300 Balken nach rechts von Anfang der Geschichte scrollen
   ChartNavigate(handle,CHART_BEGIN,300);
   first_bar=ChartGetInteger(0,CHART_FIRST_VISIBLE_BAR,0);
   comm=comm+"\r\n";
   comm=comm+"Der erste Balken auf dem Chart hat Nummer "+IntegerToString(first_bar
   Comment(comm);
   //--- 5 Sekunden warten, um zu sehen, wie das Chart bewegt
   Sleep(5000);

   //--- neuen Block von Chart-Scrollen
   comm=comm+"\r\n"+"300 Balken nach links von der rechten Chart-Grenze scrollen";
   Comment(comm);
   //--- 300 Balken nach links von der rechten Chart-Grenze scrollen
   ChartNavigate(handle,CHART_END,-300);
   first_bar=ChartGetInteger(0,CHART_FIRST_VISIBLE_BAR,0);
   comm=comm+"\r\n";
   comm=comm+"Der erste Balken auf dem Chart hat Nummer "+IntegerToString(first_bar
```

© 2000-2025, MetaQuotes Ltd.

```
      Comment(comm);
     }
;}
```

© 2000-2025, MetaQuotes Ltd.

ChartID

Gibt Identifikator des aktuellen Charts zurück.

```
long  ChartID();
```

Rückgabewert

Wert des Typs long.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Variablen für Chart-Identifikatoren
   long curr_chart=ChartFirst();
   int i=0;
   //--- Ausdruck der Daten des ersten Charts im Journal
   PrintFormat("Chart[%d] ID: %I64d,  symbol: %s", i, curr_chart, ChartSymbol(curr_cha

//--- bis zur Grenze des geöffneten Charts (CHARTS_MAX)
   while(!IsStopped() && i < CHARTS_MAX)
     {
      //--- Erhöhen des Chart-Zählers
      i++;
      //--- die nächste Chart-ID auf der Grundlage der vorherigen ermitteln
      curr_chart=ChartNext(curr_chart);

      //--- Beenden der Schleife, wenn das Ende der Chart-Liste erreicht ist
      if(curr_chart<0)
         break;

      //--- Druck der nächsten Chart-Daten im Journal
      PrintFormat("Chart[%d] ID: %I64d,  symbol: %s", i, curr_chart, ChartSymbol(curr_
     }
   /*
   Ergebnis:
   Chart[0] ID: 133246248352168440,  symbol: EURUSD
   Chart[1] ID: 133346697706632015,  symbol: USDJPY
   Chart[2] ID: 133246248352168439,  symbol: GBPUSD
   Chart[3] ID: 133346697706632009,  symbol: RU000A103661
   Chart[4] ID: 133346697706632010,  symbol: AEM4
   Chart[5] ID: 133346697706632011,  symbol: AA.SPB
   Chart[6] ID: 133346697706632012,  symbol: ALLFUTMIX
   Chart[7] ID: 133346697706632013,  symbol: EURUSD
   Chart[8] ID: 133346697706632014,  symbol: SBER
   */
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

ChartIndicatorAdd

Setzt Indikator mit dem angegebenen handle in das angegebene Fenster des Charts zu. Indikator und Chart sollten auf dem gleichen Symbol und Zeitrahmen gebaut werden.

```
bool  ChartIndicatorAdd(
   long  chart_id,                  // Identifikator des Charts
   int   sub_window                 // Nummer des Unterfensters
   int   indicator_handle           // handle des Indikators
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet das laufende Chart.

sub_window

[in] Nummer des Unterfensters des Charts. 0 bedeutet das Hauptfenster des Charts. Um ein Indikator in ein neues Fenster hinzuzufügen, der Parameter muss um eins größer als der Index des letzten existierten Fensters sein, d.h. dem CHART_WINDOWS_TOTAL gleich sein. Wen der Parameterwert mehr als CHART_WINDOWS_TOTAL ist, das neue Fenster wird nicht erstellt werden und der Indikator wird nicht hinzufügt werden.

indicator_handle

[in] Handle des Indikators.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen. Fehler 4114 bedeutet, dass der Chart und das hinzugefügte Indikator haben verschiedenen Symbol oder Zeitrahmen.

Hinweis

Wenn ein Indikator, der in einem separaten Fenster gezeichnet werden soll (z. B. integrierte IMACD oder benutzerdefinierter Indikator mit dem angegebenen #property indicator_separate_window) in den Hauptfenster des Charts hinzugefügt wird, kann dieser Indikator unsichtbar sein, obwohl er in der Liste der Indikatoren angezeigt wird. Dies bedeutet, dass Skala des Indikators von der Skala des Charts sich unterschiedet, und Werte des Indikators sind unsichtbar in angezeigte Skala des Charts. In diesen Fall gibt GetLastError() Code von Null zurück, das bedeute keinen Fehler. Die Werte dieses "unsichtbaren" Indikators wird in der "Datenfenster" sichtbar und können auch von anderen MQL5-Programme erhalten werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Expert Advisor demonstriert Funktion ChartIndicatorAdd()."
#property description "Nach dem Start auf dem Chart (und Bekommen von Fehler im Log),
#property description "Eigenschaften von Expert Advisor und geben Sie die richtigen Pa
#property description "Indikator MACD wird auf dem Chart hinzugefügt."

//--- input parameters
input string          symbol="AUDUSD";     // Symbolname
input ENUM_TIMEFRAMES period=PERIOD_M12;   // Zeitrahmen
input int    fast_ema_period=12;           // Periode der raschen MACD
input int    slow_ema_period=26;           // Periode der langsamen MACD
input int      signal_period=9;            // Mittelungsperiode der Unterschied
input ENUM_APPLIED_PRICE apr=PRICE_CLOSE; // Preisetyp für Berechnen von MACD

int indicator_handle=INVALID_HANDLE;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   indicator_handle=iMACD(symbol,period,fast_ema_period,slow_ema_period,signal_period,
//--- versuchen wir den Indikator zum Chart hinzuzufügen
   if(!AddIndicator())
     {
      //--- Funktion AddIndicator() weigerte sich, den Indikator zum Chart hinzuzufüge
      int answer=MessageBox("Trotzdem versuchen MACD auf dem Chart hinzufügen?",
                            "Falscher Symbol und/oder der Zeitrahmen für die Zugabe de
                            MB_YESNO // Auswahlschaltfläche "Yes" und "No" gezeigt wer
                            );
      //--- wenn Benutzer beharrt auf dem Missbrauch von ChartIndicatorAdd()
      if(answer==IDYES)
        {
         //--- erst anzeigen es im Log
         PrintFormat("Achtung! %s: Wir versuchen indikator MACD(%s/%s) auf dem Chart %
                     __FUNCTION__,symbol,EnumToString(period),_Symbol,EnumToString(_Pe
         //--- erhalten den Nummer von neuen Sub-Fenster, in dem wir versuchen, einen
         int subwindow=(int)ChartGetInteger(0,CHART_WINDOWS_TOTAL);
         //--- jetzt einen fehleranfälligen Versuch machen
         if(!ChartIndicatorAdd(0,subwindow,indicator_handle))
            PrintFormat("MACD konnte nicht auf dem Fenster %d des Charts hinzugefügt w
                        subwindow,GetLastError());
        }
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
// Expert Advisor mach nichts
  }
//+------------------------------------------------------------------+
//| Funktion prüft und fügt den Indikator auf den Chart zu           |
//+------------------------------------------------------------------+
bool AddIndicator()
  {
//--- Nachricht
   string message;
//--- Prüfen wir den Symbol des Indikators und Symbol des Charts
```

© 2000-2025, MetaQuotes Ltd.

```
   if(symbol!=_Symbol)
     {
      message="Demonstrieren die Verwendung der Funktion Demo_ChartIndicatorAdd():";
      message=message+"\r\n";
      message=message+"Sie können nicht den Indikator, der an anderem Symbol berechnet
      message=message+"\r\n";
      message=message+"Geben Sie den Chartsymbol in Eigenschaften des Experts ein - "-
      Alert(message);
      //--- Frühausgang, hinzufügen wir keinen Indikator zum Chart
      return false;
     }
//--- Prüfen wir Zeitrahmen des Indikators und Zeitrahmen des Charts
   if(period!=_Period)
     {
      message="Sie können nicht den Indikator, der an anderer Zeitrahmen berechnet wa
      message=message+"\r\n";
      message=message+"Geben Sie Zeitrahmen des Charts in Eigenschaften von Expert Adv
      Alert(message);
      //--- Frühausgang, hinzufügen wir keinen Indikator zum Chart
      return false;
     }
//--- alle Tests bestanden haben, Symbol und Periode des Indikators mit Chart gleich s
   if(indicator_handle==INVALID_HANDLE)
     {
      Print(__FUNCTION__,"  Erstellen wir MACD");
      indicator_handle=iMACD(symbol,period,fast_ema_period,slow_ema_period,signal_peri
      if(indicator_handle==INVALID_HANDLE)
        {
         Print("MACD konnte nicht erstellt werden. Fehlercode",GetLastError());
        }
     }
//--- reset Fehlercode
   ResetLastError();
//--- Fügen den Indikator auf dem Chart hinzu
   Print(__FUNCTION__,"  Fügen den Indikator MACD auf dem Chart hinzu");
   Print("MACD ist gebaut auf ",symbol,"/",EnumToString(period));
//--- Bekommen wir den Nummer des neuen Fenster, auf dem MACD hinzugefügt wird
   int subwindow=(int)ChartGetInteger(0,CHART_WINDOWS_TOTAL);
   PrintFormat("Fügen den Indikator auf dem Fenster %d des Charts hinzu",subwindow);
   if(!ChartIndicatorAdd(0,subwindow,indicator_handle))
     {
      PrintFormat("MACD konnte nicht auf dem Fenster %d des Chart hinzugefügt werden.
                  subwindow,GetLastError());
     }
//--- Indikator war erfolgreich auf dem Chart hinzugefügt
   return(true);
  }
```

Sehen Sie auch

ChartIndicatorDelete(), ChartIndicatorName(), ChartIndicatorsTotal(), iCustom(), IndicatorCreate()

© 2000-2025, MetaQuotes Ltd.

ChartIndicatorDelete

Löscht Indikator mit angegebenem Namen aus dem angegebenen Chartfenster.

```
bool  ChartIndicatorDelete(
   long           chart_id,              // Identifikator des Charts
   int            sub_window             // Nummer des Unterfensters
   const string   indicator_shortname    // Kurzname des Indikators
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet das laufende Chart.

sub_window

[in] Nummer des Unterfensters des Charts. 0 bedeutet das Hauptfenster des Charts.

const indicator_shortname

[in] Kurzname des Indikators, der in INDICATOR_SHORTNAME Eigenschaft bei der Funktion IndicatorSetString() angegeben wird. Sie könne den Kurzname bei der Funktion ChartIndicatorName() ermitteln.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Wenn es mehrere Indikatoren mit der gleichen Kurznamen in das angegebene Unterfenster des Charts gibt, wird das erste entfernt werden.

Wenn an den Werte des gelöschten Indikator andere Indikatoren auf dem gleichen Chart konstruiert sind, werden sie auch gelöscht werden.

Mann sollte nicht Indikators Kurzname mit dem Dateinamen der beim Erstellen des Indikators mit den Funktionen iCustom() und IndicatorCreate() gezeigt wird verwechseln. Wenn der Kurzname des Indikators nicht explizit festgelegt wird, dann bei der Kompilation wird den Namen der Quelldatei des Indikators angegeben.

Entfernung eines Indikators aus dem Chart bedeutet nicht, dass der Berechnungsteil des Indikators auch aus dem Terminalspeicher gelöscht wird. Um Handle des Indikators freizumachen, benutzen Sie die Funktion IndicatorRelease().

Sie sollen richtig einen kurzen Namen des Indikators formen, der in der Eigenschaft INDICATOR_SHORTNAME mit Hilfe der FunktionIndicatorSetString() geschrieben wird. Wir empfehlen, dass der Kurzname die Werte der Eingabeparameter des Indikators enthält, da der Indikator, der aus dem Chart gelöscht wird, in der Funktion ChartIndicatorDelete() beim Kurzname identifiziert wird.

Beispiel für das Löschen eines Indikators, wenn Initialisierung falsch war

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                    Demo_ChartIndicatorDelete.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot Histogram
#property indicator_label1  "Histogram"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input int      first_param=1;
input int      second_param=2;
input int      third_param=3;
input bool     wrong_init=true;
//--- indicator buffers
double         HistogramBuffer[];
string         shortname;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
   int res=INIT_SUCCEEDED;
//--- Den Array HistogramBuffer zum Indikatorpuffer binden
   SetIndexBuffer(0,HistogramBuffer,INDICATOR_DATA);
//--- Einen kurzen Namen des Indikators auf Eingabeparameter konstruieren
   shortname=StringFormat("Demo_ChartIndicatorDelete(%d,%d,%d)",
                          first_param,second_param,third_param);
   IndicatorSetString(INDICATOR_SHORTNAME,shortname);
//--- Wenn Zwangsendung des Indikatiors eingegeben ist, dann Wert ungleich Null zurück
   if(wrong_init) res=INIT_FAILED;
   return(res);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Ausgangsposition für Arbeit im Zyklus
   int start=prev_calculated-1;
   if(start<0) start=0;
//--- Füllen Indikator-Puffer mit Werte
   for(int i=start;i<rates_total;i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      HistogramBuffer[i]=close[i];
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Handler von Ereignis Deinit                                      |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   PrintFormat("%s: Code vom Deinitialisierungsgrund=%d",__FUNCTION__,reason);
   if(reason==REASON_INITFAILED)
     {
      PrintFormat("Der Indikator mit dem Kurznamen %s (Datei %s) löscht sich aus dem C
      int window=ChartWindowFind();
      bool res=ChartIndicatorDelete(0,window,shortname);
      //--- Analysieren wir das Ergebnis des Aufrufs von ChartIndicatorDelete()
      if(!res)
        {
         PrintFormat(Indikator %s konnte nicht aus dem Fenster #%d gelöscht werden. Fe
                     shortname,window,GetLastError());
        }
     }
  }
```

Sehen Sie auch

ChartIndicatorAdd(), ChartIndicatorName(), ChartIndicatorsTotal(), iCustom(), IndicatorCreate(), IndicatorSetString()

© 2000-2025, MetaQuotes Ltd.

ChartIndicatorGet

Gibt das Handle des Indikators mit dem angegebenen Kurznamen im angegebenen Chart-Fenster zurück.

```
int  ChartIndicatorGet(
   long           chart_id,              // ID des Charts
   int            sub_window             // Nummer des Unterfensters
   const string   indicator_shortname    // Kurzname des Indikators
   );
```

Parameter

chart_id

[in] ID des Charts. 0 bedeutet den aktuellen Chart.

sub_window

[in] Nummer des Unterfensters. 0 bedeutet das Hauptfenster des Charts.

const indicator_shortname

[in] Kurzname des Indikators, der in der Eigenschaft INDICATOR_SHORTNAME durch Funktion IndicatorSetString() angegeben wird. Ein Kurzname des Indikators kann durch Funktion ChartIndicatorName() erhalten werden.

Rückgabewert

Gibt Indikator-Handle beim Erfolg zurück, ansonsten INVALID_HANDLE. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError() an.

Hinweis

Das mit der Funktion ChartIndicatorGet() erhaltene Indikator-Handle erhöht den internen Zähler der verwendeten Indikatoren. Das Laufzeitsystem des Terminals hält alle Indikatoren, deren Zähler größer als Null ist, geladen. Daher sollte das Indikatorhandle, das nicht mehr benötigt wird, sofort und explizit mit IndicatorRelease() im gleichen Programm, das es empfangen hat, freigegeben werden, wie das folgende Beispiel zeigt. Andernfalls wird es unmöglich sein, das "herrenlose" Handle zu finden und es von einem anderen Programm korrekt freizugeben.

Sie müssen richtig den Kurzamen des Indikators währen seiner Erstellung generieren. Der Name wird durch die Funktion IndicatorSetString() in der Eigenschaft INDICATOR_SHORTNAME geschrieben werden. Es wird empfohlen, dass ein Kurznamen die Werte der Input-Parameter des Indikators enthält, da dein Indikators in der Funktion ChartIndicatorGet() nach dem Kurznamen indentifiziert wird.

Ein anderer Weg, um den Indikator zu identifizieren - eine Liste seiner Parameter für einen bestimmten Handle durch der Funktion IndicatorParameters() zu erhalten und dann die erhaltenen Werte zu analysieren.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   //--- Anzahl der Fenster auf dem Chart (es gibt immer mindestens ein Hauptfenster)
   int windows=(int)ChartGetInteger(0,CHART_WINDOWS_TOTAL);
   //--- für alle Fenster machen
   for(int w=0;w<windows;w++)
     {
      //--- Anzahl der Indikatoren in diesem Fenster/Untenfenster
      int total=ChartIndicatorsTotal(0,w);
      //--- für alle Fenster machen
      for(int i=0;i<total;i++)
        {
         //--- erhalten wir den kurzen Namen des Indikators
         string name=ChartIndicatorName(0,w,i);
         //--- erhalten wir Indikator-Handle
         int handle=ChartIndicatorGet(0,w,name);
         //--- In Log schreiben
         PrintFormat("Window=%d,  index=%d,  Name=%s,  handle=%d",w,i,name,handle);
         //--- sicherlich befreien wir den Indikatorhandle, wenn er nicht mehr benötig
         IndicatorRelease(handle);
        }
     }
  }
```

Sehen Sie auch

ChartIndicatorAdd(), ChartIndicatorName(), ChartIndicatorsTotal(), IndicatorParameters()

© 2000-2025, MetaQuotes Ltd.

ChartIndicatorName

Gibt den Kurznamen des Indikators an der Index-Position in der Indikatorenliste des Chartfensters zurück.

```
string  ChartIndicatorName(
   long   chart_id,       // Identifikator des Charts
   int    sub_window      // Nummer des Unterfensters
   int    index           // Index des Indikators in der Liste der Indikatoren, die zu d
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet das aktuelle

sub_window

[in] Nummer des Unterfensters des Charts. 0 bedeutet das Hauptfenster des Charts.

index

[in] Index des Indikators in der Liste der Indikatoren. Die Indexierung der Indikatorenliste beginnt bei Null, das heißt, der erste Indikator in der Liste hat den Index Null. Um die Anzahl der Indikatoren in der Liste zu erhalten, benutzen Sie die Funktion ChartIndicatorsTotal().

Rückgabewert

Kurzname des Indikators, der in der Eigenschaft INDICATOR_SHORTNAME mit Hilfe der Funktion IndicatorSetString() angegeben wird. Sie können den Kurznamen eines Indikators mit der Funktion ChartIndicatorName() abrufen. Um die Information über einen Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Mann darf den Kurznamen des Indikators nicht mit dem Dateinamen verwechseln, der bei der Verwendung des Indikators mit den Funktionen iCustom() und IndicatorCreate() gezeigt wird. Wenn der Kurzname des Indikators nicht explizit festgelegt wird, dann wird bei der Kompilation der Name der Quelldatei des Indikators angegeben.

Sie sollten einen richtig formierten Kurznamen des Indikators bilden, der in der Eigenschaft INDICATOR_SHORTNAME mit Hilfe der FunktionIndicatorSetString() geschrieben wird. Wir empfehlen, dass der Kurzname die Werte der Eingabeparameter des Indikators enthält, da der Indikator, der aus dem Chart gelöscht wird, mit der Funktion ChartIndicatorDelete() über den Kurznamen identifiziert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der ID des aktuellen Charts und die Anzahl seiner Unterfenster, einschl
   long chart_id = ChartID();
```

© 2000-2025, MetaQuotes Ltd.

```
   int  wnd_total= (int)ChartGetInteger(0,CHART_WINDOWS_TOTAL);

//--- Iteration über alle aktuellen Chartfenster in einer Schleife
   for(int w=0; w<wnd_total; w++)
     {
      //--- abrufen der Anzahl der Indikatoren, die in dem durch den Schleifenindex sp
      int ind_total=ChartIndicatorsTotal(chart_id, w);

      //--- Ausdruck der Kopfzeile für das ausgewählte Chartfenster
      PrintFormat("Chart window %d indicators: ", w);

      //--- In der Schleife alle Namen der Indikatoren der Variablen ausdrucken, die d
      string ind_names="";
      for(int i=0; i<ind_total; i++)
        {
         ind_names+="  "+ChartIndicatorName(chart_id, w, i)+(i<ind_total-1 ? "\n": "")
        }
      //--- die erhaltene Liste der Namen aller Indikatoren, die an das angegebene Cha
      Print(ind_names);
     }
   /*
   Ergebnis:
   Chart window 0 indicators:
     AMA(9,2,30)
     SAR(0.02,0.20)
     Fractals
   Chart window 1 indicators:
     RSI(14)
     AMA(9,2,30)
   Chart window 2 indicators:
     MFI(14)
   */
  }
```

Siehe auch

ChartIndicatorAdd(), ChartIndicatorDelete(), ChartIndicatorsTotal(), iCustom(), IndicatorCreate(), IndicatorSetString()

© 2000-2025, MetaQuotes Ltd.

ChartIndicatorsTotal

Liefert die Anzahl aller Indikatoren des angegebenen Chartfensters.

```
int  ChartIndicatorsTotal(
   long  chart_id,      // Identifikator des Charts
   int   sub_window     // Nummer des Unterfensters
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet das aktuelle

sub_window

[in] Nummer des Unterfensters des Charts. 0 bedeutet das Hauptfenster des Charts.

Rückgabewert

Anzahl aller Indikatoren des angegebenen Chartfensters. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Funktion ermöglicht alle Indikatoren des Charts durchzugehen .

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der ID des aktuellen Charts und die Anzahl seiner Unterfenster, einschli
   long chart_id = ChartID();
   int  wnd_total= (int)ChartGetInteger(0,CHART_WINDOWS_TOTAL);

//--- Iteration über alle aktuellen Chartfenster in einer Schleife
   for(int w=0; w<wnd_total; w++)
     {
      //--- abrufen der Anzahl der Indikatoren, die in dem durch den Schleifenindex sp
      int ind_total=ChartIndicatorsTotal(chart_id, w);

      //--- get the number of indicators attached to the specified chart window
      PrintFormat("Chart ID %I64d, subwindow %d. Attached indicators: %d", chart_id, w
     }
   /*
   Ergebnis:
   Chart ID 133246248352168439, subwindow 0. Attached indicators: 3
   Chart ID 133246248352168439, subwindow 1. Attached indicators: 2
   Chart ID 133246248352168439, subwindow 2. Attached indicators: 1
   */
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Siehe auch

ChartIndicatorAdd(), ChartIndicatorDelete(), iCustom(), IndicatorCreate(), IndicatorSetString()

© 2000-2025, MetaQuotes Ltd.

ChartWindowOnDropped

Gibt die Nummer des Subfensters zurück, wohin dieser Expert oder Script mit der Maus versetzt wurde. 0 bedeutet Hauptfenster des Charts.

```
int  ChartWindowOnDropped();
```

Rückgabewert

Wert des Typs int.

Beispiel:

```
   int myWindow=ChartWindowOnDropped();
   int windowsTotal=ChartGetInteger(0,CHART_WINDOWS_TOTAL);
   Print("Script ist im Fenster ausgeführt #"+myWindow+
         ". Insgesamt Fenster auf dem Chart "+ChartSymbol()+": ",windowsTotal);
```

Sehen Sie auch

ChartPriceOnDropped, ChartTimeOnDropped, ChartXOnDropped, ChartYOnDropped

© 2000-2025, MetaQuotes Ltd.

ChartPriceOnDropped

Gibt die Preiskoordinate zurück, wohin dieser Expert oder Script versetzt wurde.

```
double  ChartPriceOnDropped();
```

Rückgabewert

Wert des Typs double.

Beispiel:

```
   double p=ChartPriceOnDropped();
   Print("ChartPriceOnDropped() = ",p);
```

Sehen Sie auch

ChartXOnDropped, ChartYOnDropped

© 2000-2025, MetaQuotes Ltd.

ChartTimeOnDropped

Gibt die Zeitkoordinate zurück, wohin dieser Expert oder Script mit der Maus versetzt wurde.

```
datetime  ChartTimeOnDropped();
```

Rückgabewert

Wert des Typs datetime.

Beispiel:

```
   datetime t=ChartTimeOnDropped();
   Print("Script wasdropped on the "+t);
```

Sehen Sie auch

ChartXOnDropped, ChartYOnDropped

© 2000-2025, MetaQuotes Ltd.

ChartXOnDropped

Gibt X-Koordinate zurück, wohin dieser Expert oder Script versetzt wurde.

```
int  ChartXOnDropped();
```

Rückgabewert

Wert der X-Koordinate.

Hinweis

X-Achse ist von links nach rechts gerichtet.

Beispiel:

```
   int X=ChartXOnDropped();
   int Y=ChartYOnDropped();
   Print("(X,Y) = ("+X+","+Y+")");
```

Sehen Sie auch

ChartWindowOnDropped, ChartPriceOnDropped, ChartTimeOnDropped

© 2000-2025, MetaQuotes Ltd.

ChartYOnDropped

Gibt Y-Koordinate zurück, wo der Expert oder das Script mit der Maus platziert wurde.

```
int  ChartYOnDropped();
```

Rückgabewert

Wert der Y-Koordinate.

Hinweis

Y-Achse ist von oben nach unten gerichtet.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
/--- liefert die X- und Y-Koordinaten des Chartpunktes, auf den das Skript mit der Mau
   int x=ChartXOnDropped();
   int y=ChartYOnDropped();
   PrintFormat("Script dropped to coordinates X = %d, Y = %d", x, y);
   /*
   Ergebnis:
   Script dropped to coordinates X = 429, Y = 114
   */
  }
```

Siehe auch

ChartWindowOnDropped, ChartPriceOnDropped, ChartTimeOnDropped

© 2000-2025, MetaQuotes Ltd.

ChartSetSymbolPeriod

Damit können das Symbol und der Zeitrahmen des angegebenen Charts verändert werden. Die Funktion arbeitet asynchron, sie übergibt den Befehl und wartet nicht auf die Durchführung. Der Befehl wird in die Warteschlange des Charts gestellt und wird nach der Erledigung aller vorherigen Befehle abgearbeitet.

```
bool  ChartSetSymbolPeriod(
   long             chart_id,      // Identifikator des Charts
   string           symbol,        // Symbolname
   ENUM_TIMEFRAMES  period         // Zeitrahmen
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet denen Chart.

symbol

[in] Symbol des Charts. NULL bedeutet Symbol des aktuellen Charts (auf dem der Experte läuft)

period

[in] Zeitrahmen des Charts (Timeframe). Kann einen der Werte der Enumeration ENUM_TIMEFRAMES annehmen. 0 bedeutet Zeitrahmen des aktuellen Charts.

Rückgabewert

Gibt true zurück, wenn der Befehl in die Warteschlange des Charts gestellt wurde, andernfalls false. Um die Information über den Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Ein Wechsel von Symbol/Zeitrahmen veranlasst den Expert Advisor, der auf dem Chart läuft, sich neu zu initialisieren.

Der Aufruf von ChartSetSymbolPeriod mit demselben Symbol und Zeitrahmen kann für ein Update des Charts verwendet werden (ähnlich dem Befehl zur Aktualisierung des Charts). Ein Update des Charts wiederum löst eine Neuberechnung der mit ihm verbundenen Indikatoren aus. Auf diese Weise können Sie einen Indikator auf dem Chart berechnen, auch wenn es keine Ticks gibt (z.B. an Wochenenden).

Beispiel:

```
#define   SYMBOL    "GBPUSD"
#define   PERIOD    PERIOD_H1

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- liefert die aktuelle Chart-ID, das Symbol und den Zeitrahmen
   long           chart_id= ChartID();
```

© 2000-2025, MetaQuotes Ltd.

```
   string          symbol  = Symbol();
   ENUM_TIMEFRAMES period  = Period();

//--- Bericht im Journal über das Ersetzen des aktuellen Chartsymbols und des Zeitrahr
   PrintFormat("Change the %s symbol and the %s period of the chart %I64u to %s %s",
               symbol, TimeframeDescrioption(period), chart_id, SYMBOL, TimeframeDescr

//--- Chartsymbol und Zeitrahmen ändern
   ChartSetSymbolPeriod(chart_id, SYMBOL, PERIOD);
   /*
   Ergebnis:
   Change the EURUSD symbol and the M1 period of the chart 133246248352168440 to GBPUS
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Zeitrahmens des Charts             |
//+------------------------------------------------------------------+
string TimeframeDescrioption(const ENUM_TIMEFRAMES timeframe)
  {
   return(StringSubstr(EnumToString(timeframe), 7));
  }
```

Siehe auch

ChartSymbol, ChartPeriod

© 2000-2025, MetaQuotes Ltd.

ChartScreenShot

Die Funktion gewaehrleistet Screenshot des angegebenen Charts im laufenden Zustand in Format GIF, PNG oder BMP je nach angegebenen Erweiterung.

```
bool  ChartScreenShot(
   long             chart_id,                  // Identifikator des Charts
   string           filename,                  // Dateiname
   int              width,                     // Breite
   int              height,                    // Höhe
   ENUM_ALIGN_MODE  align_mode=ALIGN_RIGHT     // Typ der Ausrichtung
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

filename

[in] Dateiname des Screenshot. Kann nicht mehr als 63 Symbole sein. Screenshot wird m Verzeichnis \Files aufbewahren.

width

[in] Breite des Sreenshot in Pixel

height

[in] Hoehe des Screenshot in PixelDateiname des Screenshot

align_mode=ALIGN_RIGHT

[in] Ausgabemode des engen Screenshot. Wert von Enumeration ENUM_ALIGN_MODE. ALIGN_RIGHT bedeutet Ausrichtung rechtsbündig. (Ausgabe vom Ende). ALIGN_LEFT bedeutet Ausrichtung linksbündig.

Rückgabewert

Bringt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Wenn es notwendig ist, Screenshot von der bestimmten Position zu nehmen, muss der Chart vor allem durch die Funktion ChartNavigate() positioniert werden. Wenn horizontale Größe des Screenshots kleiner las Fenster des Charts ist, wird entweder der rechte Chartteil oder der linke Chartteil ausgegeben, abhängig vom Parameterwert align_mode.

Beispiel:

```
#property description "Der Expert Advisor zeigt, wie eine Reihe von Screenshots des al
#property description "mit der ChartScreenShot()-Funktion zu erstellen. Für die Bequer
#property description "auf den Chart angezeigt. Die Höhe und Breite von Bildern wird c

#define        WIDTH  800     // Breite des Bildes für Anruf von ChartScreenShot()
#define        HEIGHT 600     // Höhe des Bildes für Anruf von ChartScreenShot()
```

© 2000-2025, MetaQuotes Ltd.

```
//--- input parameters
input int      pictures=5;    // Anzahl der Bilder in Serie
int            mode=-1;       // -1 bedeutet Einrückung an den rechten Rand des Chart,
int            bars_shift=300;// Anzahl der Balken beim Scroll des Charts mit ChartNav
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- Autoscroll des Charts deaktivieren
   ChartSetInteger(0,CHART_AUTOSCROLL,false);
//--- Einrückung des rechten Rands des Charts angeben
   ChartSetInteger(0,CHART_SHIFT,true);
//--- Darstellung des Charts in Form von Kerzendiagramm
   ChartSetInteger(0,CHART_MODE,CHART_CANDLES);
//---
   Print("Vorbereitung des Expert Advisors für die Arbeit ist abgeschlossen");
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
//--- Ausgabe der Name der Funktion, der Zeit des Aufrufs und Ereignis-ID
   Print(__FUNCTION__,TimeCurrent(),"   id=",id,"   mode=",mode);
//--- Verarbeitung von Ereignissen CHARTEVENT_CLICK ("Klick mit der Maus auf dem Chart
   if(id==CHARTEVENT_CLICK)
     {
      //--- Erste Einrückung vom Rand des Charts
      int pos=0;
      //--- Arbeit mit dem linken Rand des Charts
      if(mode>0)
        {
         //--- Den Chart nach den linken Rand scrollen
         ChartNavigate(0,CHART_BEGIN,pos);
         for(int i=0;i<pictures;i++)
           {
            //--- Die Beschriftung auf dem Chart und den Dateinamen vorbereiten
```

© 2000-2025, MetaQuotes Ltd.

```
            string name="ChartScreenShot"+"CHART_BEGIN"+string(pos)+".gif";
            //--- Ausgabe des Namen auf dem Chart als Kommentar
            Comment(name);
            //--- Den Screenschot in der Datei in Terminal_Verzeichnis\MQL5\Files\ spe
            if(ChartScreenShot(0,name,WIDTH,HEIGHT,ALIGN_LEFT))
               Print("Gespeciherter Screenshot ",name);
            //---
            pos+=bars_shift;
            //--- Gib dem Benutzer Zeit, um auf neuen Abschnitt des Chart zu sehen
            Sleep(3000);
            //--- Den Chart von der aktuellen Position um bars_shift nach rechts scrol
            ChartNavigate(0,CHART_CURRENT_POS,bars_shift);
           }
         //--- Regimewechsel auf die gegenüberliegende
         mode*=-1;
        }
      else // Arbeit mit dem rechten Rand des Charts
        {
         //--- Den Chart nach den rechten Rand scrollen
         ChartNavigate(0,CHART_END,pos);
         for(int i=0;i<pictures;i++)
           {
            //--- Die Beschriftung auf dem Chart und den Dateinamen vorbereiten
            string name="ChartScreenShot"+"CHART_END"+string(pos)+".gif";
            //--- Ausgabe des Namen auf dem Chart als Kommentar
            Comment(name);
            //--- Den Screenschot in der Datei in Terminal_Verzeichnis\MQL5\Files\ spe
            if(ChartScreenShot(0,name,WIDTH,HEIGHT,ALIGN_RIGHT))
               Print("Gespeciherter Screenshot ",name);
            //---
            pos+=bars_shift;
            //--- Gib dem Benutzer Zeit, um auf neuen Abschnitt des Chart zu sehen
            Sleep(3000);
            //--- Den Chart von der aktuellen Position um bars_shift nach rechts scrol
            ChartNavigate(0,CHART_CURRENT_POS,-bars_shift);
           }
         //--- Regimewechsel auf die gegenüberliegende
         mode*=-1;
        }
     }  // Ende der Erarbeitung von CHARTEVENT_CLICK
//--- Ende von Handler OnChartEvent()
  }
```

Sehen Sie auch

ChartNavigate(), Ressourcen

© 2000-2025, MetaQuotes Ltd.

Handelsfunktionen

Funktionsgruppe für die Verwaltung der Handelstätigkeit.

Bevor Sie beginnen die Trade-Funktionen der Plattform zu nutzen, müssen Sie ein klares Verständnis der standardmäßigen Begriffe Order, Deal und Position besitzen.

- Eine Order ist die Anweisung an den Broker ein Finanzinstrument zu kaufen oder zu verkaufen. Es gibt zwei Haupt-Ordertypen: Markt und Pending. Zusätzlich gibt es Take Profit und Stop Loss Level.
- Ein Deal ist der wirtschaftliche Austausch eines Finanzinstruments (Kaufen/Verkaufen). Das Kaufen wird zum Nachfragepreis (Ask-Preis) und das Verkaufen zum Angebotspreis (Bid) durchgeführt. Ein Deal kann eröffnet werden als Ergebnis einer Market Order oder einer ausgelösten Pending Order. Beachten Sie, dass in einigen Fällen das Ausführen einer Order mit mehreren Deals geschehen kann.
- Eine Position ist eine Trading-Verpflichtung, also die Anzahl an gekauften oder verkauften Kontrakten in einem Finanzinstrument. Eine Long-Position ist ein gekauftes Handelsinstrument, mit der Erwartung, dass der Preis steigt. Eine Short-Position ist die Verpflichtung ein Instrument in der Zukunft bereitzustellen, in Erwartung eines niedrigeren Preises.

Allgemeine Informationen über Handelsgeschäfte finden Sie in der Hilfe des Terminals des Kunden.

Handelsfunktionen können in Experten und Skripts benutzt werden. Handelsfunktionen können erst dann aufgerufen werden, wenn es in Eigenschaften entsprechender Experten oder Scripts das Kontrollkästchen "Ratgeber zu handeln erlauben" gibt.

Handelserlaubnis oder Handelsverbot kann von vielen Faktoren abhängig sein. Diese Faktoren sind im Abschnitt "Handelserlaubnis" beschrieben sind.

Funktion	Handlung
OrderCalcMargin	Berechnet die Größe der Marge, die für den angegebenen Typ der Order notwendig ist, in der Währung des Kontos
OrderCalcProfit	Berechnet die Größe des Gewinns aufgrund der übertragenen Parameter, in der Währung des Kontos
OrderCheck	Prüft, ob es genug Geld ist, um die notwendige Handelsoperation auszuführen.
OrderSend	Sendet Handelsanforderungen auf das Sever
OrderSendAsync	Asynchron sendet Handelsanforderungen, ohne auf die Antwort des Handelsservers zu warten
PositionsTotal	Bringt Anzahl der offenen Positionen zurück
PositionGetSymbol	Bringt das Symbol der entsprechenden offenen Position zurück
PositionSelect	Wählt die offene Position für die weitere Arbeit damit
PositionSelectByTicket	Selects a position to work with by the ticket number specified in it
PositionGetDouble	Bringt die angeforderte Eigenschaft der offenen Position zurück (double)

© 2000-2025, MetaQuotes Ltd.

Funktion	Handlung
PositionGetInteger	Bringt die angeforderte Eigenschaft der offenen Position zurück (datetime oder int)
PositionGetString	Bringt die angeforderte Eigenschaft der offenen Position zurück (string)
PositionGetTicket	Returns the ticket of the position with the specified index in the list of open positions
OrdersTotal	Bringt die Zahl der Ordern zurück
OrderGetTicket	Bringt Ticket der entsprechenden Order zurück
OrderSelect	Wählt Order für die weitere Arbeit damit
OrderGetDouble	Bringt die angeforderte Eigenschaft der Order zurück (double)
OrderGetInteger	Bringt die angeforderte Eigenschaft der Order zurück (datetime oder int)
OrderGetString	Bringt die angeforderte Eigenschaft der Order zurück (string)
HistorySelect	Fordert die Deals- und Ordergeschichte für die angegebene Periode der Severzeit an
HistorySelectByPosition	Fordert die Geschichte der Deals und der Order mit dem angegebenen Identifikator der Position an
HistoryOrderSelect	Wählt Order in der Geschichte für die weitere Arbeit damit
HistoryOrdersTotal	Bringt die Orderzahl in der Geschichte zurück
HistoryOrderGetTicket	Bringt Ticket der entsprechenden Order in der Geschichte zurück
HistoryOrderGetDouble	Bringt die angeforderte Eigenschaft der Order in der Geschichte zurück (double)
HistoryOrderGetInteger	Bringt die angeforderte Eigenschaft der Order in der Geschichte zurück (datetime oder int)
HistoryOrderGetString	Bringt die angeforderte Eigenschaft der Order in der Geschichte zurück (string)
HistoryDealSelect	Wählt das Deal in der Geschichte für ihre weitere Anwendung durch entsprechende Funktionen
HistoryDealsTotal	Bringt die Dealszahl in der Geschichte zurück
HistoryDealGetTicket	Wählt das Deal für die weitere Verarbeitung und bringt Ticket des Deals in der Geschichte zurück
HistoryDealGetDouble	Bringt die angeforderte Eigenschaft des Deals in der Geschichte zurück (double)
HistoryDealGetInteger	Bringt die angeforderte Eigenschaft des Deals in der Geschichte zurück (datetime oder int)

© 2000-2025, MetaQuotes Ltd.

Funktion	Handlung
HistoryDealGetString	Bringt die angeforderte Eigenschaft des Deals in der Geschichte zurück (string)

© 2000-2025, MetaQuotes Ltd.

OrderCalcMargin

Berechnet die Höhe der Marge für den angegebenen Typ der Order auf dem laufenden Konto und bei der aktuellen Marktumgebung, ohne schwebenden Aufträge und offenen Positionen einzubeziehen. Ermöglicht die Höhe der Marge für die geplante Handelsoperation einzuschätzen. Wert wird in der Währung des Kontos zurückgegebenen.

```
bool  OrderCalcMargin(
   ENUM_ORDER_TYPE       action,           // Typ der Order
   string                symbol,           // Symbolname
   double                volume,           // Volumen
   double                price,            // Eröffnungspreis
   double&               margin            // Variable für den Erhalt des Wertes der M
   );
```

Parameter

action

[in] Typ der Order, kann Werte aus der Enumeration ENUM_ORDER_TYPE annehmen.

symbol

[in] Name des Finanzinstrumentes.

volume

[in] Volumen der Handelsoperation.

price

[in] Eröffnungspreis.

margin

[out] Variable, der die notwendige Höhe der Marge zugewiesen wird, wenn die Funktion erfolgreich durchgeführt wird. Berechnung erfolgt als gäbe es auf dem laufenden Konto keine schwebenden Aufträge oder offene Positionen. Der Wert der Marge hängt von vielen Faktoren ab und kann sich bei der Veränderung der Marktumgebung verändern.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false. Für den Erhalt der Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

```
#define   VOLUME     1.0   // Auftragsvolumen
#define   DEVIATION  100   // Abstand für das Platzieren von schwebenden Aufträgen
#define   STOP_LIMIT 50    // Abstand eines StopLimit-Auftrages distance

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   string currency=AccountInfoString(ACCOUNT_CURRENCY);
```

© 2000-2025, MetaQuotes Ltd.

```
   int array_types[8]={ORDER_TYPE_BUY,
                       ORDER_TYPE_SELL,
                       ORDER_TYPE_BUY_LIMIT,
                       ORDER_TYPE_SELL_LIMIT,
                       ORDER_TYPE_BUY_STOP,
                       ORDER_TYPE_SELL_STOP,
                       ORDER_TYPE_BUY_STOP_LIMIT,
                       ORDER_TYPE_SELL_STOP_LIMIT };

//--- in einer Schleife durch das Ordertyp-Array
   for(int i=0; i<(int)array_types.Size(); i++)
     {
      //--- abhängig vom Ordertyp, entweder ORDER_TYPE_BUY oder ORDER_TYPE_SELL
      ENUM_ORDER_TYPE type=MarketOrderByOrderType((ENUM_ORDER_TYPE)array_types[i]);

      //--- abhängig vom Auftragstyp den Preis abrufen
      double price=PriceByOrderType(_Symbol, type);

      //--- Abrufen des Margenbetrags, der für die „action"-angegebenen Auftragstyp be
      double margin=EMPTY_VALUE;
      ResetLastError();
      if(!OrderCalcMargin(type, _Symbol, VOLUME, price, margin))
        {
         Print("OrderCalcMargin() failed. Error ",GetLastError());
         continue;
        }
      //--- Ausdruck des Ergebnisses im Journal
      PrintFormat("Margin required for %.2f %s order at price %.*f on %s symbol: %.2f
     }
   /*
   Ergebnis:
   Margin required for 1.00 Buy order at price 1.31668 on GBPUSD symbol: 1316.68 USD
   Margin required for 1.00 Sell order at price 1.31661 on GBPUSD symbol: 1316.61 USD
   Margin required for 1.00 Buy Limit order at price 1.31568 on GBPUSD symbol: 1315.68
   Margin required for 1.00 Sell Limit order at price 1.31761 on GBPUSD symbol: 1317.6
   Margin required for 1.00 Buy Stop order at price 1.31768 on GBPUSD symbol: 1317.68
   Margin required for 1.00 Sell Stop order at price 1.31561 on GBPUSD symbol: 1315.61
   Margin required for 1.00 Buy Stop Limit order at price 1.31718 on GBPUSD symbol: 13
   Margin required for 1.00 Sell Stop Limit order at price 1.31611 on GBPUSD symbol: 1
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe des Auftragstyps „Kaufen"-oder „Verkaufen"-             |
//+------------------------------------------------------------------+
ENUM_ORDER_TYPE MarketOrderByOrderType(ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY  : case ORDER_TYPE_BUY_LIMIT  : case ORDER_TYPE_BUY_STOP  :
```

© 2000-2025, MetaQuotes Ltd.

```
         return(ORDER_TYPE_BUY);
       case ORDER_TYPE_SELL : case ORDER_TYPE_SELL_LIMIT : case ORDER_TYPE_SELL_STOP :
         return(ORDER_TYPE_SELL);
      }
    return(WRONG_VALUE);
   }
//+------------------------------------------------------------------+
//| Rückgabe des Eröffnungspreises nach Auftragsart                  |
//+------------------------------------------------------------------+
double PriceByOrderType(const string symbol, const ENUM_ORDER_TYPE order_type)
   {
    int     digits=0;
    double  point=0;
    MqlTick tick={};

//--- Abrufen vom Punktwert des Symbols
    ResetLastError();
    if(!SymbolInfoDouble(symbol, SYMBOL_POINT, point))
      {
       Print("SymbolInfoDouble() failed. Error ", GetLastError());
       return 0;
      }

//--- Abrufen der Anzahl der Dezimalen des Symbols
    long value=0;
    if(!SymbolInfoInteger(symbol, SYMBOL_DIGITS, value))
      {
       Print("SymbolInfoInteger() failed. Error ", GetLastError());
       return 0;
      }
    digits=(int)value;

//--- Aktuelle Preise eines Symbols abrufen
    if(!SymbolInfoTick(symbol, tick))
      {
       Print("SymbolInfoTick() failed. Error ", GetLastError());
       return 0;
      }

//--- Preis abhängig von der Auftragsart zurückgeben
    switch(order_type)
      {
       case ORDER_TYPE_BUY              :  return(tick.ask);
       case ORDER_TYPE_SELL             :  return(tick.bid);
       case ORDER_TYPE_BUY_LIMIT        :  return(NormalizeDouble(tick.ask - DEVIATION
       case ORDER_TYPE_SELL_LIMIT       :  return(NormalizeDouble(tick.bid + DEVIATION
       case ORDER_TYPE_BUY_STOP         :  return(NormalizeDouble(tick.ask + DEVIATION
       case ORDER_TYPE_SELL_STOP        :  return(NormalizeDouble(tick.bid - DEVIATION
       case ORDER_TYPE_BUY_STOP_LIMIT   :  return(NormalizeDouble(tick.ask + DEVIATION
```

© 2000-2025, MetaQuotes Ltd.

```
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return(NormalizeDouble(tick.bid - DEVIATION
      default                          :  return(0);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE order_type)
  {
   switch(order_type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      case ORDER_TYPE_CLOSE_BY         :  return("Close By");
      default                          :  return("Unknown order type");
     }
  }
//+------------------------------------------------------------------+
```

Siehe auch

OrderSend(), Ordereigenschaften, Typen der Handelsoperationen

© 2000-2025, MetaQuotes Ltd.

OrderCalcProfit

Berechnet die Größe des Gewinns für das laufende Konto und Marktumgebung aufgrund der übertragenen Parameter. Bestimmt für vorläufige Einschätzung der Handelsoperation. Der Wert wird in der Währung des Kontos zurückgegeben.

```
bool  OrderCalcProfit(
   ENUM_ORDER_TYPE       action,           // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYPE
   string                symbol,           // Symbolname
   double                volume,           // Volumen
   double                price_open,       // Eröffnungspreis
   double                price_close,      // Abschlusspreis
   double&               profit           // Variable für den Erhalt des Wertes des Ge
   );
```

Parameter

action

[in] Ordertyp, kann einen der zwei Werte der Enumeration ENUM_ORDER_TYPE annehemen: ORDER_TYPE_BUY oder ORDER_TYPE_SELL.

symbol

[in] Name des Finanzinstrumentes .

volume

[in] Volumen der Handelsoperation.

price_open

[in] Eröffnungspreis.

price_close

[in] Abschlusspreis.

profit

[out] Variable in die der berechnete Wert des Gewinns bei der erfolgreichen Ausführung der Funktion geschrieben wird. Wert der Gewinneinschätzung hängt von vielen Faktoren ab und kann sich bei der Veränderung der Marktumgebung verändern.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false. Wenn es einen unzulässigen Ordertyp angegeben wird, gibt die Funktion false zurück. Für den Erhalt der Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

```
#define   VOLUME     1.0   // Auftragsvolumen
#define   DEVIATION  100   // Abstand in Punkten für das Schließen

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   string currency_profit=SymbolInfoString(_Symbol,SYMBOL_CURRENCY_PROFIT);   // Gewi
   double close_price=ChartPriceOnDropped(); // als Schlusskurs dient die Preiskoordi

//---
   for(int i=0; i<=ORDER_TYPE_SELL; i++)
     {
      ENUM_ORDER_TYPE order_type=(ENUM_ORDER_TYPE)i;                  // Order-Typ: 0 -
      double open_price=PriceOpenByOrderType(_Symbol, order_type);    // Eröffnungspre

      //--- wenn der Eröffnungskurs nicht empfangen wird, Schleife fortsetzen
      if(open_price==0)
         continue;

      //--- wenn der Schlusskurs null ist (das Skript wurde nicht durch Ziehen auf das
      if(close_price==0)
         close_price=(order_type==ORDER_TYPE_BUY ? open_price + DEVIATION * _Point : c

      //--- Berechnen der ungefähren Gewinnhöhe für das aktuelle Konto und das Marktu
      double profit=0;
      ResetLastError();
      if(!OrderCalcProfit(order_type, _Symbol, VOLUME, open_price, close_price, profit
        {
         Print("OrderCalcProfit() failed. Error ", GetLastError());
         continue;
        }

      //--- den erhaltenen Gewinn im Journal ausdrucken
      PrintFormat("Estimated profit for %.2f %s position on %s with opening price of %
                  VOLUME, OrderTypeDescription(order_type), _Symbol, _Digits, open_pri

     }
   /*
   Ergebnis:
   Estimated profit for 1.00 Buy position on EURUSD with opening price of 1.10757 and
   Estimated profit for 1.00 Sell position on EURUSD with opening price of 1.10753 and
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe des Eröffnungspreises nach Auftragsart                  |
//+------------------------------------------------------------------+
double PriceOpenByOrderType(const string symbol, const ENUM_ORDER_TYPE order_type)
  {
   MqlTick tick={};

//--- Aktuelle Preise eines Symbols abrufen
   if(!SymbolInfoTick(symbol, tick))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("SymbolInfoTick() failed. Error ", GetLastError());
      return 0;
     }

//--- Preis abhängig von der Auftragsart zurückgeben
   switch(order_type)
     {
      case ORDER_TYPE_BUY              :  return(tick.ask);
      case ORDER_TYPE_SELL             :  return(tick.bid);
      default                          :  return(0);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE order_type)
  {
   switch(order_type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      default                          :  return("Unknown order type");
     }
  }
```

Siehe auch

OrderSend(), Ordereigenschaften, Typen der Handelsoperationen

© 2000-2025, MetaQuotes Ltd.

OrderCheck

OrderCheck() prüft, ob genug Geld vorhanden ist, um die notwendigen Handelsoperationen auszuführen. Die Ergebnisse der Prüfung werden den Feldern der Struktur MqlTradeCheckResult zugewiesen.

```
bool  OrderCheck(
   MqlTradeRequest&       request,      // Struktur der Anforderung
   MqlTradeCheckResult&   result        // Struktur der Antwort
   );
```

Parameter

request

[in] Zeiger auf die Struktur vom Typ MqlTradeRequest, die die erforderliche Handelsoperation beschreibt.

Ergebnis

[in,out] Zeiger auf die Struktur vom Typ MqlTradeCheckResult, der das Ergebnis der Prüfung zugewiesen wird.

Rückgabewert

Wenn es an Geldmittel fehlt oder Parameter nicht korrekt ausgefüllt sind, gibt die Funktion false zurück. Bei der erfolgreichen Grundprüfung der Strukturen (Prüfung der Zeiger) kehrt sie mit true zurück - **das zeugt noch nicht davon, dass die angeforderte Handelsoperation unbedingt erfolgreich ausgeführt wird**. Für den Erhalt der ausführlichen Beschreibung der Ausführung der Funktion muss man die Felder der Struktur *result* analysieren.

Für den Erhalt der Information über den Fehler, muss man die Funktion GetLastError() aufrufen.

Beispiel:

```
#define   DEVIATION      5              // erlaubte Abweichung vom Preis
#define   VOLUME         1.0            // Auftragsvolumen
#define   EXPERT_MAGIC   123            // MagicNumber
#define   DIRECTION      ORDER_TYPE_BUY // Richtung der zu öffnenden Position (ORDER_TY

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anforderungs-, Verifizierungs- und Ergebnisstrukturen deklarieren und initialis
   MqlTradeRequest      request={};
   MqlTradeCheckResult check  ={};
   MqlTradeResult       result ={};

//--- Handelsanforderungsparameter vorbereiten
   PrepareRequest(_Symbol, DIRECTION, VOLUME, request);

//--- Handelsanforderungsparameter prüfen
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
   bool res=OrderCheck(request, check);
   if(!res)
     {
      PrintFormat("Trade request verification completed with error %d\nServer retcode:
      return;
     }

//--- die Prüfung der Handelsanfrage war erfolgreich - Beschreibung der Felder der Har
   Print("Trade request verification completed successfully");
   MqlTradeCheckResultPrint(check, 14);

//--- Handelsanfrage senden
   if(!OrderSend(request, result))
      Print("OrderSend error ", GetLastError());    // Wenn die Anfrage nicht gesendet

//--- Information über die Operation
   PrintFormat("Trade request result: retcode=%u, deal=%I64u, order=%I64u", result.ret
   /*
   Ergebnis mit deaktiviertem Auto-Trading im Client-Terminal:
   Trade request verification completed with error 4752
   Server retcode: 10027, comment: AutoTrading disabled by client

   Automatischen Handel aktivieren und bei geschlossenem Markt erneut prüfen:
   Experts   automated trading is enabled
   Trade request verification completed successfully
   Retcode:      0
   Balance:      10779.50 USD
   Equity:       10779.50 USD
   Profit:       0.00 USD
   Margin:       1104.79 USD
   Margin free:  9674.71 USD
   Margin level: 975.71 %
   Comment:      Done
   OrderSend error 4756
   Trade request result: retcode=10018, deal=0, order=0

   Prüfen, ob der Markt offen ist:
   Trade request verification completed successfully
   Retcode:      0
   Balance:      10779.50 USD
   Equity:       10779.50 USD
   Profit:       0.00 USD
   Margin:       110.46 USD
   Margin free:  10669.04 USD
   Margin level: 9758.74 %
   Comment:      Done
   Trade request result: retcode=10009, deal=2777010968, order=2802818813
   */
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Parameter für eine Handelsanfrage vorbereiten                    |
//+------------------------------------------------------------------+
void PrepareRequest(const string symbol, const ENUM_ORDER_TYPE order_type, const doubl
  {
   ENUM_ORDER_TYPE type=(DIRECTION !=ORDER_TYPE_BUY ? ORDER_TYPE_SELL : DIRECTION);
   double price=(DIRECTION==ORDER_TYPE_BUY ? SymbolInfoDouble(Symbol(), SYMBOL_ASK) :
//--- Abfrage der Parameter
   request.action    = TRADE_ACTION_DEAL; // Typ der Handelsoperation
   request.symbol    = symbol;            // Symbol
   request.volume    = volume;            // Volumen
   request.type      = type;              // Auftragstyp
   request.price     = price;             // Eröffnungspreis
   request.deviation = DEVIATION;         // erlaubte Abweichung vom Preis
   request.magic     = EXPERT_MAGIC;      // MagicNumber des Auftrags
  }
//+------------------------------------------------------------------+
//| Felder der Handelsanfrage ausdrucken                             |
//| Prüfergebnis im Journal                                          |
//+------------------------------------------------------------------+
void MqlTradeCheckResultPrint(const MqlTradeCheckResult &check, const uint header_widt
  {
//--- Kontowährung und Anzahl der Dezimalstellen der Kontowährung abrufen
   string currency=AccountInfoString(ACCOUNT_CURRENCY);
   int    digits  =(int)AccountInfoInteger(ACCOUNT_CURRENCY_DIGITS);

//--- Definieren der Kopftextes und der Breite des Textfeldes
//--- Wenn der Funktion eine Breite von Null übergeben wird, dann ist die Breite die (
   string header="Retcode:";
   uint w=(header_width==0 ? header.Length()+1 : header_width);
//--- Anzeige des Rückgabecodes mit dem Header der angegebenen Breite im Journal
   PrintFormat("%-*s%-u", w, header, check.retcode);

//--- Anzeige des Saldos nach der Ausführung einer Handelsoperation im Journal
   header="Balance:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-.*f %s", w, header, digits, check.balance, currency);

//--- Anzeige des Eigenkapitals nach Ausführung einer Handelsoperation im Journal
   header="Equity:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-.*f %s", w, header, digits, check.equity, currency);

//--- Anzeige des schwebenden Gewinns nach der Ausführung einer Handelsoperation im Jo
   header="Profit:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-.*f %s", w, header, digits, check.profit, currency);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Anzeige der für die erforderliche Handelsoperation erforderlichen Marge im Journ
   header="Margin:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-.*f %s", w, header, digits, check.margin, currency);

//--- Anzeige des nach der Durchführung einer Handelsoperation verbleibenden Eigenkapi
   header="Margin free:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-.*f %s", w, header, digits, check.margin_free, currency);

//--- Anzeige des Margin-Levels, der nach Abschluss der erforderlichen Handelsoperatio
   header="Margin level:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-.2f %%", w, header, check.margin_level);

//--- Kommentar zum Antwortcode und Fehlerbeschreibung im Journal anzeigen
   header="Comment:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   PrintFormat("%-*s%-s", w, header, check.comment);
  }
```

Siehe auch

OrderSend(), Typen der Handelsoperationen, Struktur der Handelsanforderung, Struktur der Ergebnisse der Prüfung der Handelsanforderung, Struktur des Ergebnisses der Handelsanforderung.

© 2000-2025, MetaQuotes Ltd.

OrderSend

Die Funktion OrderSend() wird für die Durchführung von Handelsoperationen durch das Senden von Anfragen an einen Handelsserver verwendet.

```
bool  OrderSend(
   MqlTradeRequest&  request      // Struktur einer Anfrage
   MqlTradeResult&   result       // Struktur einer Antwort
   );
```

Parameter

request

[in] Pointer auf eine Struktur vom Typ MqlTradeRequest, die die Aktion des Kunden beschreibt.

result

[in,out] Pointer auf eine Struktur vom Typ MqlTradeResult, die das Ergebnis einer Handelsoperation beschreibt, wenn diese erfolgreich durchgeführt wurde (wenn true zurückgegeben wurde).

Rückgabewert

Wenn die Basisprüfung der Strukturen erfolgreich war (Prüfung der Indexe), wird true zurückgegeben - **das heißt allerdings nicht, dass eine Handelsoperation erfolgreich durchgeführt wurde**. Um eine ausführliche Beschreibung des Ergebnisses der Funktion zu bekommen, sollte man die Felder der Struktur *result* analysieren.

Hinweis

Eine Handelsanfrage durchläuft mehrere Stufen der Prüfungen auf dem Handelsserver. Zunächst wird überprüft, ob alle benötigten Felder des Parameters *request* korrekt ausgefüllt werden. Wenn es keine Fehler gibt, akzeptiert der Server die Order für die weitere Verarbeitung. Wird die Order erfolgreich vom Handelsserver akzeptiert, gibt die Funktion OrderSend() den Wert true zurück.

Es ist empfehlenswert, die Anfrage vor dem Senden an den Handelsserver selbständig zu überprüfen. Um die Anfrage zu überprüfen, verwenden Sie die Funktion OrderCheck(). Sie prüft, ob es genügend Geld für die Ausführung der Transaktion gibt und gibt viele nützliche Parameter in den Ergebnissen der Prüfung der Handelsanfrage zurück:

- Rückgabecode, der über Fehler in der zu prüfenden Anfrage informiert;
- Kontostand nach der Ausführung der Handelsoperation;
- Equity nach der Ausführung der Handelsoperation;
- Floating Profit nach der Ausführung der Handelsoperation;
- Margin, die für die Handelsoperation erforderlich ist;
- Freie Eigenmittel, die nach Ausführung der Handelsoperation bleiben;
- Level der Margin, der nach der Ausführung der Handelsoperation gesetzt wird;
- Kommentar zum Antwort-Code, Beschreibung des Fehlers.

Beim Senden einer Marktorder (MqlTradeRequest.action=TRADE_ACTION_DEAL) bedeutet das erfolgreiche Ergebnis der Funktion OrderSend() nicht, dass die Order ausgeführt wurde (die entsprechenden Trades ausgeführt wurden). In diesem Fall bedeutet true nur, dass die Order im Handelssystem erfolgreich platziert wurde. Der Handelsserver kann die Felder *deal* oder *order* in der Struktur des Ergebnisses *result* ausfüllen, wenn diese Werte im Moment der Erzeugung einer

© 2000-2025, MetaQuotes Ltd.

Antwort auf einen Aufruf von OrderSend() bekannt sein werden. Im allgemeinen Fall kann das Ereignis oder die Ereignisse der Ausführung von Trades, die einer Order entsprechen, bereits nach dem Senden einer Antwort auf den Aufruf von OrderSend() eintreten. Deswegen muss man beim Erhalten des Ergebnisses der Ausführung von OrderSend() in erster Linie den Rückgabecode des Handelsservers *retcode* und den Antwort-Code des externen Handelssystems *retcode_external* (wenn nötig) überprüfen, die in der Struktur *result* verfügbar sind.

Jede akzeptierte Order wird während des Wartens auf die Verarbeitung auf dem Handelsserver gespeichert, bis eine der Bedingungen für ihre Ausführung erfüllt wird:

- Ablaufzeit,
- Erscheinen einer entgegengesetzten Anfrage,
- Auslösung der Order beim Erscheinen des Ausführungspreises,
- Eintreffen einer Anfrage, die Order zu löschen.

Zum Zeitpunkt der Orderbearbeitung sendet der Handelsserver eine Meldung an das Terminal über das Auftreten des Handelsereignisses Trade, das mit der Funktion OnTrade() verarbeitet werden kann.

Das Ergebnis der Ausführung der Handelsanfrage, die mit der Funktion OrderSend() gesendet wurde, kann mithilfe des Handlers OnTradeTransaction auf dem Server verfolgt werden. Dabei ist es zu beachten, dass der Handler OnTradeTransaction infolge der Ausführung einer Handelsanfrage mehrmals aufgerufen wird.

Zum Beispiel, beim Senden einer Market Buy Order wird sie verarbeitet, und eine entsprechende Kauforder wird für das Konto erstellt. Die Order wird ausgeführt, von der Liste der offenen entfernt und der Orderhistorie hinzugefügt. Dann wird der entsprechende Abschluss der Historie hinzugefügt und eine neue Position wird eröffnet. Für jedes dieser Ereignisse wird die Funktion OnTradeTransaction aufgerufen.

Beispiel:

```
//--- Werte für ORDER_MAGIC
input long order_magic=55555;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- stell sicher, dass es ein Demokonto ist
   if(AccountInfoInteger(ACCOUNT_TRADE_MODE)==ACCOUNT_TRADE_MODE_REAL)
     {
      Alert("Ausführung des Scripts auf einem Realkonto verboten!");
      return;
     }
//--- platziere oder lösche eine Order
   if(GetOrdersTotalByMagic(order_magic)==0)
     {
      //--- wenn es keine aktuellen Orders gibt, platziere eine Order
      uint res=SendRandomPendingOrder(order_magic);
      Print("Rückgabewert des Handelsservers ",res);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   else // wenn es die Orders gibt, lösche diese
     {
      DeleteAllOrdersByMagic(order_magic);
     }
//---
 }
//+------------------------------------------------------------------+
//|Erhält die aktuelle Anzahl der Orders mit der angegebenen ORDER_MAGIC
//+------------------------------------------------------------------+
int GetOrdersTotalByMagic(long const magic_number)
  {
   ulong order_ticket;
   int total=0;
//--- durchlaufe alle Pending Orders
   for(int i=0;i<OrdersTotal();i++)
      if((order_ticket=OrderGetTicket(i))>0)
         if(magic_number==OrderGetInteger(ORDER_MAGIC)) total++;
//---
   return(total);
  }
//+------------------------------------------------------------------+
//| Entfernt alle Pending Orders mit der angegebenen ORDER_MAGIC     |
//+------------------------------------------------------------------+
void DeleteAllOrdersByMagic(long const magic_number)
  {
   ulong order_ticket;
//--- durchlaufe alle Pending Orders
   for(int i=0;i<OrdersTotal();i++)
      if((order_ticket=OrderGetTicket(i))>0)
         //--- Order mit der entsprechenden ORDER_MAGIC
         if(magic_number==OrderGetInteger(ORDER_MAGIC))
           {
            MqlTradeResult result={};
            MqlTradeRequest request={};
            request.order=order_ticket;
            request.action=TRADE_ACTION_REMOVE;
            OrderSend(request,result);
            //--- Antwort des Servers im Journal ausgeben
            Print(__FUNCTION__,":",result.comment,"Antwortcode",result.retcode);
           }
//---
  }
//+------------------------------------------------------------------+
//| Platziere eine Pending Order auf zufällige Weise                 |
//+------------------------------------------------------------------+
uint SendRandomPendingOrder(long const magic_number)
 {
//--- bereite eine Anfrage vor
   MqlTradeRequest request={};
```

© 2000-2025, MetaQuotes Ltd.

```
   request.action=TRADE_ACTION_PENDING;         // Platzieren einer Pending Order
   request.magic=magic_number;                  // ORDER_MAGIC
   request.symbol=_Symbol;                      // Instrument
   request.volume=0.1;                          // Volumen von 0.1 Lot
   request.sl=0;                                // Stop Loss nicht angegeben
   request.tp=0;                                // Take Profit nicht angegeben
//--- bilde den Ordertyp
   request.type=GetRandomType();                // Ordertyp
//--- bilde den Preis für die Pending Order
   request.price=GetRandomPrice(request.type);  // Eroeffnungspreis
//--- sende eine Handelsanfrage
   MqlTradeResult result={};
   OrderSend(request,result);
//--- die Antwort des Servers im Journal ausgeben
   Print(__FUNCTION__,":",result.comment);
   if(result.retcode==10016) Print(result.bid,result.ask,result.price);
//--- den Antwort-Code des Handelsservers zurückgeben
   return result.retcode;
  }
//+------------------------------------------------------------------+
//| Erhält den Typ einer Pending Order auf zufällige Weise           |
//+------------------------------------------------------------------+
ENUM_ORDER_TYPE GetRandomType()
 {
   int t=MathRand()%4;
//---   0<=t<4
   switch(t)
    {
      case(0):return(ORDER_TYPE_BUY_LIMIT);
      case(1):return(ORDER_TYPE_SELL_LIMIT);
      case(2):return(ORDER_TYPE_BUY_STOP);
      case(3):return(ORDER_TYPE_SELL_STOP);
     }
//--- ungültiger Wert
   return(WRONG_VALUE);
  }
//+------------------------------------------------------------------+
//| Erhält den Preis auf zufällige Weise                             |
//+------------------------------------------------------------------+
double GetRandomPrice(ENUM_ORDER_TYPE type)
 {
   int t=(int)type;
//--- Stoplevel für Symbol
   int distance=(int)SymbolInfoInteger(_Symbol,SYMBOL_TRADE_STOPS_LEVEL);
//--- erhalte Daten des letzten Ticks
   MqlTick last_tick={};
   SymbolInfoTick(_Symbol,last_tick);
//--- berechne den Preis entsprechend dem Typ
   double price;
```

© 2000-2025, MetaQuotes Ltd.

```
   if(t==2 || t==5) // ORDER_TYPE_BUY_LIMIT oder ORDER_TYPE_SELL_STOP
    {
      price=last_tick.bid; // gehen wir vom Preis Bid aus
      price=price-(distance+(MathRand()%10)*5)*_Point;
     }
   else             // ORDER_TYPE_SELL_LIMIT oder ORDER_TYPE_BUY_STOP
    {
      price=last_tick.ask; // gehen wir vom Preis Ask aus
      price=price+(distance+(MathRand()%10)*5)*_Point;
     }
//---
   return(price);
  }
```

Sehen Sie auch

Typen der Handelsoperationen, Struktur der Handelsanfrage, Struktur des Ergebnisses einer Handelsanfrage

© 2000-2025, MetaQuotes Ltd.

OrderSendAsync

Die OrderSendAsync()-Funktion wird für Durchführung von asynchronen Handelsoperationen ohne Warten auf eine Antwort auf eine gesendete Anfrage aus dem Server. Die Funktion wird für Hochfrequenz-Handel ausgelegt, wenn es unter den Bedingungen der Handelsalgorithmus inakzeptabel ist, Zeit für Warten auf eine Antwort aus dem Server zu verschwenden.

```
bool  OrderSendAsync(
   MqlTradeRequest&  request,      // Struktur der Anfrage
   MqlTradeResult&   result        // Struktur der Antwort
   );
```

Parameter

request

[in] Zeiger auf eine Struktur vom Typ MqlTradeRequest, die die Handelsaktion des Kunden beschreibt.

result

[in,out] Zeiger auf eine Struktur vom Typ MqlTradeResult, die das Ergebnis einer bei erfolgreicher Ausführung der Funktion (bei der Rückgabe true) beschreibt.

Rückgabewert

Gibt true zurück, wenn die Anfrage an den Handel-Server gesendet ist. Wenn die Anfrage nicht gesendet wurde, wird false zurückgegeben. Bei Erfolg erhält der Antwortcode in der Variable *result* den wert TRADE_RETCODE_PLACED (Code 10008) - "eine Order wurde platziert". Die erfolgreiche Ausführung bedeutet nur die Tatsache des Versands, aber es gibt keine Garantie, dass die Anfrage auf dem Handelsserver kam und wurde zur Bearbeitung akzeptiert. Bei der Verarbeitung der empfangenen Anfrage sendet der Handelsserver eine Antwortnachricht an den Client-Terminal über die Änderung des aktuellen Status der Positionen, Ordern und Deals, die zur Erzeugung von Ereignis Trade führt.

Das Ergebnis der Ausführung des Handelsanfrage, das mit der Funktion OrderSendAsync() gesendet wurde, kann mit Hilfe des Handlers OnTradeTransaction auf dem Server verfolgt werden. Dabei ist es zu beachten, dass als Ergebnis der Ausführung eines Handels-Anfrage der Handler OnTradeTransaction mehrmals aufgerufen wird.

Zum Beispiel, beim Senden Marktkauforder, wird Marktkauforder behandelt und eine entsprechende Marktkauforder für das Konto erstellt. Ist die Order ausgeführt, wird sie aus der Auftragsliste entfernt und zur Ordergeschichte hinzugefügt. Dann wird der entsprechende Deal zur Geschichte hinzugefügt und eine neue Position wird erstellt. Für jedes dieser Ereignisse wird die Funktion OnTradeTransaction aufgerufen. Für die ausführlichen Informationen muss man die Parameter dieser Funktion analysieren:

- **trans** - in diesen Parameter wird die Struktur MqlTradeTransaction übergeben, die die auf das Handelskonto angewendete Handelstransaktion beschreibt;
- **request** - in diesen Parameter wird die Struktur MqlTradeRequest, übergeben, die eine Handelsanfrage beschreibt, als Ergebnis deren Handeltransaktion ausgeführt wurde
- **result** - in diesen Parameter wird die Struktur MqlTradeResult übergeben, die das Ergebnis der Ausführung einer Handelsanfrage beschreibt.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Die Zweckbestimmung und die Parameter der Funktion sind der Funktion OrderSend() ähnlich, aber anders als sie, es ist asynchron, das heißt, das sie die Programmarbeit während des Wartens auf die Funktion-Ergebnis nicht aussetzt. Vergleichen Sie die die Geschwindigkeit des Handels Operationen in dieser beiden Funktionen anhand des Beispiels in gegebenen EA.

Beispiel:

```
#property description "Ein Expert Advisor für Senden von Handelsanfragen"
                      " durch die Funktion OrderSendAsync().\r\n"
#property description "Verarbeitung der Handelsereignisse mit"
                      " Handler-Funktionen OnTrade() und OnTradeTransaction() ist ange
#property description "In Expert Advisor Parameters können Sie Magic Number"
                      " (eindeutiger Identifikator) "
#property description "und Nachrichtenausgabemodus in Experten Zeitschrift angeben. St
//--- input parameters
input int  MagicNumber=1234567;      // Expert Advisor ID
input bool DescriptionModeFull=true; // Mode der detaillierten Ausgabe
//--- eine Variable für Nutzung im Aufruf von HistorySelect()
datetime history_start;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Erlaubnis für den automatisierten Handel überprüfen
   if(!TerminalInfoInteger(TERMINAL_TRADE_ALLOWED))
     {
      Alert("Automatisches Handle ist im Terminal verboten, EA wird gelöscht werden");
      ExpertRemove();
      return(-1);
     }
//--- Handle auf einem Live-Konto ist nicht erlaubt
   if(AccountInfoInteger(ACCOUNT_TRADE_MODE)==ACCOUNT_TRADE_MODE_REAL)
     {
      Alert("Dem Expert Advisor ist nicht erlaubt auf einem Live-Konto zu handeln!");
      ExpertRemove();
      return(-2);
     }
//--- ob es möglich ist, auf dem Konto zu handeln (zB ist es nicht erlaubt mit einem
   if(!AccountInfoInteger(ACCOUNT_TRADE_ALLOWED))
     {
      Alert("Der Handel auf dem Konto ist nicht erlaubt");
      ExpertRemove();
      return(-3);
     }
//--- Speichern die Laufzeit des Experts für den Handelsgeschichte
   history_start=TimeCurrent();
//---
   CreateBuySellButtons();
```

© 2000-2025, MetaQuotes Ltd.

```
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- graphische Objekte löschen
   ObjectDelete(0,"Buy");
   ObjectDelete(0,"Sell");
//---
  }
//+------------------------------------------------------------------+
//| TradeTransaction function                                        |
//+------------------------------------------------------------------+
void OnTradeTransaction(const MqlTradeTransaction &trans,
                        const MqlTradeRequest &request,
                        const MqlTradeResult &result)
  {
//--- Titel nach dem Namen der Handelsereignisse-Handler-Funktion
   Print("=> ",__FUNCTION__," at ",TimeToString(TimeCurrent(),TIME_SECONDS));
//--- Typ der Transaktion in Form von Enumerationswerte erhalten
   ENUM_TRADE_TRANSACTION_TYPE type=trans.type;
//--- wenn die Transaktion ist ein Ergebnis der Verarbeitung einer Anfrage
   if(type==TRADE_TRANSACTION_REQUEST)
     {
      //---den Name der Transaktion ausgeben
      Print(EnumToString(type));
      //--- dann geben die String-Beschreibung der verarbeiteten Anfrage
      Print("------------RequestDescription\r\n",
            RequestDescription(request,DescriptionModeFull));
      //--- und anzeigen die Beschreibung des Abfrageergebnisses
      Print("------------ ResultDescription\r\n",
            TradeResultDescription(result,DescriptionModeFull));
     }
   else // Für die Transaktionen anderen Typen geben wir eine vollständige Beschreibur
     {
      Print("------------ TransactionDescription\r\n",
            TransactionDescription(trans,DescriptionModeFull));
     }
//---
  }
//+------------------------------------------------------------------+
//| Trade function                                                   |
//+------------------------------------------------------------------+
void OnTrade()
  {
//--- Statische Mitglieder, um den Kontostand zu speichern
   static int prev_positions=0,prev_orders=0,prev_deals=0,prev_history_orders=0;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Handelsgeschichte anfragen
   bool update=HistorySelect(history_start,TimeCurrent());
   PrintFormat("HistorySelect(%s , %s) = %s",
               TimeToString(history_start),TimeToString(TimeCurrent()),(string)update)
//--- Titel nach dem Namen der Handelsereignisse-Handler-Funktion
   Print("=> ",__FUNCTION__," at ",TimeToString(TimeCurrent(),TIME_SECONDS));
//--- Name des Handlers und die Anzahl der Order in der Zeit der Verarbeitung ausgeben
   int curr_positions=PositionsTotal();
   int curr_orders=OrdersTotal();
   int curr_deals=HistoryOrdersTotal();
   int curr_history_orders=HistoryDealsTotal();
//--- Anzahl der Aufträge, Positionen, Transaktionen und Änderungen in Klammern anzeig
   PrintFormat("PositionsTotal() = %d (%+d)",
               curr_positions,(curr_positions-prev_positions));
   PrintFormat("OrdersTotal() = %d (%+d)",
               curr_orders,curr_orders-prev_orders);
   PrintFormat("HistoryOrdersTotal() = %d (%+d)",
               curr_deals,curr_deals-prev_deals);
   PrintFormat("HistoryDealsTotal() = %d (%+d)",
               curr_history_orders,curr_history_orders-prev_history_orders);
//--- Zeilenumbruch einfügen zum einfachen Ablesen des Journal
   Print("");
//--- den Zustand des Kontos speichern
   prev_positions=curr_positions;
   prev_orders=curr_orders;
   prev_deals=curr_deals;
   prev_history_orders=curr_history_orders;
//---
  }
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
//--- Verarbeitung von Ereignissen CHARTEVENT_CLICK ("Klick mit der Maus auf dem Chart
   if(id==CHARTEVENT_OBJECT_CLICK)
     {
      Print("=> ",__FUNCTION__,": sparam = ",sparam);
      //--- Mindesvolumen für Transaktion
      double volume_min=SymbolInfoDouble(_Symbol,SYMBOL_VOLUME_MIN);
      //--- wenn die Schaltfläche "Buy" ist gedrückt, dann kaufen wir
      if(sparam=="Buy")
        {
         PrintFormat("Buy %s %G lot",_Symbol,volume_min);
         BuyAsync(volume_min);
         //--- Undrücken die gedrückte Schaltfläche
```

© 2000-2025, MetaQuotes Ltd.

```
         ObjectSetInteger(0,"Buy",OBJPROP_STATE,false);
        }
      //--- wenn die Schaltfläche "Sell" ist gedrückt, dann verkaufen wir
      if(sparam=="Sell")
        {
         PrintFormat("Sell %s %G lot",_Symbol,volume_min);
         SellAsync(volume_min);
         //--- Undrücken die gedrückte Schaltfläche
         ObjectSetInteger(0,"Sell",OBJPROP_STATE,false);
        }
      ChartRedraw();
     }
//---
  }
//+------------------------------------------------------------------+
//| Gibt eine textuelle Beschreibung der Transaktion zurück          |
//+------------------------------------------------------------------+
string TransactionDescription(const MqlTradeTransaction &trans,
                              const bool detailed=true)
  {
//--- eine Zeichenfolge für Zurückgabe aus der Funktion vorbereiten
   string desc=EnumToString(trans.type)+"\r\n";
//--- im detaillierten Modus fügen wir maximale Information hinzu
   if(detailed)
     {
      desc+="Symbol: "+trans.symbol+"\r\n";
      desc+="Deal ticket: "+(string)trans.deal+"\r\n";
      desc+="Deal type: "+EnumToString(trans.deal_type)+"\r\n";
      desc+="Order ticket: "+(string)trans.order+"\r\n";
      desc+="Order type: "+EnumToString(trans.order_type)+"\r\n";
      desc+="Order state: "+EnumToString(trans.order_state)+"\r\n";
      desc+="Order time type: "+EnumToString(trans.time_type)+"\r\n";
      desc+="Order expiration: "+TimeToString(trans.time_expiration)+"\r\n";
      desc+="Price: "+StringFormat("%G",trans.price)+"\r\n";
      desc+="Price trigger: "+StringFormat("%G",trans.price_trigger)+"\r\n";
      desc+="Stop Loss: "+StringFormat("%G",trans.price_sl)+"\r\n";
      desc+="Take Profit: "+StringFormat("%G",trans.price_tp)+"\r\n";
      desc+="Volume: "+StringFormat("%G",trans.volume)+"\r\n";
     }
//--- Geben wir den den resultierenden String zurück
   return desc;
  }
//+------------------------------------------------------------------+
//| Gibt eine textuelle Beschreibung der Handelsanfrage zurück       |
//+------------------------------------------------------------------+
string RequestDescription(const MqlTradeRequest &request,
                          const bool detailed=true)
  {
//--- eine Zeichenfolge für Zurückgabe aus der Funktion vorbereiten
```

© 2000-2025, MetaQuotes Ltd.

```
   string desc=EnumToString(request.action)+"\r\n";
//--- im detaillierten Modus fügen wir maximale Information hinzu
   if(detailed)
     {
      desc+="Symbol: "+request.symbol+"\r\n";
      desc+="Magic Number: "+StringFormat("%d",request.magic)+"\r\n";
      desc+="Order ticket: "+(string)request.order+"\r\n";
      desc+="Order type: "+EnumToString(request.type)+"\r\n";
      desc+="Order filling: "+EnumToString(request.type_filling)+"\r\n";
      desc+="Order time type: "+EnumToString(request.type_time)+"\r\n";
      desc+="Order expiration: "+TimeToString(request.expiration)+"\r\n";
      desc+="Price: "+StringFormat("%G",request.price)+"\r\n";
      desc+="Deviation points: "+StringFormat("%G",request.deviation)+"\r\n";
      desc+="Stop Loss: "+StringFormat("%G",request.sl)+"\r\n";
      desc+="Take Profit: "+StringFormat("%G",request.tp)+"\r\n";
      desc+="Stop Limit: "+StringFormat("%G",request.stoplimit)+"\r\n";
      desc+="Volume: "+StringFormat("%G",request.volume)+"\r\n";
      desc+="Comment: "+request.comment+"\r\n";
     }
//--- Geben wir den den resultierenden String zurück
   return desc;
  }
//+------------------------------------------------------------------+
//| Gibt textuelle Beschreibung der Ergebnis der Anfrageverarbeitung |
//+------------------------------------------------------------------+
string TradeResultDescription(const MqlTradeResult &result,
                              const bool detailed=true)
  {
//--- eine Zeichenfolge für Zurückgabe aus der Funktion vorbereiten
   string desc="Retcode "+(string)result.retcode+"\r\n";
//--- im detaillierten Modus fügen wir maximale Information hinzu
   if(detailed)
     {
      desc+="Request ID: "+StringFormat("%d",result.request_id)+"\r\n";
      desc+="Order ticket: "+(string)result.order+"\r\n";
      desc+="Deal ticket: "+(string)result.deal+"\r\n";
      desc+="Volume: "+StringFormat("%G",result.volume)+"\r\n";
      desc+="Price: "+StringFormat("%G",result.price)+"\r\n";
      desc+="Ask: "+StringFormat("%G",result.ask)+"\r\n";
      desc+="Bid: "+StringFormat("%G",result.bid)+"\r\n";
      desc+="Comment: "+result.comment+"\r\n";
     }
//--- Geben wir den den resultierenden String zurück
   return desc;
  }
//+------------------------------------------------------------------+
//| Erstellt zwei Schaltflächen für Kauf und Verkauf                 |
//+------------------------------------------------------------------+
void CreateBuySellButtons()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- überprüfen wir ob ein namens "Buy" existiert
   if(ObjectFind(0,"Buy")>=0)
     {
      //--- wenn das gefundene Objekt keine Schaltfläche ist, löschen wir es
      if(ObjectGetInteger(0,"Buy",OBJPROP_TYPE)!=OBJ_BUTTON)
         ObjectDelete(0,"Buy");
     }
   else
      ObjectCreate(0,"Buy",OBJ_BUTTON,0,0,0); // Schaltfläche "Buy" erstellen
//--- Schaltfläche "Buy" abstimmen
   ObjectSetInteger(0,"Buy",OBJPROP_CORNER,CORNER_RIGHT_UPPER);
   ObjectSetInteger(0,"Buy",OBJPROP_XDISTANCE,100);
   ObjectSetInteger(0,"Buy",OBJPROP_YDISTANCE,50);
   ObjectSetInteger(0,"Buy",OBJPROP_XSIZE,70);
   ObjectSetInteger(0,"Buy",OBJPROP_YSIZE,30);
   ObjectSetString(0,"Buy",OBJPROP_TEXT,"Buy");
   ObjectSetInteger(0,"Buy",OBJPROP_COLOR,clrRed);
//--- überprüfen wir ob ein namens "Sell" existiert
   if(ObjectFind(0,"Sell")>=0)
     {
      //--- wenn das gefundene Objekt keine Schaltfläche ist, löschen wir es
      if(ObjectGetInteger(0,"Sell",OBJPROP_TYPE)!=OBJ_BUTTON)
         ObjectDelete(0,"Sell");
     }
   else
      ObjectCreate(0,"Sell",OBJ_BUTTON,0,0,0); // Schaltfläche "Sell" erstellen
//--- Schaltfläche "Sell" abstimmen
   ObjectSetInteger(0,"Sell",OBJPROP_CORNER,CORNER_RIGHT_UPPER);
   ObjectSetInteger(0,"Sell",OBJPROP_XDISTANCE,100);
   ObjectSetInteger(0,"Sell",OBJPROP_YDISTANCE,100);
   ObjectSetInteger(0,"Sell",OBJPROP_XSIZE,70);
   ObjectSetInteger(0,"Sell",OBJPROP_YSIZE,30);
   ObjectSetString(0,"Sell",OBJPROP_TEXT,"Sell");
   ObjectSetInteger(0,"Sell",OBJPROP_COLOR,clrBlue);
//--- Zwangsaktualisierung des Charts um die Schaltflächen sofort zu zeichnen
   ChartRedraw();
//---
  }
//+------------------------------------------------------------------+
//| Kauf über die asynchrone Funktion OrderSendAsync()               |
//+------------------------------------------------------------------+
void BuyAsync(double volume)
  {
//--- Anfrage bereiten
   MqlTradeRequest req={};
   req.action       =TRADE_ACTION_DEAL;
   req.symbol       =_Symbol;
   req.magic        =MagicNumber;
```

© 2000-2025, MetaQuotes Ltd.

```
   req.volume      =0.1;
   req.type        =ORDER_TYPE_BUY;
   req.price       =SymbolInfoDouble(req.symbol,SYMBOL_ASK);
   req.deviation   =10;
   req.comment     ="Buy using OrderSendAsync()";
   MqlTradeResult  res={};
   if(!OrderSendAsync(req,res))
     {
      Print(__FUNCTION__,": Fehler ",GetLastError(),", retcode = ",res.retcode);
     }
//---
  }
//+------------------------------------------------------------------+
//| Verkauf über die asynchrone Funktion OrderSendAsync()            |
//+------------------------------------------------------------------+
void SellAsync(double volume)
  {
//--- Anfrage bereiten
   MqlTradeRequest req={};
   req.action      =TRADE_ACTION_DEAL;
   req.symbol      =_Symbol;
   req.magic       =MagicNumber;
   req.volume      =0.1;
   req.type        =ORDER_TYPE_SELL;
   req.price       =SymbolInfoDouble(req.symbol,SYMBOL_BID);
   req.deviation   =10;
   req.comment     ="Sell using OrderSendAsync()";
   MqlTradeResult  res={};
   if(!OrderSendAsync(req,res))
     {
      Print(__FUNCTION__,": Fehler ",GetLastError(),", retcode = ",res.retcode);
     }
//---
  }
//+------------------------------------------------------------------+
```

Beispiel der Ausgabemeldungen in Protokoll "Experten":

© 2000-2025, MetaQuotes Ltd.

```
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnChartEvent: sparam = Sell
12:52:52   ExpertAdvisor (EURUSD,H1)   Sell EURUSD 0.01 lot
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTradeTransaction at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   TRADE_TRANSACTION_REQUEST
12:52:52   ExpertAdvisor (EURUSD,H1)   ------------RequestDescription
12:52:52   ExpertAdvisor (EURUSD,H1)   TRADE_ACTION_DEAL
12:52:52   ExpertAdvisor (EURUSD,H1)   Symbol: EURUSD
12:52:52   ExpertAdvisor (EURUSD,H1)   Magic Number: 1234567
12:52:52   ExpertAdvisor (EURUSD,H1)   Order ticket: 16361998
12:52:52   ExpertAdvisor (EURUSD,H1)   Order type: ORDER_TYPE_SELL
12:52:52   ExpertAdvisor (EURUSD,H1)   Order filling: ORDER_FILLING_FOK
12:52:52   ExpertAdvisor (EURUSD,H1)   Order time type: ORDER_TIME_GTC
12:52:52   ExpertAdvisor (EURUSD,H1)   Order expiration: 1970.01.01 00:00
12:52:52   ExpertAdvisor (EURUSD,H1)   Price: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Deviation points: 10
12:52:52   ExpertAdvisor (EURUSD,H1)   Stop Loss: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Take Profit: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Stop Limit: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Volume: 0.1
12:52:52   ExpertAdvisor (EURUSD,H1)   Comment: Sell using OrderSendAsync()
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   ------------ ResultDescription
12:52:52   ExpertAdvisor (EURUSD,H1)   Retcode 10009
12:52:52   ExpertAdvisor (EURUSD,H1)   Request ID: 2
12:52:52   ExpertAdvisor (EURUSD,H1)   Order ticket: 16361998
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal ticket: 15048668
12:52:52   ExpertAdvisor (EURUSD,H1)   Volume: 0.1
12:52:52   ExpertAdvisor (EURUSD,H1)   Price: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Ask: 1.29319
12:52:52   ExpertAdvisor (EURUSD,H1)   Bid: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Comment:
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistorySelect( 09:34 , 09:52) = true
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTrade at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   PositionsTotal() = 1 (+1)
12:52:52   ExpertAdvisor (EURUSD,H1)   OrdersTotal() = 0 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryOrdersTotal() = 2 (+2)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryDealsTotal() = 2 (+2)
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTradeTransaction at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   ------------ TransactionDescription
12:52:52   ExpertAdvisor (EURUSD,H1)   TRADE_TRANSACTION_ORDER_ADD
12:52:52   ExpertAdvisor (EURUSD,H1)   Symbol: EURUSD
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal ticket: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal type: DEAL_TYPE_BUY
12:52:52   ExpertAdvisor (EURUSD,H1)   Order ticket: 16361998
12:52:52   ExpertAdvisor (EURUSD,H1)   Order type: ORDER_TYPE_SELL
12:52:52   ExpertAdvisor (EURUSD,H1)   Order state: ORDER_STATE_STARTED
12:52:52   ExpertAdvisor (EURUSD,H1)   Order time type: ORDER_TIME_GTC
12:52:52   ExpertAdvisor (EURUSD,H1)   Order expiration: 1970.01.01 00:00
12:52:52   ExpertAdvisor (EURUSD,H1)   Price: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Price trigger: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Stop Loss: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Take Profit: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Volume: 0.1
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTradeTransaction at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   ------------ TransactionDescription
12:52:52   ExpertAdvisor (EURUSD,H1)   TRADE_TRANSACTION_ORDER_DELETE
12:52:52   ExpertAdvisor (EURUSD,H1)   Symbol: EURUSD
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal ticket: 0
```

© 2000-2025, MetaQuotes Ltd.

```
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal type: DEAL_TYPE_BUY
12:52:52   ExpertAdvisor (EURUSD,H1)   Order ticket: 16361998
12:52:52   ExpertAdvisor (EURUSD,H1)   Order type: ORDER_TYPE_SELL
12:52:52   ExpertAdvisor (EURUSD,H1)   Order state: ORDER_STATE_STARTED
12:52:52   ExpertAdvisor (EURUSD,H1)   Order time type: ORDER_TIME_GTC
12:52:52   ExpertAdvisor (EURUSD,H1)   Order expiration: 1970.01.01 00:00
12:52:52   ExpertAdvisor (EURUSD,H1)   Price: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Price trigger: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Stop Loss: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Take Profit: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Volume: 0.1
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistorySelect( 09:34 , 09:52) = true
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTrade at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   PositionsTotal() = 1 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   OrdersTotal() = 0 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryOrdersTotal() = 2 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryDealsTotal() = 2 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTradeTransaction at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   ------------ TransactionDescription
12:52:52   ExpertAdvisor (EURUSD,H1)   TRADE_TRANSACTION_HISTORY_ADD
12:52:52   ExpertAdvisor (EURUSD,H1)   Symbol: EURUSD
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal ticket: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal type: DEAL_TYPE_BUY
12:52:52   ExpertAdvisor (EURUSD,H1)   Order ticket: 16361998
12:52:52   ExpertAdvisor (EURUSD,H1)   Order type: ORDER_TYPE_SELL
12:52:52   ExpertAdvisor (EURUSD,H1)   Order state: ORDER_STATE_FILLED
12:52:52   ExpertAdvisor (EURUSD,H1)   Order time type: ORDER_TIME_GTC
12:52:52   ExpertAdvisor (EURUSD,H1)   Order expiration: 1970.01.01 00:00
12:52:52   ExpertAdvisor (EURUSD,H1)   Price: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Price trigger: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Stop Loss: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Take Profit: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Volume: 0
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistorySelect( 09:34 , 09:52) = true
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTrade at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   PositionsTotal() = 1 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   OrdersTotal() = 0 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryOrdersTotal() = 2 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryDealsTotal() = 2 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTradeTransaction at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   ------------ TransactionDescription
12:52:52   ExpertAdvisor (EURUSD,H1)   TRADE_TRANSACTION_DEAL_ADD
12:52:52   ExpertAdvisor (EURUSD,H1)   Symbol: EURUSD
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal ticket: 15048668
12:52:52   ExpertAdvisor (EURUSD,H1)   Deal type: DEAL_TYPE_SELL
12:52:52   ExpertAdvisor (EURUSD,H1)   Order ticket: 16361998
12:52:52   ExpertAdvisor (EURUSD,H1)   Order type: ORDER_TYPE_BUY
12:52:52   ExpertAdvisor (EURUSD,H1)   Order state: ORDER_STATE_STARTED
12:52:52   ExpertAdvisor (EURUSD,H1)   Order time type: ORDER_TIME_GTC
12:52:52   ExpertAdvisor (EURUSD,H1)   Order expiration: 1970.01.01 00:00
12:52:52   ExpertAdvisor (EURUSD,H1)   Price: 1.29313
12:52:52   ExpertAdvisor (EURUSD,H1)   Price trigger: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Stop Loss: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Take Profit: 0
12:52:52   ExpertAdvisor (EURUSD,H1)   Volume: 0.1
12:52:52   ExpertAdvisor (EURUSD,H1)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistorySelect( 09:34 , 09:52) = true
```

© 2000-2025, MetaQuotes Ltd.

```
12:52:52   ExpertAdvisor (EURUSD,H1)   => OnTrade at 09:52:53
12:52:52   ExpertAdvisor (EURUSD,H1)   PositionsTotal() = 1 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   OrdersTotal() = 0 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryOrdersTotal() = 2 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)   HistoryDealsTotal() = 2 (+0)
12:52:52   ExpertAdvisor (EURUSD,H1)
```

© 2000-2025, MetaQuotes Ltd.

PositionsTotal

Gibt die Anzahl der offenen Positionen zurück.

```
int  PositionsTotal();
```

Rückgabewert

Wert vom Typ int.

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen eines Symbols vorhanden sein.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzahl der offenen Positionen auf dem Konto im Journal abrufen und ausdrucken
   int total=PositionsTotal();
   Print("Number of open positions on account: ", total);
   /*
   Ergebnis:
   Number of open positions on account: 2
   */
  }
```

Siehe auch

PositionGetSymbol(), PositionSelect(), Eigenschaften der Positionen

© 2000-2025, MetaQuotes Ltd.

PositionGetSymbol

Gibt das Symbol der entsprechenden offenen Position zurück und wählt automatisch die Position für die weitere Arbeit mit den Funktionen PositionGetDouble,PositionGetInteger, PositionGetString aus.

```
string  PositionGetSymbol(
   int  index      // Index in der Positionenliste
   );
```

Parameter

index

[in] Positionsnummer in der Liste offener Positionen.

Rückgabewert

Wert vom Typ string. Wenn die Position nicht gefunden wird, wird eine leere Zeichenkette zurückgegeben. Um den Fehlercode zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzahl der offenen Positionen auf dem Konto abrufen
   int total=PositionsTotal();
   for(int i=0; i<total; i++)
     {
      //--- Positionssymbol über den Schleifenindex i abrufen
      ResetLastError();
      string symbol=PositionGetSymbol(i);

      //--- wenn das Positionssymbol erfolgreich empfangen wurde, wird die Position m
      //--- und wir können dann weitere Eigenschaften mit PositionGetDouble, Position
      if(symbol!="")
        {
         ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE)
         PrintFormat("Position symbol at index %d: %s, position type: %s", i, symbol,
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      else
        {
         PrintFormat("PositionGetSymbol(%d) failed. Error %d", i, GetLastError());
         continue;
        }
     }
   /*
   Ergebnis:
   Position symbol at index 0: GBPUSD, position type: SELL
   Position symbol at index 1: EURUSD, position type: BUY
   */
  }
```

Siehe auch

PositionsTotal(), PositionSelect(), Eigenschaften der Positionen

© 2000-2025, MetaQuotes Ltd.

PositionSelect

Wählt eine offene Position für die weitere Arbeit aus. Gibt true zurück, wenn die Durchführung der Funktion erfolgreich beendet wird. Gibt false zurück, wenn die Durchführung der Funktion erfolglos beendet wird. Für den Erhalt der fehlerbezogenen Information, rufen Sie die Funktion GetLastError() auf.

```
bool  PositionSelect(
   string  symbol,     // Name des Finanzinstruments
   );
```

Parameter

symbol

[in] Name des Finanzinstruments.

Rückgabewert

Wert vom Typ bool.

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen pro Symbol vorhanden sein. In diesem Fall wählt PositionSelect die Funktion mit dem kleinsten Ticket.

Die Funktion PositionSelect() kopiert die Daten der Position in die Programmumgebung und ermöglicht die weiteren Aufrufe von PositionGetDouble(), PositionGetInteger() und PositionGetString(), um die jeweiligen Daten abzurufen. Das bedeutet, dass die Position selbst nicht mehr existieren kann (oder ihr Volumen, ihre Richtung usw. haben sich verändert), aber die alten Daten dieser Position können trotzdem noch existieren. Um die neuesten Daten einer Position zu erhalten, ist es empfehlenswert, die Funktion PositionSelect() aufzurufen, bevor man diese Daten verwendet.

Beispiel:

```
#define   SYMBOL_NAME   "EURUSD"

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wähle eine Position des angegebenen Symbols
   if(!PositionSelect(SYMBOL_NAME))
     {
      PrintFormat("PositionSelect(%s) failed. Error %d",SYMBOL_NAME, GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
     }

//--- wenn eine Position ausgewählt ist, können wir seine Daten mit PositionGetDouble
//--- Abrufen der mit der Ticketnummer ausgewählten Position//--- Holen Sie sich das a
   ResetLastError();
   long ticket=PositionGetInteger(POSITION_TICKET);
   if(ticket==0)
     {
      PrintFormat("Failed to get %s position ticket. Error %d", SYMBOL_NAME, GetLastEr
      return;
     }

//--- Wenn ein Ticket erfolgreich empfangen wurde, drucke das ausgewählte Positionssyr
   PrintFormat("The position that is selected on the %s symbol has ticket %I64d", SYMB
   /*
   Ergebnis:
   The position that is selected on the EURUSD symbol has ticket 2810846623
   */
  }
```

Siehe auch

PositionGetSymbol(), PositionsTotal(), Eigenschaften der Positionen

© 2000-2025, MetaQuotes Ltd.

PositionSelectByTicket

Wählt eine offene Position nach dem angegebenen Ticket für die weitere Arbeit aus. Gibt true bei einem erfolgreichen Beenden der Funktion zurück. Gibt false zurück, wenn das Beenden der Funktion fehlgeschlagen ist. Rufen Sie die Funktion GetLastError() auf, um Details zum Fehler zu bekommen.

```
bool  PositionSelectByTicket(
   ulong    ticket     // das Ticket der Position
   );
```

Parameter

ticket

[in] Ticket der Position.

Rückgabewert

Der Wert von bool.

Hinweis

Die Funktion PositionSelectByTicket() kopiert die Daten der Position in die Programmumgebung und ermöglicht die weiteren Aufrufe von PositionGetDouble(), PositionGetInteger() und PositionGetString(), um die jeweiligen Daten abzurufen. Es kann sein, dass die Position nicht mehr vorhanden ist (oder ihr Volumen, Richtung usw. sich geändert haben), aber die alten Daten dieser Position können trotzdem noch existieren. Um aktuelle Details zu einer Position zu erhalten, sollte man unmittelbar davor die Funktion PositionSelectByTicket() aufrufen.

Beispiel:

```
#define   EXPERT_MAGIC  123456   // MagicNumber

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anforderungs- und Ergebnisstrukturen deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result ={};

//--- Handelsanforderungsparameter ausfüllen, um eine Kaufposition zu eröffnen
   request.action    = TRADE_ACTION_DEAL;                       // Typ der Handelsopera
   request.symbol    = Symbol();                                // Symbol
   request.volume    = 0.1;                                     // Volumen von 0.1 Lot
   request.type      = ORDER_TYPE_BUY;                          // Auftragstyp
   request.price     = SymbolInfoDouble(Symbol(), SYMBOL_ASK); // Eröffnungspreis
   request.deviation = 5;                                       // erlaubte Abweichung
   request.magic     = EXPERT_MAGIC;                            // MagicNumber des Auft

//--- eine Anfrage senden. Wenn das Senden einer Anfrage fehlgeschlagen ist, zeige der
   if(!OrderSend(request, result))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       PrintFormat("OrderSend error ", GetLastError());
       return;
      }

//--- Operationsdaten anzeigen
   PrintFormat("Trade request result: retcode: %u, deal: %I64u, order: %I64u", result.

//--- Abrufen des Positionstickets aus dem Ergebnis der Handelsoperation und Auswahl d
//--- Das Ticket einer neu eröffneten Position entspricht dem Ticket des Auftrags, der
   ulong ticket=result.order;
   ResetLastError();
   if(!PositionSelectByTicket(ticket))
     {
      PrintFormat("PositionSelectByTicket(%I64u) failed. Error %d", ticket, GetLastErr
      return;
     }

//--- Daten einer per Ticket ausgewählten Position im Journal anzeigen
   ENUM_POSITION_TYPE type  = (ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
   long               time  = PositionGetInteger(POSITION_TIME_MSC);
   double             price = PositionGetDouble(POSITION_PRICE_OPEN);
   double             volume= PositionGetDouble(POSITION_VOLUME);
   string             symbol= PositionGetString(POSITION_SYMBOL);
   int                digits= (int)SymbolInfoInteger(symbol, SYMBOL_DIGITS);
   PrintFormat("Current selected position: %s %.2f %s #%I64u at %.*f, %s",
               symbol, volume, (type==POSITION_TYPE_BUY ? "Buy" : "Sell"), ticket, dig
   /*
   Ergebnis:
   Trade request result: retcode: 10009, deal: 2778100901, order: 2803905975
   Current selected position: EURUSD 0.10 Buy #2803905975 at 1.10672, 2024.09.02 12:09
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe der Zeit in Millisekunden                               |
//+------------------------------------------------------------------+
string TimeMscToString(const long time_msc, int flags=TIME_DATE|TIME_MINUTES|TIME_SECO
  {
   return(TimeToString(time_msc/1000, flags) + "." + IntegerToString(time_msc %1000, 3
  }
```

Sieh auch

PositionGetSymbol(), PositionsTotal(), Eigenschaften der Position

© 2000-2025, MetaQuotes Ltd.

PositionGetDouble

Die Funktion gibt die angeforderte Eigenschaft der offenen Position zurück, die vorläufig mittels der Funktion PositionGetSymbol oder PositionSelect ausgewählt wurde. Die Podsitionseigenschaft muss vom Typ double sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
double  PositionGetDouble(
    ENUM_POSITION_PROPERTY_DOUBLE  property_id      // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wird. Bei erfolgreicher Durchführung wird der Wert der Eigenschaft der Empfangsvariablen zugewiesen, die durch Referenz als letzter Parameter übertragen wird.

```
bool  PositionGetDouble(
    ENUM_POSITION_PROPERTY_DOUBLE  property_id,     // Identifikator der Eigenschaft
   double&                         double_var       // hier nehmen wir den Wert der Ei
   );
```

Parameter

property_id

[in] Identifikator der Positionseigenschaft. Der Wert kann einer der Enumerationswerte ENUM_POSITION_PROPERTY_DOUBLE sein.

double_var

[out] Variable vom Typ double, der der Wert der angeforderten Eigenschaft zugewiesen wird.

Rückgabewert

Wert vom Typ double.

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Um aktuelle Details zu einer Position zu erhalten, sollte man unmittelbar davor die Funktion PositionSelect() aufrufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- in einer Schleife durch die Liste aller Positionen des Kontos
   int total=PositionsTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Tickets der nächsten Position, durch die automatische Auswahl
      ulong ticket=PositionGetTicket(i);
      if(ticket==0)
         continue;

      //--- Positionstyp abrufen und Kopfzeile für die Liste der „double"-Eigenschafte
      string type=(PositionGetInteger(POSITION_TYPE)==POSITION_TYPE ? "Buy" : "Sell");
      PrintFormat("Double properties of an open position %s #%I64u:", type, ticket);

      //--- alle „double"-Eigenschaften der ausgewählten Position unter der Kopfzeile
      PositionPropertiesDoublePrint(15);
     }
   /*
   Ergebnis:
   Double properties of an open position Buy #2807075208:
   Volume:        1.00
   Price open:    1.10516
   StopLoss:      0.00000
   TakeProfit:    0.00000
   Price current: 1.10518
   Swap:          0.00
   Profit:        2.00 USD
   */
  }
//+------------------------------------------------------------------+
//| „double"-Eigenschaften der gewählten Position ins Journal drucken|
//+------------------------------------------------------------------+
void PositionPropertiesDoublePrint(const uint header_width=0)
  {
   uint   w=0;
   string header="";
   double value=0;

//--- Kontowährung, Positionssymbol und Anzahl der Dezimalstellen für das Symbol abru
   string currency=AccountInfoString(ACCOUNT_CURRENCY);
   string symbol  =PositionGetString(POSITION_SYMBOL);
   int    digits  =(int)SymbolInfoInteger(symbol, SYMBOL_DIGITS);

//--- Definieren der Kopftextes und der Breite des Textfeldes
//--- Wenn der Funktion eine Breite von Null übergeben wird, dann ist die Breite die
   header="Volume:";
   w=(header_width==0 ? header.Length()+1 : header_width);
//--- Abrufen und Ausdruck von Positionsvolumen mit der angegebenen Breite der Kopfze
   if(!PositionGetDouble(POSITION_VOLUME, value))
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
   PrintFormat("%-*s%-.2f", w, header, value);

//--- Anzeige des Preises der Position im Journal
   header="Price open:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetDouble(POSITION_PRICE_OPEN, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);

//--- Wert des StopLoss im Journal anzeigen
   header="StopLoss:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetDouble(POSITION_SL, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);

//--- Wert des TakeProfit im Journal anzeigen
   header="TakeProfit:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetDouble(POSITION_TP, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);

//--- den Wert „Aktueller Preis"-im Journal anzeigen
   header="Price current:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetDouble(POSITION_PRICE_CURRENT, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);

//--- kumulierten Swap-Wert im Journal anzeigen
   header="Swap:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetDouble(POSITION_SWAP, value))
      return;
   PrintFormat("%-*s%-.2f", w, header, value);

//--- aktuellen Gewinn im Journal anzeigen
   header="Profit:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetDouble(POSITION_PROFIT, value))
      return;
   PrintFormat("%-*s%-.2f %s", w, header, value, currency);
  }
```

Siehe auch

PositionGetSymbol(), PositionSelect(), Eigenschaften der Positionen

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

PositionGetInteger

Funktion gibt die angeforderte Eigenschaft der offenen Position zurück, vorher gewaehlt durch die Funktionen PositionGetSymbol oder PositionSelect. Positionseigenschaft muss der Art , int. sein. Es gibt zwei Arten der Funktion.

1. Gibt Wert der Eigenschaft direkt zurück.

```
long  PositionGetInteger(
   ENUM_POSITION_PROPERTY_INTEGER  property_id      // Eigenschaftsidentifikator
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Bei erfolgreicher Durchführung wird der Wert der Eigenschaft in die Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  PositionGetInteger(
   ENUM_POSITION_PROPERTY_INTEGER  property_id,     // Iegenschaftsidentifikator
   long&                           long_var         // hier nehmen wir den Wert der Ei
   );
```

Parameter

property_id

[in] Identifikator der Positionseigenschaft . Wert kann einer der Enumerationswerte ENUM_POSITION_PROPERTY_INTEGER sein.

long_var

[out] Variable der Art long, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert der Art long. Im Fall der fehlerhaften Ausführung gibt 0 zurück.

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehreren Trades darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen auf einem Symbol vorhanden sein.

Für bestimmte Erhaltung der frischen Daten über die Position ist es empfehlenswert, die Funktion PositionSelect() aufzurufen, bevor man frische Daten aufruft.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Trade function                                                   |
//+------------------------------------------------------------------+
void OnTrade()
  {
//--- Überprüfen wir die Verfügbarkeit der Position und daraus Zeit ihrer Änderungen
   if(PositionSelect(_Symbol))
     {
//--- Indikator der Position für weitere Arbeit mit ihr erhalten
      ulong position_ID=PositionGetInteger(POSITION_IDENTIFIER);
      Print(_Symbol," position #",position_ID);
//--- Zeit der Positionerstellung in Millisekunden seit 01.01.1970 erhalten
      long create_time_msc=PositionGetInteger(POSITION_TIME_MSC);
      PrintFormat("Position #%d  POSITION_TIME_MSC = %i64 milliseconds => %s",positior
                  create_time_msc,TimeToString(create_time_msc/1000));
//--- Zeit der letzte Änderung der Position in Millisekunden seit 01.01.1970 erhalten
      long update_time_sec=PositionGetInteger(POSITION_TIME_UPDATE);
      PrintFormat("Position #%d  POSITION_TIME_UPDATE = %i64 seconds => %s",
                  position_ID,update_time_sec,TimeToString(update_time_sec));
//--- Zeit der letzte Änderung der Position in Millisekunden seit 01.01.1970 erhalten
      long update_time_msc=PositionGetInteger(POSITION_TIME_UPDATE_MSC);
      PrintFormat("Position #%d  POSITION_TIME_UPDATE_MSC = %i64 milliseconds => %s",
                  position_ID,update_time_msc,TimeToString(update_time_msc/1000));
     }
//---
  }
```

Sehen Sie auch

PositionGetSymbol(), PositionSelect(), Eigenschaften der Positionen

© 2000-2025, MetaQuotes Ltd.

PositionGetString

Funktion gibt die angeforderte Eigenschaft der offenen Position zurück, die vorher mit der Funktion PositionGetSymbol oder PositionSelect ausgewählt wurde. Eigenschaft der Position muss vom Typ string sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
string  PositionGetString(
   ENUM_POSITION_PROPERTY_STRING  property_id      // Identifikator der Eigenschaft
   );
```

2.Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird der Wert der Eigenschaft der Empfangsvariable zugewiesen, die per Referenz als letzter Parameter übergeben wird.

```
bool  PositionGetString(
   ENUM_POSITION_PROPERTY_STRING  property_id,     // Eigenschaftsidentifikator
   string&                        string_var       // hier nehmen wir den Wert der Eig
   );
```

Parameter

property_id

[in] Identifikator der Positionseigenschaft. Wert kann einer der Enumerationswerte ENUM_POSITION_PROPERTY_STRING sein.

string_var

[out] Variable vom Typ string, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ string.

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Um die neuesten Daten einer Position zu erhalten, ist es empfehlenswert, die Funktion PositionSelect() aufzurufen, bevor man diese Daten verwendet.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- in einer Schleife durch die Liste aller Positionen des Kontos
   int total=PositionsTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Tickets der nächsten Position, durch die automatische Auswahl
      ulong ticket=PositionGetTicket(i);
      if(ticket==0)
         continue;

      //--- Positionstyp abrufen und Kopfzeile für die Liste der „string"-Eigenschafte
      string type=(PositionGetInteger(POSITION_TYPE)==POSITION_TYPE ? "Buy" : "Sell");
      PrintFormat("String properties of an open position %s #%I64u:", type, ticket);

      //--- alle „string"-Eigenschaften der ausgewählten Position unter der Kopfzeile
      PositionPropertiesStringPrint(15);
     }
   /*
   Ergebnis:
   String properties of an open position Buy #2810798881:
   Symbol:         EURUSD
   Comment:        Test PositionGetString
   External ID:
   */
  }
//+------------------------------------------------------------------+
//| String-Eigenschaften der gewählten Position ins Journal drucken  |
//+------------------------------------------------------------------+
void PositionPropertiesStringPrint(const uint header_width=0)
  {
   uint   w=0;
   string header="";
   string value="";

//--- Definieren der Kopftextes und der Breite des Textfeldes
//--- Wenn der Funktion eine Breite von Null übergeben wird, dann ist die Breite die C
   header="Symbol:";
   w=(header_width==0 ? header.Length()+1 : header_width);
//--- Abrufen und Anzeigen des Positionssymbols mit der angegebenen Breite der Kopfzei
   if(!PositionGetString(POSITION_SYMBOL, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);

//--- Positionskommentar im Journal anzeigen
   header="Comment:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetString(POSITION_COMMENT, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Anzeige der Positions-ID eines externen Systems im Journal
   header="External ID:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!PositionGetString(POSITION_EXTERNAL_ID, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);
  }
```

Siehe auch

PositionGetSymbol(), PositionSelect(), Eigenschaften der Positionen

© 2000-2025, MetaQuotes Ltd.

PositionGetTicket

Die Funktion gibt das Ticket der Position nach dem Index in der Liste offener Positionen zurück und wählt diese Position für die weitere Arbeit mit den Funktionen PositionGetDouble, PositionGetInteger und PositionGetString automatisch aus.

```
ulong  PositionGetTicket(
   int  index      // Nummer in der Liste der Positionen
   );
```

Parameter

index

[in] Der Index der Position in der Liste offener Positionen, von 0.

Rückgabewert

Das Ticket der Position. Wenn die Ausführung fehlgeschlagen ist, liefert die Funktion Null.

Hinweis

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Um aktuelle Details zu einer Position zu erhalten, sollte man unmittelbar davor die Funktion PositionSelect() aufrufen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- in einer Schleife durch die Liste aller Positionen des Kontos
   int total=PositionsTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Tickets der nächsten Position, durch die automatische Auswahl
      ulong ticket=PositionGetTicket(i);
      if(ticket==0)
         continue;

      //--- Abrufen des Positionstyps und Ausdruck der Beschreibung der ausgewählten P
      string type=(PositionGetInteger(POSITION_TYPE)==POSITION_TYPE ? "Buy" : "Sell");
      PrintFormat("[%d] Selected position %s #%I64u", i, type, ticket);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   /*
   Ergebnis:
   [0] Selected position Sell #2810802718
   [1] Selected position Buy #2810802919
   */
  }
```

Sieh auch

PositionGetSymbol(), PositionSelect(), Eigenschaften der Position

© 2000-2025, MetaQuotes Ltd.

OrdersTotal

Gibt die Orderanzahl zurück.

```
int  OrdersTotal();
```

Rückgabewert

Wert vom Typ int.

Hinweis

Man darf gültige schwebende Aufträge nicht mit Positionen verwechseln, die auch in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Order ist ein schwebender Auftrag, eine Handelsoperation durchzuführen, und eine Position ist das Ergebnis eines oder mehrerer Deals.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzahl der aktiven schwebenden Aufträge (pending order) auf dem Konto abrufen ur
   int total=OrdersTotal();
   Print("Number of active pending orders on the account: ", total);
   /*
   Ergebnis:
   Number of active pending orders on the account: 2
   */
  }
```

Siehe auch

OrderSelect(), OrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

OrderGetTicket

Gibt Ticket der ensprechenden Order zurück und waehlt automatisch Order für die weitere Arbeit damit mittels Funktionen .

```
ulong  OrderGetTicket(
   int  index      // Nummer in Orderliste
   );
```

Parameter

index

[in] Ordernummer in Orderliste.

Rückgabewert

Wert der Art ulong.

Hinweis

Man muss nicht geltende Warteordern und Positionen verwechseln, die auch in der Registerkarte "Handel" in der Werkzeigleiste "Instrumente" dargestellt werden. Order ist ein Befehl, eine Handelsoperation durchzuführen, und Position ist das Ergebnis eines oder mehrerer Deals.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehreren Trades darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen auf einem Symbol vorhanden sein.

Funktion OrderGetTicket() kopiert Daten über die Order ins Programmumfeld und weitere Aufrufe OrderGetDouble(), OrderGetInteger(), OrderGetString() geben die früher kopierten Daten zurück. Das bedeutet, dass die Order selbst nicht mehr sein (oder darin haben sich Eröffnungspreis, Levels Stop Loss / Take Profitoder Gültigkeitsfrist verändert), aber die Daten nach dieser Order können noch erhalten werden. Für bestimmte Erhaltung der frischen Daten über die Position ist es empfehlenswert, die Funktion OrderGetTicket() aufzurufen, bevor man frische Daten aufruft

Beispiel:

```
void OnStart()
  {
//--- Variablen für Erhaltung der Werte aus Ordereigenschaften
   ulong    ticket;
   double   open_price;
   double   initial_volume;
   datetime time_setup;
   string   symbol;
   string   type;
   long     order_magic;
   long     positionID;
//--- Anzahl der laufenden Warteorder
   uint     total=OrdersTotal();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- gehen wir im Zyklus durch alle Ordern
   for(uint i=0;i<total;i++)
     {
      //--- erhalten wir Orderticket in Bezug auf seine Position in der Liste
      if((ticket=OrderGetTicket(i))>0)
        {
         //--- erhlaten wir Ordereigenschaften
         open_price    =OrderGetDouble(ORDER_PRICE_OPEN);
         time_setup    =(datetime)OrderGetInteger(ORDER_TIME_SETUP);
         symbol        =OrderGetString(ORDER_SYMBOL);
         order_magic   =OrderGetInteger(ORDER_MAGIC);
         positionID    =OrderGetInteger(ORDER_POSITION_ID);
         initial_volume=OrderGetDouble(ORDER_VOLUME_INITIAL);
         type          =EnumToString(ENUM_ORDER_TYPE(OrderGetInteger(ORDER_TYPE)));
         //--- bereiten wir die Information über Order vor und geben sie aus
         printf("#ticket %d %s %G %s at %G was set up at %s",
                ticket,                  // Orderticket
                type,                    // Typ
                initial_volume,          // das gestellte Volumen
                symbol,                  // Symbol, für das gestellt ist
                open_price,              // der angegebene Eroeffnungspreis
                TimeToString(time_setup)// Zeit der Orderstellung
                );
        }
     }
//---
  }
```

Sehen Sie auch

OrdersTotal(), OrderSelect(), OrderGetInteger()

© 2000-2025, MetaQuotes Ltd.

OrderSelect

Auswahl des Auftrags für die weitere Arbeit. Gibt true bei erfolgreicher Funktionsausführung zurück. Gibt false bei fehlerhafter Funktionsausführung zurück. Um Information zum Fehler zu bekommen, muss die Funktion GetLastError() aufgerufen werden.

```
bool  OrderSelect(
   ulong    ticket,       // Orderticket
   );
```

Parameter

ticket

[in] Orderticket.

Rückgabewert

Wert vom Typ bool.

Hinweis

Man darf gültige schwebende Aufträge nicht mit Positionen verwechseln, die auch in der Registerkarte "Handel" in der Wekzeugleiste "Instrumente" des Client-Terminals dargestellt werden.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Funktion OrderSelect() kopiert Daten über die Order in die Programmumgebung und ermöglicht die weiteren Aufrufe von OrderGetDouble(), OrderGetInteger(), OrderGetString(), um die jeweiligen Daten abzurufen. Das bedeutet, dass die Order selbst nicht mehr existiert könnte (oder sich Eröffnungspreis, Stop Loss, Take Profit oder Gültigkeitsfrist verändert haben), aber alten Daten dieser Order könnten trotzdem noch vorhanden sein. Um die neuesten Daten einer Position zu erhalten, ist es empfehlenswert, die Funktion PositionSelect() aufzurufen, bevor man diese Daten verwendet.

Beispiel:

```
#define   EXPERT_MAGIC  123456
#define   OFFSET        50                      // Abstand vom aktuellen Preis, um die
#define   DIRECTION     ORDER_TYPE_BUY_LIMIT    // Order-Typ
#define   VOLUME        1.0                     // Volumen
#define   DEVIATION     2                       // erlaubte Abweichung vom Preis

//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//-- Handelsanfrage, Ergebnis und Variablen deklarieren und initialisieren
   MqlTradeRequest request={};
   MqlTradeResult  result ={};
   double          order_price=0;

//--- Parameter der Pending-Orders
   request.action    = TRADE_ACTION_PENDING;                                  // Typ der
   request.symbol    = _Symbol;                                               // Symbol
   request.volume    = VOLUME;                                                // Volumen
   request.deviation = DEVIATION;                                             // erlaubte
   request.magic     = EXPERT_MAGIC;                                          // MagicNur

//--- Prüfen des Auftragstyps
   switch(DIRECTION)
     {
      case ORDER_TYPE_BUY_LIMIT :
        request.type = ORDER_TYPE_BUY_LIMIT;                                  // Order-Ty
        order_price  = SymbolInfoDouble(_Symbol, SYMBOL_ASK)-OFFSET*_Point;// Order-Pr
        request.price= NormalizeDouble(order_price, _Digits);                 // normalis
        break;
      case ORDER_TYPE_SELL_LIMIT :
        request.type = ORDER_TYPE_SELL_LIMIT;                                 // Order-Ty
        order_price  = SymbolInfoDouble(_Symbol, SYMBOL_BID)+OFFSET*_Point;// Order-Pr
        request.price= NormalizeDouble(order_price,_Digits);                  // normalis
        break;
      case ORDER_TYPE_BUY_STOP :
        request.type = ORDER_TYPE_BUY_STOP;                                   // Order-Ty
        order_price  = SymbolInfoDouble(_Symbol, SYMBOL_ASK)+OFFSET*_Point;// Eröffnur
        request.price= NormalizeDouble(order_price,_Digits);                  // normalis
        break;
      case ORDER_TYPE_SELL_STOP :
        request.type = ORDER_TYPE_SELL_STOP;                                  // Order-Ty
        order_price  = SymbolInfoDouble(_Symbol, SYMBOL_BID)-OFFSET*_Point;// Eröffnur
        request.price= NormalizeDouble(order_price,_Digits);                  // normalis
        break;
      default: // wenn eine nicht-schwebende oder StopLimit-Order gewählt wurde
        Alert("This example is only for placing pending orders BuyLimit, SellLimit, Bu
        break;
     }

//--- eine Anfrage senden. Wenn das Senden einer Anfrage fehlgeschlagen ist, zeige der
   if(!OrderSend(request, result))
     {
      Print("OrderSend error ", GetLastError());
      return;
     }

//--- Operationsdaten anzeigen
   PrintFormat("Trade request result: retcode=%u, order=%I64u", result.retcode, result
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Order-Ticket aus dem Handelsoperationsergebnis abrufen und die Order nach Ticket
   ulong ticket=result.order;
   ResetLastError();
   if(!OrderSelect(ticket))
     {
      PrintFormat("OrderSelect(%I64u) failed. Error %d", ticket, GetLastError());
      return;
     }

//--- Daten der per Ticket ausgewählten Order im Journal anzeigen
   ENUM_ORDER_TYPE   type  = (ENUM_ORDER_TYPE)OrderGetInteger(ORDER_TYPE);
   long              time  = OrderGetInteger(ORDER_TIME_SETUP_MSC);
   double            price = OrderGetDouble(ORDER_PRICE_OPEN);
   double            volume= OrderGetDouble(ORDER_VOLUME_CURRENT);
   string            symbol= OrderGetString(ORDER_SYMBOL);
   int               digits= (int)SymbolInfoInteger(symbol, SYMBOL_DIGITS);
   PrintFormat("Current selected order: %s %.2f %s #%I64u at %.*f, %s",
               symbol, volume, OrderTypeDescription(type), ticket, digits, price, Time
   /*
  Ergebnis:
   Trade request result: retcode=10009, order=2811006719
   Current selected order: EURUSD 1.00 Buy Limit #2811006719 at 1.10550, 2024.09.04 10
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe der Zeit in Millisekunden                               |
//+------------------------------------------------------------------+
string TimeMscToString(const long time_msc, int flags=TIME_DATE|TIME_MINUTES|TIME_SEC
  {
   return(TimeToString(time_msc/1000, flags) + "." + IntegerToString(time_msc %1000, 3
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type");
     }
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Siehe auch

OrderGetInteger(), OrderGetDouble(), OrderGetString(), OrderCalcProfit(), OrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

OrderGetDouble

Gibt die angeforderte Ordereigenschaft zurück, die vorher mit der Funktionen OrderGetTicket oder OrderSelect ausgewählt wurde. Ordereigenschaften vom Typ double. Es gibt zwei Arten der Funktion.

1. Gibt unmittelbar die Eigenschaftsgröße zurück.

```
double  OrderGetDouble(
   ENUM_ORDER_PROPERTY_DOUBLE  property_id        // Eigenschaftsidentifikator
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wird. Im Erfolgsfall wird der Eigenschaftswert der Empfangsvariable zugewiesen, die im letzten Parameter als Referenz übergeben wird.

```
bool  OrderGetDouble(
   ENUM_ORDER_PROPERTY_DOUBLE  property_id,      // Eigenschaftsidentifikator
   double&                     double_var        // hier erhalten wir den Wert der Eig
   );
```

Parameter

property_id

[in] Ordereigenschaftsidentifikator. Der Wert kann einer aus der Enumeration ENUM_ORDER_PROPERTY_DOUBLE sein.

double_var

[out] Variable vom Typ double, der der Wert der angeforderten Eigenschaft zugewiesen wird.

Rückgabewert

Wert vom Typ double.

Hinweis

Man darf gültige schwebende Aufträge nicht mit Positionen verwechseln, die auch in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" des Client-Terminals dargestellt werden.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Um die neuesten Daten einer Position zu erhalten, ist es empfehlenswert, die Funktion OrderSelect() aufzurufen, bevor man diese Daten verwendet.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- in einer Schleife durch die Liste aller Orders des Kontos
   int total=OrdersTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Order-Tickets in der Liste über den Schleifenindex
      ulong ticket=OrderGetTicket(i);
      if(ticket==0)
         continue;

      //--- Auftragstyp abrufen und Kopfzeile für die Liste der „double"-Eigenschafter
      string type=OrderTypeDescription((ENUM_ORDER_TYPE)OrderGetInteger(ORDER_TYPE));
      PrintFormat("Double properties of an active pending order %s #%I64u:", type, tic

      //--- alle „double"-Eigenschaften des ausgewählten Auftrags unter der Überschrif
      OrderPropertiesDoublePrint(16);
     }
   /*
   Ergebnis:
   Double properties of an active pending order Sell Limit #2812000714:
   Volume initial: 1.00
   Volume current: 1.00
   Price open:     145.282
   StopLoss:       0.000
   TakeProfit:     0.000
   Price current:  145.044
   StopLimit:      0.000
   Double properties of an active pending order Buy Limit #2812001112:
   Volume initial: 1.00
   Volume current: 1.00
   Price open:     144.836
   StopLoss:       0.000
   TakeProfit:     0.000
   Price current:  145.051
   StopLimit:      0.000
   Double properties of an active pending order Buy Stop #2812001488:
   Volume initial: 0.50
   Volume current: 0.50
   Price open:     1.10642
   StopLoss:       0.00000
   TakeProfit:     0.00000
   Price current:  1.10530
   StopLimit:      0.00000
   Double properties of an active pending order Sell Stop #2812001712:
   Volume initial: 0.50
   Volume current: 0.50
   Price open:     1.10374
   StopLoss:       0.00000
```

© 2000-2025, MetaQuotes Ltd.

```
   TakeProfit:      0.00000
   Price current:   1.10525
   StopLimit:       0.00000
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type");
     }
  }
//+------------------------------------------------------------------+
//| „double"-Eigenschaften der gewählten Order ausdrucken            |
//+------------------------------------------------------------------+
void OrderPropertiesDoublePrint(const uint header_width=0)
  {
//--- Orders-Symbol und Anzahl dessen Dezimalstellen abrufen
   string symbol = OrderGetString(ORDER_SYMBOL);
   int    digits = (int)SymbolInfoInteger(symbol, SYMBOL_DIGITS);

//--- Ausdruck des Anfangsvolumens bei der Platzierung einer Order mit einer Kopfzeile
   OrderPropertyPrint("Volume initial:",header_width,2,ORDER_VOLUME_INITIAL);

//--- Anzeige des nicht gefüllten Auftragsvolumens im Journal
   OrderPropertyPrint("Volume current:",header_width,2,ORDER_VOLUME_CURRENT);

//--- Anzeige des in der Order angegebenen Preises im Journal
   OrderPropertyPrint("Price open:",header_width,digits,ORDER_PRICE_OPEN);

//--- Ausdruck der StopLoss-Level im Journal
   OrderPropertyPrint("StopLoss:",header_width,digits,ORDER_SL);

//--- Ausdruck des TakeProfit-Levels im Journal
   OrderPropertyPrint("TakeProfit:",header_width,digits,ORDER_TP);

//--- Ausdruck des aktuellen Preises des Ordersymbols im Journal
   OrderPropertyPrint("Price current:",header_width,digits,ORDER_PRICE_CURRENT);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Ausdruck des Limit-Orderpreises, wenn eine StopLimit-Order aktiviert ist, im Jou
   OrderPropertyPrint("StopLimit:",header_width,digits,ORDER_PRICE_STOPLIMIT);
  }
//+------------------------------------------------------------------+
//| Die Order-Eigenschaften vom Typ „double"-im Journal anzeigen     |
//+------------------------------------------------------------------+
void OrderPropertyPrint(const string header, uint header_width, int digits, ENUM_ORDER
  {
   double value=0;
   if(!OrderGetDouble(property, value))
      PrintFormat("Cannot get property %s, error=%d", EnumToString(property), GetLastE
   else
     {
      //--- Wenn der Funktion eine Breite der Kopfzeile von Null übergeben wird, dann
      uint w=(header_width==0 ? header.Length()+1 : header_width);
      PrintFormat("%-*s%-.*f", w, header, digits, value);
     }
  }
```

Siehe auch

OrdersTotal(), OrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

OrderGetInteger

Gibt die angeforderte Eigenschaft der Order zurück, die vorher durch die Funktion OrderGetTicket oder OrderSelect gewählt wurde. Die Eigenschaft der Order muss vom Typ ganzzahlig (datetime, int,..) sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
long  OrderGetInteger(
   ENUM_ORDER_PROPERTY_INTEGER  property_id        // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gestellt, die durch Referenz auf den letzten Parameter übertragen wird.

```
bool  OrderGetInteger(
   ENUM_ORDER_PROPERTY_INTEGER  property_id,      // Identifikator der Eigenschaft
   long&                        long_var          // hier nehmen wir den Wert der Eige
   );
```

Parameter

property_id

[in] Identifikator der Ordereigenschaft. Wert kann einer der Enumerationswerte ENUM_ORDER_PROPERTY_INTEGER sein.

long_var

[out] Variable vom Typ long, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ long.

Hinweis

Man darf gültige schwebende Aufträge nicht mit Positionen verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" des Client-Terminals dargestellt werden.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Um aktuelle Daten einer Order zu erhalten, ist es empfehlenswert, die Funktion OrderSelect() aufzurufen, bevor man die Daten abruft.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//--- in einer Schleife durch die Liste aller Orders des Kontos
   int total=OrdersTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Order-Tickets in der Liste über den Schleifenindex
      ulong ticket=OrderGetTicket(i);
      if(ticket==0)
         continue;

      //--- Auftragstyp abrufen und Kopfzeile für die Liste der „double"-Eigenschafter
      string type=OrderTypeDescription((ENUM_ORDER_TYPE)OrderGetInteger(ORDER_TYPE));
      PrintFormat("Integer properties of an active pending order %s #%I64u:", type, ti

      //--- alle ganzzahligen Eigenschaften des ausgewählten Auftrags unter der Übersc
      OrderPropertiesIntegerPrint(17);
     }
   /*
   Ergebnis:
   Integer properties of an active pending order Buy Limit #2812945317:
   Ticket:           2812945317
   Time setup:       2024.09.04 19:17:16
   Type:             Buy Limit
   State:            Placed
   Time expiration:  0
   Time done:        0
   Time setup msc:   2024.09.04 19:17:16.686
   Time done msc:    0
   Type filling:     Return
   Type time:        Time GTC
   Magic:            0
   Reason:           Client
   Position ID:      0
   Position By ID:   0
   */
  }
//+------------------------------------------------------------------+
//| Ganzzahlige Eigenschaften der gewählten Order ins Journal drucken|
//+------------------------------------------------------------------+
void OrderPropertiesIntegerPrint(const uint header_width=0)
  {
   uint   w=0;
   string header="";
   long   value=0;

//--- Auftragsticket im Journal anzeigen
   OrderPropertyPrint("Ticket:", header_width, ORDER_TICKET);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Zeitpunkt der Auftragserteilung im Journal anzeigen
   OrderPropertyPrint("Time setup:", header_width, ORDER_TIME_SETUP);

//--- Anzeige des Auftragstyps im Journal
   OrderPropertyPrint("Type:", header_width, ORDER_TYPE);

//--- Anzeige des Auftragsstatus im Journal
   OrderPropertyPrint("State:", header_width, ORDER_STATE);

//--- Ablaufzeit der Bestellung im Journal anzeigen
   OrderPropertyPrint("Time expiration:", header_width, ORDER_TIME_EXPIRATION);

//--- Ausführungs-/Ablaufzeit der Order im Journal anzeigen
   OrderPropertyPrint("Time done:", header_width, ORDER_TIME_DONE);

//--- Anzeige der Zeit der Auftragserteilung zur Ausführung in Millisekunden seit dem
   OrderPropertyPrint("Time setup msc:", header_width, ORDER_TIME_SETUP_MSC);

//--- Anzeige der Auftragsausführungs-/Ablaufzeit in Millisekunden seit 01.01.1970 im
   OrderPropertyPrint("Time done msc:", header_width, ORDER_TIME_DONE_MSC);

//--- verbliebener Ordertyp im Journal anzeigen
   OrderPropertyPrint("Type filling:", header_width, ORDER_TYPE_FILLING);

//--- Lebensdauer der Order im Journal anzeigen
   OrderPropertyPrint("Type time:", header_width, ORDER_TYPE_TIME);

//--- Anzeige der ID des EA, der eine Order platziert hat, im Journal
   OrderPropertyPrint("Magic:", header_width, ORDER_MAGIC);

//--- Anzeige des Grundes oder der Quelle der Order im Journal
   OrderPropertyPrint("Reason:", header_width, ORDER_REASON);

//--- Anzeige der ID der Position, die während der Ausführung der Order festgelegt wu
   OrderPropertyPrint("Position ID:", header_width, ORDER_POSITION_ID);

//--- Anzeige der entgegengesetzten Positions-ID für Orders vom Typ ORDER_TYPE_CLOSE_
   OrderPropertyPrint("Position By ID:", header_width, ORDER_POSITION_BY_ID);
  }
//+------------------------------------------------------------------+
//| Ausdruck der ganzzahligen Eigenschaftswert der Order im Journal  |
//+------------------------------------------------------------------+
void OrderPropertyPrint(const string header, uint header_width, ENUM_ORDER_PROPERTY_IN
  {
   string svalue="";
   long   lvalue=0;
   if(!OrderGetInteger(property, lvalue))
      PrintFormat("Cannot get property %s, error=%d", EnumToString(property), GetLastE
```

© 2000-2025, MetaQuotes Ltd.

```
   else
     {
      switch(property)
        {
         case ORDER_TICKET          :
         case ORDER_MAGIC           :
         case ORDER_POSITION_ID     :
         case ORDER_POSITION_BY_ID  :
           svalue=(string)lvalue;
           break;

         case ORDER_TIME_SETUP      :
         case ORDER_TIME_EXPIRATION :
         case ORDER_TIME_DONE       :
           svalue=(lvalue!=0 ? TimeToString((datetime)lvalue, TIME_DATE|TIME_MINUTES|T
           break;

         case ORDER_TIME_SETUP_MSC  :
         case ORDER_TIME_DONE_MSC   :
           svalue=(lvalue!=0 ? TimeMscToString(lvalue) : "0");
           break;

         case ORDER_TYPE            :
           svalue=OrderTypeDescription((ENUM_ORDER_TYPE)lvalue);
           break;
         case ORDER_STATE           :
           svalue=OrderStateDescription((ENUM_ORDER_STATE)lvalue);
           break;
         case ORDER_TYPE_FILLING    :
           svalue=OrderTypeFillingDescription((ENUM_ORDER_TYPE_FILLING)lvalue);
           break;
         case ORDER_TYPE_TIME       :
           svalue=OrderTypeTimeDescription((ENUM_ORDER_TYPE_TIME)lvalue);
           break;
         case ORDER_REASON          :
           svalue=OrderReasonDescription((ENUM_ORDER_REASON)lvalue);
           break;

         default                    :
           svalue="Unknown property";
           break;
        }

      //--- Wenn der Funktion eine Breite der Kopfzeile von Null übergeben wird, dann
      uint w=(header_width==0 ? header.Length()+1 : header_width);
      PrintFormat("%-*s%-s", w, header, svalue);
     }
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type: "+(string)type);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Order-Status                       |
//+------------------------------------------------------------------+
string OrderStateDescription(ENUM_ORDER_STATE state)
  {
   switch(state)
     {
      case ORDER_STATE_STARTED         :  return("Started");
      case ORDER_STATE_PLACED          :  return("Placed");
      case ORDER_STATE_CANCELED        :  return("Canceled");
      case ORDER_STATE_PARTIAL         :  return("Partial");
      case ORDER_STATE_FILLED          :  return("Filled");
      case ORDER_STATE_REJECTED        :  return("Rejected");
      case ORDER_STATE_EXPIRED         :  return("Expired");
      case ORDER_STATE_REQUEST_ADD     :  return("Request Add");
      case ORDER_STATE_REQUEST_MODIFY  :  return("Request Modify");
      case ORDER_STATE_REQUEST_CANCEL  :  return("Request Cancel");
      default                          :  return("Unknown state: "+(string)state);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung der Füll-Politik für das Order-Volumen |
//+------------------------------------------------------------------+
string OrderTypeFillingDescription(const ENUM_ORDER_TYPE_FILLING type)
  {
   switch(type)
     {
      case ORDER_FILLING_FOK     :  return("Fill or Kill");
      case ORDER_FILLING_IOC     :  return("Immediate or Cancel");
      case ORDER_FILLING_BOC     :  return("Book or Cancel");
      case ORDER_FILLING_RETURN  :  return("Return");
      default                    :  return("Unknown type filling: "+(string)type);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Ablaufdatums der Order             |
//+------------------------------------------------------------------+
string OrderTypeTimeDescription(const ENUM_ORDER_TYPE_TIME type)
   {
    switch(type)
      {
       case ORDER_TIME_GTC           :   return("Time GTC");
       case ORDER_TIME_DAY           :   return("Time Day");
       case ORDER_TIME_SPECIFIED     :   return("Time Specified");
       case ORDER_TIME_SPECIFIED_DAY :   return("Time Specified Day");
       default                       :  return("Unknown type time: "+(string)type);
      }
   }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Order-Grundes                      |
//+------------------------------------------------------------------+
string OrderReasonDescription(const ENUM_ORDER_REASON reason)
   {
    switch(reason)
      {
       case ORDER_REASON_CLIENT    :  return("Client");
       case ORDER_REASON_MOBILE    :  return("Mobile");
       case ORDER_REASON_WEB       :  return("Web");
       case ORDER_REASON_EXPERT    :  return("Expert");
       case ORDER_REASON_SL        :  return("Stop Loss");
       case ORDER_REASON_TP        :  return("Take Profit");
       case ORDER_REASON_SO        :  return("Stop Out");
       default                     :  return("Unknown reason: "+(string)reason);
      }
   }
//+------------------------------------------------------------------+
//| Rückgabe der Zeit in Millisekunden                               |
//+------------------------------------------------------------------+
string TimeMscToString(const long time_msc, int flags=TIME_DATE|TIME_MINUTES|TIME_SEC
   {
    return(TimeToString(time_msc/1000, flags) + "." + IntegerToString(time_msc %1000, 3
   }
```

Siehe auch

OrdersTotal(), OrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

OrderGetString

Gibt die angeforderte Ordereigenschaft zurück, die vorher durch die Funktion OrderGetTicket oder OrderSelect gewählt wurde. Eigenschaft der Order muss vom Typ string sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
string  OrderGetString(
   ENUM_ORDER_PROPERTY_STRING  property_id        // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gestellt, die durch Referenz auf den letzten Parameter übertragen wird.

```
bool  OrderGetString(
   ENUM_ORDER_PROPERTY_STRING  property_id,      // Identifikator der Eigenschaft
   string&                     string_var        // hier nehmen wir den Wert der Eiger
   );
```

Parameter

property_id

[in] Identifikator der Ordereigenschaft. Wert kann einer der Enumerationswerte ENUM_ORDER_PROPERTY_STRING sein.

string_var

[out] Variable vom Typ string, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ string.

Hinweis

Man darf gültige schwebende Aufträge nicht mit Positionen verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" des Client-Terminals dargestellt werden.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Handelsgeschäfte darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht miteinander verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen für ein Symbol existieren.

Für den Erhalt der aktuellen Daten der Order ist es empfehlenswert, die Funktion OrderSelect() aufzurufen, bevor man die aktualisierten Daten aufruft.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//--- in einer Schleife durch die Liste aller Orders des Kontos
   int total=OrdersTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Order-Tickets in der Liste über den Schleifenindex
      ulong ticket=OrderGetTicket(i);
      if(ticket==0)
         continue;

      //--- Auftragstyp abrufen und Kopfzeile für die Liste der String-Eigenschaften d
      string type=OrderTypeDescription((ENUM_ORDER_TYPE)OrderGetInteger(ORDER_TYPE));
      PrintFormat("String properties of an active pending order %s #%I64u:", type, tic

      //--- alle String-Eigenschaften des ausgewählten Auftrags unter der Kopfzeile dr
      OrderPropertiesStringPrint(13);
     }
   /*
   Ergebnis:
   String properties of an active pending order Sell Limit #2813781342:
   Comment:     Test OrderGetString
   Symbol:      EURUSD
   External ID:
   */
  }
//+------------------------------------------------------------------+
//| String-Eigenschaften der ausgewählten Order im Journal anzeigen  |
//+------------------------------------------------------------------+
void OrderPropertiesStringPrint(const uint header_width=0)
  {
//--- Kommentar im Journal anzeigen
   OrderPropertyPrint("Comment:", header_width, ORDER_COMMENT);

//--- Anzeige eines Symbols, für das der Auftrag im Journal ausgegeben wurde
   OrderPropertyPrint("Symbol:", header_width, ORDER_SYMBOL);

//--- Auftrags-ID in einem externen System im Journal anzeigen
   OrderPropertyPrint("External ID:", header_width, ORDER_EXTERNAL_ID);
  }
//+------------------------------------------------------------------+
//| Display the order string property value in the journal           |
//+------------------------------------------------------------------+
void OrderPropertyPrint(const string header, uint header_width, ENUM_ORDER_PROPERTY_ST
  {
   string value="";
   if(!OrderGetString(property, value))
      PrintFormat("Cannot get property %s, error=%d", EnumToString(property), GetLastE
```

© 2000-2025, MetaQuotes Ltd.

```
   else
     {
      //--- Wenn der Funktion eine Breite der Kopfzeile von Null übergeben wird, dann
      uint w=(header_width==0 ? header.Length()+1 : header_width);
      PrintFormat("%-*s%-s", w, header, value);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type: "+(string)type);
     }
  }
```

Siehe auch

OrdersTotal(), OrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistorySelect

Fragt die Historie von Deals und Orders für den angegebenen Zeitraum der Serverzeit ab.

```
bool  HistorySelect(
   datetime  from_date,      // von
   datetime  to_date         // bis
   );
```

Parameter

from_date

[in] Anfangsdatum der Abfrage.

to_date

[in] Enddatum der Abfrage.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Hinweis

Die Funktion HistorySelect() erstellt im mql5-Programm eine Liste von Orders und eine Liste von Deals für den weiteren Zugriff auf die Elemente der Liste mithilfe der entsprechenden Funktionen. Die Größe der Liste von Deals kann mithilfe der Funktion HistoryDealsTotal() festgestellt werden, die Größe der Liste von Orders in der Historie - mit HistoryOrdersTotal(). Am besten iteriert man über die Liste von Orders mit der Funktion HistoryOrderGetTicket(), für die Elemente der Liste von Deals ist die Funktion HistoryDealGetTicket() geeignet.

Nach der Verwendung der Funktion HistoryOrderSelect() wird die Liste der Orders in der Historie, die für das mql5-Programm zugänglich sind, gelöscht und mit der gefundene Order ausgefüllt, wenn die Suche der Order nach Ticket erfolgreich abgeschlossen wurde. Das Gleiche gilt für die Liste der Deals, die für das mql5-Programm zugänglich sind: die Liste wird mit der Funktion HistoryDealSelect() gelöscht und erneut ausgefüllt, wenn ein Deal nach Ticketnummer erfolgreich gefunden wurde.

Beispiel:

```
void OnStart()
 {
   color BuyColor =clrBlue;
   color SellColor=clrRed;
//--- request trade history
   HistorySelect(0,TimeCurrent());
//--- create objects
   string   name;
   uint     total=HistoryDealsTotal();
   ulong    ticket=0;
   double   price;
   double   profit;
   datetime time;
   string   symbol;
```

© 2000-2025, MetaQuotes Ltd.

```
   long     type;
   long     entry;
//--- for all deals
   for(uint i=0;i<total;i++)
    {
      //--- try to get deals ticket
      if((ticket=HistoryDealGetTicket(i))>0)
        {
         //--- get deals properties
         price =HistoryDealGetDouble(ticket,DEAL_PRICE);
         time  =(datetime)HistoryDealGetInteger(ticket,DEAL_TIME);
         symbol=HistoryDealGetString(ticket,DEAL_SYMBOL);
         type  =HistoryDealGetInteger(ticket,DEAL_TYPE);
         entry =HistoryDealGetInteger(ticket,DEAL_ENTRY);
         profit=HistoryDealGetDouble(ticket,DEAL_PROFIT);
         //--- only for current symbol
         if(price && time && symbol==Symbol())
          {
            //--- create price object
            name="TradeHistory_Deal_"+string(ticket);
            if(entry) ObjectCreate(0,name,OBJ_ARROW_RIGHT_PRICE,0,time,price,0,0);
            else      ObjectCreate(0,name,OBJ_ARROW_LEFT_PRICE,0,time,price,0,0);
            //--- set object properties
            ObjectSetInteger(0,name,OBJPROP_SELECTABLE,0);
            ObjectSetInteger(0,name,OBJPROP_BACK,0);
            ObjectSetInteger(0,name,OBJPROP_COLOR,type?BuyColor:SellColor);
            if(profit!=0) ObjectSetString(0,name,OBJPROP_TEXT,"Profit: "+string(profit
           }
        }
     }
//--- apply on chart
   ChartRedraw();
  }
```

Siehe auch

HistoryOrderSelect(), HistoryDealSelect()

© 2000-2025, MetaQuotes Ltd.

HistorySelectByPosition

Fordert die Historie der Deals und der Order mit dem angegebenen Identifikator der Position an.

```
bool  HistorySelectByPosition(
   ulong   position_id     // Identifikator der Position - POSITION_IDENTIFIER
   );
```

Parameter

position_id

[in] Der Identifikator der Position, der für jeden ausgeführten Auftrag und auf jede Transaktion festgelegt ist.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Man darf Aufträge (order) aus der Historie und gültige schwebende Aufträge (pending order) nicht miteinander verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Die Liste der Order, die verändert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Historie"in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Die Funktion HistorySelectByPosition() erzeugt im mql5-Programm die Liste der Order und die Liste der Deals mit dem angegebenen Identifikator der Position für weitere Aufrufe der Elemente der Liste mittels der entsprechenden Funktionen. Die Größe der Liste der Deals kann man mittels der Funktion HistoryDealsTotal() erfahren, die Größe der Liste der Order kann man mittels HistoryOrdersTotal() erfahren. Die Elemente aus der Liste der Order ist es besser, sie mit der Funktion HistoryOrderGetTicket() durchzusehen, für die Elemente der Liste der Deals verwenden Sie am besten die Funktion HistoryDealGetTicket().

Nach der Verwendung der Funktion HistoryOrderSelect() wird die Liste der Order in der Historie, die für mql5-Programm zugänglich ist, gelöscht und von einer erneut gefundenen Order ausgefüllt, wenn die Suche der Order nach Ticket erfolgreich beendet hat. Dasselbe bezieht sich auf die Liste der Deals, die für mql5-Programm zugänglich ist - sie wird von der Funktion HistoryDealSelect() gelöscht und erneut ausgefüllt, wenn der Deal nach Ticketnummer erfolgreich erhalten wurde.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   long  pos_id_array[];          // Array für die IDs der Positionen

//--- die ganze Historie anfordern
   if(!HistorySelect(0, TimeCurrent()))
     {
      Print("HistorySelect() failed. Error ", GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
     }

//--- alle IDs der Positionen nur aus schwebenden Aufträgen im Array sammeln
   int total=HistoryOrdersTotal();
   for(int i=0; i<total; i++)
     {
      ulong ticket=HistoryOrderGetTicket(i);
      if(ticket==0)
         continue;
      ENUM_ORDER_TYPE type=(ENUM_ORDER_TYPE)HistoryOrderGetInteger(ticket, ORDER_TYPE)
      long pos_id=HistoryOrderGetInteger(ticket, ORDER_POSITION_ID);
      if(type<=ORDER_TYPE_SELL || pos_id==0)
         continue;

      int size=ArraySize(pos_id_array);
      if(ArrayResize(pos_id_array, size+1)==size+1)
         pos_id_array[size]=pos_id;
     }

//--- nach Liste der Positions-IDs im Array
   total=ArraySize(pos_id_array);
   for(int i=0; i<total; i++)
     {
      //--- Druckt die Kopfzeile sowie die Liste der Positionsaufträge und die Deallis
      long position_id=pos_id_array[i];
      Print("List of orders and deals for position with ID: ", position_id);
      HistorySelectByPositionProcess(position_id);
     }
   /*
   Ergebnis:
   List of orders and deals for position with ID: 1819629924
     [0] Order Sell Limit #1819629924
     [1] Order Buy #1819633194
     [0] Entry In Deal Sell #1794972472
     [1] Entry Out Deal Buy #1794975589
    List of orders and deals for position with ID: 1841753970
     [0] Order Sell Stop #1841753970
     [1] Order Buy #1842322160
     [0] Entry In Deal Sell #1817242142
     [1] Entry Out Deal Buy #1817765341
   */
  }
//+------------------------------------------------------------------+
//| Historie der Aufträge und Deals nach Positions-ID auswählen und  |
//| Ausdruck der Liste der Aufträge und Deals der Position im Journal|
//+------------------------------------------------------------------+
bool HistorySelectByPositionProcess(const long position_id)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Anfrage nach der Historie der Deals und Aufträge mit der angegebenen Positions-I
   if(!HistorySelectByPosition(position_id))
     {
      PrintFormat("HistorySelectByPosition(%I64d) failed. Error %d", position_id, GetI
      return(false);
     }

//--- Liste der Positionen drucken
   int orders_total=HistoryOrdersTotal();
   for(int i=0; i<orders_total; i++)
     {
      ulong ticket=HistoryOrderGetTicket(i);
      if(ticket==0)
         continue;
      ENUM_ORDER_TYPE order_type=(ENUM_ORDER_TYPE)HistoryOrderGetInteger(ticket, ORDEI
      PrintFormat("  [%d] Order %s #%I64u", i, OrderTypeDescription(order_type), ticke
     }

//--- eine Liste der Positionen im Journal ausdrucken
   int deals_total =HistoryDealsTotal();
   for(int i=0; i<deals_total; i++)
     {
      ulong ticket=HistoryDealGetTicket(i);
      if(ticket==0)
         continue;
      ENUM_DEAL_ENTRY deal_entry=(ENUM_DEAL_ENTRY)HistoryDealGetInteger(ticket, DEAL_E
      ENUM_DEAL_TYPE  deal_type= (ENUM_DEAL_TYPE)HistoryDealGetInteger(ticket, DEAL_TY
      if(deal_type!=DEAL_TYPE_BUY && deal_type!=DEAL_TYPE_SELL)
         continue;
      PrintFormat("  [%d] Entry %s Deal %s #%I64u", i, DealEntryDescription(deal_entry
     }
   return(true);
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type: "+(string)type);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Deals-Typs der Position            |
//+------------------------------------------------------------------+
string DealTypeDescription(const ENUM_DEAL_TYPE type)
   {
    switch(type)
      {
       //--- nur die Beschreibung der Kauf- und Verkaufsgeschäfte zurückgeben,
       //--- da alle anderen Typen nicht auf die Position zutreffen
       case DEAL_TYPE_BUY    :  return("Buy");
       case DEAL_TYPE_SELL   :  return("Sell");
       default               :  return("Unknown deal type: "+(string)type);
      }
   }
//+------------------------------------------------------------------+
//| Rückgabe der Änderungsmethode der Position                       |
//+------------------------------------------------------------------+
string DealEntryDescription(const ENUM_DEAL_ENTRY entry)
   {
    switch(entry)
      {
       case DEAL_ENTRY_IN      :  return("In");
       case DEAL_ENTRY_OUT     :  return("Out");
       case DEAL_ENTRY_INOUT   :  return("InOut");
       case DEAL_ENTRY_OUT_BY  :  return("Out by");
       case DEAL_ENTRY_STATE   :  return("Status record");
       default                 :  return("Unknown deal entry: "+(string)entry);
      }
   }
```

Siehe auch

HistorySelect(), HistoryOrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryOrderSelect

Wählt in der Historie der Order für den weiteren Zugang durch entsprechenden Funktionen aus. Gibt true zurück, wenn die Durchführung der Funktion erfolgreich beendet wird. Gibt false zurück, wenn die Durchführung der Funktion erfolglos beendet wird. Für den Erhalt der fehlerbezogenen Information rufen Sie die Funktion GetLastError() auf.

```
bool  HistoryOrderSelect(
   ulong  ticket,      // Orderticket
   );
```

Parameter

ticket

[in] Orderticket

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Man darf eine Order aus der Historie und gültige schwebende Aufträge (pending orders) nicht miteinander verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Die Liste der Order, die annulliert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Historie" in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Die Funktion HistoryOrderSelect() löscht im mql5-Programm die Liste der Order aus der Historie, die für Aufrufe zugänglich sind, und kopiert darin die einzige Order, wenn die Ausführung HistoryOrderSelect() erfolgreich beendet hat. Wenn es notwendig ist, alle Deals durchzusehen, die von der Funktion HistorySelect() gewählt wurden, ist es besser die Funktion HistoryOrderGetTicket() zu verwenden.

Beispiel:

```
#define   TICKET    1819621374   // Beispiel einer Ticketnummer einer bekannten Order
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wähle eine historische Order anhand des in TICKET angegebenen Ticketnummer
   if(!HistoryOrderSelect(TICKET))
     {
      PrintFormat("HistoryOrderSelect(%I64u) failed. Error %d", TICKET, GetLastError()
      return;
     }

//--- wenn eine Order erfolgreich ausgewählt wurde, ruf deren Daten ab und druck die C
   ENUM_ORDER_TYPE   order_type  = (ENUM_ORDER_TYPE)HistoryOrderGetInteger(TICKET, ORI
   ENUM_ORDER_STATE  order_state = (ENUM_ORDER_STATE)HistoryOrderGetInteger(TICKET, OR
   ENUM_ORDER_REASON order_reason= (ENUM_ORDER_REASON)HistoryOrderGetInteger(TICKET, C
```

© 2000-2025, MetaQuotes Ltd.

```
   long              order_time  = HistoryOrderGetInteger(TICKET, ORDER_TIME_SETUP_MSC
   string            order_symbol=HistoryOrderGetString(TICKET, ORDER_SYMBOL);
   double            order_vol_init=HistoryOrderGetDouble(TICKET, ORDER_VOLUME_INITIAL
   double            order_vol_curr=HistoryOrderGetDouble(TICKET, ORDER_VOLUME_CURRENT
   PrintFormat("%s Order %.2f/%s %s #%I64u %s by %s at %s",
               order_symbol, order_vol_init, (order_vol_curr>0 ? DoubleToString(order_
               OrderTypeDescription(order_type), TICKET, OrderStateDescription(order_s
               OrderReasonDescription(order_reason), TimeMscToString(order_time));
   /*
   Ergebnis für verschiedene angegebene Tickets:
   EURUSD Order 0.50/0.50 Buy Limit #2812894647 Canceled by Client at 2024.09.04 19:02
   EURUSD Order 0.10/0 Sell #1753011743 Filled by Take Profit at 2023.06.12 17:04:20.3
   GBPUSD Order 0.10/0 Buy #1819621374 Filled by Client at 2023.07.24 06:16:25.746
   */
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type: "+(string)type);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Order-Status                       |
//+------------------------------------------------------------------+
string OrderStateDescription(ENUM_ORDER_STATE state)
  {
   switch(state)
     {
      case ORDER_STATE_STARTED         :  return("Started");
      case ORDER_STATE_PLACED          :  return("Placed");
      case ORDER_STATE_CANCELED        :  return("Canceled");
      case ORDER_STATE_PARTIAL         :  return("Partial");
      case ORDER_STATE_FILLED          :  return("Filled");
      case ORDER_STATE_REJECTED        :  return("Rejected");
      case ORDER_STATE_EXPIRED         :  return("Expired");
      case ORDER_STATE_REQUEST_ADD     :  return("Request Add");
      case ORDER_STATE_REQUEST_MODIFY  :  return("Request Modify");
```

© 2000-2025, MetaQuotes Ltd.

```
      case ORDER_STATE_REQUEST_CANCEL  :  return("Request Cancel");
      default                          :  return("Unknown state: "+(string)state);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Order-Grundes                      |
//+------------------------------------------------------------------+
string OrderReasonDescription(const ENUM_ORDER_REASON reason)
  {
   switch(reason)
     {
      case ORDER_REASON_CLIENT    :  return("Client");
      case ORDER_REASON_MOBILE    :  return("Mobile");
      case ORDER_REASON_WEB       :  return("Web");
      case ORDER_REASON_EXPERT    :  return("Expert");
      case ORDER_REASON_SL        :  return("Stop Loss");
      case ORDER_REASON_TP        :  return("Take Profit");
      case ORDER_REASON_SO        :  return("Stop Out");
      default                     :  return("Unknown reason: "+(string)reason);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Zeit in Millisekunden                               |
//+------------------------------------------------------------------+
string TimeMscToString(const long time_msc, int flags=TIME_DATE|TIME_MINUTES|TIME_SEC
  {
   return(TimeToString(time_msc/1000, flags) + "." + IntegerToString(time_msc %1000, 3
  }
```

Siehe auch

HistorySelect(), HistoryOrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryOrdersTotal

Gibt die Anzahl der Order in der Historie zurück. Vor dem Aufruf der Funktion HistoryOrdersTotal(), müssen Sie die Historie der Transaktionen und Aufträge mit der HistorySelect() oder HistorySelectByPosition() abrufen.

```
int  HistoryOrdersTotal();
```

Rückgabewert

Wert vom Typ int.

Hinweis

Man darf Aufträge (order) aus der Historie und gültige schwebende Aufträge (pending order) nicht miteinander verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Die Liste der Order, die annulliert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Historie" in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anfordern des gesamten vorhandenen Historie des Kontos
   if(!HistorySelect(0, TimeCurrent()))
     {
      Print("HistorySelect() failed. Error ", GetLastError());
      return;
     }

//--- Anzahl der Aufträge in der Liste abrufen und im Journal ausdrucken
   int total=HistoryOrdersTotal();
   Print("Number of historical orders on the account: ", total);
   /*
   Ergebnis:
   Number of historical orders on the account: 496
   */
  }
```

Siehe auch

HistorySelect(), HistoryOrderSelect(), HistoryOrderGetTicket(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryOrderGetTicket

Gibt Ticket der entsprechenden Order in der Geschichte zurück. Vor dem Aufruf von HistorySelect() muss man die Historie der Deals und Orders mit der Funktion HistorySelectByPosition() oder HistorySelectByPosition() abrufen.

```
ulong  HistoryOrderGetTicket(
   int  index      // Nummer in der Orderliste
   );
```

Parameter

index

[in] Ordernummer in der Orderliste.

Rückgabewert

Wert des Typs ulong. Im Fall der fehlerhaften Ausführung gibt 0 zurück.

Hinweis

Man muss nicht Ordern aus der Handelsgeschichte und geltende Warteordern verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Die Liste der Ordern, die annulliert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Geschichte" in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Beispiel:

```
void OnStart()
  {
   datetime from=0;
   datetime to=TimeCurrent();
//--- fordern wir die ganze Geschichte an
   HistorySelect(from,to);
//--- Variablen für Erhaltung der Werte aus Ordereigenschaften
   ulong    ticket;
   double   open_price;
   double   initial_volume;
   datetime time_setup;
   datetime time_done;
   string   symbol;
   string   type;
   long     order_magic;
   long     positionID;
//--- Anzahl der laufenden suspendierten Order
   uint     total=HistoryOrdersTotal();
//--- Gehen wir im Zyklus durch alle Order
   for(uint i=0;i<total;i++)
     {
      //--- Erhalten wir Ticket der Order nach ihrer Position in der Liste
      if((ticket=HistoryOrderGetTicket(i))>0)
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- Erhalten wir Eigenschaften der Order
         open_price=       HistoryOrderGetDouble(ticket,ORDER_PRICE_OPEN);
         time_setup=       (datetime)HistoryOrderGetInteger(ticket,ORDER_TIME_SETUP);
         time_done=        (datetime)HistoryOrderGetInteger(ticket,ORDER_TIME_DONE);
         symbol=           HistoryOrderGetString(ticket,ORDER_SYMBOL);
         order_magic=      HistoryOrderGetInteger(ticket,ORDER_MAGIC);
         positionID =      HistoryOrderGetInteger(ticket,ORDER_POSITION_ID);
         initial_volume=   HistoryOrderGetDouble(ticket,ORDER_VOLUME_INITIAL);
         type=GetOrderType(HistoryOrderGetInteger(ticket,ORDER_TYPE));
         //--- bereiten wir die information über Order vor und geben sie aus
         printf("#ticket %d %s %G %s at %G was set up at %s => done at %s, pos ID=%d",
                ticket,                   // Orderticket
                type,                     // Typ
                initial_volume,           // das gestellte Volumen
                symbol,                   // Symbol, für das gestellt wird
                open_price,               // der angegeben Eroeffnungspreis
                TimeToString(time_setup),// Zeit der Ordereinstellung
                TimeToString(time_done), // Zeit der Durchführung oder der Entfernung
                positionID                // ID der Position, in die das Ordergeschaeft
                );
        }
     }
//---
  }
//+------------------------------------------------------------------+
//| Gibt die Zeilenbenennung des Ordertyps zurück                    |
//+------------------------------------------------------------------+
string GetOrderType(long type)
  {
   string str_type="unknown operation";
   switch(type)
     {
      case (ORDER_TYPE_BUY):             return("buy");
      case (ORDER_TYPE_SELL):            return("sell");
      case (ORDER_TYPE_BUY_LIMIT):       return("buy limit");
      case (ORDER_TYPE_SELL_LIMIT):      return("sell limit");
      case (ORDER_TYPE_BUY_STOP):        return("buy stop");
      case (ORDER_TYPE_SELL_STOP):       return("sell stop");
      case (ORDER_TYPE_BUY_STOP_LIMIT): return("buy stop limit");
      case (ORDER_TYPE_SELL_STOP_LIMIT):return("sell stop limit");
     }
   return(str_type);
  }
```

Sehen Sie auch

HistorySelect(), HistoryOrdersTotal(), HistoryOrderSelect(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryOrderGetDouble

Gibt die angeforderte Eigenschaft der Order zurück. Eigenschaft der Order muss vom Typ double sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
double  HistoryOrderGetDouble(
   ulong                      ticket_number,     // Ticket
   ENUM_ORDER_PROPERTY_DOUBLE  property_id       // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zrück, abhängig von der erfolgreichen Durchführung der Funktion. Im Erfolgsfall wird Wert der Eigenschaft der Empfangsvariablen zugewiesen, die durch Referenz als letzter Parameter übertragen wird.

```
bool  HistoryOrderGetDouble(
   ulong                      ticket_number,     // Ticket
   ENUM_ORDER_PROPERTY_DOUBLE  property_id,       // Identifikator der Eigenschaft
   double&                    double_var         // hier nehmen wir den Wert der Eige
   );
```

Parameter

ticket_number

[in] Orderticket.

property_id

[in] Identifikator der Ordereigenschaft. Wert kann einer der Enumerationswerte ENUM_ORDER_PROPERTY_DOUBLE sein.

double_var

[out] Variable vom Typ double, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ double.

Hinweis

Man darf Aufträge (order) aus der Historie und gültige schwebende Aufträge (pending order) nicht miteinander verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Die Liste der schwebenden Aufträge, die annulliert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Historie" in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Beispiel:

```
void OnStart()
  {
   datetime from=0;
   datetime to=TimeCurrent();
//--- die ganze Historie anfordern
   HistorySelect(from,to);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Variablen für die Rückgabe der Werte aus Auftragseigenschaften
   ulong    ticket;
   double   open_price;
   double   initial_volume;
   datetime time_setup;
   datetime time_done;
   string   symbol;
   string   type;
   long     order_magic;
   long     positionID;
//--- Anzahl der aktuell schwebenden Aufträge
   uint     total=HistoryOrdersTotal();
//--- in einer Schleife durch alle gehen
   for(uint i=0;i<total;i++)
     {
      //--- Ticket des Auftrags nach seiner Listenposition zurückgeben
      if((ticket=HistoryOrderGetTicket(i))>0)
        {
         //--- Rückgabe der Auftragseigenschaften
         open_price    =HistoryOrderGetDouble(ticket,ORDER_PRICE_OPEN);
         time_setup    =(datetime)HistoryOrderGetInteger(ticket,ORDER_TIME_SETUP);
         time_done     =(datetime)HistoryOrderGetInteger(ticket,ORDER_TIME_DONE);
         symbol        =HistoryOrderGetString(ticket,ORDER_SYMBOL);
         order_magic   =HistoryOrderGetInteger(ticket,ORDER_MAGIC);
         positionID    =HistoryOrderGetInteger(ticket,ORDER_POSITION_ID);
         initial_volume=HistoryOrderGetDouble(ticket,ORDER_VOLUME_INITIAL);
         type          =GetOrderType(HistoryOrderGetInteger(ticket,ORDER_TYPE));
         //--- Informationen des Auftrags vorbereiten und anzeigen
         printf("#ticket %d %s %G %s at %G was set up at %s => done at %s, pos ID=%d",
                ticket,                   // Order-Ticket
                type,                     // Typ
                initial_volume,           // platziertes Volumen
                symbol,                   // Symbol
                open_price,               // angegebener Eröffnungspreis
                TimeToString(time_setup),// Platzierungszeit der Order
                TimeToString(time_done), // Ausführungs- oder Löschzeit der Order
                positionID                // ID einer Position, auf die sich der Deal
                );
        }
     }
//---
  }
//+------------------------------------------------------------------+
//| Rückgabe vom Namen des Auftragstyps zurück                       |
//+------------------------------------------------------------------+
string GetOrderType(long type)
  {
   string str_type="unknown operation";
   switch(type)
```

© 2000-2025, MetaQuotes Ltd.

```
    {
     case (ORDER_TYPE_BUY):             return("buy");
     case (ORDER_TYPE_SELL):            return("sell");
     case (ORDER_TYPE_BUY_LIMIT):       return("buy limit");
     case (ORDER_TYPE_SELL_LIMIT):      return("sell limit");
     case (ORDER_TYPE_BUY_STOP):        return("buy stop");
     case (ORDER_TYPE_SELL_STOP):       return("sell stop");
     case (ORDER_TYPE_BUY_STOP_LIMIT): return("buy stop limit");
     case (ORDER_TYPE_SELL_STOP_LIMIT):return("sell stop limit");
    }
  return(str_type);
 }
```

Siehe auch

HistorySelect(), HistoryOrdersTotal(), HistoryOrderSelect(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryOrderGetInteger

Gibt die angeforderte Eigenschaft der Order. Eigenschaft der Order muss des Typs datetime, int sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft sofort zurück.

```
long  HistoryOrderGetInteger(
   ulong                      ticket_number,     // Ticket
   ENUM_ORDER_PROPERTY_INTEGER  property_id       // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird Wert der Eigenschaft in die Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  HistoryOrderGetInteger(
   ulong                      ticket_number,     // Ticket
   ENUM_ORDER_PROPERTY_INTEGER  property_id,       // Identifikator der Eigenschaft
   long&                      long_var           // hier nehmen wir den Wert der Eig
   );
```

Parameter

ticket_number

[in] Ticket der Order.

property_id

[in] Identifikator der Ordereigenschaft. Wert kann einer der Enumerationswerte ENUM_ORDER_PROPERTY_INTEGER.

long_var

[out] Variable des Typs long, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs long.

Hinweis

Man muss nicht Ordern aus der Handelsgeschichte und geltende Warteordern verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Liste der Ordern, die annulliert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Geschichte" in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Trade function                                                   |
//+------------------------------------------------------------------+
void OnTrade()
  {
//--- Ticket des letzten Order aus der Handelsgeschichte für eine Woche erhalten
   ulong last_order=GetLastOrderTicket();
   if(HistoryOrderSelect(last_order))
     {
      //--- Ordersetzungzeit in Millisekunden seit 01.01.1970
      long time_setup_msc=HistoryOrderGetInteger(last_order,ORDER_TIME_SETUP_MSC);
      PrintFormat("Order #%d ORDER_TIME_SETUP_MSC=%i64 => %s",
                  last_order,time_setup_msc,TimeToString(time_setup_msc/1000));
      //--- Orderausführung- oder Löschungzeit in Millisekunden seit 01.01.1970
      long  time_done_msc=HistoryOrderGetInteger(last_order,ORDER_TIME_DONE_MSC);
      PrintFormat("Order #%d ORDER_TIME_DONE_MSC=%i64 => %s",
                  last_order,time_done_msc,TimeToString(time_done_msc/1000));
     }
   else // Den Fehler melden
      PrintFormat("HistoryOrderSelect() failed for #%d. Eror code=%d",
                  last_order,GetLastError());

//---
  }
//+------------------------------------------------------------------+
//| Gibt den Ticket des letzten Orders in der Geschichte oder -1     |
//+------------------------------------------------------------------+
ulong GetLastOrderTicket()
  {
//--- Geschichte für die letzte 7 Tagen anfragen
   if(!GetTradeHistory(7))
     {
      //--- Nachricht über fehlgeschlagenen Aufruf und Rückgabe -1
      Print(__FUNCTION__," HistorySelect() hat false zurückgegeben");
      return -1;
     }
//---
   ulong first_order,last_order,orders=HistoryOrdersTotal();
//--- wenn es Ordern gibt, anfangen wir mit ihnen zu arbeiten
   if(orders>0)
     {
      Print("Orders = ",orders);
      first_order=HistoryOrderGetTicket(0);
      PrintFormat("first_order = %d",first_order);
      if(orders>1)
        {
         last_order=HistoryOrderGetTicket((int)orders-1);
         PrintFormat("last_order = %d",last_order);
         return last_order;
        }
      return first_order;
     }
//--- keine Orders waren gefunden, Rückgabe -1
   return -1;
  }
//+------------------------------------------------------------------+
//| Geschichte für letzte Tage anfragen, gibt false bei Fehler zurück|
//+------------------------------------------------------------------+
bool GetTradeHistory(int days)
  {
//--- definieren wir Wochenperiode, um eine Handelsgeschichte zu beantragen
```

© 2000-2025, MetaQuotes Ltd.

```
   datetime to=TimeCurrent();
   datetime from=to-days*PeriodSeconds(PERIOD_D1);
   ResetLastError();
//--- Machen wir eine Anfrage und überprüfen wir die Ergebnisse
   if(!HistorySelect(from,to))
     {
      Print(__FUNCTION__," HistorySelect=false. Error code=",GetLastError());
      return false;
     }
//--- Geschichte erfolgreich empfangen
   return true;
  }
```

Sehen Sie auch

HistorySelect(), HistoryOrdersTotal(), HistoryOrderSelect(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryOrderGetString

Gibt die angeforderte Eigenschaft der Order zurück. Die Ordereigenschaft muss vom Typ string sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
string  HistoryOrderGetString(
   ulong                      ticket_number,     // Ticket
   ENUM_ORDER_PROPERTY_STRING  property_id       // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig von der erfolgreichen Durchführung der Funktion. Im Erfolgsfall wird der Wert der Eigenschaft der Empfangsvariablen zugewiesen, die durch Referenz als letzter Parameter übertragen wird.

```
bool  HistoryOrderGetString(
   ulong                      ticket_number,     // Ticket
   ENUM_ORDER_PROPERTY_STRING  property_id,      // Identifikator der Eigenschaft
   string&                    string_var         // hier nehmen wir den Wert der Eige
```

Parameter

ticket_number

[in] Orderticket.

property_id

[in] Identifikator der Ordereigenschaft. Wert kann einer der Enumerationswerte ENUM_ORDER_PROPERTY_STRING sein.

string_var

[out] Variable vom Typ string, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ string.

Hinweis

Man darf Aufträge (order) aus der Historie und gültige schwebende Aufträge (pending order) nicht miteinander verwechseln, die in der Registerkarte "Handel" in der Werkzeugleiste "Instrumente" dargestellt werden. Die Liste der schwebenden Aufträge, die annulliert wurden oder zur Ausführung der Handelsoperation geführt haben, kann man in der Registerkarte "Historie" in der Werkzeugleiste "Instrumente" des Client-Terminals sehen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Handels- und Auftragshistorie anfordern
   if(!HistorySelect(0, TimeCurrent()))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("HistorySelect() failed. Error ", GetLastError());
       return;
      }

//--- in einer Schleife durch die Liste aller historischen Aufträge gehen
   int total=HistoryOrdersTotal();
   for(int i=0; i<total; i++)
     {
      //--- Abrufen des Order-Tickets in der Liste über den Schleifenindex
      ulong ticket=HistoryOrderGetTicket(i);
      if(ticket==0)
         continue;

      //--- Auftragstyp abrufen und Kopfzeile für die Liste der String-Eigenschaften d
      string type=OrderTypeDescription((ENUM_ORDER_TYPE)HistoryOrderGetInteger(ticket,
      PrintFormat("String properties of an history order %s #%I64u:", type, ticket);

      //--- alle String-Eigenschaften des ausgewählten Auftrags unter der Kopfzeile dr
      HistoryOrderPropertiesStringPrint(ticket, 16);
     }
   /*
   Ergebnis:
   String properties of an history order Buy #2646074112:
   Comment:         [tp 1.09137]
   Symbol:          EURUSD
   External ID:
   String properties of an history order Buy #2646131906:
   Comment:
   Symbol:          EURUSD
   External ID:
   */
  }
//+------------------------------------------------------------------+
//| String-Eigenschaften der                                         |
//| ausgewählten historischen Aufträge im Journal                    |
//+------------------------------------------------------------------+
void HistoryOrderPropertiesStringPrint(const long ticket, const uint header_width=0)
  {
   uint   w=0;
   string header="";
   string value="";

//--- Definieren der Kopftextes und der Breite des Textfeldes
//--- Wenn der Funktion eine Breite von Null übergeben wird, dann ist die Breite die G
   header="Comment:";
   w=(header_width==0 ? header.Length()+1 : header_width);
//--- Kommentar mit der angegebenen Kopfzeilenbreite abrufen und im Journal anzeigen
   if(!HistoryOrderGetString(ticket, ORDER_COMMENT, value))
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
   PrintFormat("%-*s%-s", w, header, value);

//--- Anzeige eines Symbols, für das der Auftrag im Journal ausgegeben wurde
   header="Symbol:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryOrderGetString(ticket, ORDER_SYMBOL, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);

//--- Auftrags-ID in einem externen System im Journal anzeigen
   header="External ID:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryOrderGetString(ticket, ORDER_EXTERNAL_ID, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Auftragstyps                       |
//+------------------------------------------------------------------+
string OrderTypeDescription(const ENUM_ORDER_TYPE type)
  {
   switch(type)
     {
      case ORDER_TYPE_BUY              :  return("Buy");
      case ORDER_TYPE_SELL             :  return("Sell");
      case ORDER_TYPE_BUY_LIMIT        :  return("Buy Limit");
      case ORDER_TYPE_SELL_LIMIT       :  return("Sell Limit");
      case ORDER_TYPE_BUY_STOP         :  return("Buy Stop");
      case ORDER_TYPE_SELL_STOP        :  return("Sell Stop");
      case ORDER_TYPE_BUY_STOP_LIMIT   :  return("Buy Stop Limit");
      case ORDER_TYPE_SELL_STOP_LIMIT  :  return("Sell Stop Limit");
      default                          :  return("Unknown order type: "+(string)type);
     }
  }
```

Siehe auch

HistorySelect(), HistoryOrdersTotal(), HistoryOrderSelect(), Ordereigenschaften

© 2000-2025, MetaQuotes Ltd.

HistoryDealSelect

Wählt den Deal in der Historie für den weiteren Zugang dazu durch die entsprechenden Funktionen. Gibt true beim erfolgreichen Beenden der Funktion zurück. Gibt false beim erfolglosen Beenden der Funktion zurück. Für den Erhalt der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

```
bool  HistoryDealSelect(
   ulong  ticket,     // Dealsticket
   );
```

Parameter

ticket

[in] Dealsticket

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Man darf Order, Deal und Position nicht miteinander verwechseln. Jeder Deal ist das Ergebnis der Ausführung eines Auftrags (order), jede Position ist das Abschlussergebnis eines oder mehrerer Deals.

Die Funktion HistoryDealSelect() löscht im mql5-Programm die Liste der Deals, die für Aufrufe zugänglich sind, und kopiert, bei erfolgreicher Ausführung von HistoryDealSelect(), den einzigen, ausgewählten Deal. Wenn es notwendig ist, alle Deals durchzusehen, die von der Funktion HistorySelect() gewählt wurden, ist es besser die Funktion HistoryDealGetTicket() zu verwenden.

Beispiel:

```
#define   TICKET    2620919264   // Ticket eines beliebigen, bekannten Deals, zum Beis

long      ExtTicket=TICKET;      // weise das angegebene Ticket der Variablen aus der
                                 // oder verwalte die Deals in der Ereignisbehandlung
//+------------------------------------------------------------------+
//| Ereignisbehandlung TradeTransaction im Experten                  |
//+------------------------------------------------------------------+
void OnTradeTransaction(const MqlTradeTransaction& trans,
                        const MqlTradeRequest& request,
                        const MqlTradeResult& result)
  {
   //--- wenn eine Transaktion einen Deal der Historie hinzufügt
   if(trans.type==TRADE_TRANSACTION_DEAL_ADD)
     {
      //--- einen Deal per Ticket auswählen, seine Daten abrufen und die Dealbeschreib
      HistoryDealSelectProcess(trans.deal);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wähle einen Deal per Ticket aus, rufe seine Daten ab und drucke die Dealbeschrei
   HistoryDealSelectProcess(ExtTicket);
   /*
   Ergebnis:
   (Position ID #2645974677) EURUSD Deal Out 0.10 Buy #2620919264 by order #264602896
   */
  }
//+------------------------------------------------------------------+
//| Einen Deal per Ticket auswählen und die Daten im Journal drucken |
//+------------------------------------------------------------------+
void HistoryDealSelectProcess(const ulong deal_ticket)
  {
//--- wähle einen historischen Deal anhand des in deal_ticket angegebenen Tickets
   ResetLastError();
   if(!HistoryDealSelect(deal_ticket))
     {
      PrintFormat("HistoryDealSelect(%I64u) failed. Error %d", deal_ticket, GetLastErr
      return;
     }

//--- wenn ein Deal erfolgreich ausgewählt wurde, ruf seine Daten ab und druck die Dea
   ENUM_DEAL_TYPE    deal_type  = (ENUM_DEAL_TYPE)HistoryDealGetInteger(ExtTicket, DEA
   ENUM_DEAL_ENTRY   deal_entry = (ENUM_DEAL_ENTRY)HistoryDealGetInteger(ExtTicket, DE
   ENUM_DEAL_REASON  deal_reason= (ENUM_DEAL_REASON)HistoryDealGetInteger(ExtTicket, I
   long              deal_time  = HistoryDealGetInteger(ExtTicket, DEAL_TIME_MSC);
   long              deal_order = HistoryDealGetInteger(ExtTicket, DEAL_ORDER);
   long              deal_pos_id= HistoryDealGetInteger(ExtTicket, DEAL_POSITION_ID);
   string            deal_symbol= HistoryDealGetString(ExtTicket, DEAL_SYMBOL);
   double            deal_volume= HistoryDealGetDouble(ExtTicket, DEAL_VOLUME);
   double            deal_price = HistoryDealGetDouble(ExtTicket, DEAL_PRICE);
   int               digits     = (int)SymbolInfoInteger(deal_symbol, SYMBOL_DIGITS);

   PrintFormat("(Position ID #%I64d) %s Deal %s %.2f %s #%I64u by order #%I64d at %.*
               deal_pos_id, deal_symbol, DealEntryDescription(deal_entry), deal_volume
               DealTypeDescription(deal_type), ExtTicket, deal_order, digits, deal_pri
               TimeMscToString(deal_time));
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Deal-Typs                          |
//+------------------------------------------------------------------+
string DealTypeDescription(const ENUM_DEAL_TYPE type)
  {
   switch(type)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      case DEAL_TYPE_BUY                     :  return("Buy");
      case DEAL_TYPE_SELL                    :  return("Sell");
      case DEAL_TYPE_BALANCE                 :  return("Balance");
      case DEAL_TYPE_CREDIT                  :  return("Credit");
      case DEAL_TYPE_CHARGE                  :  return("Additional charge");
      case DEAL_TYPE_CORRECTION              :  return("Correction");
      case DEAL_TYPE_BONUS                   :  return("Bonus");
      case DEAL_TYPE_COMMISSION              :  return("Additional commission");
      case DEAL_TYPE_COMMISSION_DAILY        :  return("Daily commission");
      case DEAL_TYPE_COMMISSION_MONTHLY      :  return("Monthly commission");
      case DEAL_TYPE_COMMISSION_AGENT_DAILY  :  return("Daily agent commission");
      case DEAL_TYPE_COMMISSION_AGENT_MONTHLY:  return("Monthly agent commission");
      case DEAL_TYPE_INTEREST                :  return("Interest rate");
      case DEAL_TYPE_BUY_CANCELED            :  return("Canceled buy deal");
      case DEAL_TYPE_SELL_CANCELED           :  return("Canceled sell deal");
      case DEAL_DIVIDEND                     :  return("Dividend operations");
      case DEAL_DIVIDEND_FRANKED             :  return("Franked (non-taxable) dividenc
      case DEAL_TAX                          :  return("Tax charges");
      default                                :  return("Unknown deal type: "+(string)t
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Änderungsmethode der Position                       |
//+------------------------------------------------------------------+
string DealEntryDescription(const ENUM_DEAL_ENTRY entry)
  {
   switch(entry)
     {
      case DEAL_ENTRY_IN      :  return("In");
      case DEAL_ENTRY_OUT     :  return("Out");
      case DEAL_ENTRY_INOUT   :  return("Reverce");
      case DEAL_ENTRY_OUT_BY  :  return("Out by");
      case DEAL_ENTRY_STATE   :  return("Status record");
      default                 :  return("Unknown deal entry: "+(string)entry);
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Zeit in Millisekunden                               |
//+------------------------------------------------------------------+
string TimeMscToString(const long time_msc, int flags=TIME_DATE|TIME_MINUTES|TIME_SECO
  {
   return(TimeToString(time_msc/1000, flags) + "." + IntegerToString(time_msc %1000, 3
  }
```

Siehe auch

HistorySelect(), HistoryDealGetTicket(), Eigenschaften der Deals

© 2000-2025, MetaQuotes Ltd.

HistoryDealsTotal

Gibt die Anzahl der Deals in der Historie zurück. Vor dem Aufruf der Funktion HistoryDealsTotal(), müssen Sie die Historie der Transaktionen und Aufträge mit der HistorySelect() oder HistorySelectByPosition() abrufen.

```
int  HistoryDealsTotal();
```

Rückgabewert

Wert vom Typ int.

Hinweis

Man darf Order, Deal und Position nicht miteinander verwechseln. Jeder Deal ist das Ergebnis der Ausführung eines Auftrags, jede Position ist das Abschlussergebnis eines oder mehrerer Deals.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anfordern des gesamten vorhandenen Historie des Kontos
   if(!HistorySelect(0, TimeCurrent()))
     {
      Print("HistorySelect() failed. Error ", GetLastError());
      return;
     }

//--- Abrufen und Ausdrucken aller Deals in der Liste
   int total=HistoryDealsTotal();
   Print("Number of historical deals on the account: ", total);
   /*
   Ergebnis:
   Number of historical deals on the account: 339
   */
  }
```

Siehe auch

HistorySelect(), HistoryDealGetTicket(), Eigenschaften der Deals

© 2000-2025, MetaQuotes Ltd.

HistoryDealGetTicket

Waehlt den Deal für die weitere Verarbeitung und gibt Ticket des Deales in der Geschichte zurück. Prior to calling HistoryDealGetTicket(), first it is necessary to receive the history of deals and orders using the HistorySelect() or HistorySelectByPosition() function.

```
ulong  HistoryDealGetTicket(
   int  index      // Nummer des Deals
   );
```

Parameter

index

[in] Nummer des Deals in der Dealsliste.

Rückgabewert

Wert des Typs ulong. Im Fall der fehlerhaften Ausführung gibt 0 zurück.

Hinweis

Man muss nicht Ordern, Deals und Positionen verwechseln. Jeder Deal ist das Ergebnis der Ausführung einer Order, jede Position ist das Abschlussergebnis eines oder mehrerer Deals.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   ulong deal_ticket;             // Ticket des Deals
   ulong order_ticket;            // Ticket der Order, für den Deal durchgeführt wurde
   datetime transaction_time;     // Zeit der Transaktion
   long deal_type ;               // Typ der Handelsoperation
   long position_ID;              // Identifikator der Position
   string deal_description;       // Beschreibung der Operation
   double volume;                 // Volumen der Operation
   string symbol;                 // Symbol des Deals
//--- stellen wir das Anfangsdatum und das Schlussdatum für Anforderung der Dealssges
   datetime from_date=0;           // vom Anfang
   datetime to_date=TimeCurrent();// bis zum laufenden Zeitpunkt
//--- fordern wir die Dealsgeschichte im angegebenen Interval an
   HistorySelect(from_date,to_date);
//--- gesamte Anzahl in der Liste der Deals
   int deals=HistoryDealsTotal();
//--- jetzt verarbeiten wir jeden Deal
   for(int i=0;i<deals;i++)
     {
      deal_ticket=               HistoryDealGetTicket(i);
      volume=                    HistoryDealGetDouble(deal_ticket,DEAL_VOLUME);
      transaction_time=(datetime)HistoryDealGetInteger(deal_ticket,DEAL_TIME);
      order_ticket=              HistoryDealGetInteger(deal_ticket,DEAL_ORDER);
      deal_type=                 HistoryDealGetInteger(deal_ticket,DEAL_TYPE);
      symbol=                    HistoryDealGetString(deal_ticket,DEAL_SYMBOL);
      position_ID=               HistoryDealGetInteger(deal_ticket,DEAL_POSITION_ID);
      deal_description=          GetDealDescription(deal_type,volume,symbol,order_tic
      //--- machen wir das schoene Formatieren für Dealnummer
      string print_index=StringFormat("% 3d",i);
      //--- geben wir Information über den Deal aus
      Print(print_index+": deal #",deal_ticket," at ",transaction_time,deal_descriptio
     }
  }
//+------------------------------------------------------------------+
//| Gibt die Zeilenbeschreibung der Operation zurück                 |
//+------------------------------------------------------------------+
string GetDealDescription(long deal_type,double volume,string symbol,long ticket,long
  {
   string descr;
//---
   switch(deal_type)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      case DEAL_TYPE_BALANCE:                  return ("balance");
      case DEAL_TYPE_CREDIT:                   return ("credit");
      case DEAL_TYPE_CHARGE:                   return ("charge");
      case DEAL_TYPE_CORRECTION:               return ("correction");
      case DEAL_TYPE_BUY:                      descr="buy"; break;
      case DEAL_TYPE_SELL:                     descr="sell"; break;
      case DEAL_TYPE_BONUS:                    return ("bonus");
      case DEAL_TYPE_COMMISSION:               return ("additional commission");
      case DEAL_TYPE_COMMISSION_DAILY:         return ("daily commission");
      case DEAL_TYPE_COMMISSION_MONTHLY:       return ("monthly commission");
      case DEAL_TYPE_COMMISSION_AGENT_DAILY:   return ("daily agent commission");
      case DEAL_TYPE_COMMISSION_AGENT_MONTHLY: return ("monthly agent commission");
      case DEAL_TYPE_INTEREST:                 return ("interest rate");
      case DEAL_TYPE_BUY_CANCELED:             descr="cancelled buy deal"; break;
      case DEAL_TYPE_SELL_CANCELED:            descr="cancelled sell deal"; break;
     }
   descr=StringFormat("%s %G %s (order #%d, position ID %d)",
                      descr,  // laufender Zustand
                      volume, // Dealsvolumen
                      symbol, // Symbol des Deals
                      ticket, // Ticket der Order, die den Deal aufgerufen hat
                      pos_ID  // ID der Position des Deals
                      );
   return(descr);
//---
  }
```

Sehen Sie auch

HistorySelect(), HistoryDealsTotal(), HistoryDealSelect(), Eigenschaften der Deals

© 2000-2025, MetaQuotes Ltd.

HistoryDealGetDouble

Gibt die angeforderte Eigenschaft des Deals zurück. Die Eigenschaft des Deals muss vom Typ double sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
double  HistoryDealGetDouble(
   ulong                      ticket_number,     // Ticket
   ENUM_DEAL_PROPERTY_DOUBLE  property_id        // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück abhängig davon, ob der Funktionsaufruf erfolgreich war. Im Erfolgsfall wird der Wert der Eigenschaft der Empfangsvariable zugewiesen, die per Referenz als letzter Parameter übergeben wird.

```
bool  HistoryDealGetDouble(
   ulong                      ticket_number,     // Ticket
   ENUM_DEAL_PROPERTY_DOUBLE  property_id,       // Identifikator der Eigenschaft
   double&                    double_var         // hier nehmen wir den Wert der Eiger
   );
```

Parameter

ticket_number

[in] Ticket des Deals.

property_id

[in] Identifikator der Eigenschaft des Deals. Wert kann einer der Enumerationswerte ENUM_DEAL_PROPERTY_DOUBLE sein.

double_var

[out] Variable vom Typ double, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ double.

Hinweis

Man darf Order, Deal und Position nicht miteinander verwechseln. Jeder Deal ist das Ergebnis der Ausführung eines Auftrags (order), jede Position ist das Abschlussergebnis eines oder mehrerer Deals.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Handels- und Auftragshistorie anfordern
   if(!HistorySelect(0, TimeCurrent()))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("HistorySelect() failed. Error ", GetLastError());
       return;
      }

//--- in einer Schleife durch die Liste der Deals in der Kontohistorie
   int total=HistoryDealsTotal();
   for(int i=0; i<total; i++)
     {
      //--- Ticket des nächsten Deals abrufen (der Deal wird automatisch ausgewählt, u
      ulong ticket=HistoryDealGetTicket(i);
      if(ticket==0)
         continue;

      //--- Typ und Richtung des Deals abrufen und die Kopfzeile für die Liste der „do
      string type=DealTypeDescription((ENUM_DEAL_TYPE)HistoryDealGetInteger(ticket, DE
      string entry=DealEntryDescription((ENUM_DEAL_ENTRY)HistoryDealGetInteger(ticket,
      PrintFormat("Double properties of an deal %s entry %s #%I64u:", type, entry, tic

      //--- alle „double"-Eigenschaften des ausgewählten Deals unter der Überschrift d
      HistoryDealPropertiesDoublePrint(ticket, 12);
     }
   /*
   real:
   Double properties of an deal Buy entry In #2785070622:
   Volume:     0.50
   Price:      1.10480
   Commission: 0.00
   Swap:       0.00
   Profit:     0.00 USD
   Fee:        0.00
   StopLoss:   0.00000
   TakeProfit: 0.00000
   Double properties of an deal Sell entry Out #2785071538:
   Volume:     0.50
   Price:      1.10491
   Commission: 0.00
   Swap:       0.00
   Profit:     5.50 USD
   Fee:        0.00
   StopLoss:   0.00000
   TakeProfit: 0.00000
   */
  }
//+------------------------------------------------------------------+
//| Ausdruck der „double"-Eigenschaften des ausgewählten Deals       |
//+------------------------------------------------------------------+
void HistoryDealPropertiesDoublePrint(const ulong ticket, const uint header_width=0)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   uint   w=0;
   string header="";
   double value=0;

//--- Abrufen von Deal-Symbol, die Gewinnwährung und die Anzahl der Dezimalstellen für
   string symbol  = HistoryDealGetString(ticket, DEAL_SYMBOL);
   string currency= SymbolInfoString(symbol, SYMBOL_CURRENCY_PROFIT);
   int    digits  = (int)SymbolInfoInteger(symbol, SYMBOL_DIGITS);

//--- Definieren der Kopftextes und der Breite des Textfeldes
//--- Wenn der Funktion eine Breite von Null übergeben wird, dann ist die Breite die G
   header="Volume:";
   w=(header_width==0 ? header.Length()+1 : header_width);
//--- Abrufen und Anzeigen des Deal-Volumens mit der angegebenen Header-Breite im Jour
   if(!HistoryDealGetDouble(ticket, DEAL_VOLUME, value))
      return;
   PrintFormat("%-*s%-.2f", w, header, value);

//--- Ausdruck des Dealpreises im Journal
   header="Price:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetDouble(ticket, DEAL_PRICE, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);

//--- Anzeige der Kommission des Deals im Journal
   header="Commission:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetDouble(ticket, DEAL_COMMISSION, value))
      return;
   PrintFormat("%-*s%-.2f", w, header, value);

//--- Anzeige der beim Schließen angesammelten Swaps im Journal
   header="Swap:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetDouble(ticket, DEAL_SWAP, value))
      return;
   PrintFormat("%-*s%-.2f", w, header, value);

//--- Ausdruck des Finanzergebnisses des Deals im Journal
   header="Profit:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetDouble(ticket, DEAL_PROFIT, value))
      return;
   PrintFormat("%-*s%-.2f %s", w, header, value, currency);

//--- Ausdruck der Transaktionsgebühr im Journal
   header="Fee:";
   w=(header_width==0 ? header.Length()+1 : header_width);
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!HistoryDealGetDouble(ticket, DEAL_FEE, value))
      return;
   PrintFormat("%-*s%-.2f", w, header, value);

//--- Ausdruck der StopLoss-Level im Journal
   header="StopLoss:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetDouble(ticket, DEAL_SL, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);

//--- Ausdruck des TakeProfit-Levels im Journal
   header="TakeProfit:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetDouble(ticket, DEAL_TP, value))
      return;
   PrintFormat("%-*s%-.*f", w, header, digits, value);
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Deal-Typs                          |
//+------------------------------------------------------------------+
string DealTypeDescription(const ENUM_DEAL_TYPE type)
  {
   switch(type)
     {
      case DEAL_TYPE_BUY                    :  return("Buy");
      case DEAL_TYPE_SELL                   :  return("Sell");
      case DEAL_TYPE_BALANCE                :  return("Balance");
      case DEAL_TYPE_CREDIT                 :  return("Credit");
      case DEAL_TYPE_CHARGE                 :  return("Additional charge");
      case DEAL_TYPE_CORRECTION             :  return("Correction");
      case DEAL_TYPE_BONUS                  :  return("Bonus");
      case DEAL_TYPE_COMMISSION             :  return("Additional commission");
      case DEAL_TYPE_COMMISSION_DAILY       :  return("Daily commission");
      case DEAL_TYPE_COMMISSION_MONTHLY     :  return("Monthly commission");
      case DEAL_TYPE_COMMISSION_AGENT_DAILY :  return("Daily agent commission");
      case DEAL_TYPE_COMMISSION_AGENT_MONTHLY:  return("Monthly agent commission");
      case DEAL_TYPE_INTEREST               :  return("Interest rate");
      case DEAL_TYPE_BUY_CANCELED           :  return("Canceled buy deal");
      case DEAL_TYPE_SELL_CANCELED          :  return("Canceled sell deal");
      case DEAL_DIVIDEND                    :  return("Dividend operations");
      case DEAL_DIVIDEND_FRANKED            :  return("Franked (non-taxable) dividenc
      case DEAL_TAX                         :  return("Tax charges");
      default                               :  return("Unknown deal type: "+(string)t
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Änderungsmethode der Position                       |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
string DealEntryDescription(const ENUM_DEAL_ENTRY entry)
  {
   switch(entry)
     {
      case DEAL_ENTRY_IN      :  return("In");
      case DEAL_ENTRY_OUT     :  return("Out");
      case DEAL_ENTRY_INOUT   :  return("Reverce");
      case DEAL_ENTRY_OUT_BY  :  return("Out by");
      case DEAL_ENTRY_STATE   :  return("Status record");
      default                 :  return("Unknown deal entry: "+(string)entry);
     }
  }
```

Siehe auch

HistoryDealsTotal(), HistorySelect(), HistoryDealGetTicket(), Eigenschaften der Deals

© 2000-2025, MetaQuotes Ltd.

HistoryDealGetInteger

Gibt die angeforderte Eigenschaft des Deals zurück. Eigenschaft des Deals muss des Typs datetime, int sein. Es gibt 2 Varianten der Funktion.

1. Gibt den wert der Eigenschaft sofort zurück.

```
long  HistoryDealGetInteger(
   ulong                      ticket_number,     // Ticket
   ENUM_DEAL_PROPERTY_INTEGER  property_id       // Identifikator der Eigeschaft
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wird. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  HistoryDealGetInteger(
   ulong                      ticket_number,     // Ticket
   ENUM_DEAL_PROPERTY_INTEGER  property_id,      // Identifikator der Eigenschaft
   long&                      long_var           // hier nehmen wir den Wert der Eige
   );
```

Parameter

ticket_number

[in] Ticket des Deals.

property_id

[in] Identifikator der Eigenschaft des Deals. Wert kann einer der Enumerationswerte ENUM_DEAL_PROPERTY_INTEGER sein.

long_var

[out] Wert des Typs long.

Rückgabewert

Wert des Typs long.

Hinweis

Man muss nicht Ordern, Deals und Positionen verwechseln. Jeder Deal ist das Ergebnis der Ausführung einer Order, jede Position ist das Abschlussergebnis eines oder mehrerer Deals.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Trade function                                                   |
//+------------------------------------------------------------------+
void OnTrade()
  {
//--- Ticket des letzten Deals aus der Handelsgeschichte für eine Woche erhalten
   ulong last_deal=GetLastDealTicket();
   if(HistoryDealSelect(last_deal))
     {
      //--- Zeit des Deals in Millisekunden seit 01.01.1970
      long deal_time_msc=HistoryDealGetInteger(last_deal,DEAL_TIME_MSC);
      PrintFormat("Deal #%d DEAL_TIME_MSC=%i64 => %s",
                  last_deal,deal_time_msc,TimeToString(deal_time_msc/1000));
     }
   else
      PrintFormat("HistoryDealSelect() failed for #%d. Eror code=%d",
                  last_deal,GetLastError());
//---
  }
//+------------------------------------------------------------------+
//| Gibt Ticket des letzten Deals in der Geschichte oder -1 zurück   |
//+------------------------------------------------------------------+
ulong GetLastDealTicket()
  {
//--- Geschichte für die letzte 7 Tagen erfordern
   if(!GetTradeHistory(7))
     {
      //--- Nachricht über fehlgeschlagenen Aufruf und Rückgabe -1
      Print(__FUNCTION__," HistorySelect() hat false zurückgegeben");
      return -1;
     }
//---
   ulong first_deal,last_deal,deals=HistoryDealsTotal();
//--- wenn es Ordern gibt, anfangen wir mit ihnen zu arbeiten
   if(deals>0)
     {
      Print("Deals = ",deals);
      first_deal=HistoryDealGetTicket(0);
      PrintFormat("first_deal = %d",first_deal);
      if(deals>1)
        {
         last_deal=HistoryDealGetTicket((int)deals-1);
         PrintFormat("last_deal = %d",last_deal);
         return last_deal;
        }
      return first_deal;
     }
//--- keine Deals waren gefunden, Rückgabe -1
   return -1;
  }
//+------------------------------------------------------------------+
//| Geschichte für letzte Tage erfordert und gibt false beim Fehler zurück |
//+------------------------------------------------------------------+
bool GetTradeHistory(int days)
  {
//--- definieren wir Wochenperiode, um eine Geschichte des Handels zu beantragen
   datetime to=TimeCurrent();
   datetime from=to-days*PeriodSeconds(PERIOD_D1);
   ResetLastError();
//--- Machen wir eine Anfrage und überprüfen wir die Ergebnisse
   if(!HistorySelect(from,to))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print(__FUNCTION__," HistorySelect=false. Error code=",GetLastError());
       return false;
      }
//--- Geschichte erfolgreich empfangen
   return true;
  }
```

Sehen Sie auch

HistoryDealsTotal(), HistorySelect(), HistoryDealGetTicket(), Eigenschaften der Deals

© 2000-2025, MetaQuotes Ltd.

HistoryDealGetString

Gibt die angeforderte Eigenschaft des Deals zurück. Die Eigenschaft des Deals muss vom Typ string sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft selbst zurück.

```
string  HistoryDealGetString(
   ulong                      ticket_number,     // Ticket
   ENUM_DEAL_PROPERTY_STRING  property_id        // Identifikator der Eigenschaft
   );
```

2. Gibt true oder false zurück, abhängig davon, ob die Funktion erfolgreich durchgeführt wird. Im Erfolgsfall wird der Wert der Variable der Empfangsvaiable zugewiesen, die durch Referenz als letzter Parameter übertragen wird.

```
bool  HistoryDealGetString(
   ulong                      ticket_number,     // Ticket
   ENUM_DEAL_PROPERTY_STRING  property_id,       // Identifikator der Eigenschaft
   string&                    string_var         // hier nehmen wir den Wert der Eiger
   );
```

Parameter

ticket_number

[in] Ticket des Deals.

property_id

[in] Identifikator der Eigenschaft des Deals. Wert kann einer der Enumerationswerte ENUM_DEAL_PROPERTY_STRING sein.

string_var

[out] Variable vom Typ string, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert vom Typ string.

Hinweis

Man darf Order, Deal und Position nicht miteinander verwechseln. Jeder Deal ist das Ergebnis der Ausführung eines Auftrags (order), jede Position ist das Abschlussergebnis eines oder mehrerer Deals.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Handels- und Auftragshistorie anfordern
   if(!HistorySelect(0, TimeCurrent()))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       Print("HistorySelect() failed. Error ", GetLastError());
       return;
      }

//--- in einer Schleife durch die Liste der Deals in der Kontohistorie
   int total=HistoryDealsTotal();
   for(int i=0; i<total; i++)
     {
      //--- Ticket des nächsten Deals abrufen (der Deal wird automatisch ausgewählt, u
      ulong ticket=HistoryDealGetTicket(i);
      if(ticket==0)
         continue;

      //--- Typ und Richtung des Deals abrufen und die Kopfzeile für die Liste der „do
      string type=DealTypeDescription((ENUM_DEAL_TYPE)HistoryDealGetInteger(ticket, DE
      string entry=DealEntryDescription((ENUM_DEAL_ENTRY)HistoryDealGetInteger(ticket,
      PrintFormat("String properties of an deal %s entry %s #%I64u:", type, entry, tic

      //--- alle „double"-Eigenschaften des ausgewählten Deals unter der Überschrift d
      HistoryDealPropertiesStringPrint(ticket, 13);
     }
   /*
   Ergebnis:
   String properties of an deal Buy entry In #2785021084:
   Symbol:      EURUSD
   Comment:     Test PositionGetString
   Extarnal ID:
   String properties of an deal Buy entry Out #2497993663:
   Symbol:      EURUSD
   Comment:     [tp 1.08639]
   Extarnal ID:
   */
  }
//+------------------------------------------------------------------+
//| Ausdruck der Texteigenschaften des ausgewählten Deals im Journal |
//+------------------------------------------------------------------+
void HistoryDealPropertiesStringPrint(const ulong ticket, const uint header_width=0)
  {
   uint   w=0;
   string header="";
   string value ="";

//--- Definieren der Kopftextes und der Breite des Textfeldes
//--- Wenn der Funktion eine Breite von Null übergeben wird, dann ist die Breite die G
   header="Symbol:";
   w=(header_width==0 ? header.Length()+1 : header_width);
//--- Abrufen und Ausdruck des Deal-Symbols mit der angegebenen Kopfzeilenbreite im Jo
   if(!HistoryDealGetString(ticket, DEAL_SYMBOL, value))
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
   PrintFormat("%-*s%-s", w, header, value);

//--- Kommentar zum Deal im Journal anzeigen
   header="Comment:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetString(ticket, DEAL_COMMENT, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);

//--- Ausdruck der Deal-ID eines externen Handelssystems
   header="Extarnal ID:";
   w=(header_width==0 ? header.Length()+1 : header_width);
   if(!HistoryDealGetString(ticket, DEAL_EXTERNAL_ID, value))
      return;
   PrintFormat("%-*s%-s", w, header, value);
  }
//+------------------------------------------------------------------+
//| Rückgabe der Beschreibung des Deal-Typs                          |
//+------------------------------------------------------------------+
string DealTypeDescription(const ENUM_DEAL_TYPE type)
  {
   switch(type)
     {
      case DEAL_TYPE_BUY                    :  return("Buy");
      case DEAL_TYPE_SELL                   :  return("Sell");
      case DEAL_TYPE_BALANCE                :  return("Balance");
      case DEAL_TYPE_CREDIT                 :  return("Credit");
      case DEAL_TYPE_CHARGE                 :  return("Additional charge");
      case DEAL_TYPE_CORRECTION             :  return("Correction");
      case DEAL_TYPE_BONUS                  :  return("Bonus");
      case DEAL_TYPE_COMMISSION             :  return("Additional commission");
      case DEAL_TYPE_COMMISSION_DAILY       :  return("Daily commission");
      case DEAL_TYPE_COMMISSION_MONTHLY     :  return("Monthly commission");
      case DEAL_TYPE_COMMISSION_AGENT_DAILY :  return("Daily agent commission");
      case DEAL_TYPE_COMMISSION_AGENT_MONTHLY: return("Monthly agent commission");
      case DEAL_TYPE_INTEREST               :  return("Interest rate");
      case DEAL_TYPE_BUY_CANCELED           :  return("Canceled buy deal");
      case DEAL_TYPE_SELL_CANCELED          :  return("Canceled sell deal");
      case DEAL_DIVIDEND                    :  return("Dividend operations");
      case DEAL_DIVIDEND_FRANKED            :  return("Franked (non-taxable) dividend
      case DEAL_TAX                         :  return("Tax charges");
      default                               :  return("Unknown deal type: "+(string)t
     }
  }
//+------------------------------------------------------------------+
//| Rückgabe der Änderungsmethode der Position                       |
//+------------------------------------------------------------------+
string DealEntryDescription(const ENUM_DEAL_ENTRY entry)
```

© 2000-2025, MetaQuotes Ltd.

```
{
 switch(entry)
   {
    case DEAL_ENTRY_IN       :  return("In");
    case DEAL_ENTRY_OUT      :  return("Out");
    case DEAL_ENTRY_INOUT    :  return("Reverce");
    case DEAL_ENTRY_OUT_BY   :  return("Out by");
    case DEAL_ENTRY_STATE    :  return("Status record");
    default                  :  return("Unknown deal entry: "+(string)entry);
   }
}
```

Siehe auch

HistoryDealsTotal(), HistorySelect(), HistoryDealGetTicket(), Eigenschaften der Deals

© 2000-2025, MetaQuotes Ltd.

Handelssignale

Eine Gruppe von Funktionen für die Steuerung der Handelssignale. Diese Funktionen erlauben Ihnen:

- Informationen über die Handelssignale, die zum Kopieren zur Verfügung stehen, erhalten,
- Kopiereinstellungen von Handelssignale lesen oder setzen,
- zu einem Signal abonnieren und Subskription durch die Sprache MQL5 abschaffen.

Funktion	Aktion
SignalBaseGetDouble	Gibt den Wert der Eigenschaft vom Typ double für das ausgewählte Signal zurück
SignalBaseGetInteger	Gibt den Wert der Eigenschaft vom Typ integer für das ausgewählte Signal zurück
SignalBaseGetString	Gibt den Wert der Eigenschaft vom Typ string für das ausgewählte Signal zurück
SignalBaseSelect	Wählt ein Signal aus der Basis der im Terminal verfügbaren Handelssignale aus
SignalBaseTotal	Gibt die Anzahl der im Terminal verfügbaren Signale zurück
SignalInfoGetDouble	Gibt den Wert der Eigenschaft vom Typ double aus der Kopiereinstellungen des Handelssignals zurück
SignalInfoGetInteger	Gibt den Wert der Eigenschaft vom Typ integer aus der Kopiereinstellungen des Handelssignals zurück
SignalInfoGetString	Gibt den Wert der Eigenschaft vom Typ string aus der Kopiereinstellungen des Handelssignals zurück
SignalInfoSetDouble	Setzt den Wert der Eigenschaft vom Typ double in Kopiereinstellungen des Handelssignals
SignalInfoSetInteger	Setzt den Wert der Eigenschaft vom Typ integer in Kopiereinstellungen des Handelssignals
SignalSubscribe	Abonniert auf ein Handelssignal
SignalUnsubscribe	Abbestellt Abonnement auf ein Handelssignal

© 2000-2025, MetaQuotes Ltd.

SignalBaseGetDouble

Gibt den Wert der Eigenschaft vom Typ double für das ausgewählte Signal zurück.

```
double  SignalBaseGetDouble(
   ENUM_SIGNAL_BASE_DOUBLE     property_id,     // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft des Signals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_BASE_DOUBLE sein.

Rückgabewert

Der Wert von double der angegebenen Eigenschaft des Signals.

© 2000-2025, MetaQuotes Ltd.

SignalBaseGetInteger

Gibt den Wert der Eigenschaft vom Typ integer für das ausgewählte Signal zurück.

```
long  SignalBaseGetInteger(
   ENUM_SIGNAL_BASE_INTEGER     property_id,     // der Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft des Signals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_BASE_INTEGER sein.

Rückgabewert

Der Wert von integer der angegebenen Eigenschaft des Signals.

© 2000-2025, MetaQuotes Ltd.

SignalBaseGetString

Gibt den Wert der Eigenschaft vom Typ string für das ausgewählte Signal zurück.

```
string  SignalBaseGetString(
   ENUM_SIGNAL_BASE_STRING     property_id,     // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft des Signals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_BASE_STRING sein.

Rückgabewert

Der Wert von string der angegebenen Eigenschaft des Signals.

© 2000-2025, MetaQuotes Ltd.

SignalBaseSelect

Wählt ein Signal aus der Datenbank der im Terminal erhältlichen Signalen aus.

```
bool  SignalBaseSelect(
   int     index     // der Index der Aufzeichnung des Signals
   );
```

Parameter

index

[in] Index der Aufzeichnung des Signals in der Datenbank von Signalen.

Rückgabewert

Gibt true bei einem erfolgreichen Beenden der Funktion zurück, und false, wenn ein Fehler aufgetreten ist. Rufen Sie die Funktion GetLastError() auf, um Details zum Fehler zu bekommen.

Beispiel:

```
void OnStart()
  {
//--- die Gesamtzahl von Signalen in der Datenbank abrufen
   int total=SignalBaseTotal();
//--- Schleife durch alle Signale
   for(int i=0;i<total;i++)
     {
      //--- ein Signal für die weitere Arbeit auswählen
      if(SignalBaseSelect(i))
        {
         //--- Eigenschaften des Signals erhalten
         long   id    =SignalBaseGetInteger(SIGNAL_BASE_ID);          //ID des Signals
         long   pips  =SignalBaseGetInteger(SIGNAL_BASE_PIPS);        //Handelsergebni
         long   subscr=SignalBaseGetInteger(SIGNAL_BASE_SUBSCRIBERS); // Anzahl der Ab
         string name  =SignalBaseGetString(SIGNAL_BASE_NAME);         // Name des Sign
         double price =SignalBaseGetDouble(SIGNAL_BASE_PRICE);        // Abo-Preis für
         string curr  =SignalBaseGetString(SIGNAL_BASE_CURRENCY);     // Währung des S
         //--- alle kostenlosen profitablen Signale mit Abonnenten anzeigen
         if(price==0.0 && pips>0 && subscr>0)
            PrintFormat("id=%d, name=\"%s\", currency=%s, pips=%d, subscribers=%d",id,
        }
      else PrintFormat("Falscher Signal ausgewählt. Fehlercode=%d",GetLastError());
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SignalBaseTotal

Gibt die Gesamtzahl der im Terminal verfügbaren Signale zurück.

```
int  SignalBaseTotal();
```

Rückgabewert

Die Gesamtzahl der im Terminal verfügbaren Signale.

© 2000-2025, MetaQuotes Ltd.

SignalInfoGetDouble

Gibt den Wert der Eigenschaft vom Typ double von den Einstellungen des Kopierens des Handelssignals zurück.

```
double  SignalInfoGetDouble(
   ENUM_SIGNAL_INFO_DOUBLE     property_id,     // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Der Identifikator der Eigenschaft in den Einstellungen des Kopierens des Handelssignals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_INFO_DOUBLE sein.

Rückgabewert

Der Wert von double der angegebenen Eigenschaft.

© 2000-2025, MetaQuotes Ltd.

SignalInfoGetInteger

Gibt den Wert der Eigenschaft vom Typ integer von den Einstellungen des Kopierens des Handelssignals zurück.

```
long  SignalInfoGetInteger(
   ENUM_SIGNAL_INFO_INTEGER     property_id,     // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Der Identifikator der Eigenschaft in den Einstellungen des Kopierens des Handelssignals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_INFO_INTEGER sein.

Rückgabewert

Der Wert von integer der angegebenen Eigenschaft von den Einstellungen des Kopierens des Handelssignals.

© 2000-2025, MetaQuotes Ltd.

SignalInfoGetString

Gibt den Wert der Eigenschaft vom Typ string von den Einstellungen des Kopierens des Handelssignals zurück.

```
string  SignalInfoGetString(
   ENUM_SIGNAL_INFO_STRING     property_id,     // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Der Identifikator der Eigenschaft in den Einstellungen des Kopierens des Handelssignals. Der Wert kann einer der Werte der Aufzählung ENUM_SIGNAL_INFO_STRING sein.

Rückgabewert

Der Wert vom Typ string der angegebenen Eigenschaft von den Einstellungen des Kopierens des Handelssignals.

© 2000-2025, MetaQuotes Ltd.

SignalInfoSetDouble

Setzt den Wert der Eigenschaft vom Typ double in den Einstellungen des Kopierens des Handelssignals.

```
bool  SignalInfoSetDouble(
   ENUM_SIGNAL_INFO_DOUBLE      property_id,     // Identifikator der Eigenschaft
   double                       value            // der Wert der Eigenschaft
   );
```

Parameter

property_id

[in] Der Identifikator der Eigenschaft in den Einstellungen des Kopierens des Handelssignals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_INFO_DOUBLE sein.

value

[in] Der Wert der Eigenschaft in den Einstellungen des Kopierens des Handelssignals.

Rückgabewert

true - wenn die Eigenschaft erfolgreich geändert wurde, sonst false. Um zusätzliche Informationen über den Fehler zu bekommen, rufen Sie die Funktion GetLastError() auf.

© 2000-2025, MetaQuotes Ltd.

SignalInfoSetInteger

Setzt den Wert der Eigenschaft vom Typ integer in den Einstellungen des Kopierens des Handelssignals.

```
bool  SignalInfoSetInteger(
   ENUM_SIGNAL_INFO_INTEGER     property_id,     // Identifikator der Eigenschaft
   long                         value            // der Wert der Eigenschaft
   );
```

Parameter

property_id

[in] Der Identifikator der Eigenschaft in den Einstellungen des Kopierens des Handelssignals. Kann einer der Werte der Aufzählung ENUM_SIGNAL_INFO_INTEGER sein.

value

[in] Der Wert der Eigenschaft in den Einstellungen des Kopierens des Handelssignals.

Rückgabewert

true - wenn die Eigenschaft erfolgreich geändert wurde, sonst false. Um zusätzliche Informationen über den Fehler zu bekommen, rufen Sie die Funktion GetLastError() auf.

© 2000-2025, MetaQuotes Ltd.

SignalSubscribe

Abonniert auf ein Handelssignal.

```
bool  SignalSubscribe(
   long     signal_id     // ID des Signals
   );
```

Parameter

signal_id

[in] ID des Signals.

Rückgabewert

Beim erfolgreichen Abonnieren gibt true zurück, ansonsten false. Für mehr Informationen über den Fehler rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

SignalUnsubscribe

Abbestellt das Abonnement auf ein Handelssignal

```
bool  SignalUnsubscribe();
```

Rückgabewert

Beim erfolgreichen Abbestellung des Abonnements gibt true zurück, ansonsten false. Für mehr Informationen über den Fehler rufen Sie die Funktion GetLastError().

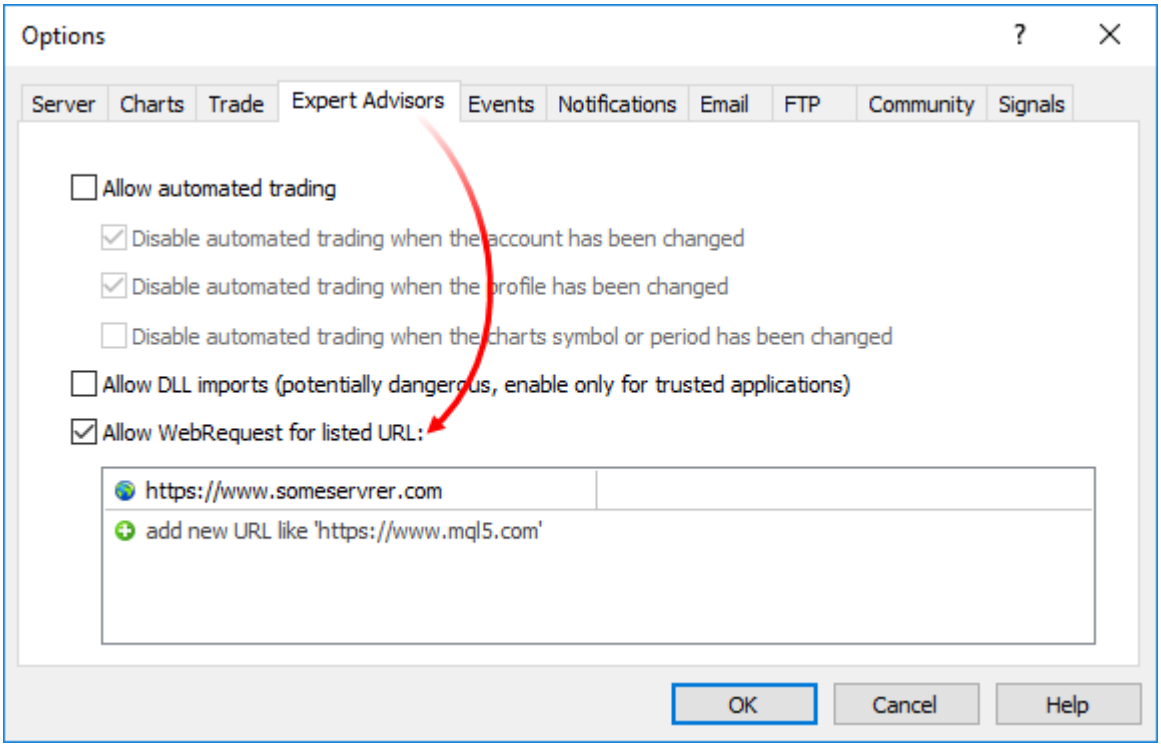
© 2000-2025, MetaQuotes Ltd.

Netzwerkfunktionen

MQL5-Programme können Daten mit entfernten Servern austauschen, sowie Push-Benachrichtigungen, E-Mails und Daten über FTP senden.

- Die Funktionsgruppe Socket* ermöglicht den Aufbau einer TCP-Verbindung (einschließlich eines sicheren TLS) mit einem entfernten Host über Systemsockets. Das Funktionsprinzip ist einfach: Erstellen eines Sockets, Verbinden mit dem Server und Starten von Lesen und Schreiben der Daten.
- Die Funktion WebRequest ist für die Arbeit mit Web-Ressourcen konzipiert und ermöglicht das einfache Senden von HTTP-Anfragen (einschließlich GET und POST).
- SendFTP, SendMail und SendNotification sind einfachere Funktionen zum Senden von Dateien, E-Mails und mobilen Push-Benachrichtigungen.

Aus Gründen der Sicherheit der Endnutzer ist die Liste der zulässigen IP-Adressen auf der Seite des Client-Terminals implementiert. Die Liste enthält die IP-Adressen, mit denen sich das MQL5-Programm über die Funktionen Socket* und WebRequest verbinden darf. Wenn das Programm beispielsweise eine Verbindung zu https://www.someserver.com herstellen soll, muss diese Adresse vom Nutzer des Terminals in der Liste explizit angegeben werden. Eine Adresse kann nicht von einem Programm hinzugefügt werden.



Hinzufügen einer expliziten Meldung an das MQL5-Programm, um einen Nutzer über die Notwendigkeit einer zusätzlichen Konfiguration zu informieren. Sie können das über #property description, Alert oder Print. tun.

Funktion	Aktion
SocketCreate	Erstellen eines Sockets mit den angegebenen Flags und der Rückgabe des Handles
SocketClose	Schließen des Sockets

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
SocketConnect	Verbinden mit einem Server inklusive dem Bestimmen eines Timeouts (Zeitkontrolle)
SocketIsConnected	Prüft, ob der Socket aktuell verbunden ist
SocketIsReadable	Abrufen einer Anzahl von Bytes, die vom Socket gelesen werden können
SocketIsWritable	Prüfen, ob jetzt Daten auf einen Socket geschrieben werden können
SocketTimeouts	Festlegen eines Timeouts für das Senden und Empfangen der Daten eines Sockets
SocketRead	Lesen der Daten von einem Socket
SocketSend	Schreiben der Daten auf einen Socket
SocketTlsHandshake	Initiieren einer sicheren (SSL) Verbindung mit einem angegebenen Host über das Protokoll TLS-Handshake
SocketTlsCertificate	Abrufen der Daten eines Zertifikates, das für eine sichere Verwendung TLS-Handshake verwandt wird
SocketTlsRead	Lesen der Daten einer sicheren TLS-Verbindung
SocketTlsReadAvailable	Lesen aller verfügbaren Daten einer sicheren TLS-Verbindung
SocketTlsSend	Senden von Daten über eine sichere TLS-Verbindung
WebRequest	Senden einer HTTP-Anfrage an den angegebenen Server
SendFTP	Senden einer Datei an die im FTP-Tab angegebene Adresse
SendMail	Senden einer E-Mail an die angegebene Adresse im Tab für die E-Mails im Fenster der Einstellungen
SendNotification	Senden einer Push-Benachrichtigung an mobile Terminals, dessen MetaQuotes IDs im Tab der Push-Benachrichtigung eingetragen ist.

© 2000-2025, MetaQuotes Ltd.

SocketCreate

Erstellen eines Sockets mit den angegebenen Flags und der Rückgabe des Handles.

```
int  SocketCreate(
   uint     flags          // Flags
   );
```

Parameter

flags

[in] Kombination der Flags, die den Arbeitsmodus des Sockets festlegt. Aktuell wird nur ein Flag unterstützt — SOCKET_DEFAULT.

Rückgabewert

Im Falle eines erfolgreichen Erstellens des Sockets, wird das Handle zurückgegeben, andernfalls INVALID_HANDLE.

Hinweis

Um den Computerspeicher eines unbenutzten Sockets freizugeben, rufen Sie SocketClose auf.

Sie können maximal 128 Sockets aus einem MQL5-Programm erstellen. Wird das Limit überschritten, wird der Fehler 5271 (ERR_NETSOCKET_TOO_MANY_OPENED) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
```

© 2000-2025, MetaQuotes Ltd.

```
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere i
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
           }
        }
     }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
        }
      //--- Schließen des Sockets nach dem Ende der Verwendung
      SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

SocketClose

Schließen eines Sockets.

```
bool  SocketClose(
   const int  socket      // Handle des Sockets
   );
```

Parameter

socket

[in] Handle des zu schließenden Sockets. Das Handle erhält man von der Funktion SocketCreate. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Hinweis

Wenn vorher eine Verbindung mit SocketConnect erstellt worden war, ist sie danach unterbrochen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere i
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
           }
        }
     }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
        }
      //--- Schließen des Sockets nach dem Ende der Verwendung
      SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

SocketConnect

Verbinden mit einem Server inklusive dem Bestimmen eines Timeouts (Zeitkontrolle).

```
bool  SocketConnect(
   int           socket,              // Socket
   const string  server,              // Adresse der Verbindung
   uint          port,                // Port der Verbindung
   uint          timeout_receive_ms   // Timeout der Verbindung
   );
```

Parameter

socket

[in] Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

server

[in] Domainname des Servers mit dem Sie sich verbinden wollen oder dessen IP-Adresse.

Port

[in] Portnummer der Verbindung.

timeout_receive_ms

[in] Timeout der Verbindung im Millisekunden. Wenn eine Verbindung sich nicht dieses Zeitintervalls entsteht, werden die versuche gestoppt.

Rückgabewert

Im Erfolgsfall wird true zurückgegeben, andernfalls false.

Hinweis

Die Adresse der Verbindung muss der Liste der erlaubten im Terminal hinzugefügt werden: Extras \ Optionen\ Experten).

Wenn keine Verbindung entstanden ist, wird der Fehler 5272 (ERR_NETSOCKET_CANNOT_CONNECT) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
```

© 2000-2025, MetaQuotes Ltd.

```
                Print("Header der HTTP-Antwort erhalten:");
                Print(StringSubstr(result,0,header_end));
                return(true);
               }
            }
         }
      }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
```

© 2000-2025, MetaQuotes Ltd.

```
         {
          Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
         }
       //--- Schließen des Sockets nach dem Ende der Verwendung
       SocketClose(socket);
      }
    else
       Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

SocketIsConnected

Prüft, ob der Socket aktuell verbunden ist.

```
bool  SocketIsConnected(
   const int  socket      // Handle des Sockets
   );
```

Parameter

socket

[in] Das Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiges Handle erstellt, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

Rückgabewert

Gibt true zurück bei einer gültigen Verbindung, andernfalls false.

Hinweis

Die Funktion SocketIsConnected() erlaubt die Überprüfung der aktuellen Verbindung des Sockets.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                            SocketIsConnected.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address    ="www.mql5.com";
input int    Port       =80;
input bool   CloseSocket=true;
bool         ExtTLS     =false;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Socket erstellen und sein Handle zuweisen
   int socket=SocketCreate();
//--- Prüfen des Handles
```

© 2000-2025, MetaQuotes Ltd.

```
   if(socket!=INVALID_HANDLE)
     {
      //--- wenn alles in Ordnung ist, verbinde
      if(SocketConnect(socket,Address,Port,1000))
        {
         PrintFormat("Established connection to %s:%d",Address,Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- wenn die Verbindung durch ein Zertifikat geschützt ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Owner:      ",subject);
            Print("   Issuer:     ",issuer);
            Print("   Number:     ",serial);
            Print("   Print:      ",thumbprint);
            Print("   Expiration: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Auslesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         PrintFormat("Connection to %s:%d failed, error %d",Address,Port,GetLastError
        }
      //--- wenn das Flag gesetzt ist, schließe den Socket nach Gebrauch
      if(CloseSocket)
         SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());

//--- Prüfen der Serververbindung
   bool connected=SocketIsConnected(socket);
//--- Beenden des Vorgangs, wenn es keine Verbindung gibt
   if(!connected)
     {
      Print("No connection to server");
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn eine Serververbindung besteht
   else
     {
      Print("Connection to the server is available\nThe connection needs to be closed.
      //--- Schließen des Sockets und erneutes Überprüfen des Verbindungsstatus
      SocketClose(socket);
      connected=SocketIsConnected(socket);
     }

//--- den aktuellen Status der Serververbindung anzeigen
   Print("Currently connected: ",(connected ? "opened" : "closed"));
   /*
   result in case CloseSocket = true:
   No connection to server
   Currently connected: closed

   result in case CloseSocket = false:
   Connection to the server is available
   The connection needs to be closed. Closing
   Currently connected: closed
   */
  }
//+------------------------------------------------------------------+
//| Befehl zum Server senden                                         |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
//--- Umwandeln der Zeichenkette in ein Zeichenarray mit dem Löschen der abschließende
   char req[];
   int  len=StringToCharArray(request,req)-1;

   if(len<0)
      return(false);
//--- wenn eine sichere TLS-Verbindung über Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- wenn eine normale TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout_ms)
  {
   char   rsp[];
   string result;
   ulong  timeout_check=GetTickCount64()+timeout_ms;
//--- Daten aus dem Socket lesen, solange welche vorhanden sind, aber nicht länger als
   do
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      uint len=SocketIsReadable(socket);

      if(len)
        {
         int rsp_len;
         //--- unterschiedliche Lesebefehle, je nachdem, ob die Verbindung sicher ist
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout_ms);
         //--- Auswerten der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- nur den Antwort-Header anzeigen
            int header_end=StringFind(result,"\r\n\r\n");

            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
            //--- Ablaufzeit-Aktualisierung der Lesezeitüberschreitung
            timeout_check=GetTickCount64()+timeout_ms;
           }
        }
     }
   while(GetTickCount64()<timeout_check && !IsStopped());

   return(false);
  }
```

Siehe auch

SocketConnect, SocketIsWritable, SocketCreate, SocketClose

© 2000-2025, MetaQuotes Ltd.

SocketIsReadable

Abrufen einer Anzahl von Bytes, die vom Socket gelesen werden können.

```
uint  SocketIsReadable(
   const int  socket      // Handle des Sockets
   );
```

Parameter

socket

[in] Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

Rückgabewert

Anzahl der lesbaren Bytes. Im Fehlerfall wird 0 zurückgegeben.

Hinweis

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Vor dem Aufruf von SocketRead, prüfen Sie das Vorhandensein von lesbaren Daten im Socket. Andernfalls, wenn es keine Daten gibt, wartet die Funktion SocketRead auf Daten für das in timeout_ms festgelegte Zeitintervall und verzögert die Programmausführung.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere 
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
           }
        }
     }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
        }
      //--- Schließen des Sockets nach dem Ende der Verwendung
      SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

SocketIsWritable

Prüfung, ob jetzt Daten auf einen Socket geschrieben werden können.

```
bool  SocketIsWritable(
   const int  socket      // Handle des Sockets
   );
```

Parameter

socket

[in] Das Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiges Handle erstellt, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

Rückgabewert

Gibt true zurück, wenn das Schreiben möglich ist, andernfalls false.

Hinweis

Die Funktion erlaubt zu prüfen, ob Daten jetzt auf den Socket geschrieben werden können.

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                              SocketIsWritable.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address    ="www.mql5.com";
input int    Port       =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Socket erstellen und sein Handle zuweisen
   int socket=SocketCreate();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- wenn alles in Ordnung ist, verbinde
      if(SocketConnect(socket,Address,Port,1000))
        {
         PrintFormat("Established connection to %s:%d",Address,Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- wenn die Verbindung durch ein Zertifikat geschützt ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Owner:      ",subject);
            Print("   Issuer:     ",issuer);
            Print("   Number:     ",serial);
            Print("   Print:      ",thumbprint);
            Print("   Expiration: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Auslesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         PrintFormat("Connection to %s:%d failed, error %d",Address,Port,GetLastError
        }
      //--- im Journal die Möglichkeit anzeigen, ob Daten zum aktuellen Zeitpunkt auf
      if(SocketIsWritable(socket))
         Print("At the current moment in time, writing data to the socket is possible"
      else
         Print("It is not possible to write data to the socket at the current time");
      //--- Schließen des Sockets nach Verwendungsende
      if(SocketClose(socket))
         Print("Now the socket is closed");
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
   /*
   Ergebnis:
```

© 2000-2025, MetaQuotes Ltd.

```
   At the current moment in time, writing data to the socket is possible
   Socket is closed now
   */
  }
//+------------------------------------------------------------------+
//| Befehl zum Server senden                                         |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
//--- Umwandeln der Zeichenkette in ein Zeichenarray mit dem Löschen der abschließende
   char req[];
   int  len=StringToCharArray(request,req)-1;

   if(len<0)
      return(false);
//--- wenn eine sichere TLS-Verbindung über Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- wenn eine normale TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout_ms)
  {
   char   rsp[];
   string result;
   ulong  timeout_check=GetTickCount64()+timeout_ms;
//--- Daten aus dem Socket lesen, solange welche vorhanden sind, aber nicht länger als
   do
     {
      uint len=SocketIsReadable(socket);

      if(len)
        {
         int rsp_len;
         //--- unterschiedliche Lesebefehle, je nachdem, ob die Verbindung sicher ist
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout_ms);
         //--- Auswerten der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- nur den Antwort-Header anzeigen
            int header_end=StringFind(result,"\r\n\r\n");
```

© 2000-2025, MetaQuotes Ltd.

```
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
            //--- Ablaufzeit-Aktualisierung der Lesezeitüberschreitung
            timeout_check=GetTickCount64()+timeout_ms;
           }
        }
     }
  while(GetTickCount64()<timeout_check && !IsStopped());

  return(false);
 }
```

© 2000-2025, MetaQuotes Ltd.

SocketTimeouts

Festlegen eines Timeouts für das Senden und Empfangen der Daten eines Sockets

```
bool  SocketTimeouts(
   int           socket,               // Socket
   uint          timeout_send_ms,      // Timeout für das Senden von Daten
   uint          timeout_receive_ms    // Timeout für das Empfangen von Daten
   );
```

Parameter

socket

[in] Das Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiges Handle erstellt, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

timeout_send_ms

[in] Timeout für das Schreiben der Daten in Millisekunden.

timeout_receive_ms

[in] Timeout für das Lesen der Daten in Millisekunden.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Hinweis

Verwechseln Sie nicht die Timeouts der Systemobjekte mit denen, die für das Lesen der Daten mit SocketRead eingestellt wurden. SocketTimeout setzt einmalig Timeouts für ein Socket-Objekt im Betriebssystem. Diese Timeouts sind auf alle Funktionen zum Lesen und Senden von Daten über diesen Sockel anzuwenden. In SocketRead wird der Timeout für einen bestimmten Datenlesevorgang eingestellt.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketTimeouts.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"

#define   TIMEOUT_SEND     5000
#define   TIMEOUT_RECEIVE  5000
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Socket erstellen und sein Handle zuweisen
   int socket=SocketCreate();

   if(socket==INVALID_HANDLE)
     {
      Print("SocketCreate() failed. Error ",GetLastError());
      return;
     }
//--- Festlegen der Timeouts für den Empfang und das Senden von Daten eines Socket-Sys
   if(SocketTimeouts(socket,TIMEOUT_SEND,TIMEOUT_RECEIVE))
      PrintFormat("timeouts were successfully set",socket,TIMEOUT_SEND,TIMEOUT_RECEIVE
   else
      PrintFormat("SocketTimeouts(%d, %I64d, %I64d) failed. Error %d",socket,TIMEOUT_S
//--- Schließen des Sockets
   SocketClose(socket);
  }
```

© 2000-2025, MetaQuotes Ltd.

SocketRead

Read data from a socket.

```
int  SocketRead(
   int           socket,              // Socket
   uchar&        buffer[],            // Puffer für die vom Socket gelesenen Daten
   uint          buffer_maxlen,       // Anzahl der zu lesenden Bytes
   uint          timeout_ms           // Timeout für den Lesevorgang
   );
```

Parameter

socket

[in] Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

buffer

[out] Referenz eines Arrays vom Typ uchar, dem die Daten zugewiesen werden sollen. Die Größe des dynamischen Arrays wird auf die Anzahl der zu lesenden Bytes gesetzt. Die Arraygröße kann INT_MAX (2147483647) nicht überschreiten.

buffer_maxlen

[in] Anzahl der Bytes, die dem Array *buffer[]* zugewiesen werden sollen. Daten, die nicht in das Array passen, verbleiben im Socket. Diese können beim nächsten Aufruf von SocketRead abgerufen werden. *buffer_maxlen* kann nicht größer sein als INT_MAX (2147483647).

timeout_ms

[in] Timeout für das Lesen der Daten in Millisekunden. Wenn innerhalb dieser Zeitspanne keinen Daten empfangen werden konnten, werden weitere Versuche gestoppt und -1 zurückgegebenen.

Rückgabewert

Bei einem Erfolg wird die Anzahl der gelesenen Bytes zurückgegeben. Im Fehlerfall wird -1 zurückgegeben.

Hinweis

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Im Falle eines Fehlers beim Lesen wird der Fehler 5273 (ERR_NETSOCKET_IO_ERROR) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere i
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
```

© 2000-2025, MetaQuotes Ltd.

```
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
           }
        }
     }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
            }
          else
             Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
         }
       else
         {
          Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastErro
         }
       //--- Schließen des Sockets nach dem Ende der Verwendung
       SocketClose(socket);
      }
    else
       Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

Siehe auch

SocketTimeouts, MathSwap

© 2000-2025, MetaQuotes Ltd.

SocketSend

Schreiben von Daten auf einen Socket.

```
int  SocketSend(
   int           socket,              // Socket
   const uchar&  buffer[],            // Datenpuffer
   uint          buffer_len           // Puffergröße
   );
```

Parameter

socket

[in] Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

buffer

[in] Referenz eines Arrays vom Typ uchar mit den Daten, die gesendet werden sollen.

buffer_len

[in] 'buffer' Arraygröße.

Rückgabewert

Bei einem Erfolg wird die Anzahl der geschriebenen Bytes zurückgegeben. Im Fehlerfall wird -1 zurückgegeben.

Hinweis

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Im Falle eines Fehlers beim Schreiben wird der Fehler 5273 (ERR_NETSOCKET_IO_ERROR) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs
```

© 2000-2025, MetaQuotes Ltd.

```
input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere 
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
               }
            }
         }
      }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Schließen des Sockets nach dem Ende der Verwendung
      SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

Siehe auch

SocketTimeouts, MathSwap, StringToCharArray

© 2000-2025, MetaQuotes Ltd.

SocketTlsHandshake

Initiieren einer sicheren (SSL) Verbindung mit einem angegebenen Host über das Protokoll des TLS-Handshakes. Während des Handshakes vereinbaren Client und Server die Verbindungsparameter: das zu verwendende Protokoll und die Verschlüsselungsmethode.

```
bool  SocketTlsHandshake(
   int           socket,               // Socket
   const string  host                  // Adresse des Hosts
   );
```

Parameter

socket

[in] Das Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiges Handle erstellt, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

host

[in] Adresse des Hosts, mit dem die sichere Verbindung hergestellt werden soll.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Hinweis

Vor einer sicheren Verbindung sollte das Programm eine standardmäßige TCP-Verbindung mit dem Host über SocketConnect aufbauen.

Kommt keine die sichere Verbindung zustande, wird der Fehler 5274 (ERR_NETSOCKET_HANDSHAKE_FAILED) der Variablen _LastError zugewiesen.

Es ist nicht notwendig, bei einer Verbindung zum Port 443 diese Funktion aufzurufen. Dies ist ein standardmäßiger TCP-Port, der für sichere (SSL) TLS-Verbindungen verwendet wird.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                          SocketTlsHandshake.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"

#define   SERVER    "smtp.gmail.com"
#define   PORT      587
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Manuelles Umschalten auf eine sichere Verbindung                 |
//+------------------------------------------------------------------+
bool TlsHandshake(int socket)
  {
//--- Empfang der Begrüßung des Servers
   string rsp;

   if(!RecvString(socket,rsp))
      return(false);
//--- Begrüßung des Servers
   if(!SendString(socket,"EHLO my.domain.com\r\n"))
      return(false);
//--- Erhalt der Serverantwort mit einer Liste der unterstützten Befehle
   if(!RecvString(socket,rsp))
      return(false);
//--- Ausgabe der Begrüßung
   Print("SERVER: ",rsp);
//--- dem Server mitteilen, dass wir von einer unsicheren Verbindung zu einer sicherer
   if(!SendString(socket,"STARTTLS\r\n"))
      return(false);
//--- Empfang der Antwort des Servers
   if(!RecvString(socket,rsp))
      return(false);
//--- im Beispiel wird die Antwort des Servers über die Bereitschaft zum Umschalten au

//--- eine sichere Verbindung mit TLS (SSL) zum angegebenen Host unter Verwendung des
   if(SocketTlsHandshake(socket,InpTimeout))
      return(true);

   Print("SocketTlsHandshake() failed. Error ",GetLastError());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Socket erstellen und sein Handle zuweisen
   int socket=SocketCreate();

   if(socket==INVALID_HANDLE)
     {
      Print("SocketCreate() failed. Error ",GetLastError());
      return;
     }
//--- Verbindung zu SERVER über PORT
   if(!SocketConnect(socket,SERVER,PORT,10000))
     {
      Print("SocketConnect() failed. Error ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   else
      {
       //--- unsichere Verbindung hergestellt
       PrintFormat("%s connection has been established to %s:%d",(PORT==443 ? "A secure
       //--- Wechsel zu einer sicheren Verbindung
       if(PORT!=443 && TlsHandshake(socket))
         {
          PrintFormat("Unsecured connection to %s:%d switched to secured",SERVER,PORT);
          //--- wenn die Verbindung durch ein Zertifikat geschützt ist, zeige dessen Da
          string   subject,issuer,serial,thumbprint;
          datetime expiration;

          if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
            {
             Print("TLS Zertifikat:");
             Print("   Owner:      ",subject);
             Print("   Issuer:     ",issuer);
             Print("   Number:     ",serial);
             Print("   Print:      ",thumbprint);
             Print("   Expiration: ",expiration);
            }
         }
      }
//--- Schließen des Sockets nach Gebrauch
   SocketClose(socket);
   /*
  Ergebnis:
   An unsecured connection has been established to smtp.gmail.com:587
   SERVER: 220 smtp.gmail.com ESMTP a640c23a62f3a-a9b1f298319sm82305866b.105 - gsmtp

   SERVER: 250-smtp.gmail.com at your service, [37.193.40.122]
   250-SIZE 35882577
   250-8BITMIME
   250-STARTTLS
   250-ENHANCEDSTATUSCODES
   250-PIPELINING
   250-CHUNKING
   250 SMTPUTF8

   SERVER: 220 2.0.0 Ready to start TLS

   SocketTlsHandshake(): A secure connection to smtp.gmail.com:587 is now established
   TLS certificate:
      Owner:  /CN=smtp.gmail.com
      Issuer:  /C=US/O=Google Trust Services/CN=WR2
      Number:     1f:f4:db:2a:5a:e6:dc:52:0a:4c:05:ce:81:cc:c3:f7
      Print: d6be8af229b5329cd3d4c2789c02aa94f89b421c
      Expiration: 2024.12.30 08:25:30
```

© 2000-2025, MetaQuotes Ltd.

```
   */
  }
//+------------------------------------------------------------------+
//| Senden einer Zeichenkette an den Server                          |
//+------------------------------------------------------------------+
bool SendString(int socket,const string str)
  {
//--- die Zeichenkette in ein Zeichenarray umwandeln
   uchar data[];
   int   size=StringToCharArray(str,data,0,str.Length(),CP_UTF8);
//--- Senden der Daten an den Socket
   ResetLastError();
   if(SocketSend(socket,data,size)==size)
      return(true);
//--- Fehler während des Sendens der Daten
   Print("Failed to send data to server. Error ",GetLastError());
   return false;
  }
//+------------------------------------------------------------------+
//| Abrufen einer Zeichenkette vom Server                            |
//+------------------------------------------------------------------+
bool RecvString(int socket,string& result,uint timeout_ms=1000)
  {
//--- warten, bis Daten auf dem Socket erscheinen
   ulong wait_time_end=GetMicrosecondCount()+timeout_ms*1000;

   while(!SocketIsReadable(socket))
     {
      Sleep(10);
      //--- Zeitüberschreitung beim Warten auf Daten - als Antwort wird dann NULL zurü
      if(wait_time_end<GetMicrosecondCount())
        {
         Print("ERROR: No response from server");
         return(false);
        }
     }
//--- Lesen der Daten vom Socket
   uchar  data[128];
   uint   size=0;
   string resp=NULL;

   do
     {
      uchar b[1];
      int   n=SocketRead(socket,b,1,1000);

      if(n < 0)
         break;
```

© 2000-2025, MetaQuotes Ltd.

```
      if(n)
        {
         data[size++]=b[0];

         if(size==data.Size())
           {
            resp += CharArrayToString(data,0,data.Size(),CP_UTF8);
            size = 0;
           }
        }
     }
   while(SocketIsReadable(socket));
//--- Kopieren der gelesenen Daten in die Zeichenkette
   if(size)
      resp+=CharArrayToString(data,0,size,CP_UTF8);
//--- ist die Zeichenkette leer, dann Fehler ausgeben
   if(!resp.Length())
     {
      Print("ERROR: No response from server");
      return(false);
     }
//--- Rückgabe der Zeichenkette
   result=resp;
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

SocketTlsCertificate

Abrufen der Daten eines Zertifikates, das für eine sichere Verwendung TLS-Handshake verwandt wird.

```
bool  SocketTlsCertificate(
   int           socket,              // Socket
   string&       subject,             // Besitzer des Zertifikats
   string&       issuer,              // Aussteller des Zertifikats
   string&       serial,              // Seriennummer des Zertifikats
   string&       thumbprint,          // Ausdruck des Zertifikats
   datetime&     expiration           // Ablaufdatum des Zertifikats
   );
```

Parameter

socket

[in] Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

subject

[in] Besitzer des Zertifikats. Korrespondiert mit dem Feld subject.

issuer

[in] Aussteller des Zertifikats. Korrespondiert mit dem Feld issuer.

serial

[in] Seriennummer des Zertifikats. Korrespondiert mit dem Feld serial.

thumbprint

[in] Ausdruck des Zertifikats. Korrespondiert mit dem SHA-1 Hash der ganzen Datei des Zertifikats (all Felder umfassen die Signatur des Ausstellers).

expiration

[in] Ablaufdatum des Zertifikats im Format datetime.

Rückgabewert

Gibt true im Erfolgsfall zurück, andernfalls false.

Hinweis

Die Zertifikatsdaten können nur nach dem Aufbau einer sicheren Verbindung über SocketTlsHandshake angefordert werden.

Im Falle eines fehlerhaften Zertifikats wird der Fehler 5275 (ERR_NETSOCKET_NO_CERTIFICATE) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere 
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
           }
        }
     while(GetTickCount()<timeout_check && !IsStopped());
     return(false);
    }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
```

© 2000-2025, MetaQuotes Ltd.

```
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
        }
      else
        {
         Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
        }
      //--- Schließen des Sockets nach dem Ende der Verwendung
      SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

SocketTlsRead

Liest die Daten von einer sicheren TLS-Verbindung.

```
int  SocketTlsRead(
   int           socket,              // Socket
   uchar&        buffer[],            // Puffer für die vom Socket gelesenen Daten
   uint          buffer_maxlen        // Anzahl der zu lesenden Bytes
   );
```

Parameter

socket

[in] Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiger Handle übergeben wird, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

buffer

[out] Referenz eines Arrays vom Typ uchar, dem die Daten zugewiesen werden sollen. Die Größe des dynamischen Arrays wird auf die Anzahl der zu lesenden Bytes gesetzt. Die Arraygröße kann INT_MAX (2147483647) nicht überschreiten.

buffer_maxlen

[in] Anzahl der Bytes, die dem Array *buffer[]* zugewiesen werden sollen. Daten, die nicht in das Array passen, verbleiben im Socket. Diese können beim nächsten Aufruf von SocketTLSRead abgerufen werden. *buffer_maxlen* kann nicht größer sein als INT_MAX (2147483647).

Rückgabewert

Bei einem Erfolg wird die Anzahl der gelesenen Bytes zurückgegeben. Im Fehlerfall wird -1 zurückgegeben.

Hinweis

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Die Funktion wird ausgeführt bis der angegebenen Umfang von Daten oder der Timeout erreicht wurde (SocketTimeouts).

Im Falle eines Fehlers beim Lesen wird der Fehler 5273 (ERR_NETSOCKET_IO_ERROR) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketExample.mq5 |
//|                        Copyright 2018, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
#property copyright   "Copyright 2000-2024, MetaQuotes Ltd."
#property link        "https://www.mql5.com"
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs

input string Address="www.mql5.com";
input int    Port   =80;
bool         ExtTLS =false;
//+------------------------------------------------------------------+
//| Einen Befehl zum Server schicken                                 |
//+------------------------------------------------------------------+
bool HTTPSend(int socket,string request)
  {
   char req[];
   int  len=StringToCharArray(request,req)-1;
   if(len<0)
      return(false);
//--- Wenn eine sichere TLS-Verbindung über den Port 443 verwendet wird
   if(ExtTLS)
      return(SocketTlsSend(socket,req,len)==len);
//--- Wenn eine standardmäßige TCP-Verbindung verwendet wird
   return(SocketSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout)
  {
   char   rsp[];
   string result;
   uint   timeout_check=GetTickCount()+timeout;
//--- Lesen der Daten vom Socket solange es welche gibt, aber nicht länger als der Tir
   do
     {
      uint len=SocketIsReadable(socket);
      if(len)
        {
         int rsp_len;
         //--- Verschiedene Lesebefehle, je nach dem, ob die Verbindung eine sichere 
         if(ExtTLS)
            rsp_len=SocketTlsRead(socket,rsp,len);
         else
            rsp_len=SocketRead(socket,rsp,len,timeout);
         //--- Analysieren der Antwort
         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- Ausdruck nur des Headers der Antwort
```

© 2000-2025, MetaQuotes Ltd.

```
            int header_end=StringFind(result,"\r\n\r\n");
            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
            }
          }
      }
   while(GetTickCount()<timeout_check && !IsStopped());
   return(false);
  }
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- Verbinden, wenn alles ok ist
      if(SocketConnect(socket,Address,Port,1000))
        {
         Print("Verbindung hergestellt mit ",Address,":",Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- Wenn die Verbindung mit einem Zertifikat gesichert ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Besitzer:  ",subject);
            Print("   Aussteller:  ",issuer);
            Print("   Seriennummer:     ",serial);
            Print("   Ausdruck: ",thumbprint);
            Print("   Ablaufdatum: ",expiration);
            ExtTLS=true;
           }
         //--- Senden einer GET-Anforderung an den Server
         if(HTTPSend(socket,"GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r
           {
            Print("GET-Anforderung gesendet");
            //--- Lesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
```

© 2000-2025, MetaQuotes Ltd.

```
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
         }
       else
         {
          Print("Verbindung mit ",Address,":",Port," schlug fehl, Fehler ",GetLastError
         }
       //--- Schließen des Sockets nach dem Ende der Verwendung
       SocketClose(socket);
      }
    else
       Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
```

Siehe auch

SocketTimeouts, MathSwap

© 2000-2025, MetaQuotes Ltd.

SocketTlsReadAvailable

Lesen aller verfügbaren Daten einer sicheren TLS-Verbindung

```
int  SocketTlsReadAvailable(
   int           socket,              // Socket
   uchar&        buffer[],            // Puffer für die vom Socket gelesenen Daten
   const uint    buffer_maxlen        // Anzahl der zu lesenden Bytes
   );
```

Parameter

socket

[in] Das Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiges Handle erstellt, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

buffer

[out] Referenz eines Arrays vom Typ uchar, dem die Daten zugewiesen werden sollen. Die Größe des dynamischen Arrays wird auf die Anzahl der zu lesenden Bytes gesetzt. Die Arraygröße kann INT_MAX (2147483647) nicht überschreiten.

buffer_maxlen

[in] Anzahl der Bytes, die dem Array buffer[] zugewiesen werden sollen. Daten, die nicht in das Array passen, verbleiben im Socket. Diese können beim nächsten Aufruf von SocketTlsReadAvailable oder SocketTlsRead abgerufen werden. buffer_maxlen darf nicht größer sein als INT_MAX (2147483647).

Rückgabewert

Bei einem Erfolg wird die Anzahl der gelesenen Bytes zurückgegeben. Im Fehlerfall wird -1 zurückgegeben.

Hinweis

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Im Falle eines Fehlers beim Lesen wird der Fehler 5273 (ERR_NETSOCKET_IO_ERROR) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                      SocketTlsReadAvailable.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
```

© 2000-2025, MetaQuotes Ltd.

```
#property version     "1.00"
#property script_show_inputs
//+------------------------------------------------------------------+
//| Script Eingaben                                                  |
//+------------------------------------------------------------------+
input string InpMethod ="GET";             // Method (HEAD,GET)
input string InpServer ="www.google.com"; // Server
input uint   InpPort   =443;               // Port
input uint   InpTimeout=5000;              // Timeouts
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
   Print("Server: ",InpServer);
   Print("Port: ",InpPort);
//--- Socket erstellen und sein Handle zuweisen
   const int socket=SocketCreate();

   if(socket==INVALID_HANDLE)
     {
      Print("SocketCreate() failed. Error ",GetLastError());
      return;
     }
//--- Festlegen der Timeouts für den Empfang und das Senden von Daten eines Socket-Sys
   if(!SocketTimeouts(socket,InpTimeout,InpTimeout))
     {
      PrintFormat("SocketTimeouts(%u, %u) failed. Error %d",InpTimeout,InpTimeout,GetL
      SocketClose(socket);
      return;
     }
//--- connect to Server via Port
   if(!SocketConnect(socket,InpServer,InpPort,InpTimeout))
     {
      PrintFormat("SocketConnect('%s', %u, %u) failed. Error %d",InpServer,InpPort,Inp
      SocketClose(socket);
      return;
     }
//--- Daten über das zur Sicherung der Netzwerkverbindung verwendete Zertifikat abrufe
   string   subject,issuer,serial,thumbprint;
   datetime expiration;

   if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
     {
      Print("TLS certificate:");
      Print("   Owner:       ",subject);
      Print("   Issuer:      ",issuer);
      Print("   Number:      ",serial);
      Print("   Print:       ",thumbprint);
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("   Expiration: ",expiration);
     }
   else
     {
      //--- der Server stellt kein Zertifikat bereit - eine ungesicherte Verbindung me
      Print("The connection is not secured by a certificate");
      SocketClose(socket);
      return;
     }
//--- Senden einer Anfrage an den Server
   string request=StringFormat("%s / HTTP/1.1\r\nHost: %s\r\nUser-Agent: MetaTrader 5\

   if(HTTPSendTLS(socket,request))
     {
      //--- Request gesendet - Response erhalten
      Print("\nRequest sent. Starting page loading...");
      uchar response[]; // alle erhaltenen Daten (Header und Haupteil des Dokuments)

      if(!HTTPRecvTLS(socket,response,InpTimeout))
        {
         Print("There were errors while reading the page");
         SocketClose(socket);
         return;
        }
      //--- Ausgabe der Anzahl der empfangenen Datenbytes
      PrintFormat("%u bytes received",response.Size());
      //--- Ausgabe nur des Headers der erhaltenen Seite
      string result    =CharArrayToString(response,0,WHOLE_ARRAY,CP_UTF8);
      int    header_end=StringFind(result,"\r\n\r\n");

      if(header_end>0)
        {
         Print("\nHTTP answer header received:");
         Print(StringSubstr(result,0,header_end));
        }
     }
//--- Schließen des Sockets nach Gebrauch
   SocketClose(socket);
   /*
   Ergebnis:
   Server: www.google.com
   Port: 443
   TLS certificate:
      Owner: /CN=www.google.com
      Issuer: /C=US/O=Google Trust Services/CN=WR2
      Number: 0d:43:b1:4a:bb:9c:15:96:10:e1:3d:55:23:9f:25:4e
      Print: 89167618e5017f813aff981c88ce422dc1016bdf
      Expiration: 2024.12.30 08:26:35
```

© 2000-2025, MetaQuotes Ltd.

```
   Request sent. Starting page loading...
   HTTPRecvTLS: Document received within 27 attempts
   25185 bytes received

   HTTP answer header received:
   HTTP/1.1 200 OK
   Date: Fri, 25 Oct 2024 17:12:42 GMT
   Expires: -1
   Cache-Control: private, max-age=0
   Content-Type: text/html; charset=ISO-8859-1
   Content-Security-Policy-Report-Only: object-src 'none';base-uri 'self';script-src '
   Accept-CH: Sec-CH-Prefers-Color-Scheme
   P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
   Server: gws
   X-XSS-Protection: 0
   X-Frame-Options: SAMEORIGIN
   Set-Cookie: AEC=AVYB7coyYMCdweTDTaWeGYzmRnxzKGqsOEosH_VkbCn8xhWkFz6v0kxQFw; expires
   Set-Cookie: NID=518=J02X02Ff4v_9sMcNoUz-1SolmuG08E26Gs438ik0J_SOJUMy7of-P-qup-LaNSW
   Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
   Accept-Ranges: none
   Vary: Accept-Encoding
   Transfer-Encoding: chunked
   */
  }
//+------------------------------------------------------------------+
//| Senden einer HTTP-Anfrage über eine sichere Verbindung           |
//+------------------------------------------------------------------+
bool HTTPSendTLS(int socket,const string request)
  {
//--- Umwandeln der Zeichenkette in ein Zeichenarray mit dem Löschen der abschließende
   char req[];
   int  len=StringToCharArray(request,req,0,WHOLE_ARRAY,CP_UTF8)-1;

   if(len<0)
      return false;

   return(SocketTlsSend(socket,req,len)==len);
  }
//+------------------------------------------------------------------+
//| Abrufen einer Web-Seite über eine sichere Verbindung             |
//+------------------------------------------------------------------+
bool HTTPRecvTLS(int socket,uchar &response[],const uint timeout_ms)
  {
//--- Lesen der verfügbaren Daten einer sicheren TLS-Verbindung, bevor die Zeitübersch
   ulong timeout_check=GetTickCount64()+timeout_ms;
   uchar block[1024];   // Puffer für den Datenblock, gelesen vom Soket
   uint  attempt=0;     // Angeforderte Anzahl von Datenblocks
   int   err    =0;     // Fehlernummer
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();

   do
     {
      //--- Lesen der Blöcke, maximal 1024 Bytes
      int len=SocketTlsReadAvailable(socket,block,1024);

      if(len>0)
        {
         attempt++;
         //--- Zusammenführen der erhaltenen Datenblöcke
         ArrayCopy(response,block,response.Size());
         //--- Auswerten der erhaltenen Daten, Bestimmen des Headers, dem Hauptteil de
         //...
         //...
         //...
         timeout_check=GetTickCount64()+timeout_ms;
        }
      else
         Sleep(10);

      err=GetLastError();
     }
   while(!IsStopped() && GetTickCount()<timeout_check && !err);
//--- Gab es irgendwelche Fehler während des Lesens?
   if(err)
     {
      Print("Error ",err);
      return(false);
     }

   PrintFormat("%s: Document received within %d attempts",__FUNCTION__,attempt);
   return(true);
  }
```

Siehe auch

SocketTimeouts, MathSwap

© 2000-2025, MetaQuotes Ltd.

SocketTlsSend

Senden von Daten über eine sichere TLS-Verbindung

```
int  SocketTlsSend(
   int           socket,              // Socket
   const uchar&  buffer[],            // Datenpuffer
   uint          buffer_len           // Puffergröße
   );
```

Parameter

socket

[in] Das Handle des Sockets, das von der Funktion SocketCreate erzeugt worden war. Wurde ein ungültiges Handle erstellt, wird der Fehler 5270 (ERR_NETSOCKET_INVALIDHANDLE) der Variablen _LastError zugewiesen.

buffer

[in] Referenz des Arrays vom Typ uchar mit den Daten, die gesendet werden sollen.

buffer_len

[in] 'buffer' Arraygröße.

Rückgabewert

Bei einem Erfolg wird die Anzahl der geschriebenen Bytes zurückgegeben. Im Fehlerfall wird -1 zurückgegeben.

Hinweis

Im Fehlerfall eines System-Sockets wird, beim Aufruf der Funktion, die Verbindung, die mit SocketConnect erstellt wurde, beendet.

Im Falle eines Fehlers beim Schreiben wird der Fehler 5273 (ERR_NETSOCKET_IO_ERROR) der Variablen _LastError zugewiesen.

Die Funktion kann nur von Expert Advisors und Skripten aufgerufen werden, da sie in ihrem eigenen Ausführungsthread laufen. Wenn sie ein Indikator aufruft, wird von GetLastError() der Fehler 4014 - "Funktionsaufruf ist nicht erlaubt" ausgeworfen.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                SocketTlsSend.mq5 |
//|                                Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"
#property description "Fügen Sie die Adresse der Liste mit den erlaubten in der Einste
#property script_show_inputs
```

© 2000-2025, MetaQuotes Ltd.

```
input string Address="www.mql5.com";
input int    Port   =443;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
   int socket=SocketCreate();
//--- Prüfen des Handles
   if(socket!=INVALID_HANDLE)
     {
      //--- wenn alles in Ordnung ist, verbinde
      if(SocketConnect(socket,Address,Port,1000))
        {
         PrintFormat("Established connection to %s:%d",Address,Port);

         string   subject,issuer,serial,thumbprint;
         datetime expiration;
         //--- wenn die Verbindung durch ein Zertifikat geschützt ist, zeige dessen Da
         if(SocketTlsCertificate(socket,subject,issuer,serial,thumbprint,expiration))
           {
            Print("TLS Zertifikat:");
            Print("   Owner:      ",subject);
            Print("   Issuer:     ",issuer);
            Print("   Number:     ",serial);
            Print("   Print:      ",thumbprint);
            Print("   Expiration: ",expiration);
           }
         //--- Senden einer GET-Anforderung an den Server
         string request="GET / HTTP/1.1\r\nHost: www.mql5.com\r\nUser-Agent: MT5\r\n\r
         char   req[];
         int    len=StringToCharArray(request,req)-1;

         if(len<0)
           {
            Print("StringToCharArray() failed. Error ", GetLastError());
            SocketClose(socket);
            return;
           }
         //--- wenn eine sichere TLS-Verbindung über Port 443 verwendet wird
         if(SocketTlsSend(socket,req,len)==len)
           {
            Print("GET-Anforderung gesendet");
            //--- Auslesen der Antwort
            if(!HTTPRecv(socket,1000))
               Print("Fehlerhafte Antwort, Fehler ",GetLastError());
           }
         else
            Print("Fehler bei der GET-Anforderung, Fehler ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
        }
      else
        {
         PrintFormat("Connection to %s:%d failed, error %d",Address,Port,GetLastError
        }
      //--- Schließen des Sockets nach Verwendungsende
      SocketClose(socket);
     }
   else
      Print("Fehler beim Erstellen des Sockets, Fehler ",GetLastError());
  }
//+------------------------------------------------------------------+
//| Lesen der Antwort vom Server                                     |
//+------------------------------------------------------------------+
bool HTTPRecv(int socket,uint timeout_ms)
  {
//--- Daten aus dem Socket lesen, solange welche vorhanden sind, aber nicht länger als
   char   rsp[];
   string result;
   ulong  timeout_check=GetTickCount64()+timeout_ms;

   do
     {
      uint len=SocketIsReadable(socket);

      if(len)
        {
         //--- die Daten einer sicheren TLS-Verbindung lesen und auswerten
         int rsp_len=SocketTlsRead(socket,rsp,len);

         if(rsp_len>0)
           {
            result+=CharArrayToString(rsp,0,rsp_len);
            //--- nur den Antwort-Header anzeigen
            int header_end=StringFind(result,"\r\n\r\n");

            if(header_end>0)
              {
               Print("Header der HTTP-Antwort erhalten:");
               Print(StringSubstr(result,0,header_end));
               return(true);
              }
            //--- Ablaufzeit-Aktualisierung der Lesezeitüberschreitung
            timeout_check=GetTickCount64()+timeout_ms;
           }
        }
     }
   while(GetTickCount64()<timeout_check && !IsStopped());
```

© 2000-2025, MetaQuotes Ltd.

```
   return(false);
  }
```

Siehe auch

SocketTimeouts, MathSwap, StringToCharArray

© 2000-2025, MetaQuotes Ltd.

WebRequest

Die Funktion sendet eine HTTP-Anfrage an den angegebenen Server. Es gibt zwei Varianten der Funktion:

1. **Für das Senden einer einfachen Anfrage**vom Typ "Schlüssel=Wert" mit dem Header Content-Type: application/x-www-form-urlencoded.

```
int  WebRequest(
   const string      method,           // HTTP-Methode
   const string      url,              // URL-Adresse
   const string      cookie,           // Cookie
   const string      referer,          // Referer
   int               timeout,          // Timeout
   const char        &data[],          // Array des Body einer HTTP-Nachricht
   int               data_size,        // Größe des data[]-Arrays in Bytes
   char              &result[],        // Array mit den Daten der Serverantwort
   string            &result_headers   // Headers der Serverantwort
   );
```

2. **Für das Senden einer Anfrage beliebigen Typs** mit einem eigenen Set von Headers für eine flexiblere Interaktion mit verschiedenen Webservices.

```
int  WebRequest(
   const string      method,           // HTTP-Methode
   const string      url,              // URL-Adresse
   const string      headers,          // Headers
   int               timeout,          // Timeout
   const char        &data[],          // Array des Body einer HTTP-Nachricht
   char              &result[],        // Array mit den Daten der Serverantwort
   string            &result_headers   // Headers der Serverantwort
   );
```

Parameter

method

[in] HTTP-Methode.

url

[in] URL-Adresse

headers

[in] Headers vom Type "Schlüssel=Wert", die durch Zeilenumbruch "\r\n" getrennt sind.

cookie

[in] Wert von Cookie.

referer

[in] Wert von Referer der HTTP-Anfrage.

timeout

© 2000-2025, MetaQuotes Ltd.

[in] Timeout in Millisekunden.

data[]

[in] Array für die Daten des Body von HTTP-Nachrichten.

data_size

[in] Größe des data[]-Arrays.

result[]

[out] Array mit den Daten der Serverantwort

result_headers

[out] Headers der Serverantwort.

Rückgabewert

Antwortcode des HTTP-Servers oder -1 im Fehlerfall.

Hinweis

Um die WebRequest-Funktion zu verwenden, schreiben Sie die Serveradressen in die Liste der erlaubten URLs auf der Registerkarte "Experten" im Fenster "Einstellungen". Der Serverport wird auf der Grundlage des angegebenen Protokolls automatisch ausgewählt - 80 für "http://" und 443 für "https://".

Die WebRequest()-Funktion ist synchron, das heißt, sie unterbricht die Ausführung des Programms und wartet auf eine Antwort von dem angefragten Server. Da die Verzögerungen beim Erhalten einer Antwort auf die gesendete Anfrage groß sein können, ist es verboten, die Funktion von Indikatoren aus aufzurufen, denn die Indikatoren arbeiten in einem Thread, der für alle Indikatoren und Charts dieses Symbols gemeinsam ist. Die Verzögerung bei der Ausführung eines Indikators auf einem der Symbolcharts kann die Aktualisierung aller Charts dieses Symbols stoppen.

Die Funktion kann nur von Expert Advisors und Skripts aus aufgerufen werden, denn diese arbeiten in einem eigenen Thread. Wenn die Funktion aus einem Indikator aufgerufen wird, gibt GetLastError() den Fehler 4014 zurück - "Function is not allowed for call" (Die Funktion darf nicht aufgerufen werden.

Die WebRequest()-Funktion kann nicht im Strategietester ausgeführt werden.

Beispiel:

```
void OnStart()
 {
   string cookie=NULL,headers;
   char   post[],result[];
   string url="https://finance.yahoo.com";
//--- für die Arbeit mit dem Server muss die URL "https://finance.yahoo.com"
//--- zur Liste der erlaubten URLs hinzugefügt werden (Hauptmenü->Extras->Optionen, Re
//--- Setzen des Codes des letzten Fehlers auf Null
   ResetLastError();
//--- Herunterladen der HTML-Seite von Yahoo Finance
   int res=WebRequest("GET",url,cookie,NULL,500,post,0,result,headers);
   if(res==-1)
    {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("Fehler in WebRequest. Fehlercode  =",GetLastError());
      //--- kann sein, dass die URL in der Liste nicht vorhanden ist, geben wir eine M
      MessageBox("Die URL '"+url+"' muss zur Liste der erlaubten URLs im Reiter 'Exper
     }
   else
    {
     if(res==200)
      {
       //--- Herunterladen erfolgreich
       PrintFormat("Die Datei wurde erfolgreich heruntergeladen, Größe %d Byte.",Arr
       //PrintFormat("Headers des Servers: %s",headers);
       //--- Speichern der Daten in einer Datei
       int filehandle=FileOpen("url.htm",FILE_WRITE|FILE_BIN);
       if(filehandle!=INVALID_HANDLE)
        {
         //--- Speichern des Inhaltes des Arrays result[] in einer Datei
         FileWriteArray(filehandle,result,0,ArraySize(result));
         //--- Datei schließen
         FileClose(filehandle);
        }
       else
          Print("Fehler in FileOpen. Fehlercode =",GetLastError());
      }
     else
        PrintFormat("Herunterladen von '%s' fehlgeschlagen, Fehlercode %d",url,res);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SendFTP

Sendet die Datei an die im Fenster der Terminal-Optionen im Karteireiter "FTP" angegebenen Adresse.

```
bool  SendFTP(
   string  filename,          // Datei, die per FTP gesendet werden soll
   string  ftp_path=NULL      // Pfad für das Hochladen auf den FTP-Server
   );
```

Parameter

filename

[in] Name der zu sendenden Datei.

ftp_path=NULL

[in] Verzeichnispfad der FTP-Datei. Wenn das Verzeichnis nicht angegeben wird, wird das Verzeichnis verwendet, das in Einstellungen beschrieben wird.

Rückgabewert

Beim Misserfolg wird false zurückgegeben.

Hinweis

Die gesendete Datei muss im Ordner *Terminal_Verzeichnis\MQL5\files* oder einem Unterordner sein. Senden wird nicht durchgeführt, wenn in Einstellungen keine FTP Adresse und/oder kein Zugangspasswort angegeben wurden.

Bei einer Verwendung im Strategie-Tester wird die SendFTP()-Funktion nicht ausgeführt.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                     SendFTP.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"

#define   FILENAME  "SomeFile.bin"
#define   PATH      NULL
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Prüfen der Berechtigung zum Senden von Dateien an die FTP-Adresse im Terminal
   if(!TerminalInfoInteger(TERMINAL_FTP_ENABLED))
     {
      Print("Error. The client terminal does not have permission to send messages to t
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
     }
//--- Senden der Datei
   ResetLastError();
   if(!SendFTP(FILENAME, PATH))
      Print("SendFTP() failed. Error ",GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

SendMail

Sendet die Email an die Adresse, angegeben im Fenster der Terminal-Optionen im Karteireiter "E-Mail".

```
bool  SendMail(
   string  subject,        // Betreff
   string  some_text       // Text der Email
   );
```

Parameter

subject

[in] Betreff der Email.

some_text

[in] Text der Email.

Rückgabewert

true - wenn die E-Mail in die Sendequeue gestellt wird, sonst false.

Hinweis

Senden kann in Einstellungen verboten sein, es kann aber auch die E-Mail-Adresse nicht angegeben worden sein. Für Fehlerinformation muss man die Funktion GetLastError() aufrufen.

Bei einer Verwendung im Strategie-Tester wird die SendMail()-Funktion nicht ausgeführt.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                     SendMail.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"

#define   SUBJECT   "Test SendMail"
#define   TEXT      "Text for SendMail() function"
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Berechtigung zum Senden von E-Mails im Terminal prüfen
   if(!TerminalInfoInteger(TERMINAL_EMAIL_ENABLED))
     {
      Print("Error. The client terminal does not have permission to send email message
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Senden der E-Mail
   ResetLastError();
   if(!SendMail(SUBJECT, TEXT))
      Print("SendMail() failed. Error ",GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

SendNotification

Sendet eine Benachrichtigung an mobilen Terminals, deren MetaQuotes-IDs in dem Optionen-Fenster auf der Registerkarte "Benachrichtigungen" angegeben sind.

```
bool  SendNotification(
   string  text          // Text der Benachrichtigung
   );
```

Parameter

text

[in] Text der Benachrichtigung. Die Benachrichtigung darf nicht länger als 255 Zeichen sein.

Rückgabewert

true, wenn Benachrichtigung erfolgreich vom Terminal gesendet wurde, ansonsten wird false zurückgegeben. Bei der Überprüfung nach dem Fehlschlagen der benachrichtigen GetLastError() kann ein der folgenden Fehler ausgeben:

- 4515 - ERR_NOTIFICATION_SEND_FAILED,
- 4516 - ERR_NOTIFICATION_WRONG_PARAMETER,
- 4517 - ERR_NOTIFICATION_WRONG_SETTINGS,
- 4518 - ERR_NOTIFICATION_TOO_FREQUENT.

Hinweis

Der Funktion SendNotification() sind enge Einschränkungen bei der Verwendung gesetzt: nicht mehr als 2 Anrufe pro Sekunde und nicht mehr als 10 Anrufe pro Minute. Die Überwachung der Häufigkeit der Nutzung ist dynamisch, und die Funktion kann bei der Verletzung deaktiviert werden.

Bei der Anwendung im Strategie-Tester wird die SendNotification()-Funktion nicht ausgeführt.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                             SendNotification.mq5 |
//|                                  Copyright 2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com
#property version     "1.00"

#define   MESSAGE   "Test Message"
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart(void)
  {
//--- Prüfen der Berechtigung zum Senden von Benachrichtigungen im Terminal
   if(!TerminalInfoInteger(TERMINAL_NOTIFICATIONS_ENABLED))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      Print("Error. The client terminal does not have permission to send notifications
      return;
     }
//--- Senden der Benachrichtigung
   ResetLastError();
   if(!SendNotification(MESSAGE))
      Print("SendNotification() failed. Error ",GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

Globalvariablen des Client-Terminals

Gruppe der Funktionen, die für Arbeit mit Globalvariablen bestimmt sind.

Globale Variablen des Client-Terminals muessen nicht mit den im globalen Massstab erklärten Variablen des mql5-Programms verwechselt werden.

Globale Variablen befinden sich im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugang, dann werden sie automatisch entfernt. Zugang zur globalen Variable ist nicht nur die Einstellung des neuen Wertes, sondern auch Lesen des Wertes der globalen Variable.

Globale Variablen des Client-Terminals sind gleichzeitig aus allen mql5-Programmen zugaenglich, die im Client-Terminal gestartet sind.

Funktion	Massnahme
GlobalVariableCheck	Prueft, ob die globale Variable mit dem angegebenen Namen existiert
GlobalVariableTime	Gibt die Zeit des letzten Zuganges zur globalen Variable zurück
GlobalVariableDel	entfernt globale Variable
GlobalVariableGet	Fordert den Wert der globalen Variable an
GlobalVariableName	Gibt den Namen der globalen Variable nach der laufenden Nummer in der Liste der globalen Variable
GlobalVariableSet	Stellt den neuen Wert der globalen Variable fest
GlobalVariablesFlush	Speichert den Inhalt aller Variablen auf der Platte zwangsmaessig
GlobalVariableTemp	Stellt den neuen wert der globalen Variable fest, die nur während der laufenden Terminalsitzung existiert
GlobalVariableSetOnCondition	Stellt den neuen Wert der existierenden globalen Variable nach Bedingung
GlobalVariablesDeleteAll	Entfernt globale Variablen mit dem angegebenen Praefix im Namen
GlobalVariablesTotal	Gibt die ganze Zahl der globalen Variablen zurück

© 2000-2025, MetaQuotes Ltd.

GlobalVariableCheck

Prueft das Existieren der globalen Variable des Client-Terminals.

```
bool  GlobalVariableCheck(
   string  name      // Name
   );
```

Parameter

name

[in] Name der globalen Variable.

Rückgabewert

Gibt de Wert true zurück, wenn die globale Variable existiert, anderenfalls gibt false zurück.

Hinweis

Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugriffs, danach werden sie automatisch entfernt.

Sehen Sie auch

GlobalVariableTime()

© 2000-2025, MetaQuotes Ltd.

GlobalVariableTime

Gibt die Zeit des letzten Zugangs zur globalen Variable zurück.

```
datetime  GlobalVariableTime(
   string  name      // Name
   );
```

Parameter

name

[in] Name der globalen Variable.

Rückgabewert

Gibt die Zeit des letzten Zugangs zu angegebenen globalen Variable. Zugriff auf den Wert einer Variablen, z.B. mit der Funktion GlobalVariableGet() und GlobalVariableCheck(), verändert auch die Zeit des letzten Zugriffs. Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugriffs, danach werden sie automatisch entfernt.

Sehen Sie auch

GlobalVariableCheck()

© 2000-2025, MetaQuotes Ltd.

GlobalVariableDel

Entfernt globale Variable des Client-Terminals.

```
bool  GlobalVariableDel(
   string  name      // Name
   );
```

Parameter

name

[in] Name der globalen Variable.

Rückgabewert

Bei der erfolgreichen Entfernung gibt die Funktion true zurück, anderenfalls false. Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugriffs, danach werden sie automatisch entfernt.

© 2000-2025, MetaQuotes Ltd.

GlobalVariableGet

Gibt de Wert der globalen Variable des Client-Terminals zurück. Es gibt zwei Varianten der Funktion.

1. Gibt den wert der Eigenschaft sofort zurück.

```
double  GlobalVariableGet(
   string  name      // Name
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird der Wert der Variable des Client-Terminals in die Enpfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  GlobalVariableGet(
   string  name               // Name
   double& double_var         // hier nehmen wir den Wert der globalen Variable auf
   );
```

Parameter

name

[in] Name der globalen Variable.

double_var

[out] Variable des Typs double, die den Wert annimmt, der in der globalen Veriable des Client-Terminals aufbewahren wird.

Rückgabewert

Wert der existierenden globalen Variable oder 0 beim Fehler. Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugriffs, danach werden sie automatisch entfernt.

© 2000-2025, MetaQuotes Ltd.

GlobalVariableName

Gibt den Wert der globalen Variable nach der Ordnungsnummer zurück.

```
string  GlobalVariableName(
   int  index      // Nummer in der Liste der globalen Variablen
   );
```

Parameter

index

[in] Ordnungsnummer in der Liste der globalen Variablen Muss groesser als oder 0 sein und weniger als GlobalVariablesTotal() .

Rückgabewert

Name der globalen Variable nach der Ordnungsnummer in der Liste der globalen Variablen. Fur die Erhaltung der fehlerbezogenen Information muss die Funtion GetLastError() aufgerufen werden .

Hinweis

Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem lezten Zugriff, danach werden sie automatisch entfernt.

© 2000-2025, MetaQuotes Ltd.

GlobalVariableSet

Stellt einen neuen Wert der globalen Variable. Wenn die Variable existiert nicht, erzeugt das System eine neue Variable.

```
datetime  GlobalVariableSet(
   string  name,       // Name
   double  value       // der festgestellte Wert
   );
```

Parameter

name

[in] Name der globalen Variable.

value

[in] Neuer numerische Wert.

Rückgabewert

Bei der erfolgreichen Durchführung gibt die Funktion die Zeit des letzten Zugangs, anderenfalls 0. Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Der Name der globalen Variablen darf nicht länger als 63 Symbole sein. Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugriff,danach werden sie automatisch entfernt.

© 2000-2025, MetaQuotes Ltd.

GlobalVariablesFlush

Speichert den Inhalt aller globalen Variablen in der Platte zwangsmaessig.

```
void  GlobalVariablesFlush();
```

Rückgabewert

Keinen Rückgabewert.

Hinweis

Das Terminal speichert alle globalen Variablen, wenn die Arbeit zu Ende ist, aber Daten können verloren werden beim ploetzlichen Fehler des Computers. Diese Funktion erlaubt das Prozess der Speicherung der globalen Variablen selbststaendig zu verwalten, falls unvorhergesehene Ereignisse vorkommen.

© 2000-2025, MetaQuotes Ltd.

GlobalVariableTemp

Versucht, temporaere Variable zu erzeugen. Wenn die Variable existiert nicht, erzeugt das System eine neue temporaere Variable.

```
bool  GlobalVariableTemp(
   string  name,      // Name
   );
```

Parameter

name

[in] Name der temporaeren globalen Variable.

Rückgabewert

Bei der erfolgreichen Durchführung gibt die Funktion true zurück, anderenfalls false. Für die Erhaltung der fehlerbezogenen Information, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Temporaere globale Variablen existieren nur während der Session des Client-Terminals, nach Schliessen des Terminlas werden sie automatisch entfernt. Bei der Durchführung der Operation GlobalVariablesFlush() werden globale Variablen in der Platte nicht gespeichert.

Nach Erzeugung der temporaeren globalen Variable wird der Zugang zu ihr und ihre Modifikation ebenso, wie zur normalen globalen Variable des Client-Terminals durchgeführt..

© 2000-2025, MetaQuotes Ltd.

GlobalVariableSetOnCondition

Stellt einen neuen Wert der existierenden globalen Variable, wenn der laufende Wert der Variable dem Wert des dritten Parameters check_value gleich wird. Wenn es keine globale Variable gibt, wird die Funktion den Fehler ERR_GLOBALVARIABLE_NOT_FOUND (4501) generieren und false zurückgeben.

```
bool  GlobalVariableSetOnCondition(
   string  name,             // Name
   double  value,            // Wert bei der Erfuellung der Bedingung
   double  check_value       // die gepruefte Bedingung
   );
```

Parameter

name

[in] Name der globalen Variable.

value

[in] Neuer Wert.

check_value

[in] Wert für die Pruefung des laufenden Wertes der globalen Variable.

Rückgabewert

hführung gibt die Funktion true zurück, anderenfalls false. Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.Wenn sich der laufende Wert der globalen Variable von check_value unterscheidet, gibt die Funktion false zurück.

Hinweis

Funktion gewaehrt automaren Zugang zur globalen Variable, darum kann sie Semaphor gewaehren bei der Interaktion der Experten, die gleichzeitig innerhalb eines Client-Terminals arbeiten.

© 2000-2025, MetaQuotes Ltd.

GlobalVariablesDeleteAll

Entfernt globale Variablen des Client-Terminals.

```
int  GlobalVariablesDeleteAll(
   string     prefix_name=NULL      // alle globalen Variablen, deren Namen mit Praefi
   datetime   limit_data=0          // alle globalen Variablen, die vor diesem Datum v
   );
```

Parameter

prefix_name=NULL

[in] Praefixname der entfernten globalen Variablen. Wenn das Praefix NULL oder Leerzeile angegeben wird, werden alle globalen Variablen, die dem Datumkriterium entsprechen, entfernt.

limit_data=0

[in] Datum für Auswahl der globalen Variablen nach der Zeit der letzten Modifikation. Alle Variablen, die vor dem angegebenen Datum verändert wurden, werden entfernt. Wenn der Parameter gleich Null ist, werden alle Variablen, die dem ersten Kriterium entsprechen (Praefix), entfernt.

Rückgabewert

Anzahl der entfernten Variablen.

Hinweis

Wenn die beiden Parameter gleich Null sind (prefix_name=NULL und limit_data=0), werden alle globale Variablen des Client-Terminals entfernt. Wenn beide Parameter angegeben werden, werden globale Variablen, die beiden Parametern entsprechen, entfernt.

Globale Variablen existieren im Client-Terminal innerhalb von 4 Wochen seit dem letzten Zugriffs, danach werden sie automatisch entfernt.

© 2000-2025, MetaQuotes Ltd.

GlobalVariablesTotal

Gibt die gesamte Anzahl der globalen Variablen des Client-Terminals zurück.

```
int  GlobalVariablesTotal();
```

Rückgabewert

Anzahl der globalen Variablen.

Hinweis

Globale Variablen existieren im Client-Terminal innerhab von 4 Wochen seit dem letzten Zugriffs, danach werden sie automatisch entfernt. Zugriff zur globalen Variable ist nicht nur die Einstellung des neuen wertes, sondern auch das Lesen des Wertes der globalen Variable.

© 2000-2025, MetaQuotes Ltd.

Dateioperationen

Gruppe der Funktionen für Arbeit mit Dateien.

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Es gibt zwei Verzeichnisse (mit Subverzeichnissen), in denen Arbeitsdateien befinden können:

- Terminal_data_folder\MQL5\FILES\ (Im Menue des Terminals wählen Sie für Durchsicht Manüpunkte "Datei"-"Datenverzeichnis öffnen");
- der gesamter Ordner aller im Computer installierten Terminals - befinden sich normelerweise im Verzeichnis C:\Documents and Settings\All Users\Application Data\MetaQuotes\Terminal\Common\Files.

Durch das Programmverzeichnis kann man Benennungen dieser dieser Verzeichnisse durch die Funktion TerminalInfoString() erhalten werden, beim Verwenden von Enumerationen ENUM_TERMINAL_INFO_STRING:

```
//--- Ordner, in dem Terminaldaten aufbewahren werden
   string terminal_data_path=TerminalInfoString(TERMINAL_DATA_PATH);
//--- der Gesamtordner aller Client-Terminals
   string common_data_path=TerminalInfoString(TERMINAL_COMMONDATA_PATH);
```

Arbeit mit Dateien aus anderen Verzeichnissen wird untersagt.

Die Datei Funktionen können Sie mit sogenannten "die genannten Kanäle" arbeiten. Dazu ist genügend es, die Funktion FileOpen() mit den entsprechenden Parametern zu rufen</t2.

Funktion	Massnahme
FileSelectDialog	Dialog zum Erstellen einer Datei oder eines Verzeichnisses
FileFindFirst	Fängt die Suche der Dateien im entsprechenden Verzeichnis nach dem angegebenen Filter an
FileFindNext	Setzt die Suche fort, die durch die Funktion FileFindFirst() angefangen wurde
FileFindClose	Schließt die Suche Handle
FileOpen	Öffnet die Datei mit dem angegebenen Namen und mit der angegebenen Flagge
FileDelete	Entfernt die angegebene Datei
FileFlush	Speicher auf der Platte alle Daten, die im Dateipuffer Eingabe-Ausgabe noch bleiben
FileGetInteger	Ruft eine ganzzahlige Eigenschaft der Datei
FileIsEnding	Bestimmt das Ende der Datei im während des Lesens
FileIsLineEnding	Bestimmt Zeilenende während des Lesens
FileClose	Schließt die früher geöffnete Datei

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
FileIsExist	Prüft, ob die Datei existiert
FileCopy	Kopiert die Ausgangsdatei aus dem lokalen oder gesamten Ordner in eine andere Datei
FileMove	Bewegt oder benennt die Datei um
FileReadArray	Liest Arrays verschiedener Typen außer Zeilentypen (es kann ein Strukturarray sein ohne Zeilen und die dynamische Arrays) aus der binären Datei von der laufenden Position des Dateiindikators
FileReadBool	Liest aus der Datei des Typs CSV die Zeile von der laufenden Position bis zum Begrenzer (oder bis zum Ende der Textzeile) und wandelt die gelesene Zeile in den Wert des Typs bool
FileReadDatetime	Liest aus der Datei des Typs CSV die Zeile eines der Formate: "YYYY.MM.DD HH:MI:SS", "YYYY.MM.DD" oder "HH:MI:SS" - und wandelt sie in den Wert des Typs datetime um
FileReadDouble	Liest die Zahl der doppelten Genauigkeit mit dem Fließpunkt (double) aus der binären Datei von der laufenden Position des Dateiindikators
FileReadFloat	Liest den Wert float von der laufenden Position des Dateianzeigers
FileReadInteger	Liest den Wert des Typs int, short oder charvon der laufenden Position des Dateianzeigers abhängig von der angegebenen Länge in Bytes
FileReadLong	Liest den Wert des Typs long von der laufenden Position des Dateianzeigers
FileReadNumber	Liest aus der Datei des Typs CSV die Zeile von der laufenden Position zum Begrenzer (oder bis zum Ende der Textzeile) und wandelt die gelesene Zeile in den Wert des Typs double um
FileReadString	Liest die Zeile aus der Datei von der laufenden Position des Dateianzeigers
FileReadStruct	Liest Inhalt aus der binären Datei in die Struktur, die als Parameter übertragen wurde
FileSeek	Bewegt die Position des Dateiindikators um die angegebene Zahl der Bytes gegen die angegebene Position
FileSize	Gibt die Größe der entsprechenden geöffneten Datei zurück
FileTell	Gibt die laufende Position des Indikators der entsprechenden geöffneten Datei zurück
FileWrite	Schreibt Daten in die Datei des Typs CSV oder TXT
FileWriteArray	Schreibt in die Datei des Typs BIN Arrays aller Typen außer Zeilentypen

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
FileWriteDouble	Schreibt den Parameterwert des Typs double von der angegebenen Position des Dateianzeigers in die binäre Datei
FileWriteFloat	Schreibt den Parameterwert des Typs float von der angegebenen Position des Dateianzeigers
FileWriteInteger	Schreibt den Parameterwert des Typs int in die binäre Datei von der laufenden Position des Dateianzeigers
FileWriteLong	Schreibt den Parameterwert des Typs long in die binäre Datei von der laufenden Position des Dateianzeigers
FileWriteString	Schreibt den Parameterwert des Typs string in die Datei des Typs BIN oder TXT von der laufenden Position des Dateianzeigers
FileWriteStruct	Schreibt den Inhalt der Struktur, die als Parameter übertragen wurde, in die binäre Datei von der laufenden Position des Dateianzeigers
FileLoad	Liest alle Daten der angegebenen binären Datei in das Array numerischer Typen oder einfacher Strukturen
FileSave	Schreibt alle Elemente des als Parameter übermittelten Arrays in eine binäre Datei
FolderCreate	Erzeugt ein Verzeichnis im Ordner Files (abhängig von dem Wert common_flag)
FolderDelete	Entfernt das angegebene Verzeichnis. Wenn der Ordner nicht leer ist, kann er entfernt werden
FolderClean	Entfernt alle Dateien im angegebenen Ordner

Wenn Datei für Schreiben durch die Funktion FileOpen() geöffnet wird, werden alle im Pfad angegebenen Unterordner erzeugt werden, wenn sie fehlen.

© 2000-2025, MetaQuotes Ltd.

FileSelectDialog

Dialog zum Erstellen einer Datei oder eines Verzeichnisses.

```
int  FileSelectDialog(
   string   caption,              // Kopfzeile des Fensters
   string   initial_dir,          // Anfangsverzeichnis
   string   filter,               // Erweiterungsfilter
   uint     flags,                // Kombination der Flags
   string&  filenames[],          // Array mit den Dateinamen
   string   default_filename      // standardmäßiger Dateiname
   );
```

Parameter

caption

[in] Kopfzeile des Dialogfensters.

initial_dir

[in] Name des Anfangsverzeichnisses relativ zu MQL5\Files, dessen Inhalt im Dialogfenster angezeigt werden soll. Ist dieser Wert NULL, wird MQL5\Files im Dialogfenster angezeigt.

filter

[in] Filtert die Dateien gemäß der angegebenen Erweiterung, die im Dialogfenster angezeigt werden soll. Dateien in anderem Format werden ausgeblendet.

flags

[in] Kombination der Flags, die den Modus des Dialogfensters festlegen. Die Flags sind wie folgt definiert:
FSD_WRITE_FILE - Dialog zum Öffnen einer Datei;
FSD_SELECT_FOLDER - nur die Auswahl von Ordnern erlauben;
FSD_ALLOW_MULTISELECT - die Auswahl mehrerer Dateien erlauben;
FSD_FILE_MUST_EXIST - ausgewählte Dateien müssen existieren;
FSD_COMMON_FOLDER - die Datei befindet sich im gemeinsamen Ordner aller Client-Terminals \Terminal\Common\Files.

filenames[]

[out] Array mit den Namen als Zeichenketten der ausgewählten Dateien/Verzeichnisse.

default_filename

[in] Standardname von Datei/Verzeichnis. Falls angegeben, wird dem Öffnungsdialog automatisch der Name hinzugefügt und beim Testen im Array *filenames[]* zurückgegeben.

Rückgabewert

Im Falle einer erfolgreichen Beendigung gibt die Funktion die Anzahl der ausgewählten Dateien zurück, deren Namen in *filenames[]* eingetragen wurden. Wenn ein Nutzer den Dialog schließt, ohne eine Datei auszuwählen, gibt die Funktion 0 zurück. Im Falle einer nicht erfolgreichen Ausführung wird ein Wert kleiner 0 zurückgegeben. Der Fehlercode kann mit GetLastError(). abgerufen werden.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Aus Sicherheitsgründen wird die Arbeit mit Dateien in der Sprache MQL5 streng kontrolliert. Dateien, die bei Dateioperationen mittels der Sprache MQL5 verwendet werden, können nicht außerhalb der Datei-Sandbox (d.h. außerhalb des MQL5\Files-Verzeichnisses) liegen.

Der Name *initial_dir* wird im Verzeichnis des Client-Terminals in MQL5\Files gesucht (oder im Falle von Tests im Verzeichnis test_agent_directory\MQL5\Files). Wenn in der Kombination der Flags FSD_COMMON_FOLDER gesetzt ist, wird die Suche nach dem Anfangsverzeichnis im gemeinsamen Ordner aller Client-Terminals \Terminal\Common\Files. durchgeführt.

Der Parameter *filter* gibt gültige Dateien an und sollte im Format "<Beschreibung-1>|<Erweiterung-1>|<Beschreibung-2>|<Erweiterung-2>..." gesetzt werden, z.B. "Text files (*.txt)|*.txt|All files (*.*)|*.*". Die erste Erweiterung "Textdateien (*.txt)|*.txt" wird als Standarddateityp ausgewählt.

Wenn *filter*=NULL, lautet die Maske der Dateiauswahl im Dialogfenster "All Files (*.*)|*.*|"

Wenn der Parameter *default_filename* gesetzt ist, gibt der Aufruf FileSelectDialog() den Wert 1 zurück, während der Wert von *default_filename* selbst während des nicht-visualisierten Tests in das Array *Dateinamen[]* kopiert wird.

Die Funktion kann nicht in nutzerdefinierten Indikatoren verwendet werden, da durch den Aufruf von FileSelectDialog() der gerade ausgeführte Thread für die gesamte Zeit, während der auf die Antwort des Nutzers gewartet wird, unterbrochen wird. Da alle Indikatoren für jedes Symbol in einem einzigen Thread ausgeführt werden, macht eine solche Unterbrechung den Arbeitsablauf aller Diagramme in allen Zeitrahmen für dieses Symbol unmöglich.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Abrufen der Namen der Textdateien zum Laden aus dem Common-Verzeichnis des Klier
   string filenames[];
   if(FileSelectDialog("Dateien zum Laden auswählen", NULL,
                       "Text files (*.txt)|*.txt|All files (*.*)|*.*",
                       FSD_ALLOW_MULTISELECT|FSD_COMMON_FOLDER, filenames, "data.txt")
     {
      //--- Anzeige des Namens von jeder ausgewählten Datei
      int total=ArraySize(filenames);
      for(int i=0; i<total; i++)
         Print(i, ": ", filenames[i]);
     }
   else
     {
     Print("Keine Dateien ausgewählt");
     }
//---
  }
```

Siehe auch

FileOpen, FileIsExist, FileDelete, FileMove, FolderCreate, FolderDelete, FolderClean, Flags für das Öffnen von Dateien

© 2000-2025, MetaQuotes Ltd.

FileFindFirst

Starten die Sortierung der Dateien und der Unterverzeichnisse im entsprechenden Verzeichnis mit dem angegebenen Filter.

```
long  FileFindFirst(
   const string   file_filter,          // Zeile - Suchefilter
   string&        returned_filename,    // der Name von gefundenen Datei oder Unterver
   int            common_flag=0         // bestimmt den Suchbereich
   );
```

Parameter

file_filter

[in] Suchfilter. Im Filter kann Unterverzeichnis (oder Folge der verschachtelten Unterverzeichnisse) gegenüber dem Verzeichnis angegeben werden \Files,in dem Dateien gesucht werden müssen.

returned_filename

[out] Der zurückgegebene Parameter, wohin im Falle des Erfolges der Name von gefundenen Datei oder Unterverzeichnis unterbracht wird. Only the file name is returned (including the extension), the directories and subdirectories are not included no matter if they are specified or not in the search filter.

common_flag

[in] Flagge, die Dateilage bestimmt. Wenncommon_flag=FILE_COMMON ist, dann befindet sich die Datei im gesamten Ordner aller Client-Terminals \Terminal\Common\Files. Anderenfall befindet sich die Datei im lokalen Ordner.

Rückgabewert

Gibt Handle des Objektes der Suche zurück, das man für die weitere Sortierung der Dateien und der Unterverzeichnisse der FileFindNext() Funktion verwendet werden, oder INVALID_HANDLE wenn es keine Dateien und Unterverzeichnissen gibt, die dem Filter entsprechend (insbesondere, das Verzeichnis ist leer). Nach dem Abschluss der Suche muss man Handle mit Hilfe der FileFindClose(). Funktion schließen.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- display the window of input parameters when launching the script
#property script_show_inputs
//--- filter
input string InpFilter="Dir1\\*";
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string file_name;
   string int_dir="";
   int    i=1,pos=0,last_pos=-1;
//--- search for the last backslash
   while(!IsStopped())
     {
      pos=StringFind(InpFilter,"\\",pos+1);
      if(pos>=0)
         last_pos=pos;
      else
         break;
     }
//--- the filter contains the folder name
   if(last_pos>=0)
      int_dir=StringSubstr(InpFilter,0,last_pos+1);
//--- get the search handle in the root of the local folder
   long search_handle=FileFindFirst(InpFilter,file_name);
//--- check if the FileFindFirst() is executed successfully
   if(search_handle!=INVALID_HANDLE)
     {
      //--- in a cycle, check if the passed strings are the names of files or directo
      do
        {
         ResetLastError();
         //--- if it's a file, the function returns true, and if it's a directory, it
         FileIsExist(int_dir+file_name);
         PrintFormat("%d : %s name = %s",i,GetLastError()==ERR_FILE_IS_DIRECTORY ? "Di
         i++;
        }
      while(FileFindNext(search_handle,file_name));
      //--- close the search handle
      FileFindClose(search_handle);
     }
   else
      Print("Files not found!");
  }
```

Sehen Sie auch

FileFindNext, FileFindClose

© 2000-2025, MetaQuotes Ltd.

FileFindNext

Setzt die Suche fort, die durch die FileFindFirst() Funktion angefangen wurde.

```
bool  FileFindNext(
   long      search_handle,          // die Suche Handle
   string&   returned_filename       // der Name von gefundenen Datei oder Unterverzeic
   );
```

Parameter

search_handle

[in] Die Suche Handle, die durch die FileFindFirst() Funktion erhalten wurde.

returned_filename

[out] Name der nächsten gefundenen Datei oder des Verzeichnisses. Only the file name is returned (including the extension), the directories and subdirectories are not included no matter if they are specified or not in the search filter.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false.

Beispiel:

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Filter
input string InpFilter="*";
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string file_name;
   int    i=1;
//--- erhalten Sie die Suche Handle von Grund auf den lokaler Ordner
   long search_handle=FileFindFirst(InpFilter,file_name);
//--- prüfen Sie, ob die FileFindFirst() Funktion erfolgreich durchgearbeitet hat
   if(search_handle!=INVALID_HANDLE)
     {
      //--- im Zyklus prüfen Sie, ob gesendet Strings Dateinamen oder Verzeichnissen s
      do
        {
         ResetLastError();
         //--- Wenn es die Datei, so wird die Funktion true zurückgeben, und wenn das
         FileIsExist(file_name);
         PrintFormat("%d : %s Name = %s",i,GetLastError()==ERR_FILE_IS_DIRECTORY ? "Ve
         i++;
        }
      while(FileFindNext(search_handle,file_name));
      //--- schließen Sie die Suche Handle
      FileFindClose(search_handle);
     }
   else
      Print("Keine Dateien gefunden!");
  }
```

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

FileFindFirst, FileFindClose

© 2000-2025, MetaQuotes Ltd.

FileFindClose

Schließen Sie die Suche Handle.

```
void  FileFindClose(
   long  search_handle      // die Suche Handle
   );
```

Parameter

search_handle

[in] Die Suche Handle, die durch die FileFindFirst() Funktion erhalten wurde.

Rückgabewert

Keinen Rückgabewert.

Hinweis

Funktion muss aufgerufen werden, um Systemmittel zu befreien.

Beispiel:

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Filter
input string InpFilter="*";
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string file_name;
   int    i=1;
//--- erhalten Sie die Suche Handle von Grund auf den lokaler Ordner
   long search_handle=FileFindFirst(InpFilter,file_name);
//--- prüfen Sie, ob die FileFindFirst()Funktion erfolgreich durchgearbeitet hat
   if(search_handle!=INVALID_HANDLE)
     {
      //--- im Zyklus prüfen Sie, ob gesendet Strings Dateinamen oder Verzeichnissen s
      do
        {
         ResetLastError();
         //--- Wenn es die Datei, so wird die Funktion true zurückgeben, und wenn das
         FileIsExist(file_name);
         PrintFormat("%d : %s Name = %s",i,GetLastError()==5018 ? "Verzeichnis" : "Dat
         i++;
        }
      while(FileFindNext(search_handle,file_name));
      //--- schließen Sie die Suche Handle
      FileFindClose(search_handle);
     }
   else
      Print("Keine Dateien gefunden!");
  }
```

Sehen Sie auch

FileFindFirst, FileFindNext

© 2000-2025, MetaQuotes Ltd.

FileIsExist

Prueft, ob die Datei existiert.

```
bool  FileIsExist(
   const string  file_name,        // der Dateiname
   int           common_flag=0     // der Suchbereich
   );
```

Prameter

file_name

[in] Name der geprüften Datei.

common_flag=0

[in] Flagge, die Dateilage bestimmt. Wenn common_flag=FILE_COMMON ist, befindet sich die Datei im gesamten Ordner aller Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich die Datei im lokalen Ordner.

Rückgabewert

Gibt true zurück, wenn die angegebene Datei existiert.

Hinweis

Die prüfen Datei kann ein Unterverzeichnis werden. in diesem Fall gibt die FileIsExist() Funktion false, und in die _LastError Variable wird Fehler 5018 geschrieben - "Das ist keine Datei, nur Verzeichnis" (sehen Sie das Beispiel für die FileFindFirst Funktion).

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Wenn common_flag=FILE_COMMON ist, sucht die Funktion die angegebene Datei im Gesamtordner aller Client-Terminals \Terminal\Common\Files, anderenfalls sucht die Funktion die Datei im lokalen Ordner (MQL5\Files oder MQL5\Tester\Files beim Testen).

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//---Datum für alte Dateien
input datetime InpFilesDate=D'2013.01.01 00:00';
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string   file_name;      // Variable für Speichern die Dateinamen
   string   filter="*.txt"; // Der Filter für die Suche nach Dateien
   datetime create_date;    // das Dateierstellungsdatum
   string   files[];        // Die Liste von Dateinamen
   int      def_size=25;    // die standardmässige Größe des Arrays
   int      size=0;         // die Anzahl der Dateien
//--- trennen Sie den Speicher für den Array ab
   ArrayResize(files,def_size);
//--- erhalten Sie die Suche Handle von Grund auf den lokaler Ordner
   long search_handle=FileFindFirst(filter,file_name);
//--- prüfen Sie, ob die FileFindFirst()Funktion erfolgreich durchgearbeitet hat
   if(search_handle!=INVALID_HANDLE)
     {
      //--- durchlaufen Sie im Zyklus Dateien
      do
        {
         files[size]=file_name;
         //--- erhöhen Sie die Größe des Arrays
         size++;
         if(size==def_size)
           {
            def_size+=25;
            ArrayResize(files,def_size);
           }
         //--- stürzen Sie den Fehlerwert
         ResetLastError();
         //--- haben Sie das Dateierstellungsdatum
         create_date=(datetime)FileGetInteger(file_name,FILE_CREATE_DATE,false);
         //--- prüfen Sie, ob die Datei alt ist
         if(create_date<InpFilesDate)
           {
            PrintFormat("Die %s Datei gelöscht!",file_name);
            //---löschen Sie die alte Datei
            FileDelete(file_name);
           }
        }
      while(FileFindNext(search_handle,file_name));
      //--- schließen Sie die Suche Handle
      FileFindClose(search_handle);
     }
   else
     {
      Print("Keine Dateien gefunden!");
      return;
     }
//--- prüfen Sie, ob welche Dateien blieben
   PrintFormat("Resultate:");
   for(int i=0;i<size;i++)
     {
      if(FileIsExist(files[i]))
         PrintFormat("Die %s Datei existiert!",files[i]);
      else
```

© 2000-2025, MetaQuotes Ltd.

```
            PrintFormat("Die %s Datei gelöscht!",files[i]);
        }
    }
```

Sehen Sie auch

FileFindFirst

© 2000-2025, MetaQuotes Ltd.

FileOpen

Funktion öffnet die Datei mit dem angegebenen Namen und angegebenen Flaggen.

```
int  FileOpen(
   string  file_name,           // der Dateiname
   int     open_flags,          // die Kombination der Flaggen
   short   delimiter='\t',      // der Begrenzer
   uint    codepage=CP_ACP      // die Kodeseite
   );
```

Parameter

file_name

[in] Name der geöffneten Datei, kann Unterordner haben. Wenn die Datei für Schreiben geöffnet wird, werden die angegebenen Unterordner erzeugt werden, wenn sie es nicht gibt.

open_flags

[in] Kombination der Flaggen, die Arbeitsmode mit Datei bestimmt. Flaggen werden folgenderweise bestimmt:
FILE_READ Datei wird für Lesen geöffnet
FILE_WRITE Datei wird für Schreiben geöffnet
FILE_BIN Binärmode Lesen-Schreiben (ohne Umwandlung der Zeile in die Zeile)
FILE_CSV Date des Typs csv (alle geschriebenen Elemente werden in die Zeilen des entsprechenden Typs umgewandelt, unicode oder ansi, und durch Begrenzer getrennt)
FILE_TXT einfache Textdatei (dieselbe csv, aber Begrenzer wird nicht berücksichtigt)
FILE_ANSI Zeilen des Typs ANSI (Einbyte-Symbole)
FILE_UNICODE Zeilen des Typs UNICODE (Zweibyte-Symbole)
FILE_SHARE_READ Gesamter Lesezugang von mehreren Programmen
FILE_SHARE_WRITE Gesamter Schreibezugang von mehreren Programmen
FILE_COMMON Lage der Datei im Gesamtordner aller Client-Terminals \Terminal\Common\Files

delimiter='\t'

[in] Wert, der als Begrenzer in txt oder csv-Datei verwendet wird.Wenn der Begrenzer für csv-Datei nicht angegeben wird, wird als Default-Wert Tab-Symbol verwendet. Wenn der Begrenzer für txt-Datei nicht angegeben wird, wird keiner Begrenzer verwendet. Wenn Begrenzer auf Null gestellt wird, wird keiner Begrenzer verwendet.

codepage=CP_ACP

[in] Wert der Kodeseite. Für die am meisten verwendeten Kodeseiten werden entsprechende Konstanten vorausgesehen.

Rückgabewert

Beim erfolgreichen Schließen gibt die Funktion der Datei Handle zurück, das danach für den Zugang zu Dateidaten verwendet wird). Beim Misserfolg gibt die Funktion INVALID_HANDLE zurück.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

© 2000-2025, MetaQuotes Ltd.

Wenn eine Datei in einer bestimmten Codierung gelesen werden muss (es wurde der Parameter codepage mit einem Wert der Codepage angegeben), muss das Flag FILE_ANSI gesetzt werden. Wenn das Flag FILE_ANSI nicht angegeben wurde, wird die Textdatei in Unicode ohne Konvertierung gelesen.

Datei wird im Ordner des Client-Terminals im Subordner MQL5\files geöffnet (oder Ordner_des_Testagents\MQL5\files beim Testen). Wenn unter Flaggen FILE_COMMON angegeben wird, wird die Datei im Gesamtordner aller Client-Terminals MetaTrader5 geöffnet.

Man kann "die genannten Kanäle" nach den einhaltenden Regeln öffnen:

- Der Name des Kanals ist die Zeile, die aussehen soll: "\\servername\pipe\pipename", wo Servername ist der Name eines Servers im Netzwerk und Pipename ist der Name des Kanals. Wenn die Kanäle auf einem und derselbe Computer verwendet werden, kann der Name des Servers gesenkt sein, aber muss man anstelle seiner den Punkt machen: "\\.\pipe\pipename". Der Kunde, der versucht mit dem Kanal zu verbinden, soll seinen Namen wissen.
- Man muss FileFlush() und FileSeek() auf den Anfang der Datei zwischen aufeinander folgenden Operationen von lesen und schreiben aufgerufen werden.

In den Zeilen verwenden Sie einen umgekehrten Schrägstrich '\', also wenn Sie das Programm in MQL5 schreiben, Sie Backslash müssen '\' deshalb das obige Beispiel Ihren Code als "\\\\servername\\pipe\\pipename" schreiben.

Details zum Arbeiten mit die genannten Kanäle lesen Sie im Artikel "Communicating With MetaTrader 5 Using Named Pipes Without Using DLLs".

Beispiel:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- unkorrektes Verfahren der Dateiöffnung
   string terminal_data_path=TerminalInfoString(TERMINAL_DATA_PATH);
   string filename=terminal_data_path+"\\MQL5\\Files\\"+"fractals.csv";
   int filehandle=FileOpen(filename,FILE_WRITE|FILE_CSV);
   if(filehandle<0)
     {
      Print("erfolgloser Versuch, die Datei durch absoluten Pfad zu öffnen");
      Print("Fehlercode ",GetLastError());
     }
//--- korrektes Verahren der Arbeit in "Sandkasten"
   ResetLastError();
   filehandle=FileOpen("fractals.csv",FILE_WRITE|FILE_CSV);
   if(filehandle!=INVALID_HANDLE)
     {
      FileWrite(filehandle,TimeCurrent(),Symbol(), EnumToString(_Period));
      FileClose(filehandle);
      Print("FileOpen OK");
    }
   else Print("Operation FileOpen erfolglos, Fehler ",GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
//--- ein anderes Beispiel der Erzeugung des verschachtelten Verzeichnisses in MQL5\Fi
   string subfolder="Research";
   filehandle=FileOpen(subfolder+"\\fractals.txt",FILE_WRITE|FILE_CSV);
      if(filehandle!=INVALID_HANDLE)
     {
      FileWrite(filehandle,TimeCurrent(),Symbol(), EnumToString(_Period));
      FileClose(filehandle);
      Print("Datei muss im Ordner "+terminal_data_path+"\\"+subfolder+" erzeugt werden
    }
   else Print("Fehler beim Öffnen der Datei, Fehler ",GetLastError());
  }
```

Sehen Sie auch

Kodeseite Verwenden, FileFindFirst, FolderCreate, Flaggen der Dateioeffnung

© 2000-2025, MetaQuotes Ltd.

FileClose

Schließen Sie die Datei, die früher durch die FileOpen() Funktion geöffnet wurde.

```
void  FileClose(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die FileOpen() Funktion zurückgegeben wird.

Rückgabewert

Keinen Rückgabewert.

Beispiel:

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpFileName="file.txt";     // der DateinName
input string InpDirectoryName="Data";    // der Verzeichnisname
input int    InpEncodingType=FILE_ANSI; // ANSI=32 oder UNICODE=64
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- drucken Sie den Weg zum Folder in dem Sie arbeiten werden
   PrintFormat("Arbeiten in dem %s\\Files\\ Folder ",TerminalInfoString(TERMINAL_DATA_
//--- der Fehlerwert zu stürzen
   ResetLastError();
//--- öffnen Sie die Datei für Lesen (wenn die Datei nicht existiert, tritt ein Fehler
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_TXT|InpEr
   if(file_handle!=INVALID_HANDLE)
     {
      //--- drucken Sie den Dateiinhalt
      while(!FileIsEnding(file_handle))
         Print(FileReadString(file_handle));
      //--- schließen Sie die Datei
      FileClose(file_handle);
     }
   else
      PrintFormat("Fehler, Kode = %d",GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

FileCopy

Kopiert Ausgangsdatei aus einem Lokale Ordner oder Gesamtordner in eine andere Datei.

```
bool  FileCopy(
   const string  src_file_name,      // der Name der Quelle-Datei
   int           common_flag,        // die Lage
   const string  dst_file_name,      // der Name der Zieldatei
   int           mode_flags          // der Zugangsmode
   );
```

Parameter

src_file_name

[in] Datename für Kopieren.

common_flag

[in] Flagge, die Dateilage bestimmt. Wenn common_flag=FILE_COMMON, befindet sich Datei im Gesamtordner aller Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich die Datei im Lokalen Ordner (zum Beispiel, *common_flag=0*).

dst_file_name

[in] Name der ergebenden Datei.

mode_flags

[in] Zugangsflaggen. Parameter kann erst 2 Flaggen enthalten: FILE_REWRITE und/oder FILE_COMMON - andere Flaggen werden ignoriert. Wenn Date existiert schon und dabei die Flagge FILE_REWRITE nicht angegeben wurde, wird die Datei nicht neugeschrieben und die Funktion gibt false zurück.

Rückgabewert

Beim Misserfolg gibt die Funktion false zurück.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Wenn die neue Datei existiert, wird Kopieren abhängig davon durchgeführt, ob es die Flagge FILE_REWRITE im Wert des parameters gibt mode_flags oder nicht.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpSrc="source.txt";        // die Quelle
input string InpDst="destination.txt";   // Kopie
input int    InpEncodingType=FILE_ANSI; // ANSI=32 oder UNICODE=64
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- zeigen den Inhalt der Quelle an (er muss existieren sein)
   if(!FileDisplay(InpSrc))
      return;
//--- prüfen Sie, ob schon die Datei der Kopie existiert (es ist nicht verpflichtet, e
   if(!FileDisplay(InpDst))
     {
      //--- die Datei der Kopie existiert nicht, das Kopieren ohne Fahne FILE_REWRITE
      if(FileCopy(InpSrc,0,InpDst,0))
         Print("Die Datei ist abgeschrieben!");
      else
         Print("Die Datei ist nicht abgeschrieben!");
     }
   else
     {
      //--- die Datei der Kopie existiert schon, wir werden versuchen, ohne FILE_REWRI
      if(FileCopy(InpSrc,0,InpDst,0))
         Print("Die Datei ist abgeschrieben!");
      else
         Print("Die Datei ist nicht abgeschrieben!");
      //--- der InpDst Dateiinhalt bleibt vorig
      FileDisplay(InpDst);
      //--- Wir werden noch einmal mit der FILE_REWRITE Fahne abschreiben (das richtig
      if(FileCopy(InpSrc,0,InpDst,FILE_REWRITE))
         Print("Die Datei ist abgeschriebened!");
      else
         Print("Die Datei ist nicht abgeschrieben!");
     }
//--- haben die Kopie der Datei bekommen InpSrc
   FileDisplay(InpDst);
  }
//+------------------------------------------------------------------+
//| Lesen des Inhalts einer Datei                                    |
//+------------------------------------------------------------------+
bool FileDisplay(const string file_name)
  {
//--- der Fehlerwert zu stürzen
   ResetLastError();
//--- öffnen Sie die Datei
   int file_handle=FileOpen(file_name,FILE_READ|FILE_TXT|InpEncodingType);
   if(file_handle!=INVALID_HANDLE)
     {
      //--- darstellen Sie den Dateiinhalt im Zyklus
      Print("+----------------------+");
      PrintFormat("der Dateiname = %s",file_name);
      while(!FileIsEnding(file_handle))
         Print(FileReadString(file_handle));
      Print("+----------------------+");
      //--- schließen Sie die Datei
      FileClose(file_handle);
      return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- es ist damit gescheitert, die Datei zu öffnen
   PrintFormat("%s ist nicht geöffnet, Fehler = %d",file_name,GetLastError());
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

FileDelete

Entfernt die angegebene Datei im Lokalen Ordner des Client-Terminals.

```
bool  FileDelete(
   const string  file_name,     // der Name der entfernten Datei
   int           common_flag=0  // die Lage der entfernten Datei
   );
```

Parameter

file_name

[in] Dateiname.

common_flag=0

[in] Flagge, die Dateilage bestimmt. Wenn common_flag=FILE_COMMON, befindet sich die Datei im Gesamtordner aller Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich die Datei im lokalen Ordner.

Rückgabewert

Beim Misserfolg gibt die Funktion false zurück.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Entfernt die angegebene Datei im Lokalen Ordner des Client-Terminals (MQL5\Files oder MQL5\Tester\Files beim Testen). Wenn aber common_flag=FILE_COMMON angegeben wird, entfernt die Funktion die Datei aus dem Gesamtordner aller Client-Terminals.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//---Datum für alte Dateien
input datetime InpFilesDate=D'2013.01.01 00:00';
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string   file_name;      // Variable für Speichern die Dateinamen
   string   filter="*.txt"; // Der Filter für die Suche nach Dateien
   datetime create_date;    // das Dateierstellungsdatum
   string   files[];        // Die Liste von Dateinamen
   int      def_size=25;    // die standardmässige Größe des Arrays
   int      size=0;         // die Anzahl der Dateien
//--- trennen Sie den Speicher für den Array ab
   ArrayResize(files,def_size);
//--- erhalten Sie die Suche Handle von Grund auf den lokaler Ordner
   long search_handle=FileFindFirst(filter,file_name);
//--- prüfen Sie, ob die FileFindFirst() Funktion erfolgreich durchgearbeitet hat
   if(search_handle!=INVALID_HANDLE)
     {
      //--- durchlaufen Sie im Zyklus Dateien
      do
        {
         files[size]=file_name;
         //--- erhöhen Sie die Größe des Arrays
         size++;
         if(size==def_size)
           {
            def_size+=25;
            ArrayResize(files,def_size);
           }
         //--- der Fehlerwert zu stürzen
         ResetLastError();
         //--- haben Sie das Dateierstellungsdatum
         create_date=(datetime)FileGetInteger(file_name,FILE_CREATE_DATE,false);
         //--- prüfen Sie, ob die Datei alt ist
         if(create_date<InpFilesDate)
           {
            PrintFormat("Die %s Datei gelöscht!",file_name);
            //---löschen Sie die alte Datei
            FileDelete(file_name);
           }
        }
      while(FileFindNext(search_handle,file_name));
      //--- schließen Sie die Suche Handle
      FileFindClose(search_handle);
     }
   else
     {
      Print("Files not found!");
      return;
     }
//--- prüfen Sie, ob welche Dateien blieben
   PrintFormat("Resultate:");
   for(int i=0;i<size;i++)
     {
      if(FileIsExist(files[i]))
         PrintFormat("Die %s Datei existiert!",files[i]);
      else
```

© 2000-2025, MetaQuotes Ltd.

```
         PrintFormat("Die %s Datei gelöscht!",files[i]);
      }
   }
```

© 2000-2025, MetaQuotes Ltd.

FileMove

überträgt die Datei aus einem lokalen oder Gesamtordner in einen anderen Ordner.

```
bool  FileMove(
   const string  src_file_name,     // der Dateiname für übertragung
   int           common_flag,       // die Lage
   const string  dst_file_name,     // der Name der Zieldatei
   int           mode_flags         // Zugangsverfahren
   );
```

Parameter

src_file_name

[in] Dateiname für übertragung/Umbanennung.

common_flag

[in] Flagge, die die Lage der Datei bestimmt. Wenn common_flag=FILE_COMMON ist, befindet sich die Datei im Gesamtordner aller Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich die Datei im lokalen Ordner (*common_flag=0*).

dst_file_name

[in] Name der ergebenden Datei.

mode_flags

[in] Zugangsflaggen. Parameter kann nur zwei Flaggen enthalten: FILE_REWRITE und/oder FILE_COMMON - andere Flaggen werden ignoriert. Wenn die Datei existiert schon und dabei die Flagge FILE_REWRITE nicht angegeben wurde, wird die Datei nicht neugeschrieben, und die Funktion gibt false zurück.

Rückgabewert

Beim Misserfolg gibt die Funktion false zurück.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Wenn die neue Datei existiert schon, wird das Kopieren abhängig davon durchgeführt werden, ob die Flagge FILE_REWRITE im Parameterwert mode_flags gibt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpSrcName="data.txt";
input string InpDstName="newdata.txt";
input string InpSrcDirectory="SomeFolder";
input string InpDstDirectory="OtherFolder";
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string local=TerminalInfoString(TERMINAL_DATA_PATH);
   string common=TerminalInfoString(TERMINAL_COMMONDATA_PATH);
//--- erhalten Sie die Dateipfade
   string src_path;
   string dst_path;
   StringConcatenate(src_path,InpSrcDirectory,"//",InpSrcName);
   StringConcatenate(dst_path,InpDstDirectory,"//",InpDstName);
//--- prüfen Sie, ob die Quelle Datei existiert (wenn es gibt keine - Ausgabe)
   if(FileIsExist(src_path))
      PrintFormat("%s Datei existiert in der %s\\Files\\%s Ordner nicht",InpSrcName,lo
   else
     {
      PrintFormat("Fehler, %s Hauptdatei nicht gefunden",InpSrcName);
      return;
     }
//--- prüfen Sie, ob die Ergebnisdatei existiert
   if(FileIsExist(dst_path,FILE_COMMON))
     {
      PrintFormat("%s Datei existiert in der %s\\Files\\%s Ordner nicht",InpDstName,co
      //--- die Datei existiert, die Umstellung muss man mit der Fahne FILE_REWRITE du
      ResetLastError();
      if(FileMove(src_path,0,dst_path,FILE_COMMON|FILE_REWRITE))
         PrintFormat("%s Datei  ist versetzt",InpSrcName);
      else
         PrintFormat("Fehler! Kode = %d",GetLastError());
     }
   else
     {
      PrintFormat("%s Datei existiert in der %s\\Files\\%s Ordner nicht",InpDstName,co
      //--- die Datei existiert nicht, die Umstellung muss man ohne der Fahne FILE_REW
      ResetLastError();
      if(FileMove(src_path,0,dst_path,FILE_COMMON))
         PrintFormat("%s Datei  ist versetzt",InpSrcName);
      else
         PrintFormat("Fehler! Kode = %d",GetLastError());
     }
//--- jetzt ist die Datei versetzt, überprüfen Sie es
   if(FileIsExist(dst_path,FILE_COMMON) && !FileIsExist(src_path,0))
      Print("Erfolgreich!");
   else
      Print("Fehler!");
  }
```

Sehen Sie auch

FileIsExist

© 2000-2025, MetaQuotes Ltd.

FileFlush

Speichert in der Platte alle Daten, die im Dateipuffer Eingabe-Ausgabe noch bleiben.

```
void  FileFlush(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die FileOpen() Funktion zurückgegeben wird.

Rückgabewert

Keinen Rückgabewert.

Hinweis

Wenn Sie in eine Datei schreiben, können die Daten physisch nur nach einiger Zeit sein. Damit die Daten sofort in die Datei erhalten, müssen Sie die FileFlush() Funktion verwenden. Wenn Sie nicht der Daten-Funktion verwenden, die auf den Datenträger nicht noch aufgeholt hat gibt es nur, wenn Sie die FileClose() Funktion schließen.

Die Funktion sollte verwendet werden, wenn die Aufnahmedaten einige Wert darstellt. Es sollte berücksichtigt werden, dass ein häufige Funktionsaufruf die Geschwindigkeit des Programms beeinflussen kann.

Die FileFlush() Funktion muss zwischen der Operationen des Lesens aus der Datei und Schreibens in die Datei aufgerufen werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- der Name der Datei zum Schreiben
input string InpFileName="example.csv"; // der Dateiname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Stürzen Sie den Fehlerwert
   ResetLastError();
//--- öffnen Sie die Datei
   int file_handle=FileOpen(InpFileName,FILE_READ|FILE_WRITE|FILE_CSV);
   if(file_handle!=INVALID_HANDLE)
     {
      //--- Daten in eine Datei schreiben
      for(int i=0;i<1000;i++)
        {
         //--- Aufnahmefunktion aufrufen
         FileWrite(file_handle,TimeCurrent(),SymbolInfoDouble(Symbol(),SYMBOL_BID),Sy
         //--- Zurücksetzen der Daten auf der Festplatte bei jeder 128 Iteration
         if((i & 127)==127)
           {
            //---Nun wird die Daten in der Datei, und wenn das Terminal wir schwerwieg
            FileFlush(file_handle);
           }
         //---die Verzögerung von 0,01 Sekunden
         Sleep(10);
        }
      //--- Schließen Sie die Datei
      FileClose(file_handle);
     }
   else
      PrintFormat("Fehler, Kode = %d",GetLastError());
  }
```

Sehen Sie auch

FileClose

© 2000-2025, MetaQuotes Ltd.

FileGetInteger

Ruft eine ganzzahlige Eigenschaft der Datei. Es gibt 2 Varianten der Funktion.

1. Eigenschaften beim Handle bekommen.

```
long  FileGetInteger(
   int                          file_handle,    // Datei-Handle
   ENUM_FILE_PROPERTY_INTEGER   property_id     // ID der Eigenschaft
   );
```

2. Eigenschaften beim Dateinamen bekommen.

```
long  FileGetInteger(
   const string                 file_name,             // der Dateiname
   ENUM_FILE_PROPERTY_INTEGER   property_id,           // ID der Eigenschaft
   bool                         common_folder=false    // Die Datei ist in einem lokaler
   );                                                  // oder in einem freigegebenen Or
```

Parameter

file_handle

[in] Dateideskriptor der durch die FileOpen() Funktion zurückgegeben wird.

file_name

[in] Der Dateiname.

property_id

[in] ID der Datei-Eigenschaft. Der Wert kann der ENUM_FILE_PROPERTY_INTEGER Enumerationswerte sein. Wenn die zweite Version der Funktion benutzt wird, können Sie die Werte nur für die folgenden Eigenschaften erhalten: FILE_EXISTS, FILE_CREATE_DATE, FILE_MODIFY_DATE, FILE_ACCESS_DATE und FILE_SIZE.

common_folder=false

[in] Verweist auf den Speicherort der Datei. Wenn der Parameter falsch ist, dann das Data-Ordner des Terminals durchgeschaut wird, ansonsten wird es davon ausgegangen, dass die Datei in einem freigegebenen Ordner von Kunden-Terminals liegt \Terminal\Common\Files (FILE_COMMON).

Rückgabewert

Wert der Eigenschaft. Im Falle eines Fehlers, gibt die Funktion -1 zurück. Um die Information über Fehler zu erhalten, muss man die Funktion GetLastError() aufrufen.

Wenn ein Ordner angegeben wird, wenn Sie die Eigenschaften durch den Namen erhalten, hat die Funktion Fehler 5018 (ERR_MQL_FILE_IS_DIRECTORY) in jedem Fall, während der Rückgabewert korrekt ist.

Hinweis

Die Funktion verändert immer den Fehlercode. Wenn es gelingt, wird der Fehlercode auf Null zurückgesetzt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpFileName="data.csv";
input string InpDirectoryName="SomeFolder";
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string path=InpDirectoryName+"//"+InpFileName;
   long   l=0;
//--- öffnen Sie die Datei
   ResetLastError();
   int handle=FileOpen(path,FILE_READ|FILE_CSV);
   if(handle!=INVALID_HANDLE)
     {
      //--- drucken Sie alle Informationen über die Datei
      Print(InpFileName," file info:");
      FileInfo(handle,FILE_EXISTS,l,"bool");
      FileInfo(handle,FILE_CREATE_DATE,l,"date");
      FileInfo(handle,FILE_MODIFY_DATE,l,"date");
      FileInfo(handle,FILE_ACCESS_DATE,l,"date");
      FileInfo(handle,FILE_SIZE,l,"other");
      FileInfo(handle,FILE_POSITION,l,"other");
      FileInfo(handle,FILE_END,l,"bool");
      FileInfo(handle,FILE_IS_COMMON,l,"bool");
      FileInfo(handle,FILE_IS_TEXT,l,"bool");
      FileInfo(handle,FILE_IS_BINARY,l,"bool");
      FileInfo(handle,FILE_IS_CSV,l,"bool");
      FileInfo(handle,FILE_IS_ANSI,l,"bool");
      FileInfo(handle,FILE_IS_READABLE,l,"bool");
      FileInfo(handle,FILE_IS_WRITABLE,l,"bool");
      //--- schließen Sie die Datei
      FileClose(handle);
     }
   else
      PrintFormat("%s Datei ist nicht geöffnet, FehlerCode = %d",InpFileName,GetLastEr
  }
//+------------------------------------------------------------------+
//| Anzeige den Wert der Dateieigenschaft                            |
//+------------------------------------------------------------------+
void FileInfo(const int handle,const ENUM_FILE_PROPERTY_INTEGER id,
              long l,const string type)
  {
//--- erhalten Sie den Wert der Eigenschaft
   ResetLastError();
   if((l=FileGetInteger(handle,id))!=-1)
     {
      //--- der Wert erhaltet wurde, zeigen Sie es im richtigen Format
      if(!StringCompare(type,"bool"))
         Print(EnumToString(id)," = ",l ? "true" : "false");
      if(!StringCompare(type,"date"))
         Print(EnumToString(id)," = ",(datetime)l);
      if(!StringCompare(type,"other"))
         Print(EnumToString(id)," = ",l);
     }
   else
      Print("Fehler, Kode = ",GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

Dateioperationen, Dateieigenschaften

© 2000-2025, MetaQuotes Ltd.

FileIsEnding

Bestimmt Dateiende während des Lesens.

```
bool  FileIsEnding(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die FileOpen() Funktion zurückgegeben wird.

Rückgabewert

Funktion gibt true zurück, wenn während des Lesens oder übertragung des Dateianzeigers das Dateiende erreicht wurde.

Hinweis

Um das Dateiende zu ermitteln, versucht die Funktion lesen auf die nächste Zeile aus der Datei lesen. Wenn es nicht vorhanden ist, gibt die Funktion true, andernfalls false.

Beispiel:

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpFileName="file.txt";     // der Dateiname
input string InpDirectoryName="Data";    // der Verzeichnisname
input int    InpEncodingType=FILE_ANSI; // ANSI=32 oder UNICODE=64
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- drucken Sie den Weg zum Folder in dem Sie arbeiten werden
   PrintFormat("Arbeiten in dem %s\\Files\\ Ordner ",TerminalInfoString(TERMINAL_DATA_
//--- der Fehlerwert zu stürzen
   ResetLastError();
//--- öffnen Sie die Datei für Lesen (wenn die Datei nicht existiert, tritt ein Fehler
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_TXT|InpEr
   if(file_handle!=INVALID_HANDLE)
     {
      //--- drucken Sie den Dateiinhalt
      while(!FileIsEnding(file_handle))
         Print(FileReadString(file_handle));
      //--- schließen Sie die Datei
      FileClose(file_handle);
     }
   else
      PrintFormat("Fehler, Kode = %d",GetLastError());
  }
```

© 2000-2025, MetaQuotes Ltd.

FileIsLineEnding

Bestimmt Zeilenende während des Lesens.

```
bool  FileIsLineEnding(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Gibt true zurück wenn beim Lesen der txt oder csv-Datei das Zeilenende erreicht wird (Symbole CR-LF).

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteString bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                        Demo_FileIsLineEnding.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 5
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Overbought & Oversold"
#property indicator_type1   DRAW_COLOR_BARS
#property indicator_color1  clrRed, clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  2
//--- Parameter zum Lesen von Daten
input string InpFileName="RSI.csv";   // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- die Indikator-Buffers
double   open_buff[];
double   high_buff[];
double   low_buff[];
double   close_buff[];
double   color_buff[];
//--- überkauft Variablen
int      ovb_ind=0;
int      ovb_size=0;
datetime ovb_time[];
//--- überverkauft Variablen
int      ovs_ind=0;
int      ovs_size=0;
datetime ovs_time[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- die standartmäßige Variable Größe von Arrays
   int ovb_def_size=100;
   int ovs_def_size=100;
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(ovb_time,ovb_def_size);
   ArrayResize(ovs_time,ovs_def_size);
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_CSV|FILE_
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat(" %s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      double value;
      //--- lesen Sie die Dateidaten
      while(!FileIsEnding(file_handle))
        {
         //--- lesen Sie den ersten Wert in eine Zeichenfolge
         value=FileReadNumber(file_handle);
         //--- lesen Sie die verschiedenen Arrays, abhängig vom Ergebnis der Funktion
         if(value>=70)
            ReadData(file_handle,ovb_time,ovb_size,ovb_def_size);
```

© 2000-2025, MetaQuotes Ltd.

```
         else
            ReadData(file_handle,ovs_time,ovs_size,ovs_def_size);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten werden gelesen, die Datei %s geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Arrays-Bindung
   SetIndexBuffer(0,open_buff,INDICATOR_DATA);
   SetIndexBuffer(1,high_buff,INDICATOR_DATA);
   SetIndexBuffer(2,low_buff,INDICATOR_DATA);
   SetIndexBuffer(3,close_buff,INDICATOR_DATA);
   SetIndexBuffer(4,color_buff,INDICATOR_COLOR_INDEX);
//---- erstellen Sie die Anzeige-Werte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Lesen Sie die Zeilen einer Datei                                 |
//+------------------------------------------------------------------+
void ReadData(const int file_handle,datetime &arr[],int &size,int &def_size)
  {
   bool flag=false;
//--- Lesen Sie, bis das Ende der Zeichenfolge oder Datei erreichen
   while(!FileIsLineEnding(file_handle) && !FileIsEnding(file_handle))
     {
      //--- bewegen Sie den Wagen, nach dem Lesen der Anzahl
      if(flag)
         FileReadNumber(file_handle);
      //--- das aktuelle Datum speichern
      arr[size]=FileReadDatetime(file_handle);
      size++;
      //--- Falls erforderlich, vergrößern Sie die Größe des Arrays
      if(size==def_size)
        {
         def_size+=100;
         ArrayResize(arr,def_size);
        }
      //--- übergeben die erste iteration
      flag=true;
     }
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
{
```

© 2000-2025, MetaQuotes Ltd.

```
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(open,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
   ArraySetAsSeries(close,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      open_buff[i]=0;
      high_buff[i]=0;
      low_buff[i]=0;
      close_buff[i]=0;
      color_buff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ovb_ind<ovb_size)
         for(int j=ovb_ind;j<ovb_size;j++)
           {
            //--- wenn die Daten identisch sind, ist die Bar in überkauft Zone
            if(time[i]==ovb_time[j])
              {
               open_buff[i]=open[i];
               high_buff[i]=high[i];
               low_buff[i]=low[i];
               close_buff[i]=close[i];
               //--- 0 - die rote Farbe
               color_buff[i]=0;
               //--- vergrössern Sie den Zähler
               ovb_ind=j+1;
               break;
              }
           }
      //--- prüfen Sie, ob es noch Daten gibt
      if(ovs_ind<ovs_size)
         for(int j=ovs_ind;j<ovs_size;j++)
           {
            //--- wenn die Daten identisch sind, ist die Bar in überverkauft Zone
            if(time[i]==ovs_time[j])
              {
               open_buff[i]=open[i];
               high_buff[i]=high[i];
               low_buff[i]=low[i];
               close_buff[i]=close[i];
               //--- 1 - die blau Farbe
               color_buff[i]=1;
               //--- vergrössern Sie den Zähler
               ovs_ind=j+1;
               break;
              }
           }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Der Bearbeiter des Ereignisses ChartEvent                        |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam
```

© 2000-2025, MetaQuotes Ltd.

```
                  )
  {
//--- ändern Sie die Breite des Indikators je nach dem Ausmaß
   if(ChartGetInteger(0,CHART_SCALE)>3)
      PlotIndexSetInteger(0,PLOT_LINE_WIDTH,2);
   else
      PlotIndexSetInteger(0,PLOT_LINE_WIDTH,1);
  }
```

Sehen Sie auch

FileWriteString

© 2000-2025, MetaQuotes Ltd.

FileReadArray

Liest Felder aller Typen ausser Zeilentypen (kann Feld der Strukturen ohne Zeilen und dynamische Felder), aus Binärdatei von der laufenden Position des Dateianzeigers.

```
uint  FileReadArray(
   int     file_handle,               // Datei-Handle
   void&   array[],                   // Feld für Aufzeichnung
   int     start=0,                   // Anfangsposition für Schreiben
   int     count=WHOLE_ARRAY          // Anzahl für Lesen
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

array[]

[out] Feld, wohin Daten geladen werden.

start=0

[in] Anfangsposition für Schreiben in das Feld.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für Lesen.

Rückgabewert

Anzahl der gelesenen Elemente. Als Default liest das ganze Feld (count=WHOLE_ARRAY).

Hinweis

Zeilenfeld kann nur aus Datei des Typs TXT gelesen werden. Wenn notwendig versucht die Funktion die Größe des Feldes zu vergrössern.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteArray bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpFileName="data.bin";
input string InpDirectoryName="SomeFolder";
//+------------------------------------------------------------------+
//| Struktur zum Speichern von Daten auf die Preise                  |
//+------------------------------------------------------------------+
struct prices
  {
   datetime          date; // Datum
   double            bid;  // der Bid Preis
   double            ask;  // der Ask Preis
  };
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- ein Array von Struktur
   prices arr[];
//--- Pfad zur Datei
   string path=InpDirectoryName+"//"+InpFileName;
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(path,FILE_READ|FILE_BIN);
   if(file_handle!=INVALID_HANDLE)
     {
      //--- lesen Sie alle Dateteidaten in ein Array
      FileReadArray(file_handle,arr);
      //--- erhalten Sie die Größe des Arrays
      int size=ArraySize(arr);
      //--- drucken Sie die Daten aus dem Array
      for(int i=0;i<size;i++)
         Print("Daten = ",arr[i].date," Bid = ",arr[i].bid," Ask = ",arr[i].ask);
      Print("Gesamtdaten = ",size);
      //--- schließen Sie die Datei
      FileClose(file_handle);
     }
   else
      Print("Fehler beim Öffnen der Datei, Fehler ",GetLastError());
  }
```

Sehen Sie auch

Variablen, FileWriteArray

© 2000-2025, MetaQuotes Ltd.

FileReadBool

Liest aus der Datei des Typs CSV die Zeile von der laufenden Position bis zum Begrenzer (oder bis zum Ende der Textzeile) und wandelt die gelesene Zeile in den Wert des Typs bool um.

```
bool  FileReadBool(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen().

Rückgabewert

Die gelesene Zeile kann die Werte "true", "false" oder Symboldarstellung der Ganzzahlen "0" oder "1" haben. Der Nichtnullwert wird in den logischen Wert true umgewandelt. Die Funktion gibt den erhaltenen umgewandelten Wert zurück.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWrite bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                              Demo_FileReadBool.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   2
//---- plot Label1
#property indicator_label1  "UpSignal"
#property indicator_type1   DRAW_ARROW
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  4
//---- plot Label2
#property indicator_label2  "DownSignal"
#property indicator_type2   DRAW_ARROW
#property indicator_color2  clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  4
//--- Parameter zum Lesen von Daten
input string InpFileName="MACD.csv";  // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- globale Variablen
int      ind=0;        // Index
double   upbuff[];     // der Indikator-Puffer der Zeiger nach oben
double   downbuff[];   // der Indikator-Puffer der Zeiger nach unten
bool     sign_buff[]; // ein Array von Signalen (true - kaufen, false - verkaufen)
datetime time_buff[]; // ein Array der Zeit des Eintritts der Signale
int      size=0;       // die Größe der Array von Signalen
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_CSV);
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      //--- Lesen Sie zuerst die Anzahl der Signale
      size=(int)FileReadNumber(file_handle);
      //--- trennen Sie den Speichern für den Arrays
      ArrayResize(sign_buff,size);
      ArrayResize(time_buff,size);
      //--- lesen Sie die Dateidaten
      for(int i=0;i<size;i++)
        {
         //--- Die Zeit des Signals
         time_buff[i]=FileReadDatetime(file_handle);
         //--- Der Signalwert
         sign_buff[i]=FileReadBool(file_handle);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
     }
   else
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
       return(INIT_FAILED);
      }
//--- Arrays-Bindung
   SetIndexBuffer(0,upbuff,INDICATOR_DATA);
   SetIndexBuffer(1,downbuff,INDICATOR_DATA);
//--- geben Sie den Zeichencode für das Rendering in PLOT_ARROW vor
   PlotIndexSetInteger(0,PLOT_ARROW,241);
   PlotIndexSetInteger(1,PLOT_ARROW,242);
//---- erstellen Sie die Indikatorwerte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
   PlotIndexSetDouble(1,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      upbuff[i]=0;
      downbuff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- Wenn die Daten übereinstimmen, dann verwenden Sie den Wert aus einer
            if(time[i]==time_buff[j])
              {
               //--- zeichnen Sie den Zeiger je nach dem Signal
               if(sign_buff[j])
                  upbuff[i]=high[i];
               else
                  downbuff[i]=low[i];
               //--- vergrößern Sie den Zähler
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Typ bool, FileWrite

© 2000-2025, MetaQuotes Ltd.

FileReadDatetime

Liest von der Datei des Typs CSVeine Zeile eines der Formate: "YYYY.MM.DD HH:MI:SS", "YYYY.MM.DD" oder "HH:MI:SS" - und wandelt sie in den Wert des Typs datetime um.

```
datetime  FileReadDatetime(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Wert des Typs datetime.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWrite bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                        Demo_FileReadDateTime.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   2
//---- plot Label1
#property indicator_label1  "UpSignal"
#property indicator_type1   DRAW_ARROW
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  4
//---- plot Label2
#property indicator_label2  "DownSignal"
#property indicator_type2   DRAW_ARROW
#property indicator_color2  clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  4
//--- Parameter zum Lesen von Daten
input string InpFileName="MACD.csv";  // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- globale Variablen
int      ind=0;        // Index
double   upbuff[];     // der Indikator-Puffer der Zeiger nach oben
double   downbuff[];   // der Indikator-Puffer der Zeiger nach unten
bool     sign_buff[]; // ein Array von Signalen (true - kaufen, false - verkaufen)
datetime time_buff[]; // ein Array der Zeit des Eintritts der Signale
int      size=0;       // die Größe der Array von Signalen
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_CSV);
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      //--- Lesen Sie zuerst die Anzahl der Signale
      size=(int)FileReadNumber(file_handle);
      //--- trennen Sie den Speichern für den Arrays
      ArrayResize(sign_buff,size);
      ArrayResize(time_buff,size);
      //--- lesen Sie die Dateidaten
      for(int i=0;i<size;i++)
        {
         //--- Die Zeit des Signals
         time_buff[i]=FileReadDatetime(file_handle);
         //--- Der Signalwert
         sign_buff[i]=FileReadBool(file_handle);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
     }
   else
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
       return(INIT_FAILED);
      }
//--- Arrays-Bindung
   SetIndexBuffer(0,upbuff,INDICATOR_DATA);
   SetIndexBuffer(1,downbuff,INDICATOR_DATA);
//--- geben Sie den Zeichencode für das Rendering in PLOT_ARROW vor
   PlotIndexSetInteger(0,PLOT_ARROW,241);
   PlotIndexSetInteger(1,PLOT_ARROW,242);
//---- erstellen Sie die Indikatorwerte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
   PlotIndexSetDouble(1,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      upbuff[i]=0;
      downbuff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- Wenn die Daten übereinstimmen, dann verwenden Sie den Wert aus einer
            if(time[i]==time_buff[j])
              {
               //--- zeichnen Sie den Zeiger je nach dem Signal
               if(sign_buff[j])
                  upbuff[i]=high[i];
               else
                  downbuff[i]=low[i];
               //--- vergrößern Sie den Zähler
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Typ datetime, StringToTime, TimeToString, FileWrite

© 2000-2025, MetaQuotes Ltd.

FileReadDouble

Liest die Zahl der doppelten Genauigkeit mit dem Fliesspunkt (double) aus der Binärdatei von der laufenden Position des Dateianzeigers.

```
double  FileReadDouble(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Wert des Typs double.

Hinweis

Für die Erhlatung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteDouble bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileReadDouble.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "MA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  2
#property indicator_separate_window
//--- Parameter zum Lesen von Daten
input string InpFileName="MA.csv";     // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- globale Variablen
int      ind=0;
int      size=0;
double   ma_buff[];
datetime time_buff[];
//--- Indikator Buffer
double   buff[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_BIN);
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- lesen Sie zuerst wie viele Daten im Datei sind
      size=(int)FileReadDouble(file_handle);
      //--- trennen Sie den Speichern für den Arrays
      ArrayResize(ma_buff,size);
      ArrayResize(time_buff,size);
      //--- lesen Sie den Dateidaten
      for(int i=0;i<size;i++)
        {
         time_buff[i]=(datetime)FileReadDouble(file_handle);
         ma_buff[i]=FileReadDouble(file_handle);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten werden gelesen, %s Datei geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Array-Bindung zum Indikator Puffer mit 0 Index
```

© 2000-2025, MetaQuotes Ltd.

```
   SetIndexBuffer(0,buff,INDICATOR_DATA);
//---- erstellen Sie die Indikatorwerte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   ArraySetAsSeries(time,false);
//--- der Zyklus für noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      buff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- wenn die Daten identisch sind, dann verwenden Sie den Wert aus einer
            if(time[i]==time_buff[j])
              {
               buff[i]=ma_buff[j];
               //--- vergrößern Sie den Zähler
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Realtypen (double, float), StringToDouble, DoubleToString, FileWriteDouble

© 2000-2025, MetaQuotes Ltd.

FileReadFloat

Liest Zahl mit einfacher Genauigkeit mit dem Fliesspunkt von der laufenden Position der binaeren Datei.

```
float  FileReadFloat(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattibut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Wert des Typs float.

Hinweis

für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteFloat bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                            Demo_FileReadFloat.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "CloseLine"
#property indicator_type1   DRAW_COLOR_LINE
#property indicator_color1  clrRed,clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  2
//--- Parameter zum Lesen von Daten
input string InpFileName="Close.bin"; // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- globale Variablen
int      ind=0;
int      size=0;
double   close_buff[];
datetime time_buff[];
//--- Indicator Buffers
double   buff[];
double   color_buff[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
   int def_size=100;
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(close_buff,def_size);
   ArrayResize(time_buff,def_size);
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_BIN);
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- lesen Sie die Dateidaten
      while(!FileIsEnding(file_handle))
        {
         //--- lesen Sie die Zeitwert und Preisewert
         time_buff[size]=(datetime)FileReadDouble(file_handle);
         close_buff[size]=(double)FileReadFloat(file_handle);
         size++;
         //--- die Größe des Arrays zu erhöhen, wenn sie überfüllt sind
         if(size==def_size)
           {
            def_size+=100;
            ArrayResize(close_buff,def_size);
            ArrayResize(time_buff,def_size);
           }
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("Die Daten werden gelesen, %s Datei geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Arrays-Bindung zum Indikator Puffers
   SetIndexBuffer(0,buff,INDICATOR_DATA);
   SetIndexBuffer(1,color_buff,INDICATOR_COLOR_INDEX);
//---- erstellen Sie die Anzeige-Werte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   ArraySetAsSeries(time,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      buff[i]=0;
      color_buff[i]=0; // die rote Farbe als Voreinstellung
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- wenn die Daten identisch sind, dann verwenden Sie den Wert aus einer
            if(time[i]==time_buff[j])
              {
               //--- erhalten Sie einen Preis
               buff[i]=close_buff[j];
               //--- wenn der aktuelle Preis größer als die vorherige, so Farbe blau i
               if(buff[i-1]>buff[i])
                  color_buff[i]=1;
               //--- vergrößern Sie den Zähler
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Realtypen (double, float), FileReadDouble, FileWriteFloat

© 2000-2025, MetaQuotes Ltd.

FileReadInteger

Liest den Wert des Typs int, short oder char aus der binären Datei abhängig von der angegebenen Länge in Bytes. Das Lesen wird von der laufenden Position des Dateianzeigers durchgeführt.

```
int  FileReadInteger(
   int  file_handle,         // Datei-Handle
   int  size=INT_VALUE       // die Größe des ganzzahligen Typs in Bytes
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

size=INT_VALUE

[in] Die Anzahl der Bytes (bis zu 4), die Sie lesen möchten. Die folgenden Konstanten sind verfügbar: CHAR_VALUE=1, SHORT_VALUE=2 und INT_VALUE=4, so kann die Funktion den ganzzahligen Wert des Typs char, short oder int lesen.

Rückgabewert

Ein Wert vom Typ int. Das Ergebnis dieser Funktion muss deutlich in den Zieltyp, d.h. in den Typ, den Sie lesen möchten, geführt werden. Da der Rückgabewert vom Typ int ist, dann kann er leicht an jede Integer-Wert umgewandelt werden. Der Dateizeiger wird durch die Anzahl der gelesenen Bytes verschoben.

Hinweis

Wenn weniger als 4 Bytes gelesen werden, wird das Ergebnis immer positiv sein. Wenn ein oder zwei Bytes gelesen werden, können Sie das Vorzeichen durch eine explizite Umwandlung bzw. in den Typ char (1 Byte) oder short (2 Byte) bestimmen. Bestimmung vom Verzeichnis für die Drei-Byte-Zahl ist nicht trivial, da es keine entsprechenden Basiswerttyp gibt.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteInteger bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileReadInteger.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Trends"
#property indicator_type1   DRAW_SECTION
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Parameter zum Lesen von Daten
input string InpFileName="Trend.bin"; // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- globale Variablen
int      ind=0;
int      size=0;
datetime time_buff[];
//--- indicator buffers
double   buff[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
   int def_size=100;
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(time_buff,def_size);
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_BIN);
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- Hilfsvariablen
      int    arr_size;
      uchar  arr[];
      //--- lesen Sie die Dateidaten
      while(!FileIsEnding(file_handle))
        {
         //--- erkennen Sie wieviele Symbole dienen für Zeiterfassung
         arr_size=FileReadInteger(file_handle,INT_VALUE);
         ArrayResize(arr,arr_size);
         for(int i=0;i<arr_size;i++)
            arr[i]=(char)FileReadInteger(file_handle,CHAR_VALUE);
         //--- merken Sie den Zeitwert
         time_buff[size]=StringToTime(CharArrayToString(arr));
         size++;
         //--- die Größe des Arrays zu erhöhen, wenn sie überfüllt sind
         if(size==def_size)
           {
            def_size+=100;
            ArrayResize(time_buff,def_size);
           }
```

© 2000-2025, MetaQuotes Ltd.

```
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind gelesen, die Datei %s geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Arrays-Bindung zum Indikator Puffers
   SetIndexBuffer(0,buff,INDICATOR_DATA);
//---- erstellen Sie die Indikatorwerte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(close,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      buff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- wenn die Daten identisch sind, dann verwenden Sie den Wert aus einer
            if(time[i]==time_buff[j])
              {
               //--- erhalten Sie einen Preis
               buff[i]=close[i];
               //--- vergrößern Sie den Zähler
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

IntegerToString, StringToInteger, Ganzzahlige Typen, FileWriteInteger

© 2000-2025, MetaQuotes Ltd.

FileReadLong

Liest die Ganzzahl des Typs long (8 Byte) von der laufenden Position der binären Datei.

```
long  FileReadLong(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Wert des Typs long.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteLong bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                              Demo_FileReadLong.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Volume"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrYellow
#property indicator_style1  STYLE_SOLID
#property indicator_width1  2
#property indicator_separate_window
//--- Parameter zum Lesen von Daten
input string InpFileName="Volume.bin"; // der Dateiname
input string InpDirectoryName="Data";  // der Verzeichnisname
//--- globale Variablen
int      ind=0;
int      size=0;
long     volume_buff[];
datetime time_buff[];
//--- indicator buffers
double   buff[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_BIN);
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- lesen Sie zuerst wie viele Daten im Datei sind
      size=(int)FileReadLong(file_handle);
      //--- trennen Sie den Speichern für den Arrays
      ArrayResize(volume_buff,size);
      ArrayResize(time_buff,size);
      //--- lesen Sie den Dateidaten
      for(int i=0;i<size;i++)
        {
         time_buff[i]=(datetime)FileReadLong(file_handle);
         volume_buff[i]=FileReadLong(file_handle);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind gelesen, die Datei %s geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Array-Bindung zum Indikator Puffer mit 0 Index
   SetIndexBuffer(0,buff,INDICATOR_DATA);
```

© 2000-2025, MetaQuotes Ltd.

```
//---- erstellen Sie die Indikatorwerte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   ArraySetAsSeries(time,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- Voreinstellung 0
      buff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- wenn die Daten identisch sind, dann verwenden Sie den Wert aus einer
            if(time[i]==time_buff[j])
              {
               buff[i]=(double)volume_buff[j];
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Ganzzahlige Typen, FileReadInteger, FileWriteLong)

© 2000-2025, MetaQuotes Ltd.

FileReadNumber

Liest aus der Datei des Typs CSV die Zeile von der laufenden Position bis zum Begrenzer (oder bis zum Ende der Textzeile) und wandelt die gelesene Zeile in den Wert des Typs double um.

```
double  FileReadNumber(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Wert des Typs double.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteString bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileReadNumber.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 5
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Overbought & Oversold"
#property indicator_type1   DRAW_COLOR_BARS
#property indicator_color1  clrRed, clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  2
//--- Parameter zum Lesen von Daten
input string InpFileName="RSI.csv";   // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//--- die Indikator-Buffers
double   open_buff[];
double   high_buff[];
double   low_buff[];
double   close_buff[];
double   color_buff[];
//--- überkauft Variablen
int      ovb_ind=0;
int      ovb_size=0;
datetime ovb_time[];
//--- überverkauft Variablen
int      ovs_ind=0;
int      ovs_size=0;
datetime ovs_time[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Die standartmäßige Variable Größe von Arrays
   int ovb_def_size=100;
   int ovs_def_size=100;
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(ovb_time,ovb_def_size);
   ArrayResize(ovs_time,ovs_def_size);
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_CSV|FILE_
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat(" %s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      double value;
      //--- lesen Sie die Dateidaten
      while(!FileIsEnding(file_handle))
        {
         //--- lesen Sie den ersten Wert in eine Zeichenfolge
         value=FileReadNumber(file_handle);
         //--- lesen Sie die verschiedenen Arrays, abhängig vom Ergebnis der Funktion
         if(value>=70)
            ReadData(file_handle,ovb_time,ovb_size,ovb_def_size);
```

© 2000-2025, MetaQuotes Ltd.

```
         else
            ReadData(file_handle,ovs_time,ovs_size,ovs_def_size);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten werden gelesen, die Datei %s geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Arrays-Bindung
   SetIndexBuffer(0,open_buff,INDICATOR_DATA);
   SetIndexBuffer(1,high_buff,INDICATOR_DATA);
   SetIndexBuffer(2,low_buff,INDICATOR_DATA);
   SetIndexBuffer(3,close_buff,INDICATOR_DATA);
   SetIndexBuffer(4,color_buff,INDICATOR_COLOR_INDEX);
//---- erstellen Sie die Anzeige-Werte, die im Diagramm nicht sichtbar sein werden
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Lesen Sie die Zeilen einer Datei                                 |
//+------------------------------------------------------------------+
void ReadData(const int file_handle,datetime &arr[],int &size,int &def_size)
  {
   bool flag=false;
//--- Lesen Sie, bis das Ende der Zeichenfolge oder Datei erreichen
   while(!FileIsLineEnding(file_handle) && !FileIsEnding(file_handle))
     {
      //--- bewegen Sie den Wagen, nach dem Lesen der Anzahl
      if(flag)
         FileReadNumber(file_handle);
      //--- das aktuelle Datum speichern
      arr[size]=FileReadDatetime(file_handle);
      size++;
      //--- Falls erforderlich, vergrößern Sie die Größe des Arrays
      if(size==def_size)
        {
         def_size+=100;
         ArrayResize(arr,def_size);
        }
      //--- übergeben die erste iteration
      flag=true;
     }
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
{
```

© 2000-2025, MetaQuotes Ltd.

```
   ArraySetAsSeries(time,false);
   ArraySetAsSeries(open,false);
   ArraySetAsSeries(high,false);
   ArraySetAsSeries(low,false);
   ArraySetAsSeries(close,false);
//--- der Zyklus für die noch unbearbeiteten Bars
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      open_buff[i]=0;
      high_buff[i]=0;
      low_buff[i]=0;
      close_buff[i]=0;
      color_buff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ovb_ind<ovb_size)
         for(int j=ovb_ind;j<ovb_size;j++)
           {
            //--- wenn die Daten identisch sind, ist die Bar in überkauft Zone
            if(time[i]==ovb_time[j])
              {
               open_buff[i]=open[i];
               high_buff[i]=high[i];
               low_buff[i]=low[i];
               close_buff[i]=close[i];
               //--- 0 - die rote Farbe
               color_buff[i]=0;
               //--- vergrössern Sie den Zähler
               ovb_ind=j+1;
               break;
              }
           }
      //--- prüfen Sie, ob es noch Daten gibt
      if(ovs_ind<ovs_size)
         for(int j=ovs_ind;j<ovs_size;j++)
           {
            //--- wenn die Daten identisch sind, ist die Bar in überverkauft Zone
            if(time[i]==ovs_time[j])
              {
               open_buff[i]=open[i];
               high_buff[i]=high[i];
               low_buff[i]=low[i];
               close_buff[i]=close[i];
               //--- 1 - die blau Farbe
               color_buff[i]=1;
               //--- vergrössern Sie den Zähler
               ovs_ind=j+1;
               break;
              }
           }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Der Bearbeiter des Ereignisses ChartEvent                        |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam
```

© 2000-2025, MetaQuotes Ltd.

```
                  )
  {
//--- ändern Sie die Breite des Indikators je nach dem Ausmaß
   if(ChartGetInteger(0,CHART_SCALE)>3)
      PlotIndexSetInteger(0,PLOT_LINE_WIDTH,2);
   else
      PlotIndexSetInteger(0,PLOT_LINE_WIDTH,1);
  }
```

Sehen Sie auch

FileWriteString

© 2000-2025, MetaQuotes Ltd.

FileReadString

Liest aus der Datei die Zeile von der laufenden Position des Dateianzeigers.

```
string  FileReadString(
   int  file_handle,     // Datei-Handle
   int  length=-1        // Zeilenlänge
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

length=-1

[in] Anzahl der Symbole für Lesen.

Rückgabewert

Gelesene Zeile(string).

Hinweis

Beim Lesen aus der bin-Datei ist es notwendig, die Länge der gelesenen Zeile anzugeben. Beim Lesen aus der txt-Datei braucht man nicht, die Zeilenlänge zu spezifizieren, die Zeile wird von der laufenden Position bis zum Zeilenvorschubszeichen "\r\n gelesen". Beim Lesen aus der csv-Datei braucht man auch nicht die Zeilenlänge anzugeben, die Zeile wird von der laufenden Position bis zum nächsten Begrenzer oder bis zum Endezeichen der Textzeile.

Wenn die Datei mit Flagge FILE_ANSI geöffnet wird, wird die gelesene Zeile in Unicode umgewandelt.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteInteger bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Parameter zum Lesen von Daten
input string InpFileName="Trend.bin"; // der Dateiname
input string InpDirectoryName="Data"; // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_BIN|FILE_
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- Hilfsvariablen
      int    str_size;
      string str;
      //--- lesen Sie die Dateidaten
      while(!FileIsEnding(file_handle))
        {
         //--- erkennen Sie wieviele Symbole dienen für Zeiterfassung
         str_size=FileReadInteger(file_handle,INT_VALUE);
         //--- lesen Sie die Zeile
         str=FileReadString(file_handle,str_size);
         //--- drucken Sie die Zeile
         PrintFormat(str);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind gelesen, die Datei %s geschlossen",InpFileName);
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

Typ string, Datenverarbeitung, FileWriteInteger

© 2000-2025, MetaQuotes Ltd.

FileReadStruct

Liest den Inhalt aus der binären Datei in die Struktur, die als Parameter übertragen wird. Lesen wird von der laufenden Position des Dateianzeigers durchgeführt.

```
uint  FileReadStruct(
   int           file_handle,         // Datei-Handle
   const void&   struct_object,       // Zielstruktur für Einlesen
   int           size=-1              // Strukturgröße in Bytes
   );
```

Parameter

file_handle

[in] Dateittribut der geöffneten binären Datei .

struct_object

[out] Objekt der angegebenen Struktur. Die Struktur darf keine Strings, dynamischen Arrays, virtuellen Funktionen, Objekte von Klassen sowie keine Pointer auf Objekte und Funktionen beinhalten.

size=-1

[in] Anzahl der Bytes, die eingelesen werden müssen. Wenn die Größe nicht angegeben wird oder die Anzahl der angegebenen Bytes größer als die Strukturgröße ist, wird die genaue Größe der angegebenen Struktur verwendet.

Rückgabewert

Im Erfolgsfall gibt die Funktion die Anzahl der eingelesenen Bytes zurück. Dateianzeiger wird um dieselbe Zahl der Bytes verschoben.

Beispiel (die Datei wird verwendet, die infolge der Arbeit des Beispiels für die Funktion FileWriteStruct bekommen ist)

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileReadStruct.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_separate_window
#property indicator_buffers 4
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Candles"
#property indicator_type1   DRAW_CANDLES
#property indicator_color1  clrOrange
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
#property indicator_separate_window
//--- Die Parameter für das Erhalten der Daten
input string  InpFileName="EURUSD.txt"; // der Dateiname
input string  InpDirectoryName="Data";  // der Verzeichnisname
//+------------------------------------------------------------------+
//| Struktur für die Speicherung von Daten-Kerzen                    |
//+------------------------------------------------------------------+
struct candlesticks
  {
   double            open;  // Eröffnungspreis
   double            close; // Schlußpreis
   double            high;  // der maximale Preis
   double            low;   // der minimale Preis
   datetime          date;  // Datum
  };
//--- Indikator Buffers
double       open_buff[];
double       close_buff[];
double       high_buff[];
double       low_buff[];
//--- globale Variablen
candlesticks cand_buff[];
int          size=0;
int          ind=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
   int default_size=100;
   ArrayResize(cand_buff,default_size);
//--- öffnen Sie die Datei
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_BIN|FILE_
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_COMMONDATA
      //--- lesen Sie die Dateidaten
      while(!FileIsEnding(file_handle))
        {
         //--- zeichnen Sie die Dateien in den Array auf
         FileReadStruct(file_handle,cand_buff[size]);
         size++;
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- prüfen Sie das Array für Überbelegung
         if(size==default_size)
           {
            //--- erhöhen die Dimensionalität des Arrays
            default_size+=100;
            ArrayResize(cand_buff,default_size);
           }
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind gelesen, die Datei %s geschlossen",InpFileName);
     }
   else
     {
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
      return(INIT_FAILED);
     }
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,open_buff,INDICATOR_DATA);
   SetIndexBuffer(1,high_buff,INDICATOR_DATA);
   SetIndexBuffer(2,low_buff,INDICATOR_DATA);
   SetIndexBuffer(3,close_buff,INDICATOR_DATA);
//--- der Leerwert
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   ArraySetAsSeries(time,false);
//--- der Zyklus für die noch unbearbeiteten Kerzen
   for(int i=prev_calculated;i<rates_total;i++)
     {
      //--- die Voreinstellung 0
      open_buff[i]=0;
      close_buff[i]=0;
      high_buff[i]=0;
      low_buff[i]=0;
      //--- prüfen Sie, ob es noch Daten gibt
      if(ind<size)
        {
         for(int j=ind;j<size;j++)
           {
            //--- wenn die Daten identisch sind, dann verwenden Sie den Wert aus einer
            if(time[i]==cand_buff[j].date)
              {
               open_buff[i]=cand_buff[j].open;
               close_buff[i]=cand_buff[j].close;
               high_buff[i]=cand_buff[j].high;
               low_buff[i]=cand_buff[j].low;
               //--- vergrößern Sie den Zähler
               ind=j+1;
               break;
              }
           }
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
```

Sehen Sie auch

Strukturen und Klassen, FileWriteStruct

© 2000-2025, MetaQuotes Ltd.

FileSeek

Verschiebt die Position des Dateianzeigers um die angegebene Anzahl der Bytes gegen die angegebene Position.

```
bool  FileSeek(
   int                  file_handle,     // Datei-Handle
   long                 offset,          // in Bytes
   ENUM_FILE_POSITION   origin           // Position für Abzählung
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

offset

[in] übertragung in Bytes (kann auch negativen Wert annehmen).

origin

[in] Anfangspunkt für übertragung. Kann einen der Enumerationswerte ENUM_FILE_POSITION annehmen.

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false. Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Wenn das Ergebnis der Funktionsdurchführung FileSeek() eine negative Zahl ist (ausserhalb der "linken Grenze" der Datei), wird Dateianzeiger am Dateianfang gestellt.

Wenn die Position ausserhalb der "rechten Grenze" der Datei (mehr als Dateigröße) gestellt wird , wird das nächste Schreiben in die Datei nicht von der Dateiende, sondern von der gestellten Position durchgeführt wird. Dabei werden unbestimmte Werte zwischen dem frühren Dateiende und der gestellten Position geschrieben.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                Demo_FileSeek.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpFileName="file.txt";     // der Dateiname
input string InpDirectoryName="Data";    // der Verzeichnisname
input int    InpEncodingType=FILE_ANSI; // ANSI=32 oder UNICODE=64
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- feststellen Sie den Variablenwert um Zufallszahlen zu generieren
   _RandomSeed=GetTickCount();
//--- Variabel für die Positionen des Anfanges der Zeilen
   ulong pos[];
   int   size;
//--- stürzen Sie den Fehlerwert
   ResetLastError();
//--- öffnen Sie die Datei
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_TXT|InpEr
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      //--- bekommen Sie die Position des Anfanges für jede Zeile in der Datei
      GetStringPositions(file_handle,pos);
      //--- ermitteln Sie wieviel allen der Zeilen in der Datei sind
      size=ArraySize(pos);
      if(!size)
        {
         //--- Wenn es in der Datei keine Zeilen gibt, so beenden wir die Arbeit
         PrintFormat("Die Datei %s ist leer!",InpFileName);
         FileClose(file_handle);
         return;
        }
      //--- wählen Sie zufällig die Nummer der Zeile
      int ind=MathRand()%size;
      //--- verschieben Sie die Position an den Anfang der Zeile
      if(FileSeek(file_handle,pos[ind],SEEK_SET)==true)
        {
      //--- lesen und drucken Sie eine Zeile mit der Nummer ind
         PrintFormat("Der Text der Zeile mit der Nummer %d: \"%s\"",ind,FileReadString
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("die Datei %s ist geschlossen",InpFileName);
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
//+------------------------------------------------------------------+
//| Funktion bestimmt die Startposition für jede der Zeilen in der   |
```

© 2000-2025, MetaQuotes Ltd.

```
//| Datei und legt sie in das Array arr                              |
//+------------------------------------------------------------------+
void GetStringPositions(const int handle,ulong &arr[])
  {
//--- die standardmässige Größe des Arrays
   int def_size=127;
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(arr,def_size);
//--- Der Zähler der Zeilen
   int i=0;
//--- Wenn nicht das Ende der Datei, dann es gibt mindestens eine Zeile
   if(!FileIsEnding(handle))
     {
      arr[i]=FileTell(handle);
      i++;
     }
   else
      return; // die Datei ist leer, gehen
//--- bestimmen Sie die Verschiebung in den Bytes je nach der Kodierung
   int shift;
   if(FileGetInteger(handle,FILE_IS_ANSI))
      shift=1;
   else
      shift=2;
//--- im Zyklus lesen Sie die Zeilen aus
   while(1)
     {
      //--- lesen Sie die Zeile
      FileReadString(handle);
      //--- Die Prüfung auf dem Dateiende
      if(!FileIsEnding(handle))
        {
         //--- merken Sie die Position der nächsten Zeile
         arr[i]=FileTell(handle)+shift;
         i++;
         //--- die Größe des Arrays zu erhöhen, wenn es überfüllt ist
         if(i==def_size)
           {
            def_size+=def_size+1;
            ArrayResize(arr,def_size);
           }
        }
      else
         break; // der Dateiende, gehen
     }
//--- feststellen Sie den wahrhaften Umfang des Arrays
   ArrayResize(arr,i);
  }
```

© 2000-2025, MetaQuotes Ltd.

FileSize

Gibt die Größe der Datei in Bytes zurück.

```
ulong  FileSize(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Wert des Typs int.

Hinweis

Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input ulong  InpThresholdSize=20;        // die Grenze von Dateigröße im Kilobyte
input string InpBigFolderName="big";     // Ordner für die größen Dateien
input string InpSmallFolderName="small"; // Ordner für die kleinen Dateien
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string   file_name;      // Variable für Speichern die Dateinamen
   string   filter="*.csv"; // Der Filter für die Suche nach Dateien
   ulong    file_size=0;    // die Größe der Datei in Bytes
   int      size=0;         // die Anzahl der Dateien
//--- drucken Sie der Weg zum Ordner in der arbeiten werden
   PrintFormat("Arbeiten in dem Ordner %s\\Files\\",TerminalInfoString(TERMINAL_COMMON
//--- erhalten Sie die Suche Handle von Grund auf den Gesamtordner aller Client-Termi
   long search_handle=FileFindFirst(filter,file_name,FILE_COMMON);
//--- prüfen Sie, ob die FileFindFirst() Funktion erfolgreich durchgearbeitet hat
   if(search_handle!=INVALID_HANDLE)
     {
      //--- im Zyklus versetzen Sie die Dateien je nach ihrer Größe
      do
        {
         //--- öffnen Sie die Datei
         ResetLastError();
         int file_handle=FileOpen(file_name,FILE_READ|FILE_CSV|FILE_COMMON);
         if(file_handle!=INVALID_HANDLE)
           {
            //--- erhalten Sie die Größe der Datei
            file_size=FileSize(file_handle);
            //--- schließen Sie die Datei
            FileClose(file_handle);
           }
         else
           {
            PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",file_name,G
            continue;
           }
         //--- drucken Sie die Größe der Datei
         PrintFormat("die Größe der Datei %s ist %d Byte gleich",file_name,file_size);
         //--- ermitteln Sie den Pfad um die Datei zu versetzen
         string path;
         if(file_size>InpThresholdSize*1024)
            path=InpBigFolderName+"//"+file_name;
         else
            path=InpSmallFolderName+"//"+file_name;
         //--- versetzen Sie die Datei
         ResetLastError();
         if(FileMove(file_name,FILE_COMMON,path,FILE_REWRITE|FILE_COMMON))
            PrintFormat("%s Datei ist versetzt",file_name);
         else
            PrintFormat("Fehler, Kode = %d",GetLastError());
        }
      while(FileFindNext(search_handle,file_name));
      //--- schließen Sie die Suche Handle
      FileFindClose(search_handle);
     }
   else
      Print("Dateien sind nicht geöffnet!");
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

FileTell

Gibt die laufende Position des Dateianzeigers der entsprechenden geöffneten Datei.

```
ulong  FileTell(
   int  file_handle      // Datei-Handle
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

Rückgabewert

Laufende Position des Dateiattributs in Bytes vom Anfang der Datei.

Hinweis

Für die Erhaltung der fehlerbezogenen Information muss die Funktion GetLastError() aufgerufen werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                Demo_FileTell.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameters
input string InpFileName="file.txt";     // der Dateiname
input string InpDirectoryName="Data";    // der Verzeichnisname
input int    InpEncodingType=FILE_ANSI; // ANSI=32 oder UNICODE=64
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- feststellen Sie den Variablenwert um Zufallszahlen zu generieren
   _RandomSeed=GetTickCount();
//--- Variabel für die Positionen des Anfanges der Zeilen
   ulong pos[];
   int   size;
//--- stürzen Sie den Fehlerwert
   ResetLastError();
//--- öffnen Sie die Datei
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_TXT|InpEr
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      //--- bekommen Sie die Position des Anfanges für jede Zeile in der Datei
      GetStringPositions(file_handle,pos);
      //--- ermitteln Sie wieviel allen der Zeilen in der Datei sind
      size=ArraySize(pos);
      if(!size)
        {
         //--- Wenn es in der Datei keine Zeilen gibt, so beenden wir die Arbeit
         PrintFormat("Die Datei %s ist leer!",InpFileName);
         FileClose(file_handle);
         return;
        }
      //--- wählen Sie zufällig die Nummer der Zeile
      int ind=MathRand()%size;
      //--- verschieben Sie die Position an den Anfang der Zeile
      FileSeek(file_handle,pos[ind],SEEK_SET);
      //--- lesen und drucken Sie eine Zeile mit der Nummer ind
      PrintFormat("Der Text der Zeile mit der Nummer %d: \"%s\"",ind,FileReadString(fi
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("die Datei %s ist geschlossen",InpFileName);
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
//+------------------------------------------------------------------+
//| Funktion bestimmt die Startposition für jede der Zeilen in der   |
```

© 2000-2025, MetaQuotes Ltd.

```
//| Datei und legt sie in das Array arr                              |
//+------------------------------------------------------------------+
void GetStringPositions(const int handle,ulong &arr[])
  {
//--- die standardmässige Größe des Arrays
   int def_size=127;
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(arr,def_size);
//--- Der Zähler der Zeilen
   int i=0;
//--- Wenn nicht das Ende der Datei, dann es gibt mindestens eine Zeile
   if(!FileIsEnding(handle))
     {
      arr[i]=FileTell(handle);
      i++;
     }
   else
      return; // die Datei ist leer, gehen
//--- bestimmen Sie die Verschiebung in den Bytes je nach der Kodierung
   int shift;
   if(FileGetInteger(handle,FILE_IS_ANSI))
      shift=1;
   else
      shift=2;
//--- im Zyklus lesen Sie die Zeilen aus
   while(1)
     {
      //--- lesen Sie die Zeile
      FileReadString(handle);
      //--- Die Prüfung auf dem Dateiende
      if(!FileIsEnding(handle))
        {
         //--- merken Sie die Position der nächsten Zeile
         arr[i]=FileTell(handle)+shift;
         i++;
         //--- die Größe des Arrays zu erhöhen, wenn es überfüllt ist
         if(i==def_size)
           {
            def_size+=def_size+1;
            ArrayResize(arr,def_size);
           }
        }
      else
         break; // der Dateiende, gehen
     }
//--- feststellen Sie den wahrhaften Umfang des Massives
   ArrayResize(arr,i);
  }
```

© 2000-2025, MetaQuotes Ltd.

FileWrite

Aufzeichnung der Daten in die Datei des Typs CSV oder TXT, der Datenbegrenzer wird automatisch eingefügt, wenn er nicht 0 gleich ist. Nach Aufzeichnung in die Datei wird Zeilenendesymbol "\r\n" hinzugefügt.

```
uint  FileWrite(
   int  file_handle,      // Datei-Handle
   ...                    // Liste der geschriebenen Parameter
   );
```

Parameter

file_handle

[in] Dateidescriptor, der durch die Funktion FileOpen() zurückgegeben wird.

...

[in] Liste der Parameter, die durch Kommas getrennt werden, für Schreiben in die Datei. Anzahl der in die Datei geschriebenen Parameter muss nicht mehr als 63 sein.

Rückgabewert

Anzahl der geschriebenen Bytes.

Hinweis

Bei der Ausgabe werden numerische Daten in Textformat umgewandelt (s. die Funktion Print()). Daten des Typs double werden mit der Genauigkeit bis 16 Dezimalzeichen ausgegeben, dabei können Daten entweder im traditionellen oder im wissenschaftlichen Format ausgegeben - abhängig davon, welche Aufzeichnung kompakter ist. Daten des Typs werden mit 5 Dezimalzeichen ausgegeben. Für Ausgabe der Realzahlen mit anderer Genauigkeit oder im explizit angegebenen Format muss die Funktion DoubleToString() verwendet werden.

Zahlen des Typs bool werden als die Zeilen "true" oder "false" ausgegeben. Zahlen des Typs datetime werden im Format "YYYY.MM.DD HH:MI:SS" ausgegeben.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                Demo_FileWrite.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string             InpSymbolName="EURUSD";           // Währungspaar
input ENUM_TIMEFRAMES    InpSymbolPeriod=PERIOD_H1;        // Zeitspanne
input int                InpFastEMAPeriod=12;              // die Periode des schnelle
input int                InpSlowEMAPeriod=26;              // die Periode des langsame
input int                InpSignalPeriod=9;                // die durchschnittliche Pe
input ENUM_APPLIED_PRICE InpAppliedPrice=PRICE_CLOSE;      // der Typ des Preises
input datetime           InpDateStart=D'2012.01.01 00:00'; // das Anfangsdatum zum Kop
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string             InpFileName="MACD.csv";  // der Dateiname
input string             InpDirectoryName="Data"; // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date_finish; // das Enddatum zum Kopieren von Daten
   bool     sign_buff[]; // ein Array von Signalen (true - kaufen, false - verkaufen
   datetime time_buff[]; // ein Array der Zeit des Eintritts der Signale
   int      sign_size=0; // die Größe der Array von Signalen
   double   macd_buff[]; // ein Array von Werten des Indikators
   datetime date_buff[]; // ein Array von Daten des Indikators
   int      macd_size=0; // die Größe der Array von Indikator
//--- Die Zeit des Abschlusses - laufend
   date_finish=TimeCurrent();
//--- bekommen Sie die Handle des Indikators MACD
   ResetLastError();
   int macd_handle=iMACD(InpSymbolName,InpSymbolPeriod,InpFastEMAPeriod,InpSlowEMAPeri
   if(macd_handle==INVALID_HANDLE)
     {
      //--- Es misslang, die Handle des Indikators zu bekommen
      PrintFormat("Fehler beim Erhalten der Handle des Indikators. Fehlercode = %d",Ge
      return;
     }
//--- befinden Sie sich im Zyklus, bis der Indikator alle Werten rechnen wird
   while(BarsCalculated(macd_handle)==-1)
      Sleep(10); // die Verzögerungszeit, damit der Indikator dazugekommen ist die Wer
//--- Kopieren Sie die Werte eines Indikators für eine bestimmte Periode
   ResetLastError();
   if(CopyBuffer(macd_handle,0,InpDateStart,date_finish,macd_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren der Werte des Indikators. Fehlercode = %d",Get
      return;
     }
//--- kopieren Sie die entsprechende Zeit für die Werte des Indikators
   ResetLastError();
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,date_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Zeitwerten. Fehlercode = %d",GetLastError
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- befreien Sie das Gedächtnis, das mit dem Indikator eingenommen wird
   IndicatorRelease(macd_handle);
//--- erhalten Sie die Größe des Buffers
   macd_size=ArraySize(macd_buff);
//--- Analysieren der Daten und speichern die Indikator-Signale in Arrays
   ArrayResize(sign_buff,macd_size-1);
   ArrayResize(time_buff,macd_size-1);
   for(int i=1;i<macd_size;i++)
     {
      //--- Das Signal auf den Kauf
      if(macd_buff[i-1]<0 && macd_buff[i]>=0)
        {
         sign_buff[sign_size]=true;
         time_buff[sign_size]=date_buff[i];
         sign_size++;
        }
      //--- Das Signal auf den Verkauf
      if(macd_buff[i-1]>0 && macd_buff[i]<=0)
        {
         sign_buff[sign_size]=false;
         time_buff[sign_size]=date_buff[i];
         sign_size++;
        }
     }
//--- öffnen Sie die Datei zum Schreiben die Indikatorwerte (wenn nicht, dann sie auto
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- schreiben Sie zuerst die Anzahl der Signale
      FileWrite(file_handle,sign_size);
      //--- zeichnen Sie die Zeit der Signale und ihre Werte in der Datei auf
      for(int i=0;i<sign_size;i++)
         FileWrite(file_handle,time_buff[i],sign_buff[i]);
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

Comment, Print, StringFormat

© 2000-2025, MetaQuotes Ltd.

FileWriteArray

Schreibt in die Datei des Typs BIN Arrays verschiedener Typen ausser Zeilentypen (kann Feld der Strukturen sein, das keine Zeilen und dynamische Arrays enthält).

```
uint  FileWriteArray(
   int           file_handle,          // Datei-Handle
   const void&   array[],              // Feld
   int           start=0,              // Anfangsindex im Feld
   int           count=WHOLE_ARRAY     // Anzahl der Elemente
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

array[]

[out] Feld für Aufzeichnung.

start=0

[in] Anfangsindex im Feld (Nummer des ersten aufgezeichneten Elementes).

count=WHOLE_ARRAY

[in] Anzahl der zu aufgezeichneten Elemente (WHOLE_ARRAY bedeutet, dass alle Elemente seit der Nummer start bis zum Arraysende aufgezeichnet werden).

Rückgabewert

Anzahl der aufgezeichneten Elemente.

Hinweis

Zeilenfeld kann nur in die Datei des Typs TXT geschrieben werden. In diesem Fall werden die Zeilen automatischb mit Symbolen der Zeilenende "\r\n" beendet. Abhängig vom Dateityp ANSI oder UNICODE, werden die Zeilen zur ansi-Kodierung reduziert oder nicht.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileWriteArray.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Eingabeparameters
input string InpFileName="data.bin";
input string InpDirectoryName="SomeFolder";
//+------------------------------------------------------------------+
//| Struktur für die Speicherung von Daten auf die Preise            |
//+------------------------------------------------------------------+
struct prices
  {
   datetime          date; // Datum
   double            bid;  // der Bid Preis
   double            ask;  // der Ask Preis
  };
//--- globale Variablen
int    count=0;
int    size=20;
string path=InpDirectoryName+"//"+InpFileName;
prices arr[];
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- trennen Sie den Speichern für den Arrays
   ArrayResize(arr,size);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Aufzeichnung die verbleibenden count Zeilen, wenn count<n
   WriteData(count);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//--- Speichern Sie die Daten in einem Array
   arr[count].date=TimeCurrent();
   arr[count].bid=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   arr[count].ask=SymbolInfoDouble(Symbol(),SYMBOL_ASK);
//--- zeigen Sie aktuelle Daten an
   Print("Date = ",arr[count].date," Bid = ",arr[count].bid," Ask = ",arr[count].ask);
//--- vergrössern Sie den Zähler
   count++;
//--- Wenn das Array ausgefüllt wurde, dann die Daten in einer Datei speichern und nul
   if(count==size)
     {
      WriteData(size);
      count=0;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   }
//+------------------------------------------------------------------+
//| Aufzeichnung die n Elementen des Arrays in der Datei             |
//+------------------------------------------------------------------+
void WriteData(const int n)
   {
//--- öffnen Sie die Datei
    ResetLastError();
    int handle=FileOpen(path,FILE_READ|FILE_WRITE|FILE_BIN);
    if(handle!=INVALID_HANDLE)
      {
       //--- schreiben Sie des Arrays Daten an das Ende der Datei
       FileSeek(handle,0,SEEK_END);
       FileWriteArray(handle,arr,0,n);
       //--- schließen Sie die Datei
       FileClose(handle);
      }
    else
       Print("Fehler beim Öffnen der Datei, Fehler ",GetLastError());
   }
```

Sehen Sie auch

Variablen, FileSeek

© 2000-2025, MetaQuotes Ltd.

FileWriteDouble

Schreibt in bin-Datei den Wert des Parameters des Typs double von der laufenden Position des Dateianzeigers.

```
uint  FileWriteDouble(
   int     file_handle,     // Datei-Handle
   double  value            // Werte für Aufzeichnung
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

value

[in] Wert des Typs double.

Rückgabewert

Im Erfolgsfall gibt die Funktion die Anzahl der geschriebenen Bytes zurück (in diesem Fall sizeof(double)=8). Dateianzeiger wird um dieselbe Anzahl der Bytes verschoben.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileWriteDouble.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string             InpSymbolName="EURJPY";           // Währungspaar
input ENUM_TIMEFRAMES    InpSymbolPeriod=PERIOD_M15;       // Zeitspanne
input int                InpMAPeriod=10;                   // die Glätten Periode
input int                InpMAShift=0;                     // die Absetzung des Indika
input ENUM_MA_METHOD     InpMAMethod=MODE_SMA;             // der Glätten Typ
input ENUM_APPLIED_PRICE InpAppliedPrice=PRICE_CLOSE;      // der Typ des Preises
input datetime           InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum zum Kop
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string             InpFileName="MA.csv";     // der Dateiname
input string             InpDirectoryName="Data"; // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date_finish=TimeCurrent();
   double   ma_buff[];
   datetime time_buff[];
   int      size;
//--- bekommen Sie die Handle des Indikators MA
   ResetLastError();
   int ma_handle=iMA(InpSymbolName,InpSymbolPeriod,InpMAPeriod,InpMAShift,InpMAMethod,
   if(ma_handle==INVALID_HANDLE)
     {
      //--- Es misslang, die Handle des Indikators zu bekommen
      PrintFormat("Fehler beim Erhalten der Handle des Indikators. Fehlercode = %d",Ge
      return;
     }
//--- befinden Sie sich im Zyklus, bis der Indikator alle Werten rechnen wird
   while(BarsCalculated(ma_handle)==-1)
      Sleep(20); // die Verzögerungszeit, damit der Indikator dazugekommen ist die Wer
   PrintFormat("die Werte des Indikators, da seit %s beginnen, werden in die Datei auf
//--- Kopieren Sie die Werte eines Indikators
   ResetLastError();
   if(CopyBuffer(ma_handle,0,InpDateStart,date_finish,ma_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren der Werte des Indikators. Fehlercode = %d",Get
      return;
     }
//--- Kopieren Sie die Zeit des Erscheinens der entsprechenden Bars
   ResetLastError();
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,time_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Zeitwerte. Fehlercode = %d",GetLastError()
      return;
     }
//--- erhalten Sie die Größe des Buffers
   size=ArraySize(ma_buff);
//--- befreien Sie das Gedächtnis, das mit dem Indikator eingenommen wird
   IndicatorRelease(ma_handle);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- öffnen Sie die Datei zum Schreiben die Indikatorwerte (wenn nicht, dann sie auto
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- schreiben Sie zuerst die Große der Beispieldaten
      FileWriteDouble(file_handle,(double)size);
      //--- schreiben Sie die Zeit und die Indikatorwerte in der Datei
      for(int i=0;i<size;i++)
        {
         FileWriteDouble(file_handle,(double)time_buff[i]);
         FileWriteDouble(file_handle,ma_buff[i]);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

Realtypen (double, float)

© 2000-2025, MetaQuotes Ltd.

FileWriteFloat

Schreibt den Parameterwert des Typs foat in bin-Datei von der laufenden Position des Dateianzeigers.

```
uint  FileWriteFloat(
   int    file_handle,     // Datei-Handle
   float  value            // geschriebener Wert
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

value

[in] Wert des Typs float.

Rückgabewert

Im Erfolgsfall gibt die Funktion die Anzahl der geschriebenen Bytes (in diesem Fall sizeof(float)=4). Dateianzeiger wird um dieselbe Zahl der Bytes verschiebt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileWriteFloat.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string          InpSymbolName="EURUSD";           // Währungspaar
input ENUM_TIMEFRAMES InpSymbolPeriod=PERIOD_M15;       // Zeitspanne
input datetime        InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum zum Kopier
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string          InpFileName="Close.bin"; // der Dateiname
input string          InpDirectoryName="Data"; // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date_finish=TimeCurrent();
   double   close_buff[];
   datetime time_buff[];
   int      size;
//--- stürzen Sie den Fehlerwert
   ResetLastError();
//--- Kopieren Sie der Schlußpreis für jede Bar
   if(CopyClose(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,close_buff)==-1
     {
      PrintFormat("Fehler beim Kopieren von Werte des Schlußpreises. Fehlercode = %d",
      return;
     }
//--- Kopieren Sie die Zeit für jede Bar
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,time_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Zeitwerte. Fehlercode = %d",GetLastError()
      return;
     }
//--- erhalten Sie die Größe des Buffers
   size=ArraySize(close_buff);
//--- öffnen Sie die Datei zum Schreiben die Werte (wenn nicht, dann sie automatisch e
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- zeichnen Sie die Zeit und die Werte der Schlußpreis in die Datei auf
      for(int i=0;i<size;i++)
        {
         FileWriteDouble(file_handle,(double)time_buff[i]);
         FileWriteFloat(file_handle,(float)close_buff[i]);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Sehen Sie auch

Realtypen (double, float), FileWriteDouble

© 2000-2025, MetaQuotes Ltd.

FileWriteInteger

Schreibt in eine bin-Datei den Wert des Parameters des Typs int von der angegebenen Position des Dateianzeigers.

```
uint  FileWriteInteger(
   int  file_handle,         // Datei-Handle
   int  value,               // geschriebener Wert
   int  size=INT_VALUE       // Größe in Bytes
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

value

[in] Gazzahliger Wert.

size=INT_VALUE

[in] Anzahl der Bytes (bis 4 einschliesslich), die geschrieben werden müssen. Entsprechende Konstanten werden vorausgesehen: CHAR_VALUE=1, SHORT_VALUE=2 und INT_VALUE=4, so kann die Funktion den ganzzahligen Wert des Typs char, uchar, short, ushort, int oder uint schreiben.

Rückgabewert

Im Erfolgsfall gibt die Funktion die Anzahl der geschriebenen Bytes zurück. Dateianzeiger wird um dieselbe Anzahl der Bytes verschoben.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                        Demo_FileWriteInteger.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string             InpSymbolName="EURUSD";           // Währungspaar
input ENUM_TIMEFRAMES    InpSymbolPeriod=PERIOD_H1;        // Zeitspanne
input datetime           InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum zum Kop
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string             InpFileName="Trend.bin"; // der Dateiname
input string             InpDirectoryName="Data"; // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date_finish=TimeCurrent();
   double   close_buff[];
   datetime time_buff[];
   int      size;
//--- stürzen Sie den Fehlerwert
   ResetLastError();
//--- Kopieren Sie der Schlußpreis für jede Bar
   if(CopyClose(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,close_buff)==-1
     {
      PrintFormat("Fehler beim Kopieren von Werte des Schlußpreises. Fehlercode = %d",
      return;
     }
//--- Kopieren Sie die Zeit für jede Bar
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,time_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Zeitwerte. Fehlercode = %d",GetLastError()
      return;
     }
//--- erhalten Sie die Größe des Buffers
   size=ArraySize(close_buff);
//--- öffnen Sie die Datei zum Schreiben die Werte (wenn nicht, dann sie automatisch e
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //---
      int   up_down=0; // Die Fahne der Tendenz
      int   arr_size;  // die Größe des Arrays arr
      uchar arr[];     // das Array vom Typ uchar
      //--- schreiben Sie die Zeitwerte in die Datei
      for(int i=0;i<size-1;i++)
        {
         //--- Vergleichen Sie die Schlußpreis an den aktuellen und folgenden Bars
         if(close_buff[i]<=close_buff[i+1])
           {
            if(up_down!=1)
              {
```

© 2000-2025, MetaQuotes Ltd.

```
               //--- schreiben Sie den Datumswert in einer Datei mit FileWriteInteger
               StringToCharArray(TimeToString(time_buff[i]),arr);
               arr_size=ArraySize(arr);
               //--- schreiben Sie zuerst die Anzahl der Symbolen im Array
               FileWriteInteger(file_handle,arr_size,INT_VALUE);
               //--- schreiben Sie die Symbole selbst
               for(int j=0;j<arr_size;j++)
                  FileWriteInteger(file_handle,arr[j],CHAR_VALUE);
               //--- ändern Sie die Fahne der Tendenz
               up_down=1;
              }
           }
         else
           {
            if(up_down!=-1)
              {
               //--- schreiben Sie den Datumswert in einer Datei mit FileWriteInteger
               StringToCharArray(TimeToString(time_buff[i]),arr);
               arr_size=ArraySize(arr);
               //--- schreiben Sie zuerst die Anzahl der Symbolen im Array
               FileWriteInteger(file_handle,arr_size,INT_VALUE);
               //--- schreiben Sie die Symbole selbst
               for(int j=0;j<arr_size;j++)
                  FileWriteInteger(file_handle,arr[j],CHAR_VALUE);
               //--- ändern Sie die Fahne der Tendenz
               up_down=-1;
              }
           }
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

IntegerToString, StringToInteger, Ganzzahlige Typen

© 2000-2025, MetaQuotes Ltd.

FileWriteLong

Schreibt den Parameterwert des Typs long in eine bin-Datei von der laufenden Position des Dateianzeigers.

```
uint  FileWriteLong(
   int   file_handle,     // Datei-Handle
   long  value            // der zu geschriebene Wert
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

value

[in] Wert des Typs long.

Rückgabewert

Im Erfolgsfall gibt die Funktion Anzahl der geschriebenen Bytes zurück (in diesem Fall sizeof(long) =8). Dateianzeiger wird um dieselbe Anzahl der Bytes verschoben.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileWriteLong.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string          InpSymbolName="EURUSD";           // Währungspaar
input ENUM_TIMEFRAMES InpSymbolPeriod=PERIOD_H1;        // Zeitspanne
input datetime        InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum zum Kopier
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string          InpFileName="Volume.bin"; // der Dateiname
input string          InpDirectoryName="Data";  // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date_finish=TimeCurrent();
   long     volume_buff[];
   datetime time_buff[];
   int      size;
//--- stürzen Sie den Fehlerwert
   ResetLastError();
//--- kopieren Sie die Tick Volumen für jede Bar
   if(CopyTickVolume(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,volume_buf
     {
      PrintFormat("Fehler beim Kopieren die Werte von Tick-Volumen. Fehlercode = %d",G
      return;
     }
//--- Kopieren Sie die Zeit für jede Bar
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,time_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Zeitwerte. Fehlercode = %d",GetLastError()
      return;
     }
//--- erhalten Sie die Größe des Puffers
   size=ArraySize(volume_buff);
//--- öffnen Sie die Datei zum Schreiben die Indikatorwerte (wenn nicht, dann sie auto
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- schreiben Sie zuerst die Große der Beispieldaten
      FileWriteLong(file_handle,(long)size);
      //--- schreiben Sie die Zeit und die Volumenwerte in die Datei
      for(int i=0;i<size;i++)
        {
         FileWriteLong(file_handle,(long)time_buff[i]);
         FileWriteLong(file_handle,volume_buff[i]);
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

Ganzzahlige Typen, FileWriteInteger

© 2000-2025, MetaQuotes Ltd.

FileWriteString

Zeichnet in die Datei des Typs BIN, CSV oder TXT, die Wert des Parameters des Typs string von der laufenden Position des Dateianzeigers auf. Bei der Aufzeichnung in die Datei des Typs CSV oder TXT, wenn in der Zeile das Symbol '\n ' (LF) ohne vorangehendes Symbol '\r ' (CR) anwesend ist, so wird vor dem Symbol '\n ' das fehlende Symbol '\r ' fertiggeschrieben.

```
uint  FileWriteString(
   int           file_handle,     // Datei-Handle
   const string  text_string,     // die geschriebene Zeile
   int           length=-1        // Anzahl der Symbole
   );
```

Parameter

file_handle

[in] Dateiattribut, das durch die Funktion FileOpen() zurückgegeben wird.

text_string

[in] Zeile.

length=-1

[in] Anzahl der Symbole, die geschrieben werden müssen. Parameter ist für Schreiben der Zeile in die Datei des Typs BIN notwendig. Wenn die Größe nicht angegeben wird, wird die ganze Zeile ohne terminale 0 geschrieben. Wenn die angegebene Größe weniger als Zeilenlänge ist, wird ein Teil der Zeile ohne terminale 0 geschrieben. Wenn die angegebene Größe mehr als die Zeilenlänge ist, wird die Zeile durch entsprechende Anzahl der Nullen ergänzt. Für die Dateien des Typs CSV oder TXT wird dieser Parameter auch die Zeile ignoriert schreibt sich vollständig ein.

Rückgabewert

Im Erfolgsfall gibt die Funktion die Anzahl der geschriebenen Bytes zurück. Dateianzeiger wird um dieselbe Anzahl der Bytes verschoben.

Hinweis

Es ist bemerkenswert, dass beim Schreiben in die Datei, die durch die Flagge Flagge FILE_UNICODE (oder ohne Flagge FILE_ANSI) geöffnet wurde, wird die Anzahl der geschriebenen Bytes um das Doppelte mehr als die Anzahl der geschriebenen Zeilensymbole. Beim Schreiben in die Datei, die durch die Flagge FILE_ANSI eröffnet wurde, wird die Anzahl der geschriebenen Bytes mit der Anzahl der geschriebenen Zeilensymbole zusammenfallen.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                        Demo_FileWriteString.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string             InpSymbolName="EURUSD";           // Währungspaar
input ENUM_TIMEFRAMES    InpSymbolPeriod=PERIOD_H1;        // Zeitspanne
input int                InpMAPeriod=14;                   // die durchschnittliche Pe
input ENUM_APPLIED_PRICE InpAppliedPrice=PRICE_CLOSE;      // der Typ des Preises
input datetime           InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum zum Kop
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string             InpFileName="RSI.csv";    // der Dateiname
input string             InpDirectoryName="Data"; // der Verzeichnisname
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime date_finish; // das Enddatum zum Kopieren von Daten
   double   rsi_buff[];  // ein Array von Werten des Indikators
   datetime date_buff[]; // ein Array von Daten des Indikators
   int      rsi_size=0;  // die Größe der Array von Indikator
//--- die Zeit des Abschlusses - laufend
   date_finish=TimeCurrent();
//--- bekommen Sie die Handle des Indikators RSI
   ResetLastError();
   int rsi_handle=iRSI(InpSymbolName,InpSymbolPeriod,InpMAPeriod,InpAppliedPrice);
   if(rsi_handle==INVALID_HANDLE)
     {
      //--- es misslang, die Handle des Indikators zu bekommen
      PrintFormat("Fehler beim Erhalten der Handle des Indikators. Fehlercode = %d",Ge
      return;
     }
//--- befinden Sie sich im Zyklus, bis der Indikator alle Werten rechnen wird
   while(BarsCalculated(rsi_handle)==-1)
      Sleep(10); // die Verzögerungszeit, damit der Indikator dazugekommen ist die Wer
//--- kopieren Sie die Werte eines Indikators für eine bestimmte Periode
   ResetLastError();
   if(CopyBuffer(rsi_handle,0,InpDateStart,date_finish,rsi_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren der Werte des Indikatos. Fehlercode = %d",GetI
      return;
     }
//--- kopieren Sie die entsprechende Zeit für die Werte des Indikators
   ResetLastError();
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,date_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Zeitwerten. Fehlercode = %d",GetLastError
      return;
     }
//--- befreien Sie das Gedächtnis, das mit dem Indikator eingenommen wird
   IndicatorRelease(rsi_handle);
//--- erhalten Sie die Größe des Puffers
   rsi_size=ArraySize(rsi_buff);
//--- öffnen Sie die Datei zum Schreiben die Indikatorwerte (wenn nicht, dann sie auto
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_DATA_PATH)
      //--- bereiten Sie die Hilfsvariabelen vor
      string str="";
      bool   is_formed=false;
      //--- schreiben Sie das Datum auf die überkauft und überverkauft Zonen
      for(int i=0;i<rsi_size;i++)
        {
         //--- die Prüfung der Werten des Indikators
         if(rsi_buff[i]>=70 || rsi_buff[i]<=30)
           {
            //--- der erste Wert in dieser Zone
            if(!is_formed)
              {
               //--- fügen Sie den Wert und das Datum ein
               str=(string)rsi_buff[i]+"\t"+(string)date_buff[i];
               is_formed=true;
              }
            else
               str+="\t"+(string)rsi_buff[i]+"\t"+(string)date_buff[i];
            //--- der Übergang auf die nächste Iteration des Zyklus
            continue;
           }
         //--- die Prüfung der Fahne
         if(is_formed)
           {
            //---die Zeile ist gebildet, wir werden sie in die Datei aufzeichnen
            FileWriteString(file_handle,str+"\r\n");
            is_formed=false;
           }
        }
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

Typ string, StringFormat

© 2000-2025, MetaQuotes Ltd.

FileWriteStruct

Schreibt in die bin-Datei Inhalt der Struktur, die als Parameter von der laufenden Position des Dateianzeigers übertragen wurde.

```
uint  FileWriteStruct(
   int           file_handle,       // Datei-Handle
   const void&   struct_object,     // Referenz auf das Objekt
   int           size=-1            // die aufzugezeichnete Größe in Bytes
   );
```

Parameter

file_handle

[in] Dateidescriptor, der durch die Funktion FileOpen() zurückgegeben wird.

struct_object

[in] Referenz auf das Objekt der angegebenen Struktur. Die Struktur darf keine Strings, dynamischen Arrays, virtuellen Funktionen, Objekte von Klassen sowie keine Pointer auf Objekte und Funktionen beinhalten.

size=-1

[in] Anzahl der Bytes, die geschrieben werden müssen. Wenn die Größe nicht angegeben wird oder die angegebene Anzahl der Bytes mehr als die Strukturgröße ist, wird die ganze Struktur vollstaendig geschrieben.

Rückgabewert

Im Erfolgsfall gibt die Funktion die Anzahl der geschriebenen Bytes zurück. Dateianzeiger wird um dieselbe Anzahl der Bytes verschoben.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          Demo_FileWiteStruct.mq5 |
//|                        Copyright 2013, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2013, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- die Parameter für das Erhalten der Daten aus dem Terminal
input string          InpSymbolName="EURUSD";           // Währungspaar
input ENUM_TIMEFRAMES InpSymbolPeriod=PERIOD_H1;        // Zeitspanne
input datetime        InpDateStart=D'2013.01.01 00:00'; // das Anfangsdatum zum Kopier
//--- die Parameter für die Aufzeichnung der Daten in die Datei
input string          InpFileName="EURUSD.txt";         // der Dateiname
input string          InpDirectoryName="Data";          // der Verzeichnisname
//+------------------------------------------------------------------+
//| Struktur für die Speicherung von Daten-Kerzen                    |
//+------------------------------------------------------------------+
struct candlesticks
  {
   double            open;  // Eröffnungspreis
   double            close; // Schlußpreis
   double            high;  // der maximale Preis
   double            low;   // der minimale Preis
   datetime          date;  // Datum
  };
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   datetime     date_finish=TimeCurrent();
   int          size;
   datetime     time_buff[];
   double       open_buff[];
   double       close_buff[];
   double       high_buff[];
   double       low_buff[];
   candlesticks cand_buff[];
//--- stürzen Sie den Fehlerwert
   ResetLastError();
//---erhalten Sie die Zeit des Erscheinens der Bars aus dem Umfang
   if(CopyTime(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,time_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren der Zeitwerte. Fehlercode = %d",GetLastError()
      return;
     }
//--- erhalten Sie die maximalen Preise der Bars aus dem Umfang
   if(CopyHigh(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,high_buff)==-1)
     {
      PrintFormat("Fehler beim Lopieren der maximalen Preise. Fehlercode = %d",GetLast
      return;
     }
//--- erhalten Sie die minimalen Preise der Bars aus dem Umfang
   if(CopyLow(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,low_buff)==-1)
     {
      PrintFormat("Fehler beim Lopieren der minimalen Preise. Fehlercode = %d",GetLast
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- erhalten Sie die Eröffnungspreise der Bars aus dem Umfang
   if(CopyOpen(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,open_buff)==-1)
     {
      PrintFormat("Fehler beim Kopieren von Werte des Eröffnungspreises. Fehlercode =
      return;
     }
//--- erhalten Sie die Schlüßpreise der Bars aus dem Umfang
   if(CopyClose(InpSymbolName,InpSymbolPeriod,InpDateStart,date_finish,close_buff)==-1
     {
      PrintFormat("Fehler beim Kopieren von Werte des Schlußpreises. Fehlercode = %d",
      return;
     }
//--- bestimmen Sie die Dimension des Arrays
   size=ArraySize(time_buff);
//--- Speichern Sie alle Daten in der Arraystruktur
   ArrayResize(cand_buff,size);
   for(int i=0;i<size;i++)
     {
      cand_buff[i].open=open_buff[i];
      cand_buff[i].close=close_buff[i];
      cand_buff[i].high=high_buff[i];
      cand_buff[i].low=low_buff[i];
      cand_buff[i].date=time_buff[i];
     }

//--- öffnen Sie die Datei zum Schreiben des Arrays der Struktur in die Datei (wenn ni
   ResetLastError();
   int file_handle=FileOpen(InpDirectoryName+"//"+InpFileName,FILE_READ|FILE_WRITE|FI
   if(file_handle!=INVALID_HANDLE)
     {
      PrintFormat("%s Datei ist zum Lesen geöffnet",InpFileName);
      PrintFormat("Pfad zur Datei: %s\\Files\\",TerminalInfoString(TERMINAL_COMMONDATA
      //--- bereiten Sie den Zähler der Zahl das Byte vor
      uint counter=0;
      //--- zeichnen Sie die Werte des Arrays im Zyklus
      for(int i=0;i<size;i++)
         counter+=FileWriteStruct(file_handle,cand_buff[i]);
      PrintFormat("In die Datei %s ist es %d das Byte der Informationen aufgezeichnet"
      PrintFormat("Die Gesamtanzahl der Bytes: %d * %d * %d = %d, %s",size,5,8,size*5*
      //--- schließen Sie die Datei
      FileClose(file_handle);
      PrintFormat("Die Daten sind aufgezeichnet, die Datei %s geschlossen",InpFileName
     }
   else
      PrintFormat("Fehler beim Öffnen der Datei %s, Fehlercode = %d",InpFileName,GetLa
  }
```

Sehen Sie auch

[Strukturen und Klassen]

© 2000-2025, MetaQuotes Ltd.

FileLoad

Liest alle Daten der angegebenen binären Datei in das Array numerischer Typen oder einfacher Strukturen. Die Funktion ermöglicht es, die Daten von einem bekannten Datentyp in das entsprechende Array zu lesen.

```
long  FileLoad(
   const string  file_name,          // Dateiname
   void&         buffer[],           // Array numerischer Typen oder einfacher Struktur
   int           common_flag=0       // Flag der Datei, standardmäßig wird nach der Dat
   );
```

Parameter

file_name

[in] Name der Datei, aus welcher die Daten gelesen werden.

buffer

[out] Array numerischer Typen oder einfacher Strukturen.

common_flag=0

[in] Flag der Datei, zeigt den Modus an. Wenn der Parameter nicht angegeben ist, dann wird nach der Datei im Unterordner MQL5\Files (oder in <testing_agent_directory>\MQL5\Files beim Testen) gesucht.

Rückgabewert

Anzahl gelesener Elemente oder -1 im Fehlerfall.

Hinweis

Die Funktion FileLoad() liest die Anzahl der Bytes aus der Datei, die der Arraygröße aliquot ist. Zum Beispiel: die Dateigröße beträgt 10 Byte, und sie wird in ein Array vom Typ double (sizeof(double)=8) gelesen. In diesem Fall wird die Funktion nur 8 Byte lesen, 2 Byte vom Ende der Datei werden weggelassen, und die Funktion FileLoad() liefert 1 (1 Element wurde gelesen).

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_FileLoad.mq5 |
//|                        Copyright 2016, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property script_show_inputs
//--- input parameters
input int      bars_to_save=10; // Anzahl der Balken
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string  filename=_Symbol+"_rates.bin";
   MqlRates rates[];
//---
   int copied=CopyRates(_Symbol,_Period,0,bars_to_save,rates);
   if(copied!=-1)
     {
      PrintFormat(" CopyRates(%s) copied %d bars",_Symbol,copied);
      //---  schreiben wir Kurse in eine Datei
      if(!FileSave(filename,rates,FILE_COMMON))
         PrintFormat("FileSave() failed, error=%d",GetLastError());
     }
   else
      PrintFormat("Failed CopyRates(%s), error=",_Symbol,GetLastError());
//--- nun lesen wir diese Kurse aus der Datei
   ArrayFree(rates);
   long count=FileLoad(filename,rates,FILE_COMMON);
   if(count!=-1)
     {
      Print("Time\tOpen\tHigh\tLow\tClose\tTick Voulme\tSpread\tReal Volume");
      for(int i=0;i<count;i++)
        {
         PrintFormat("%s\t%G\t%G\t%G\t%G\t%I64u\t%d\t%I64u",
                     TimeToString(rates[i].time,TIME_DATE|TIME_SECONDS),
                     rates[i].open,rates[i].high,rates[i].low,rates[i].close,
                     rates[i].tick_volume,rates[i].spread,rates[i].real_volume);
        }
     }
  }
```

Sieh auch

Strukturen und Klassen, FileReadArray, FileReadStruct, FileSave

© 2000-2025, MetaQuotes Ltd.

FileSave

Schreibt alle Elemente des als Parameter übermittelten Arrays in eine binäre Datei. Die Funktion ermöglicht es, Arrays numerischer Typen oder einfacher Strukturen als einen String zu schreiben.

```
bool  FileSave(
   const string  file_name,          // Dateiname
   void&         buffer[],           // Array numerischer Typen oder einfacher Struktur
   int           common_flag=0       // Flag der Datei, standardmäßig werden die Datei
   );
```

Parameter

file_name

[in] Name der Datei, in welche das Array geschrieben wird.

buffer

[in] Array numerischer Typen oder einfacher Strukturen.

common_flag=0

[in] Flag der Datei, zeigt den Modus an. Wenn der Parameter nicht angegeben wurde, dann wird die Datei in den Unterordner MQL5\Files (oder in <testing_agent_directory>\MQL5\Files beim Testen) geschrieben.

Rückgabewert

Im Fehlerfall liefert die Funktion false.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_FileSave.mq5 |
//|                        Copyright 2016, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property script_show_inputs
//--- input parameters
input int      ticks_to_save=1000; // Anzahl der Ticks
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string  filename=_Symbol+"_ticks.bin";
   MqlTick ticks[];
//---
   int copied=CopyTicks(_Symbol,ticks,COPY_TICKS_ALL,0,ticks_to_save);
   if(copied!=-1)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat(" CopyTicks(%s) copied %d ticks",_Symbol,copied);
      //--- wenn die Tick-Historie synchronisiert ist, ist der Fehlercode gleich Null
      if(!GetLastError()==0)
         PrintFormat("%s: Ticks are not synchronized, error=%d",_Symbol,copied,_LastEr
      //---  schreiben wir Ticks in eine Datei
      if(!FileSave(filename,ticks,FILE_COMMON))
         PrintFormat("FileSave() failed, error=%d",GetLastError());
     }
   else
      PrintFormat("Failed CopyTicks(%s), Error=",_Symbol,GetLastError());
//--- nun lesen wir diese Ticks aus der Datei
   ArrayFree(ticks);
   long count=FileLoad(filename,ticks,FILE_COMMON);
   if(count!=-1)
     {
      Print("Time\tBid\tAsk\tLast\tVolume\tms\tflags");
      for(int i=0;i<count;i++)
        {
         PrintFormat("%s.%03I64u:\t%G\t%G\t%G\t%I64u\t0x%04x",
         TimeToString(ticks[i].time,TIME_DATE|TIME_SECONDS),ticks[i].time_msc%1000,
         ticks[i].bid,ticks[i].ask,ticks[i].last,ticks[i].volume,ticks[i].flags);
        }
     }
  }
```

Sieh auch

Strukturen und Klassen, FileWriteArray, FileWriteStruct, FileLoad, FileWrite

© 2000-2025, MetaQuotes Ltd.

FolderCreate

Erzeugt ein Verzeichnis im Verzeichnis Files (abhängig vom Wert common_flag)

```
bool  FolderCreate(
   string  folder_name,       // Zeile mit dem Namen des neuen Ordners
   int     common_flag=0      // Geltungsbereich
   );
```

Parameter

folder_name

[in] der Name des Verzeichnisses, das erzeugt werden muss. Beinhaltet den relativen Pfad zum Ordner.

common_flag=0

[in] Flagge, die die Lage des Verzeichnisses bestimmt. Wenn common_flag=FILE_COMMON ist, befindet sich das Verzeichnis im gemeinsamen Ordner für alle Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich das Verzeichnis im lokalen Ordner (MQL5\files oder MQL5\tester\files beim Testen).

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Beispiel:

```
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- Beschreibung
#property description "Das Skript zeigt ein Beispiel für die Anwendung von FolderCreat
#property description "Der externe Parameter legt das Verzeichnis für die Erstellung v
#property description "Nach der Ausführung des Scripts wird eine Ordnerstruktur erstel

//--- das Fenster der Eingabeparameter beim Starten des Scripts anzeigen
#property script_show_inputs
//--- der Eingabeparameter definiert den Ordner, in welchem das Skript ausgeführt wird
input bool     common_folder=false; // gemeinsamer Ordner für alle Terminals
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
 {
//--- Ordner, den wir in MQL5\Files erstellen
   string root_folder="Folder_A";
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CreateFolder(root_folder,common_folder))
    {
      //--- den Unterordner Child_Folder_B1 in diesem Ordner erstellen
      string folder_B1="Child_Folder_B1";
      string path=root_folder+"\\"+folder_B1;          // den Namen des Ordners unter
      if(CreateFolder(path,common_folder))
       {
         //--- drei weitere Unterordner in diesem Ordner erstellen
         string folder_C11="Child_Folder_C11";
         string child_path=root_folder+"\\"+folder_C11;// den Namen des Ordners unter
         CreateFolder(child_path,common_folder);
         //--- zweiter Unterordner
         string folder_C12="Child_Folder_C12";
         child_path=root_folder+"\\"+folder_C12;
         CreateFolder(child_path,common_folder);

         //--- dritter Unterordner
         string folder_C13="Child_Folder_C13";
         child_path=root_folder+"\\"+folder_C13;
         CreateFolder(child_path,common_folder);
        }
     }
+//---+
  }
//+------------------------------------------------------------------+
//| Versucht einen Ordner zu erstellen und gibt eine Meldung darüber |
//+------------------------------------------------------------------+
bool CreateFolder(string folder_path,bool common_flag)
 {
   int flag=common_flag?FILE_COMMON:0;
   string working_folder;
//--- den vollständigen Pfad je nach dem Parameter common_flag feststellen
   if(common_flag)
      working_folder=TerminalInfoString(TERMINAL_COMMONDATA_PATH)+"\\MQL5\\Files";
   else
      working_folder=TerminalInfoString(TERMINAL_DATA_PATH)+"\\MQL5\\Files";
//--- Debugging-Meldung
   PrintFormat("folder_path=%s",folder_path);
//--- Versuch, einen Ordner hinsichtlich des MQL5\Files Pfads zu erstellen
   if(FolderCreate(folder_path,flag))
    {
      //--- den vollständigen Pfad zum erstellen Ordner ausgeben
      PrintFormat("Der Ordner %s wurde erstellt",working_folder+"\\"+folder_path);
      //--- Fehlercode zurücksetzen
      ResetLastError();
      //--- erfolgreiche Ausführung
      return true;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   else
      PrintFormat("Die Erstellung des Ordners %s fehlgeschlagen. Fehlercode %d",workir
//--- Ausführung fehlgeschlagen
   return false;
  }
```

Sehen Sie auch

FileOpen(), FolderClean(), FileCopy()

© 2000-2025, MetaQuotes Ltd.

FolderDelete

Entfernt das angegeben Verzeichnis. Wenn der Ordner leer ist, kann er nicht entfernt werden.

```
bool  FolderDelete(
   string  folder_name,       // Zeile mit dem Namen des entfernten Ordners
   int     common_flag=0      // Geltungsbereich
   );
```

Parameter

folder_name

[in] Name des Verzeichnisses, das entfernt werden muss. Hat den ganzen Pfad zum Ordner.

common_flag=0

[in] Flagge, die die Lage des Verzeichnisses bestimmt. Wenn common_flag=FILE_COMMON, befindet sich das Verzeichnis im Gesamtordner aller Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich das Verzeichnis im lokalen Ordner (MQL5\files oder MQL5\tester\files beim Testen).

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Wenn es im Verzeichnis wenigstens eine Datei und / oder ein Unterverzeichnis gibt, ist die Entfernung solches Verzeichnisses unmöglich, es muss vorerst abgeklärt werden. durch die Funktion FolderClean() kann der Ordner von allen Dateien und verschachtelten Unterordnern abgeklärt werden.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                              Demo_FolderDelete.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                    https://team.metaquotes.ru/email/view/599588 |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://team.metaquotes.ru/email/view/599588"
#property version   "1.00"
//--- Beschreibung
#property description "Das Skript zeigt ein Beispiel der Funktion FolderDelete()."
#property description "Erstellen Sie zunächst zwei Ordner, einer ist leer und der ande
#property description "Beim Versuch einen nicht-leeren Ordner zu löschen, erhalten wi

//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameter
input string   firstFolder="empty";     // Leerer Ordner
input string   secondFolder="nonempty";// Ein Ordner mit einer Datei
string filename="delete_me.txt";       // der Dateiname, die Sie im Ordner secondFolde
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Schreiben Sie die Datei-Handle hier
   int handle;
//--- Finden Sie in welchem Ordner Sie arbeiten
   string working_folder=TerminalInfoString(TERMINAL_DATA_PATH)+"\\MQL5\\Files";
//--- Debug-Meldung
   PrintFormat("working_folder=%s",working_folder);
//--- Versuchen Sie einen leeren Ordner auf Pfad MQL5\Files zu erstellen
   if(FolderCreate(firstFolder,0)) // 0 bedeutet, dass Sie in lokalem Ordner des Termi
     {
      //--- Zeigen Sie den vollen Pfad zum erstellten Ordner
      PrintFormat("Ordner %s wurde erstellt",working_folder+"\\"+firstFolder);
      //--- Fehlercode rücksetzen
      ResetLastError();
     }
   else
      PrintFormat("Ordner %s konnte nicht erstellt werden. Fehlercode %d",working_fold

//--- Erstellen Sie nun einen nicht-leeren mit der FileOpen()-Funktion
   string filepath=secondFolder+"\\"+filename;  // Formen Sie den Pfad zur Datei, die
   handle=FileOpen(filepath,FILE_WRITE|FILE_TXT); // Flag FILE_WRITE ist in diesem Fal
   if(handle!=INVALID_HANDLE)
      PrintFormat("Datei %s wurde für lesen eröffnet",working_folder+"\\"+filepath);
   else
      PrintFormat("Date %s konnte nicht im Ordner %s erstellt werden. Fehlercode=",fil

   Comment(StringFormat("Vorbereiten auf Löschung der Dateien %s und %s", firstFolder,
//--- Eine kurze Pause von 5 Sekunden, so dass Sie die Meldung auf Chart lesen können
   Sleep(5000); // Sleep() kann nicht in den Indikators verwendet werden!

//--- Im Dialogfenster fragen Sie den Benutzer
   int choice=MessageBox(StringFormat("Möchten Sie Ordner %s und %s löschen?", firstFo
                         "Löschung der Ordner",
                         MB_YESNO|MB_ICONQUESTION); //  Zwei Schaltflächen - "Yes" und

//--- Aktionen je nach der gewählten Option
   if(choice==IDYES)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Den Kommentar aus dem Chart löschen
      Comment("");
      //---  Eine Meldung in "Experts" Journal hinzufügen
      PrintFormat("Versuch die Ordner %s und %s zu löschen",firstFolder, secondFolder)
      ResetLastError();
      //--- Löschen den leeren Ordner
      if(FolderDelete(firstFolder))
         //--- Diese Meldung soll erscheinen, weil der Ordner leer ist
         PrintFormat("Ordner %s wurde erfolgreich gelöscht",firstFolder);
      else
         PrintFormat("Ordner %s konnte nicht gelöscht werden. Fehlercode=%d", firstFol

      ResetLastError();
      //--- Löschen Sie den Ordner mit einer Datei
      if(FolderDelete(secondFolder))
         PrintFormat("Ordner %s wurde erfolgreich gelöscht", secondFolder);
      else
         //--- Diese Meldung soll erscheinen, weil der Ordner hat eine Datei
         PrintFormat("Ordner %s konnte nicht gelöscht werden. Fehlercode=%d", secondFo
     }
   else
      Print("Löschung wurde abgebrochen");
//---
  }
```

Sehen Sie auch

FileOpen(), FolderClean(), FileMove()

© 2000-2025, MetaQuotes Ltd.

FolderClean

Entfernt alle Dateien im angegebenen Ordner.

```
bool  FolderClean(
   string  folder_name,       // Zeile mit dem Ordnernamen
   int     common_flag=0      // Geltungsbereich
   );
```

Parameter

folder_name

[in] Name des Verzeichnisses, in dem alle Dateien entfernt werden müssen. Enthält den gesamten Pfad zum Ordner.

common_flag=0

[in] Flagge, die die Lage des Verzeichnisses bestimmt. Wenn common_flag=FILE_COMMON ist, befindet sich das Verzeichnis im Gesamtfolder aller Client-Terminals \Terminal\Common\Files. Anderenfalls befindet sich das Verzeichnis im lokalen Ordner (MQL5\files oder MQL5\tester\files beim Testen).

Rückgabewert

Gibt true im Erfolgsfall zurück, anderenfalls false.

Hinweis

Aus Sicherheitsgründen ist die Arbeit mit Dateien in MQL5 streng gesteuert. Dateien, mit denen Datei-Operationen mittels MQL5 durchgeführt werden, können nicht außerhalb der Datei-Sandbox sein.

Verwenden Sie diese Funktion vorsichtig, denn alle Dateien und verschachtelte Unterverzeichnisse werden unwiederbringlich entfernt.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                Demo_FolderClean.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                     https://team.metaquotes.ru/email/view/599588 |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://team.metaquotes.ru/email/view/599588"
#property version   "1.00"
//--- Beschreibung
#property description "Das Skript zeigt ein Beispiel der Funktion FolderClean()."
#property description "Zuerst erstellen Sie Dateien in einem bestimmten Ordner mit Fil
#property description "Dann, bevor Löschung der Dateien, erscheint die Warnung Message

//--- Beim Starten des Scripts zeigen Sie den Fenster mit den Eingabeparametern an
#property script_show_inputs
//--- Eingabeparameter
input string   foldername="demo_folder";  // Erstellen wir einen Ordner in MQL5/Files/
input int      files=5;                   // Wie viele Dateien erstellen und löschen S
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   string name="testfile";
//--- Zuerst erstellen oder öffnen Sie Dateien im Datenordner des Terminals
   for(int N=0;N<files;N++)
     {
      //--- Dateiname in der Form von 'demo_folder\testfileN.txt'
      string filemane=StringFormat("%s\\%s%d.txt",foldername,name,N);
      //--- Öffnen Sie eine Date mot dem Flag zu schreiben, in diesem Fall wird Ordner
      int handle=FileOpen(filemane,FILE_WRITE);
      //--- Finden Sie, ob FileOpen() erfolgreich war
      if(handle==INVALID_HANDLE)
        {
         PrintFormat("Datei %s konnte nicht erstellt werden. Fehlercode ",filemane,Get
         ResetLastError();
        }
      else
        {
         PrintFormat("Datei %s wurde erfolgreich geöffnet",filemane);
         //--- Die offene Datei brauchen Sie nicht mehr, so schließen Sie sie
         FileClose(handle);
        }
     }

//--- Überprüfen Sie, wie viele Dateien der Ordner hat
   int k=FilesInFolder(foldername+"\\*.*",0);
   PrintFormat("%d Dateien im Ordner %s gefunden",foldername,k);

//--- Im Dialogfenster fragen Sie den Benutzer
   int choice=MessageBox(StringFormat("Sie sind dabei, %d Dateien aus dem Ordner %s zu
                         "Löschung der Dateien aus dem Ordner",
                         MB_YESNO|MB_ICONQUESTION); //  Zwei Schaltflächen - "Yes" und
   ResetLastError();

//--- Aktionen je nach der gewählten Option
   if(choice==IDYES)
     {
      //--- Sie beginnen zu löschen
      PrintFormat("Der Versuch, alle Dateien aus dem Ordner %s zu löschen",foldername)
      if(FolderClean(foldername,0))
```

© 2000-2025, MetaQuotes Ltd.

```
         PrintFormat("Die Dateien wurden erfolgreich gelöscht, %d Dateien bleiben im O
                     foldername,
                     FilesInFolder(foldername+"\\*.*",0));
      else
         PrintFormat("Dateien konnten nicht aus dem Ordner %s gelöscht werden. Fehlerc
     }
   else
      PrintFormat("Löschung wurde abgebrochen");
//---
  }
//+------------------------------------------------------------------+
//| Gibt die Anzahl der Dateien im Ordner zurück                     |
//+------------------------------------------------------------------+
int FilesInFolder(string path,int flag)
  {
   int count=0;
   long handle;
   string filename;
//---
   handle=FileFindFirst(path,filename,flag);
//--- Wenn eine Datei gefunden wird, dann suchen Sie nach anderen
   if(handle!=INVALID_HANDLE)
     {
      //--- Dateinamen zeigen
      PrintFormat("File %s wurde gefunden",filename);
      //--- Erhöhen Sie den Zähler der gefundenen Dateien/Ordner
      count++;
      //--- Beginnen Sie suchen nach allen Dateien/Ordnern
      while(FileFindNext(handle,filename))
        {
         PrintFormat("Datei %s wurde gefunden",filename);
         count++;
        }
      //--- Vergessen Sie nicht die Suche Handle zu schließen
      FileFindClose(handle);
     }
   else // Handle konnte nicht erhalten werden
     {
      PrintFormat("Suche nach Dateien im Ordner %s fehlgeschlagen",path);
     }
//--- Ergebnis zurückgeben
   return count;
  }
```

Sehen Sie auch

FileFindFirst, FileFindNext, FileFindClose

© 2000-2025, MetaQuotes Ltd.

Benutzerindikatoren

Gruppe der Funktionen, die für die Erzeugung der Benutzerindikatoren verwendet werden. Diese Funktionen können nicht beim Schreiben von Scripts und Ratgebern verwendet werden.

Funktion	Massnahme
SetIndexBuffer	Verbindet den Indikatorpuffer mit dem eindimensionalen dynamischen Feld Feld des Typs double
IndicatorSetDouble	Gibt den Wert des Indikators vor, der den Typ double hat
IndicatorSetInteger	Gibt Eigenschaftswert des Indikators vor, der den Typ int hat
IndicatorSetString	Gibt Eigenschaftswert des Indikators vor, der den Typ string hat
PlotIndexSetDouble	Gibt Zeilenwert des Indikators vor, der den Typ double hat
PlotIndexSetInteger	Gibt Zeilenwert des Indikators vor,der den Typ int hat
PlotIndexSetString	Gibt Zeilenwert des Indikators vor, der den Typ string hat
PlotIndexGetInteger	Gibt Zeilenwert des Indikators zurück, der den ganzzahligen Typ hat

Eigenschaften der Indikator können mit Hilfe der Compiler-Direktiven oder mit Funktionen angegeben werden. Um dies besser zu verstehen, ist es empfehlenswert, dass Sie Stile der Indikator in den Beispielen studieren.

Alle notwendigen Berechnungen der Benutzerindikatoren muessen in der vorbestimmten Funktion OnCalculate() untergeordnet werden .Wenn die kurze Aufrufform der Funktion OnCalculate() der Art

```
int OnCalculate (const int rates_total, const int prev_calculated, const int begin, co
```

verwendet wird, enthält die Variable *rates_total* den Wert der gesamten Anzahl der Elemente des Feldes price[], das als Eingabe-Parameter für Berechnung der Anzeigerwerte übertragen wurde.

Parameter *prev_calculated* - ist das Ergebnis der Durchführung der Funktion OnCalculate() auf dem vorangehenden Level und ermöglicht sparsamen Algorythmus für Berechnung der Indikatorwerte zu gestalten. ZB wenn der laufende Wert rates_total=1000, und prev_calculated=999, dann ist es vielleicht genug, Berechnungen nur für einen Wert des Anzeigerpuffers zu machen.

Wenn die Information über die Größe des Eingabefeldes price unzugaenglich waere, wuerde es zur Notwendigkeit führen, Berechnungen für 1000 Werte jedes Indikatorpuffers zu machen. Beim ersten Aufruf der Funktion OnCalculate() ist der Wert prev_calculated=0. Wenn sich das Feld price[] irgendwie verändert hat, ist prev_calculated in diesem Fall auch 0.

Parameter *begin* zeigt die Anzahl der Initialwerte des Feldes price, die keine Daten für die Berechnung enthalten. ZB wenn als Eingabefeld die Werte des Indikators Accelerator Oscillator verwendet wurden (für den die ersten 37 Werte nicht berechnet werden), ist begin=37. Als Beispiel betrachten wir einen einfachen Anzeiger:

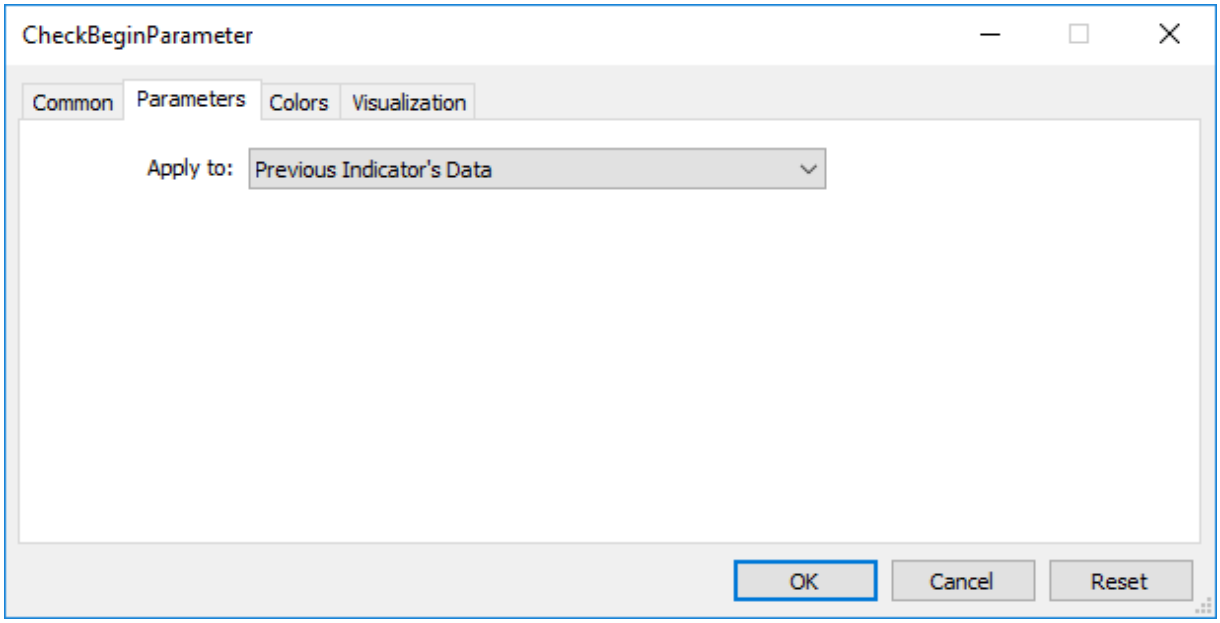
```
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Label1
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_label1  "Label1"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- indicator buffers
double         Label1Buffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,Label1Buffer,INDICATOR_DATA);
//---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const int begin,
                const double &price[])

  {
//---
   Print("begin = ",begin,"  prev_calculated = ",prev_calculated,"  rates_total = ",ra
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Ziehen wir das aus dem Fenster "Navigator" ins Fenster des Indikators Accelerator Oscillator und geben wir an, dass Berechnungen auf der Grund der Werte des früheren Indikators durchgeführt werden:

© 2000-2025, MetaQuotes Ltd.



Im Ergebnis wird der Wert prev_calculated beim ersten Aufruf der Funktion OnCalculate() 0 sein, bei weiteren Aufrufen wird der Wert dem Werten rates_total gleich sein (bis die Anzahl der Bars am Preischart größer wird).



Wert des Parameters begin wird der Anzahl der Initialbars gleich, für die Werte des Indikators Accelerator nicht berechnet werden auf Grund der Logik dieses Indikators. Wenn wir den Ausgangskode des Benutzerindikators Accelerator.mq5 ansehen, sehen wir in der Funktion OnInit() die folgenden Zeilen:

© 2000-2025, MetaQuotes Ltd.

```
//--- sets first bar from what index will be drawn
   PlotIndexSetInteger(0,PLOT_DRAW_BEGIN,37);
```

Gerade durch die Funktion PlotIndexSetInteger(0, PLOT_DRAW_BEGIN, empty_first_values) stellen wir die Anzahl der unbedeutenden ersten Werte im Nullindikatorfeld, die wir für die Berechnungen (empty_first_values) nicht zu beachten brauchen. So haben wir Mechanismen um:

1. die Anzahl der Initialwerte des Indikators anzugeben, die für Berechnungen im anderen Benutzerindikator nicht zu verwendet sind;
2. die Information über die Anzahl der ersten Werte zu bekommen, die beim Aufruf eines anderen Benutzerundikatots ignoriert werden muessen, ohne in Logik der Berechnungen zu gehen.

© 2000-2025, MetaQuotes Ltd.

Stile der Indikator in den Beispielen

MetaTrader 5 Client Terminal enthielt 38 eingebaute Technische Indikatoren, die in MQL5-Programmen mithilfe der entsprechenden Funktionen verwendet werden können. Aber der Hauptvorteil der MQL5-Sprache ist die Fähigkeit, eigene Indikatoren, die dann in Expert Advisor für Erhaltung der Werte oder für die technische Analyse verwendet werden, zu erstellen.

Das ganze Reihe von Indikatoren kann aus mehreren grundlegenden Zeichenstile, die grafische graphische Konstruktionen genannt sind, abgeleitet werden. Eine Konstruktion ist eine Weise der Darstellung von Daten, die der Indikator berechnet, speichert und zeigt auf Aufruf. Es gibt sieben solcher grundlegenden Konstruktionen:

1. Linie
2. Sektion (Segment)
3. Histogramm
4. Pfeil (Symbol)
5. Die gefärbte Fläche (Kanal zu füllen)
6. Balken
7. Japanische Kerzen

Jede Darstellung benötigt 1-5 Arrays des Typs double, in deren Indikatorwerte gespeichert werden. Für die Bequemlichkeit werden diese Arrays mit den Indikator-Puffern verbunden. Die Anzahl der Puffer im Indikator muss im Voraus durch Compiler-Direktiven deklariert werden, zum Beispiel:

```
#property indicator_buffers 3 // Anzahl der Puffer
#property indicator_plots   2 // Anzahl der graphischen Konstruktionen
```

Die Anzahl der Puffer im Indikator ist immer größer oder gleich der Anzahl von Konstrukten im Indikator.

Da jede grundlegende grafische Konstruktion kann Farbvariationen oder spezielle Darstellung haben, dann die tatsächliche Anzahl der Konstruktionen in der MQL5-Sprache ist 18:

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
DRAW_NONE	Wird nicht auf dem Chart visuell dargestellt, aber man kann die Werte des entsprechenden Puffers in "Data	1	-

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
	Window" finden		
DRAW_LINE	Eine Linie wird auf die Werte des entsprechenden Puffers gezeichnet (leere Werte im Puffer sind unerwünscht)	1	-
DRAW_SECTION	Liniensegmente werden zwischen den Werten des jeweiligen Puffers (in der Regel eine Menge von leeren Werten) gezeichnet	1	-
DRAW_HISTOGRAM	Ein Histogramm wird von der Null-Linie bis den Werte	1	-

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
	des jeweiliges Puffer gezeichnet (leere Werte sind erlaubt)		
DRAW_HISTOGRAM2	Ein Histogramm wird auf zwei Indikator-Puffer gezeichnet (leere Werte sind erlaubt)	2	-
DRAW_ARROW	Wird als Symbols gezeichnet (leere Werte sind erlaubt)	1	-
DRAW_ZIGZAG	Ist dem Stil DRAW_SECTION ähnlich, aber im Gegensatz zu ihm kann er vertikale Segmente auf einem Balken bauen	2	-
DRAW_FILLING	Farbfüllung	2	-

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
	zwischen zwei Linien. In "Data Window" werden zwei Werte der entsprechenden Puffer angezeigt		
DRAW_BARS	Wird als Balken auf dem Chart gezeichnet. In "Data Window" werden vier Werte der entsprechenden Puffer angezeigt	4	-
DRAW_CANDLES	Wird als Kerzen auf dem Chart gezeichnet. In "Data Window" werden vier Werte der entsprechenden Puffer	4	-

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
	angezeigt		
DRAW_COLOR_LINE	Eine Linie, für die Sie Farben auf verschiedenen Bars ändern sowie ihre Farbe jederzeit wechseln können	1	1
DRAW_COLOR_SECTION	Ist dem Stil DRAW_SECTION ähnlich, aber die Farbe jeder Sektion kann individuell eingegeben werden; die Farbe kann dynamisch angegeben werden	1	1
DRAW_COLOR_HISTOGRAM	Ist dem Stil DRAW_	1	1

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
	HISTOGRAM ähnlich, aber jede Band kann ihre eigene Farbe haben; die Farbe kann dynamisch angegeben werden		
DRAW_COLOR_HISTOGRAM2	Ist dem Stil DRAW_HISTOGRAM2 ähnlich, aber jede Band kann ihre eigene Farbe haben; die Farbe kann dynamisch angegeben werden	2	1
DRAW_COLOR_ARROW	Ist dem Stil DRAW_ARROW ähnlich, aber	1	1

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
	jedes Symbol kann seine eigene Farbe haben. Farbe kann dynamisch geändert werden		
DRAW_COLOR_ZIGZAG	Stil DRAW_ZIGZAG mit den Fähigkeiten der individuellen Farbgebung der Sektionen und dynamischen Farbwechsel	2	1
DRAW_COLOR_BARS	Stil DRAW_BARS mit den Fähigkeiten der individuellen Farbgebung der Balken und dynamischen Farbwechsel	4	1

© 2000-2025, MetaQuotes Ltd.

Konstruktion	Beschreibung	Werte-Puffer	Farbpuffer
DRAW_COLOR_CANDLES	Stil DRAW_CANDLES mit den Fähigkeiten der individuellen Farbgebung der Kerzen und dynamischen Farbwechsel	4	1

Der Unterschied zwischen einem Indikator-Puffer und einem Array

In jedem Indikator deklarieren wir ein oder mehr Arrays vom Typ double auf die globale Ebene, die dann als Indikator-Puffer mit der SetIndexBuffer()-Funktion verwendet werden. Für Zeichnung der graphischen Konstruktionen eines Indikators sind nur die Werte der Indikator-Puffer verwendeten, alle anderen Arrays können nicht dafür genutzt werden. Darüber hinaus sind die Werte der Puffer in "Data Window" angezeigt.

Der Indicator-Puffer sollte dynamisch sein und erfordert keine Angabe der Größe - die Größe des Arrays, das als Indikator-Puffer verwendet wird, wird automatisch durch die Laufzeit-Subsystem des Terminals angegeben.

Indizierung-Richtung nach der Bindung des Arrays mit dem Indikator-Puffer ist standardmäßig wie in einem gewöhnlichen Array aufgestellt, aber Sie können die Funktion ArraySetAsSeries() verwenden, um die Methode für den Zugriff zu Elemente eines Arrays zu ändern. Standardmäßig wird der Indikator-Puffer verwendet, um Daten für die Darstellung zu speichern (INDICATOR_DATA).

Wenn Berechnung der Indikator-Werte Zwischen-Berechnungen und Speicherung von untergeordneter Wert für jedem Balken erfordert, so können Sie dieses Array während Bindung als Berechnung-Puffer deklarieren (INDICATOR_CALCULATIONS). Für die Zwischen-Werte kann ein regelmäßiges Array verwenden werden, aber in diesem Fall muss der Programmierer selbst die Größe des Arrays steuern.

Einige Darstellungen erlauben Farbe für jeden Balken anzugeben. Um Informationen über die Farbe zu speichern, werden Farbpuffer verwendet (INDICATOR_COLOR_INDEX). Die Farbe ist ein Integer-Typ color, aber nicht alle Indikator-Puffer müssen vom Typ double sein. Werte von der Farbpuffer und Hilfspuffer (INDICATOR_CALCULATIONS) können nicht mit der CopyBuffer() function erhalten werden.

Die Anzahl der Indikator Puffer muss durch den Compiler-Direktive #property indicator_buffers Anzahl_der_Puffers angegeben werden:

```
#property indicator_buffers 3  //  Der Indikator hat 3 Puffer
```

© 2000-2025, MetaQuotes Ltd.

Die maximale Anzahl der Puffer in einem Indikator ist 512.

Relevanz von Indikator-Puffer und grafische Konstruktionen

Jede grafische Konstruktion basiert auf einem oder mehreren Indikator-Puffer. Zum Beispiel, Kerze brauchen vier Werte - die Preise von Open, High, Low und Close. Dementsprechend müssen Sie für den Indikator in Form von Kerzen 4 Indikator-Puffer und 4 Arrays vom Typ double für sie deklarieren. Zum Beispiel:

```
//--- Ein Indikator hat vier Puffer
#property indicator_buffers 4
//--- Der Indikator hat eine graphische Konstruktion
#property indicator_plots   1
//--- Graphische Konstruktion Nummer 1 wird als Kerzen gezeichnet
#property indicator_type1   DRAW_CANDLES
//--- Die Farbe der Kerzen ist clrDodgerBlue
#property indicator_color1  clrDodgerBlue
//--- 4 Arrays für die Puffer
double OBuffer[];
double HBuffer[];
double LBuffer[];
double CBuffer[];
```

Grafische Darstellungen verwenden automatisch die Indikator-Puffer in Übereinstimmung mit der Darstellung-Nummer. Nummer der Darstellungen beginnen von eins, Nummer der Puffer beginnen von Null. Wenn die erste Darstellung 4 Indikator-Puffer benötigt, dann werden erste 4 Indikator-Puffer für Zeichnung verwendet. Diese vier Puffer sollten durch die Funktion SetIndexBuffer() mit den entsprechenden Arrays mit der richtigen Indizierung gebunden werden.

```
//--- Bindung von Arrays und Indikator-Puffer
   SetIndexBuffer(0,OBuffer,INDICATOR_DATA);  // Der erste Puffer entspricht den Index
   SetIndexBuffer(1,HBuffer,INDICATOR_DATA);  // Der zweite Puffer entspricht den Inde
   SetIndexBuffer(2,LBuffer,INDICATOR_DATA);  // Der dritte Puffer entspricht den Inde
   SetIndexBuffer(3,CBuffer,INDICATOR_DATA);  // Der vierte Puffer entspricht den Inde
```

Beim Zeichnung von Kerzen, verwendet der Indikator die erste vier Puffer, da sie Darstellung "Kerze" von der ersten Reihe deklariert wurde.

Wir ändern das Beispiel und fügen eine Konstruktion von einer einfacher Linie hinzu - DRAW_LINE. Nehmen wir nun an, dass die Linie Nummer 1 hat und die Kerzen Nummer 2 haben. Die Anzahl der Puffer und die Anzahl der Konstruktionen haben erhöht.

```
//--- Der Indikator hat fünf Indikator-Puffer
#property indicator_buffers 5
//--- Der Indikator hat zwei graphischen Konstruktionen
#property indicator_plots   2
//--- Graphische Konstruktion Nummer 1 wird als eine Linie gezeichnet
#property indicator_type1   DRAW_LINE
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Die Farbe der Linie ist clrDodgerRed
#property indicator_color1  clrDodgerRed
//--- Graphische Konstruktion Nummer 2 wird als Kerzen gezeichnet
#property indicator_type2   DRAW_CANDLES
//--- Die Farbe der Kerzen ist clrDodgerBlue
#property indicator_color2  clrDodgerBlue
//--- 5 Arrays für die Puffer
double LineBuffer[];
double OBuffer[];
double HBuffer[];
double LBuffer[];
double CBuffer[];
```

Die Reihenfolge der Konstruktionen hat verändert, jetzt kommt eine Linie und dann Kerzen. Daher wird die Reihenfolge der Puffer die gleiche sein - zum ersten Mal deklarieren wir den Linie-Puffer für Index Null, und dann vier Puffer für Kerzen.

```
   SetIndexBuffer(0,LineBuffer,INDICATOR_DATA);  // Der erste Puffer entspricht den Ir
//--- Bindung von Arrays und Indikator-Puffer für Kerzen
   SetIndexBuffer(1,OBuffer,INDICATOR_DATA);     // Der zweite Puffer entspricht den 1
   SetIndexBuffer(2,HBuffer,INDICATOR_DATA);     // Der dritte Puffer entspricht den 1
   SetIndexBuffer(3,LBuffer,INDICATOR_DATA);     // Der vierte Puffer entspricht den 1
   SetIndexBuffer(4,CBuffer,INDICATOR_DATA);     // Der fünfte Puffer entspricht den 1
```

Die Anzahl der Puffer und grafischen Konstruktionen können nur mit Compiler-Direktiven angegeben werden, die dynamischen Veränderungen dieser Eigenschaften mit Funktionen ist unmöglich.

Farbversionen von Stilen

Stile sind in zwei Gruppen eingeteilt. Die erste Gruppe enthält die Stile, in denen es kein Wort **COLOR** im Titel gibt, wir nennen diese Stile grundlegende Stile:

- DRAW_LINE
- DRAW_SECTION
- DRAW_HISTOGRAM
- DRAW_HISTOGRAM2
- DRAW_ARROW
- DRAW_ZIGZAG
- DRAW_FILLING
- DRAW_BARS
- DRAW_CANDLES

Die zweite Gruppe enthält die Stile mit dem Wort **COLOR** in Titel, wer nennen sie Farbversionen:

- DRAW_COLOR_LINE
- DRAW_COLOR_SECTION
- DRAW_COLOR_HISTOGRAM

© 2000-2025, MetaQuotes Ltd.

- DRAW_COLOR_HISTOGRAM2
- DRAW_COLOR_ARROW
- DRAW_COLOR_ZIGZAG
- DRAW_COLOR_BARS
- DRAW_COLOR_CANDLES

Alle Farbversionen der Stile unterscheiden sich von den grundlegenden Stile in dass sie eine Farbe für jeden Teil der graphischen Konstruktion anzugeben erlauben. Minimaler Teil der Konstruktion ist ein Balken, so kann man sagen, dass die Farbversionen die Farbe der Konstruktion auf jedem Balken einzugeben erlauben.

Um die Farbe der Konstruktion auf jeden Balken anzugeben, erhalten die Farbversionen der Stile ein zusätzlicher spezieller Puffer, um die Farbindexes zu speichern. Diese Indizes zeigen die Nummer der Farbe in einem speziellen Array mit einem vordefinierten Set von Farben. Die Größe des Arrays von Farben ist 64. Dies bedeutet, dass jede Farbversion erlaubt Färbung mit 54 verschiedenen Farben.

Das Set und die Anzahl der Farben im speziellen Array kann dynamisch durch Compiler-Direktive #property indicator_color angegeben werden (durch Kommata werden alle notwendigen Farben angegeben). Zum Beispiel, der Eintrag im Indikator:

```
//--- Wir definieren 8 Farben zum Einfärben von Kerzen (sie sind in dem speziellen Ar
#property indicator_color1  clrRed,clrBlue,clrGreen,clrYellow,clrMagenta,clrCyan,clrL
```

Hier ist angegeben, dass für die grafische Konstruktion Nummer acht Farben, die in einem speziellen Array gestellt werden, definiert sind. Als nächstes werden wir nicht die Farbe angeben, sondern nur ihr Index. Wenn wir rot für den Balken angeben sollen, dann müssen wir den Index der roten Farbe aus einem Array in Farbpuffer angeben. Die rote Farbe hat Nummer 1 in Direktive, Indexnummer 0 entspricht sie.

```
  //--- Definieren wir die Farbe der Kerze clrRed
  col_buffer[buffer_index]=0;
```

Ein Set von Farben für die Färbung kann dynamisch durch PlotIndexSetInteger()-Funktion verändert sein. Beispiel:

```
      //--- Wir geben die Farbe für jeden Index als die Eigenschaft PLOT_LINE_COLOR
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Eigenschaftbezeichnung
                          plot_color_ind,       //  Index der Farbe, in dem wir die Fa
                          color_array[i]);      //  Neue Farbe
```

Eigenschaften des Indikators und der grafischen Konstruktionen

Eigenschaften für graphischen Konstruktionen können durch Compiler-Direktiven und mit den entsprechenden Funktionen angegeben werden. Lesen Sie mehr darüber in der Sektion Zusammenhang zwischen Eigenschaften des Indikators und entsprechenden Funktionen. Dynamische Veränderung der Indikator-Eigenschaften durch Funktionen erlaubt es flexibler benutzerdefinierte Indikatoren zu erstellen.

© 2000-2025, MetaQuotes Ltd.

Start der Zeichnung des Indikators im Chart

In vielen Fällen, unter den Bedingungen des Algorithmus, kann die Berechnung der Indikatorwerte nicht sofort mit dem aktuellen Balken gestartet werden, es ist erforderlich, die minimale Anzahl von vorgehenden Balken in der Geschichte zu versorgen. Zum Beispiel bedeuten viele Arten von Glättung, dass ein Array von Preisen für das vorgehenden N Balken erforderlich ist, und auf der Grundlage dieser Werte wird der Indikatorwert für den aktuellen Balken berechnet.

In solchen Fällen entweder gibt es keine Möglichkeit, die Indikatorwerte für die ersten N Balken zu berechnen, oder diese Werte sollen nicht auf dem Chart angezeigt werden und sind nur subsidiär für die Berechnung der folgenden Werte. Um nichts auf die erste N Balken in der Geschichte zu zeichnen, geben Sie den N Wert der Eigenschaft PLOT_DRAW_BEGIN für die entsprechende grafische Konstruktion ein:

```
//--- Bindung von Arrays und Indikator-Puffer für Kerzen
PlotIndexSetInteger(Nummer_der_graphischen_Konstruktion,PLOT_DRAW_BEGIN,N);
```

Hier:

- Nummer_der_graphischen_Konstruktion - Wert von 0 bis indicator_plots-1 (Nummerierung der Konstruktionen beginnt mit Null).
- N - Anzahl der ersten Balken in der Geschichte, auf denen der Indikator nicht auf dem Chart angezeigt werden soll.

© 2000-2025, MetaQuotes Ltd.

DRAW_NONE

Stil DRAW_NONE ist für diejenigen Fälle, in denen der Puffer-Wert berechnet und in "Data Window" dargestellt werden sollte, aber die Anzeige auf dem Chart ist nicht erforderlich. Um die Genauigkeit der Anzeige zu konfigurieren, verwenden Sie Ausdruck IndicatorSetInteger(INDICATOR_DIGITS,Anzahl_der_Zeichen) in der OnInit()-Funktion:

```
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,InvisibleBuffer,INDICATOR_DATA);
//--- Wir stellen die Genauigkeit, mit der der Wert im Daten-Fenster angezeigt werden
   IndicatorSetInteger(INDICATOR_DIGITS,0);
//---
   return(INIT_SUCCEEDED);
  }
```

Die erforderliche Anzahl der Puffer für den Bau von DRAW_NONE ist 1.

Ein Beispiel für einen Indikator, der in "Data Window" die Nummer des Balkens, auf dem die Maus ist, zeigt. Die Nummerierung entspricht der Zeit-Serie, d.h. der aktuelle unvollendete Balken hat Null-Index, und der älteste Balken verfügt über den größten Index.



Beachten Sie, dass trotz der Tatsache, dass für die grafische Konstruktion **Nummer 1** rote Farbe der Darstellung angegeben ist, zeichnet der Indikator nichts auf dem Chart.

```
//+------------------------------------------------------------------+
//|                                                   DRAW_NONE.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot Invisible
#property indicator_label1  "Bar Index"
#property indicator_type1   DRAW_NONE
#property indicator_style1  STYLE_SOLID
#property indicator_color1  clrRed
#property indicator_width1  1
//--- indicator buffers
double         InvisibleBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung vom Array und Indikator-Puffer
   SetIndexBuffer(0,InvisibleBuffer,INDICATOR_DATA);
//--- Wir stellen die Genauigket, mit der der Wert in Data Window angezeigt werden
   IndicatorSetInteger(INDICATOR_DIGITS,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static datetime lastbar=0;
//--- Wenn dies die erste Berechnung des Indikators ist
   if(prev_calculated==0)
     {
      //--- Durchnumerieren Balken zum ersten Mal
      CalcValues(rates_total,close);
      //--- Speichern Öffnungszeit des aktuellen Balkens in lastbar
      lastbar=(datetime)SeriesInfoInteger(_Symbol,_Period,SERIES_LASTBAR_DATE);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
    else
      {
       //--- Wenn ein neuer Balken erscheint hat, sien Öffnungszeit fällt nicht mit las
       if(lastbar!=SeriesInfoInteger(_Symbol,_Period,SERIES_LASTBAR_DATE))
         {
          //--- Durchnummerieren Balken noch einmal
          CalcValues(rates_total,close);
          //--- Aktuallisieren Öffnungszeit des aktuellen Balkens in lastbar
          lastbar=(datetime)SeriesInfoInteger(_Symbol,_Period,SERIES_LASTBAR_DATE);
         }
      }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Numeriert Balken wie in Zeitreihen                               |
//+------------------------------------------------------------------+
void CalcValues(int total,double const  &array[])
  {
//--- Wir definieren Indexierung des Indikator-Puffers als in Zeitreihen
   ArraySetAsSeries(InvisibleBuffer,true);
//--- Wir füllen Nummer jedem Balken
   for(int i=0;i<total;i++) InvisibleBuffer[i]=i;
  }
```

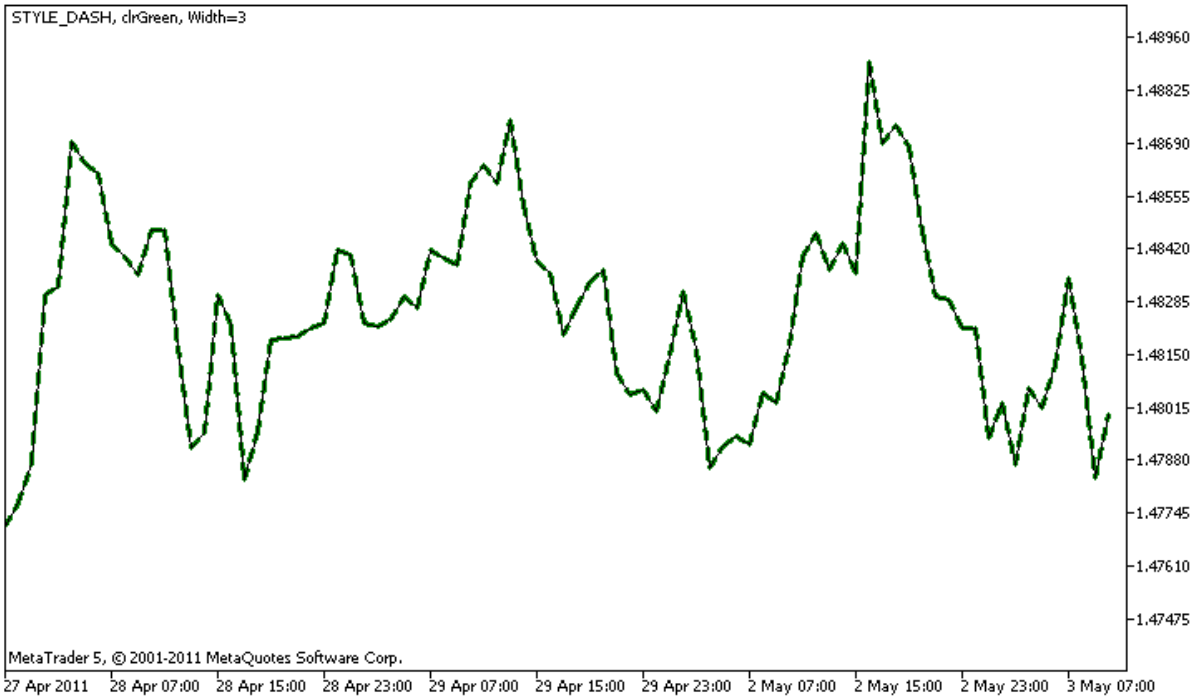
© 2000-2025, MetaQuotes Ltd.

DRAW_LINE

Stil DRAW_LINE zeichnet eine Linie der angegebenen Farbe auf die Werte des Indikator-Puffers. Die Breite, Stil und Farbe der Linie können durch Compiler-Direkriven, oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_LINE ist 1.

Ein Beispiel für einen Indikator, der eine Linie auf die Close Priese der Balken zeichnet. Farbe, Breite und Linienart werden nach dem Zufallsprinzip jede N=5 Ticks geändert.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_LINE-Stil werden Eigenschaften mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden diese drei Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                    DRAW_LINE.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_LINE"
#property description "Zeichnet eine Linie der angegebenen Farbe auf die Close Preise"
#property description "Farbe, Breite und Linienart werden nach dem Zufallsprinzip"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "jede N Ticks verändert"

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- Eigenschaften der Linie werden durch Compiler-Direktiven definiert
#property indicator_label1  "Line"      // Name der Darstellung für Data Window
#property indicator_type1   DRAW_LINE   // Typ der graphischen Darstellung ist die Lin
#property indicator_color1  clrRed      // Linienfarbe
#property indicator_style1  STYLE_SOLID // Linienstil
#property indicator_width1  1           // Linienbreite
//--- Eingabeparameter
input int      N=5;          // Anzahl der Ticks für Änderung
//--- Indicator-Puffer
double         LineBuffer[];
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen};
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung vom Array und Indikator-Puffer
   SetIndexBuffer(0,LineBuffer,INDICATOR_DATA);
//--- Initialisierung des Zufallszahlengenerators
   MathSrand(GetTickCount());
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
```

© 2000-2025, MetaQuotes Ltd.

```
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Block mit Berechnung der Indikatorwerte
   for(int i=0;i<rates_total;i++)
     {
      LineBuffer[i]=close[i];
     }

//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der angezeigten Linie im Indikator           |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienfarbe
//--- Wir erhalten eine Zufallszahl
   int number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+(string)colors[color_index];

//--- Block mit Änderungen der Linienbreite
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5; // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+", Width="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   size=ArraySize(styles);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm=EnumToString(styles[style_index])+", "+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_SECTION

Stil DRAW_SECTION zeichnet Segmente der angegebenen Farbe auf die Werte des Indikator-Puffers. Die Breite, Stil und Farbe der Linie können ebenso wie für den Stil DRAW_LINE - durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

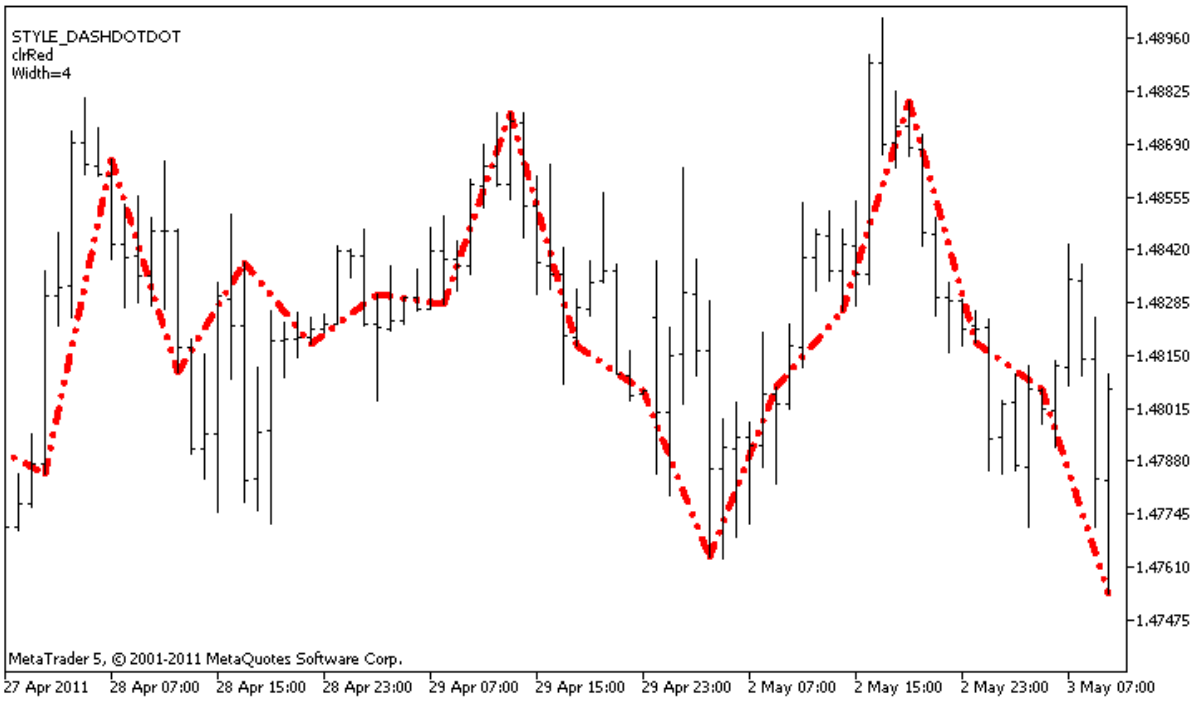
Die Sektionen werden aus einem nicht leeren Wert zu einem anderen leeren Wert des Indikator-Puffers gezeichnet, leere Werte werden ignoriert. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE. Zum Beispiel, wenn der Indikator wie Sektionen auf nicht leere Werte gezeichnet werden soll, müssen Sie einen Nullwert als leer angeben:

```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(Index_der_Zeichnung_DRAW_SECTION,PLOT_EMPTY_VALUE,0);
```

Füllen Sie immer alle Elemente des Indikator-Puffers mit den Werten, geben sie leere Werte für nicht-gezeichnete Elemente.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_SECTION ist 1.

Ein Beispiel für einen Indikator, der Segmente zwischen den Preisen von High und Low zeichnet. Farbe, Breite und Stil **aller** Sektionen werden zufällig jede **N** Ticks geändert.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_SECTION-Stil werden Eigenschaften mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden diese drei Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                  DRAW_SECTION.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_SECTION"
#property description "Zeichnet gerade Linien durch jede bars Balken"
#property description "Farbe, Breite und Stil der Sektionen werden zufällig"
#property description "jede N Ticks verändert"

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot Section
#property indicator_label1  "Section"
#property indicator_type1   DRAW_SECTION
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      bars=5;           // Länge der Sektionen in Balken
input int      N=5;              // Anzahl der Ticks um Stil der Sektionen zu ändern
//--- Indicator-Puffer
double         SectionBuffer[];
//--- Hilfsvariable, um Ende der Sektionen zu berechnen
int            divider;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen};
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung vom Array und Indikator-Puffer
   SetIndexBuffer(0,SectionBuffer,INDICATOR_DATA);
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//--- Überprüfen wir die Indikator-Parameter
   if(bars<=0)
     {
      PrintFormat("Invalid Parameterwert bar=%d",bars);
      return(INIT_PARAMETERS_INCORRECT);
    ;}
   else divider=2*bars;
```

© 2000-2025, MetaQuotes Ltd.

```
//---+
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Anzahl der Balken, mit dem die Berechnung des Indikators beginnt
   int start=0;
//--- Wenn der Indikator bevor berechnet wurde, dann setzen wir start bei dem vorherig
   if(prev_calculated>0) start=prev_calculated-1;
//--- alle Berechnungen der Indikatorwerte
   for(int i=start;i<rates_total;i++)
     {
      //--- Wir bekommen Rest nach der Division der Nummer des Balkens durch 2*bars
      int rest=i%divider;
      //--- Wenn die Nummer des Balken ohne Rest durch 2*bars teilbar ist
      if(rest==0)
        {
         //--- Ende der Segment ist auf den High-Preis dieses Balkens gesetzt
         SectionBuffer[i]=high[i];
        }
      //--- Wenn der Rest der Division gleich bars ist,
      else
        {
         //--- Ende der Segment ist auf den High-Preis dieses Balkens gesetzt
         if(rest==bars) SectionBuffer[i]=low[i];
```

© 2000-2025, MetaQuotes Ltd.

```
 //--- Wenn etwas nicht passt, dann überspringen wir diesen Balken - setzen 0
         else SectionBuffer[i]=0;
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
 ;}
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Sektionen im Indikator                   |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+"\r\n"+(string)colors[color_index];

//--- Block mit Änderungen der Linienbreite
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Breite angeben
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Linienart angeben
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm="\r\n"+EnumToString(styles[style_index])+""+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

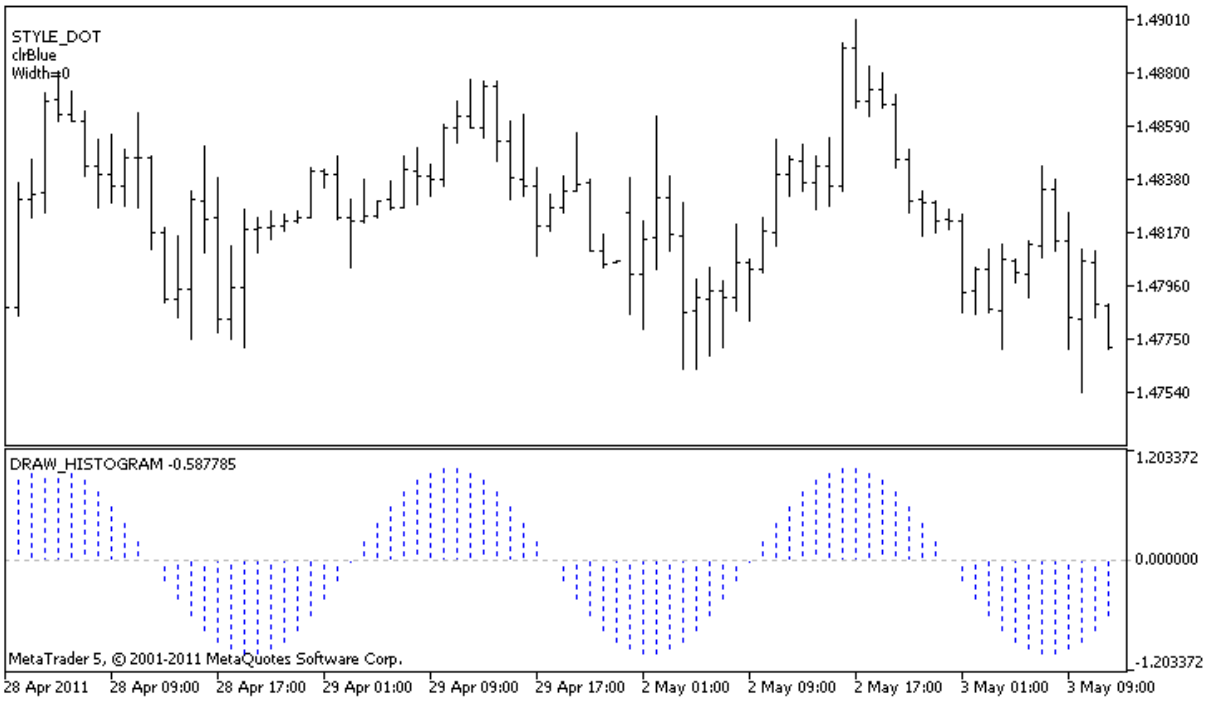
DRAW_HISTOGRAM

DRAW_HISTOGRAM-Stil zeichnet ein Histogramm der angegebenen Farbe von Null bis den angegebenen Wert. Die Werte werden aus den Indikator-Puffer aufgenommen. Die Breite, Farbe und Stil der Säule können wie für DRAW_LINE durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, den Aussehens des Histogramms je nach der aktuellen Situation zu ändern.

Da wird auf jeden Balken eine Säule von Null-Ebene gezeichnet, dann wird DRAW_HISTOGRAM besser für Zeichnung in einem separaten Fenster verwendet. Meistens wird diese Art der graphischen Konstruktion an einen Oszillator Art von Indikatoren, z.B. Bears Power oder OsMA verwendet. Für leere nicht gezeichnete Werte ist es genug Null-Werte anzugeben.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_HISTOGRAM ist 1.

Ein Beispiel für einen Indikator, der eine Sinuskurve der angegebenen Farbe auf Funktion MathSin() zeichnet. Die Farbe, Breite und Stil **aller** Säule des Histograms werden nach dem Zufallsprinzip jede **N** Ticks verändert. Parameter bars gibt den Zeitraum der Sinuskurve, d.h. in einer bestimmten Anzahl von Balken wird die Sinuskurve ihren Zyklus wiederholen.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_HISTOGRAM-Stil werden Eigenschaften mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden diese drei Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                DRAW_HISTOGRAM.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_HISTOGRAM"
#property description "Zeichnet die Sinuskurve als ein Histogramm in einem separaten F
#property description "Farbe und Breite der Säulen werden zufällig"
#property description "jede N Ticks verändert"
#property description "Parameter bars setzt die Anzahl der Balken im Zyklus der Sinusk

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot Histogram
#property indicator_label1  "Histogram"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      bars=30;          // Zeitraum der Sinuskurve in Balken
input int      N=5;              // Anzahl der Ticks um Histogram zu ändern
//--- indicator buffers
double         HistogramBuffer[];
//--- Faktor für 2Pi Winkel im Bogenmaß durch Multiplikation mit der Parameter bars
double    multiplier;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen};
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,HistogramBuffer,INDICATOR_DATA);
//--- Berechnen wir den Multiplikator
   if(bars>1)multiplier=2.*M_PI/bars;
   else
     {
      PrintFormat("Geben Sie den Wert von bars=%d größer als 1",bars);
      //--- Vorzeitige Beendigung des Indikators
      return(INIT_PARAMETERS_INCORRECT);
     }
//---
   return(INIT_SUCCEEDED);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Berechnen der Indikatorwerte
   int start=0;
//--- Wenn die Berechnung auf einem vorherigen Lauf von OnCalculate gemacht war
   if(prev_calculated>0) start=prev_calculated-1; // Setzen wir den Beginn der Berechr
//--- Füllen den Indikator-Puffer mit Werte
   for(int i=start;i<rates_total;i++)
     {
      HistogramBuffer[i]=sin(i*multiplier);
     }
//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Linien im Indikator                      |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+"\r\n"+(string)colors[color_index];

//--- Block mit Änderungen der Linienbreite
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Breite angeben
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Linienart angeben
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm="\r\n"+EnumToString(styles[style_index])+""+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_HISTOGRAM2

DRAW_HISTOGRAM2-Stil zieht ein Histogramm der angegebenen Farbe - die vertikalen Segmente auf die Werte der zwei Indikator-Puffer. Die Breite, Farbe und Stil der Segmenten können wie für DRAW_LINE durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, den Aussehens des Histogramms je nach der aktuellen Situation zu ändern.

DRAW_HISTOGRAM2-Stil kann in einem separaten Grafikfenster und inm Hauptfenster verwendet werden. Leere Werte werden nicht gezeichnet, alle Werte in Indikator-Puffern müssen explizit angegeben werden. Puffer werden nicht mit einem leeren Wert initialisiert.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_HISTOGRAM2 ist 2.

Ein Beispiel für einen Indikator, der zeichnet eine vertikale Säule der angegebenen Farbe und Breite zwischen Open und Close. Die Farbe, Breite und Stil **aller** Säule des Histogramms werden nach dem Zufallsprinzip jede **N** Ticks verändert. Beim Start des Indikators in der OnInit()-Funktion wird die Nummer des Wochentags, für den das Histogramm nicht gezeichnet wird, zufällig eingegeben - invisible_day. Dazu wird ein leerer Wert eingegeben PLOT_EMPTY_VALUE=0:

```
//--- Leerer Wert setzen
   PlotIndexSetDouble(Index_der_Zeichnung_DRAW_SECTION,PLOT_EMPTY_VALUE,0);
```



Bitte beachten Sie, dass für **plot1** mit dem DRAW_HISTOGRAM2-Stil werden Eigenschaften mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden diese drei Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                DRAW_HISTOGRAM2.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                 https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_HISTOGRAM2"
#property description "Zeichnet ein Segment auf jedem Balken zwischen Open und Close"
#property description "Farbe, Breite und Stil werden zufällig"
#property description "jede N Ticks verändert"

#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   1
//--- plot Histogram_2
#property indicator_label1  "Histogram_2"
#property indicator_type1   DRAW_HISTOGRAM2
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input int      N=5;              // Anzahl der Ticks um Histogram zu ändern
//--- indicator buffers
double         Histogram_2Buffer1[];
double         Histogram_2Buffer2[];
//--- der Wochentag, für den der Indikator nicht gezeichnet wird
int invisible_day;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen};
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,Histogram_2Buffer1,INDICATOR_DATA);
   SetIndexBuffer(1,Histogram_2Buffer2,INDICATOR_DATA);
//--- Leerer Wert setzen
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//--- Erhalten eine Zufallszahl von 0 bis 5
   invisible_day=MathRand()%6;
//---
   return(INIT_SUCCEEDED);
 ;}
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Berechnen der Indikatorwerte
   int start=0;
//--- Um den Wochentag durch den Öffnungszeiten jedes Balkens
   MqlDateTime dt;
//--- Wenn die Berechnung auf einem vorherigen Lauf von OnCalculate gemacht war
   if(prev_calculated>0) start=prev_calculated-1; // Setzen wir den Beginn der Berechr
//--- Füllen den Indikator-Puffer mit Werte
   for(int i=start;i<rates_total;i++)
     {
      TimeToStruct(time[i],dt);
      if(dt.day_of_week==invisible_day)
        {
         Histogram_2Buffer1[i]=0;
         Histogram_2Buffer2[i]=0;
        }
      else
        {
         Histogram_2Buffer1[i]=open[i];
         Histogram_2Buffer2[i]=close[i];
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Linien im Indikator                      |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Fark
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+"\r\n"+(string)colors[color_index];

//--- Block mit Änderungen der Linienbreite
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Linienbreite angeben
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Linienart angeben
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm="\r\n"+EnumToString(styles[style_index])+""+comm;
//--- Informationen über den Tag, der in den Berechnungen ignoriert wird, hinzufügen
   comm="\r\nUngezeichnetert Tag - "+EnumToString((ENUM_DAY_OF_WEEK)invisible_day)+cor
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_ARROW

Der DRAW_ARROW-Stil zeichnet Pfeile (Zeichen aus dem Wingdings-Satz) der angegebenen Farbe auf den Wert des Indikator-Puffers. Die Breite und die Farbe der Symbole können wie für DRAW_LINE durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, den Aussehens eines Indikators zu ändern je nach der aktuellen Situation.

Der Symbolcode wird mit der PLOT_ARROW-Eigenschaft eingegeben.

```
//--- Den Szmbolcode aus Wingdings-Satz für PLOT_ARROW definieren
   PlotIndexSetInteger(0,PLOT_ARROW,code);
```

Der Standardwert ist PLOT_ARROW=159 (Kreis).

Jeder Pfeil ist eigentlich ein Zeichen, dass die Höhe und den Ankerpunkt hat, und kann einige wichtige Information über den Chart (z. B. der Schlusskurs am Balken) bedecken. Daher können Sie optional den vertikalen Verschiebung in Pixel, die nicht auf der Skala des Charts abhängig ist, angeben. Bei dieser Anzahl von Pixeln werden die Pfeile visuell vertikal verschoben, obwohl der Wert des Indikators gleich bleiben wird:

```
//--- Definieren wir vertikale Verschiebung der Pfeile in Pixel
   PlotIndexSetInteger(0,PLOT_ARROW_SHIFT,shift);
```

Ein negativer Wert von PLOT_ARROW_SHIFT bedeutet Verschiebung des Pfeils nach oben, ein positiver Wert verschiebt den Pfeil nach unten.

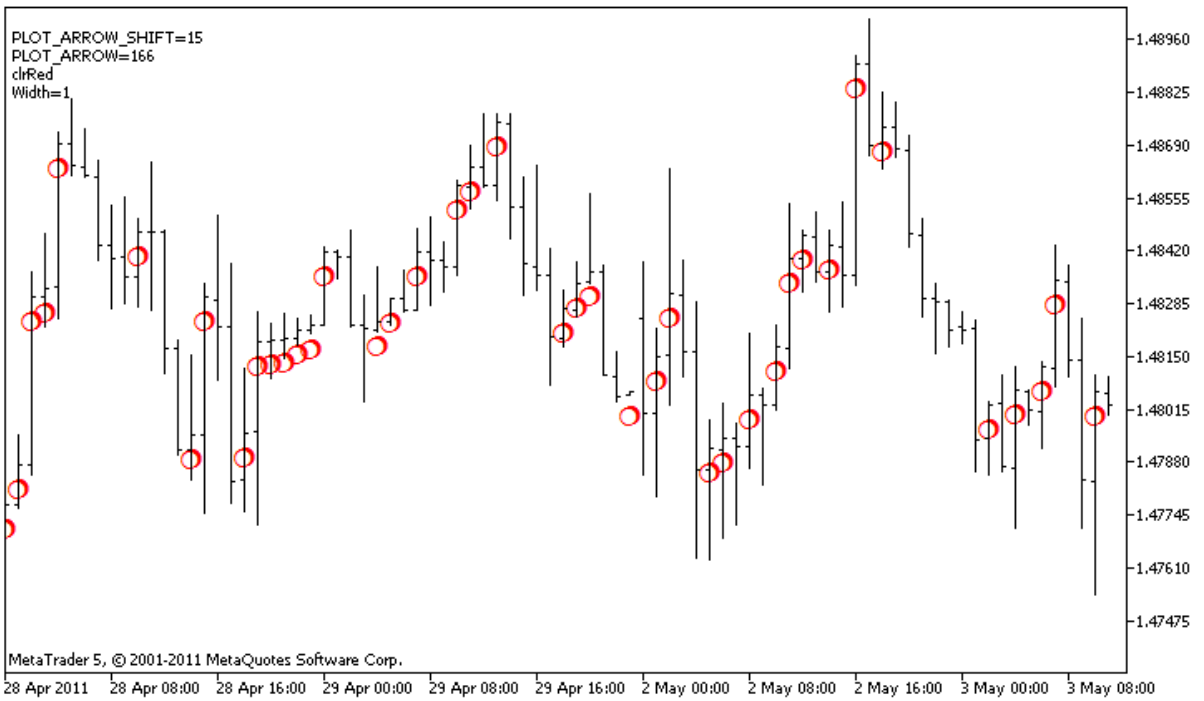
DRAW_ARROW-Stil kann in einem separaten Grafikfenster und im Hauptfenster verwendet werden. Leere Werte werden nicht gezeichnet und nicht in den "Data Window" angezeigt, alle Werte in Indikator-Puffern müssen explizit angegeben werden. Puffer werden nicht mit einem leeren Wert initialisiert.

```
//--- Leerer Wert setzen
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_ARROW,PLOT_EMPTY_VALUE,0);
```

Die erforderliche Anzahl der Puffer für den Bau von DRAW_ARROW ist 1.

Ein Beispiel des Indikators, der Pfeile auf jeder Balken, dessen Schlussprice Close mehr als Schlusspries des vorherigen Balkens ist, zeichnet. Die Farbe, Breite, Verschiebung und Zeichencode **aller** Pfeile werden nach dem Zufallsprinzip jede **N** Ticks verändert.

© 2000-2025, MetaQuotes Ltd.



In diesem Beispiel, für **plot1** mit dem DRAW_ARROW-Stil werden Farbe und Größe mithilfe Compiler-Direktive #Property angegeben, und dann in der Funktion OnCalculate() werden Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                   DRAW_ARROW.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Indikator zeigt DRAW_ARROW"
#property description "Auf dem Chart zeichnet es Pfeile, die durch Unicode-Symbole de
#property description "Die Farbe, Größe, Verschiebung und Zeichencode des Pfeils werde
#property description "jede N Ticks verändert"
#property description "Parameter code gibt den Basiswert an: Code=159 (Kreis)"

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot Arrows
#property indicator_label1  "Arrows"
#property indicator_type1   DRAW_ARROW
#property indicator_color1  clrGreen
#property indicator_width1  1
//--- Eingabeparameter
```

© 2000-2025, MetaQuotes Ltd.

```
input int      N=5;          // Anzahl der Ticks für Änderung
input ushort   code=159;     // Zeichencode zu zeichnen in DRAW_ARROW
//--- Indicator-Puffer
double         ArrowsBuffer[];
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen};
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,ArrowsBuffer,INDICATOR_DATA);
//--- Geben wir Zeichencode um in PLOT_ARROW zu zeichnen an
   PlotIndexSetInteger(0,PLOT_ARROW,code);
//--- Definieren wir vertikale Verschiebung der Pfeile in Pixel
   PlotIndexSetInteger(0,PLOT_ARROW_SHIFT,5);
//--- Setzen wir 0 als ein leerer Wert
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um die Farbe, Größe, Verschiebung und Code des Pfeils zu ände
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Block mit Berechnung der Indikatorwerte
```

© 2000-2025, MetaQuotes Ltd.

```
   int start=1;
   if(prev_calculated>0) start=prev_calculated-1;
//--- Berechnungzyklus
   for(int i=1;i<rates_total;i++)
     {
 //--- Wenn der aktuelle Close-Preis ist höher als der vorherige Close-price, dann set
      if(close[i]>close[i-1])
         ArrowsBuffer[i]=close[i];
      //---Ansonsten geben einen leeren Wert an
      else
         ArrowsBuffer[i]=0;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Zeichen im Indikator                     |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften des Indikators zu erstellen
   string comm="";
//--- Block mit Änderungen der Pfeilfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+"\r\n"+(string)colors[color_index];

//--- Block mit Änderungen der Pfeilgröße
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // die Größe wird von 0 bis 4 angegeben
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Größe der Pfeile
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Block mit Änderungen von Pfeilcode (PLOT_ARROW)
   number=MathRand();
//--- Erhalten wir den Rest aus Integer-Division, um den neuen Pfeilcode zu berechnen
   int code_add=number%20;
//--- Setzen wir einen neuen Zeichnencode als die Summe von code+code_add
   PlotIndexSetInteger(0,PLOT_ARROW,code+code_add);
//--- Schreiben wir den Zeichencode PLOT_ARROW
```

© 2000-2025, MetaQuotes Ltd.

```
   comm="\r\n"+"PLOT_ARROW="+IntegerToString(code+code_add)+comm;

//--- Block mit vertikaler Verschiebung der Pfeile in Pixel
   number=MathRand();
//--- Erhalten wir die Verschiebung als den Rest der ganzzahligen Division
   int shift=20-number%41;
//--- Einstellen wir wine neue Verschiebung von -20 bis 20
   PlotIndexSetInteger(0,PLOT_ARROW_SHIFT,shift);
//--- Screiben wir die Verschiebung PLOT_ARROW_SHIFT
   comm="\r\n"+"PLOT_ARROW_SHIFT="+IntegerToString(shift)+comm;

//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_ZIGZAG

Stil DRAW_ZIGZAG zeichnet Segmente der angegebenen Farbe auf die Werte der zwei Indikator-Puffer. Dieser Stil ist dem Stil DRAW_SECTION sehr ähnlich, aber im Gegensatz zu diesen können Sie die vertikale Segmente innerhalb eines Balkens zeichnen, wenn für diesen Balken Werte für die beiden Indikator-Puffer angegeben sind. Die Segmente werden vom dem Wert im ersten Puffer bis dem Wert im zweiten Indikator-Puffer gezeichnet. Keiner der Puffer darf nicht nur leere Werte erhalten, da in diesem Fall nichts gezeichnet wird.

Die Breite, Stil und Farbe der Linie können ebenso wie für den Stil DRAW_SECTION – durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

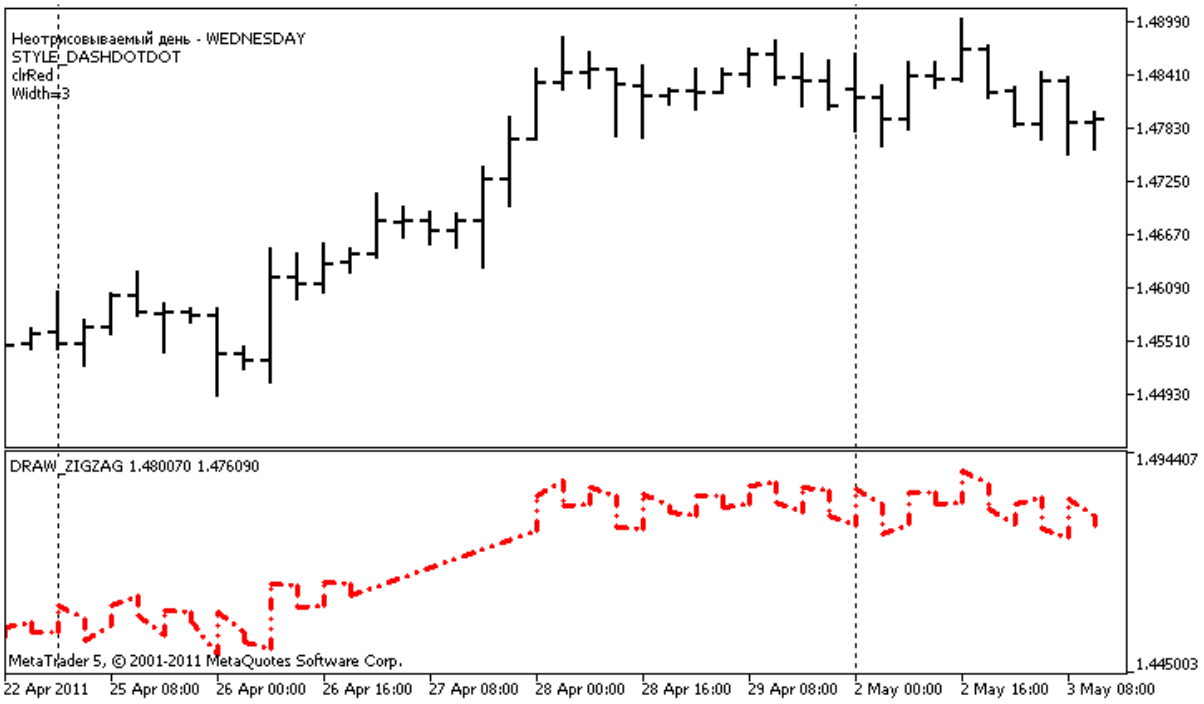
Die Segmente werden vom nicht leeren Wert eines Puffers zum nicht leeren Wert anderes Indikator-Puffers gezeichnet. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE:

```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(Index_der_Zeichnung_DRAW_ZIGZAG,PLOT_EMPTY_VALUE,0);
```

Immer füllen Sie die Indikator-Puffer mit Werten expliziert aus, geben Sie den leeren Wert in Puffer der Balken zu überspringen.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_ZIGZAG ist 2.

Ein Beispiel für einen Indikator, der eine Säge auf die High und Low Priese zeichnet. Farbe, Breite und Linienart des Zigzags werden nach dem Zufallsprinzip jede **N** Ticks geändert.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_ZIGZAG-Stil werden Eigenschaften mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden diese drei Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des

© 2000-2025, MetaQuotes Ltd.

Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                   DRAW_ZIGZAG.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_ZIGZAG"
#property description "Zeichnet "Säge" aus den geraden Linien, überspringt Balken eine
#property description "Der Tag zu überspringen wird zufällig beim Start des Indikators
#property description "Farbe, Breite und Stil der Segmente variieren zufällig"
#property description " jede N Ticks"

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- plot ZigZag
#property indicator_label1  "ZigZag"
#property indicator_type1   DRAW_ZIGZAG
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks für Änderung
//--- indicator buffers
double         ZigZagBuffer1[];
double         ZigZagBuffer2[];
//--- der Wochentag, für den der Indikator nicht gezeichnet wird
int invisible_day;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen};
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays und Indikator-Puffer
   SetIndexBuffer(0,ZigZagBuffer1,INDICATOR_DATA);
   SetIndexBuffer(1,ZigZagBuffer2,INDICATOR_DATA);
//--- Erhalten eine Zufallszahl von 0 bis 6, für diesen Tag wird der Indikator nicht g
   invisible_day=MathRand()%6;
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
```

© 2000-2025, MetaQuotes Ltd.

```
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetString(0,PLOT_LABEL,"ZigZag1;ZigZag2");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Die Zeitstruktur ist benötigt, um den Wochentag jedes Balkens zu erhalten
   MqlDateTime dt;

//--- Startposition der Berechnungen
   int start=0;
//--- Wenn der Indikator wurde auf dem vorherigen Tick berechnet, dann beginnen wir Be
   if(prev_calculated!=0) start=prev_calculated-1;
//--- Berechnungzyklus
   for(int i=start;i<rates_total;i++)
     {
      //--- Schreiben die Balkenöffnungszeit in die Struktur
      TimeToStruct(time[i],dt);
      //--- Wen der Wochentag dieses Balken invisible_day ist
      if(dt.day_of_week==invisible_day)
        {
         //--- Wir schreiben leere Werte des Puffers für diesen Balken
         ZigZagBuffer1[i]=0;
```

© 2000-2025, MetaQuotes Ltd.

```
         ZigZagBuffer2[i]=0;
        }
      //--- Wen der Wochentag entspricht, füllen wir die Puffer
      else
        {
         //--- Wenn die Nummer des Balkens ist gerade
         if(i%2==0)
           {
            //---  Wir schreiben High in den ersten Puffer und Low in den zweiten Puf
            ZigZagBuffer1[i]=high[i];
            ZigZagBuffer2[i]=low[i];
           }
         //--- Die Nummer des Balkens ist ungerade
         else
           {
            //--- Wir füllen den Balken in der umgekehrten Reihenfolge
            ZigZagBuffer1[i]=low[i];
            ZigZagBuffer2[i]=high[i];
           }
        }
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Segmente in Zigzag                       |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über Egenschaften von ZigZag zu erstellen
   string comm="";
//--- Block mit Änderungen der Farbe des ZigZags
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Far
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+"\r\n"+(string)colors[color_index];

//--- Block mit Änderungen der Linienbreite
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
```

© 2000-2025, MetaQuotes Ltd.

```
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm="\r\n"+EnumToString(styles[style_index])+""+comm;
//--- Informationen über den Tag, der in den Berechnungen ignoriert wird, hinzufügen
   comm="\r\nUngezeichneter Tag - "+EnumToString((ENUM_DAY_OF_WEEK)invisible_day)+comm
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_FILLING

Stil DRAW_FILLING zeichnet Farbbereich zwischen den Werten der zwei Indikator-Puffer. In der Tat zeichnet dieser Stil zwei Zeilen und füllt den Raum zwischen ihnen mit einem der zwei angegebenen Farben. Es wird für Indikatoren, diw die Kanäle zeichnen, benutzt. Keiner der Puffer darf nicht nur leere Werte erhalten, da in diesem Fall auch für die Erbringung nicht der Fall ist.

Sie können zwei Farbe für Füllen angeben:

- die erste Farbe ist für die Bereiche, wo die Werte im ersten Indikatorpuffer größer als jene im zweiten Indikator-Puffer sind;
- die zweite Farbe ist für die Bereiche, wo die Werte im zweiten Indikatorpuffer größer als jene im ersten Indikator-Puffer sind.

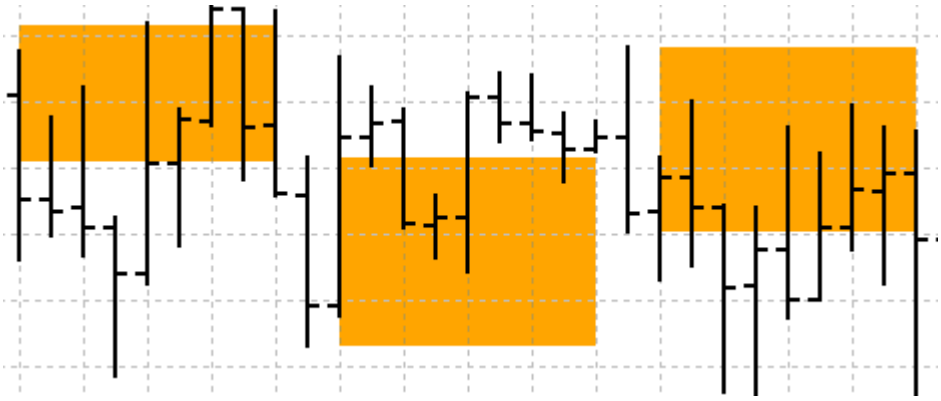
Füllfarbe kann durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Der Indikator wird für alle Bars, für die die Werte der beiden Indikatorpuffer nicht gleich 0 sind und nicht gleich dem leeren Wert sind, berechnet. Um anzugeben, welcher Wert als "leer" betrachtet werden sollte, setzen Sie diesen Wert in Eigenschaft PLOT_EMPTY_VALUE:

```
   #define INDICATOR_EMPTY_VALUE -1.0
   ...
//--- Wert INDICATOR_EMPTY_VALUE (leerer Wert) wird nicht in Berechnungen teil nehmen
   PlotIndexSetDouble(Plot_Index_DRAW_FILLING,PLOT_EMPTY_VALUE,INDICATOR_EMPTY_VALUE);
```

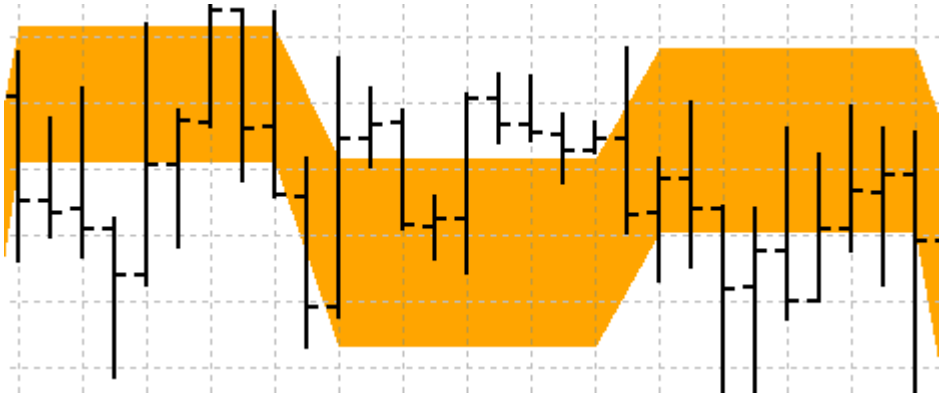
Zeichnung auf die Bars, die nicht in die Berechnung des Indikators beteiligt sind, sind von den Werten in den Indikatorpuffern abhängig:

- Die Bars, für die die Werte der beiden Indikatorpuffer 0 sind, beteiligen nicht in der Zeichnung des Indikators. Das ist ein Bereich mit Null-Werte wird nicht übermalt werden.



- Die Bars, für die die Werte der Indikatorpuffer dem leeren Wert gleich sind, beteiligen in der Zeichnung des Indikators. Bereich mit leeren Werten wird übermalt, um die Bereiche mit sinnvollen Werten zu verbinden.

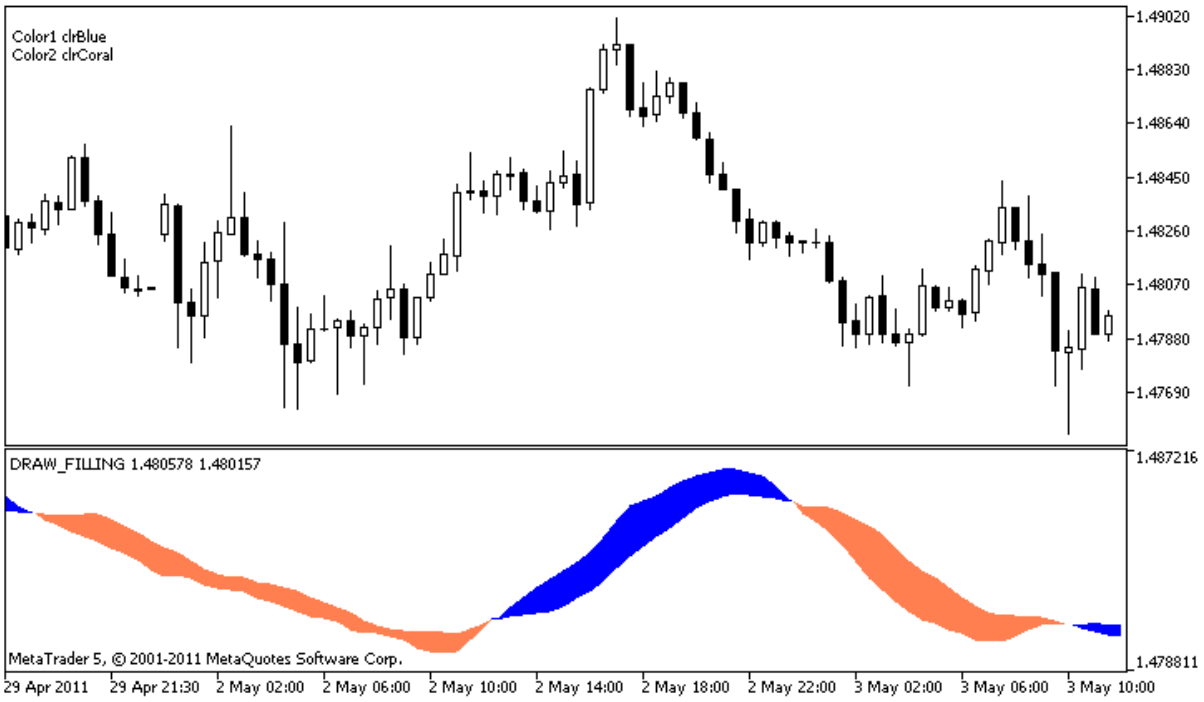
© 2000-2025, MetaQuotes Ltd.



Es ist wichtig zu beachten, dass, wenn ein "leerer Wert" Null gleich ist, dann werden die Bars, die nicht in die Berechnung des Indikators beteiligen, auch übermalt.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_FILLING ist 2.

Ein Beispiel für einen Indikator, der einen Kanal zwischen zwei MAs mit verschiedenen Perioden der Mitteilung in einem separaten Fenster zeichnet. Farbwechsel beim Überschreiten der MAs zeigt Veränderung der aufsteigenden und absteigenden Trends. Farben variieren zufällig jede **N** Ticks. Der Parameter N wird in externen Parameter des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).



Bitte beachten Sie, dass für **plot1** mit dem DRAW_FILLING-Stil werden zwei Farben mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden neue Farben nach dem Zufallsprinzip angegeben.

```
//+------------------------------------------------------------------+
//|                                                 DRAW_FILLING.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_FILLING"
#property description "Zeichnet in einem separaten Fenster einen Kanal zwischen zwei M
#property description "Füllfarbe des Kanals variiert zufällig"
#property description "jede N Ticks verändert"

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- plot Intersection
#property indicator_label1  "Intersection"
#property indicator_type1   DRAW_FILLING
#property indicator_color1  clrRed,clrBlue
#property indicator_width1  1
//--- Eingabeparameter
input int      Fast=13;          // Periode der schnellen MA
input int      Slow=21;          // Periode der schnellen MA
input int      shift=1;          // Verschiebung nach Zukunft (positive)
input int      N=5;              // Anzahl der Ticks für Änderung
//--- Indicator-Puffer
double         IntersectionBuffer1[];
double         IntersectionBuffer2[];
int fast_handle;
int slow_handle;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen,clrAquamarine,clrBlanchedAlmond,clrBrown,clrCo
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,IntersectionBuffer1,INDICATOR_DATA);
   SetIndexBuffer(1,IntersectionBuffer2,INDICATOR_DATA);
//---
   PlotIndexSetInteger(0,PLOT_SHIFT,shift);
//---
   fast_handle=iMA(_Symbol,_Period,Fast,0,MODE_SMA,PRICE_CLOSE);
   slow_handle=iMA(_Symbol,_Period,Slow,0,MODE_SMA,PRICE_CLOSE);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
```

© 2000-2025, MetaQuotes Ltd.

```
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Die erste Berechnung des Indikators, oder Daten haben geändert haben und eine vo
   if(prev_calculated==0)
     {
      //--- Kopieren wir alle Werte der Indikatoren in die entsprechende Puffer
      int copied1=CopyBuffer(fast_handle,0,0,rates_total,IntersectionBuffer1);
      int copied2=CopyBuffer(slow_handle,0,0,rates_total,IntersectionBuffer2);
     }
   else // sparsam füllen nur die Daten, die aktualisiert wurden
     {
      //--- Erhalten wir die Differenz in Balken zwischen aktuellen und früheren Lauf
      int to_copy=rates_total-prev_calculated;
      //--- Wenn es keinen Unterschied gibt, kopieren wir nur einen Wert - das Wert au
      if(to_copy==0) to_copy=1;
      //--- Koperen to_copy Werte an das Ende der Indikator-Puffern
      int copied1=CopyBuffer(fast_handle,0,0,to_copy,IntersectionBuffer1);
      int copied2=CopyBuffer(slow_handle,0,0,to_copy,IntersectionBuffer2);
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert die Füllfarbe des Kanals                                  |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Block mit Änderungen der Linienfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);

//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index1=number%size;
//--- Geben wir die erste Farbe als Eigenschaft PLOT_LINE_COLOR
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,0,colors[color_index1]);
//--- Schreiben wir die erste Farbe
   comm=comm+"\r\nColor1 "+(string)colors[color_index1];

//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   number=MathRand(); // Erhalten eine Zufallszahl
   int color_index2=number%size;
//--- Geben wir die zweite Farbe als Eigenschaft PLOT_LINE_COLOR
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,1,colors[color_index2]);
//--- Schreiben wir die zweite Farbe
   comm=comm+"\r\nColor2 "+(string)colors[color_index2];
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_BARS

DRAW_BARS-Stil zeichnet Balken auf den Werten der vier Indikator-Puffer, die die Preise Open, High, Low und Close enthalten. Es erlaubt individuelle Indikatoren in Form von Balken, darunter ein in separates Grafikfenster und andere Finanzinstrumente, zu erstellen.

Balkenfarbe kann durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Der Indikator ist nur für die Balken, fur die nicht leer Werte **aller** vier Indikator-Puffern angegeben wird, gezeichnet. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE:

```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDoubleIndex_der_Darstellung_DRAW_BARS,PLOT_EMPTY_VALUE,0);
```

Immer füllen Sie die Indikator-Puffer mit Werten expliziert aus, geben Sie den leeren Wert in Puffer der Balken zu überspringen.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_BARS ist 4. Alle Puffer sollen eine nach der anderen in dieser Reihenfolge gehen: Open, High, Low und Close. Keiner der Puffer darf nicht nur leere Werte erhalten, da in diesem Fall nichts gezeichnet wird.

Ein Beispiel für einen Indikator, der Balken des Finanzinstrumenten in einem separaten Fenster zeichnet. Die Balkenfarbe wird zufällig alle **N** Ticks verändert. Der Parameter N wird in externen Parameter des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).



© 2000-2025, MetaQuotes Ltd.

Bitte beachten Sie, dass für **plot1** mit dem DRAW_BARS-Stil die Farbe mit Compiler-Direktiven #property angegeben wird, und dann in der Funktion OnCalculate() wird die Farbe nach dem Zufallsprinzip aus einer vorbereiteten Liste ausgewählt.

```
//+------------------------------------------------------------------+
//|                                                    DRAW_BARS.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator zeigt DRAW_BARS"
#property description "Zeichnet in einem separaten Fenster Balken des ausgewählten Sym
#property description "Die Farbe und Größe der Balken und der Symbol werden nach dem Z
#property description "jede N Ticks verändert

#property indicator_separate_window
#property indicator_buffers 4
#property indicator_plots   1
//--- plot Bars
#property indicator_label1  "Bars"
#property indicator_type1   DRAW_BARS
#property indicator_color1  clrGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks um Art zu ändern
input int      bars=500;         // Anzahl der Balken zu zeigen
input bool     messages=false;   // Ausgabe der Meldungen ins"Experts"-Journal
//--- Indicator-Puffer
double         BarsBuffer1[];
double         BarsBuffer2[];
double         BarsBuffer3[];
double         BarsBuffer4[];
//--- Symbolname
string symbol;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen,clrPurple,clrBrown,clrIndianRed};
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- wenn es gibt zu wenig bars, Arbeit früher vollenden
   if(bars<50)
     {
      Comment("Geben Sie eine größere Anzahl von Bars! Arbeit des Indikators ist beend
```

© 2000-2025, MetaQuotes Ltd.

```
      return(INIT_PARAMETERS_INCORRECT);
     }
//--- indicator buffers mapping
   SetIndexBuffer(0,BarsBuffer1,INDICATOR_DATA);
   SetIndexBuffer(1,BarsBuffer2,INDICATOR_DATA);
   SetIndexBuffer(2,BarsBuffer3,INDICATOR_DATA);
   SetIndexBuffer(3,BarsBuffer4,INDICATOR_DATA);
//--- Name des Symbols, für das die Balken gezeichnet werden
   symbol=_Symbol;
//--- Einstellen wir Anzeige des Symbols
   PlotIndexSetString(0,PLOT_LABEL,symbol+" Open;"+symbol+" High;"+symbol+" Low;"+symb
   IndicatorSetString(INDICATOR_SHORTNAME,"DRAW_BARS("+symbol+")");
//--- Leerer Wert
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0.0);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
 //--- Wählen wir ein neues Symbol aus dem "Market Watch"
      symbol=GetRandomSymbolName();
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();

      int tries=0;
      //--- Machen wir 5 Versuche, um den Puffer mit Preise aus symbol zu füllen
      while(!CopyFromSymbolToBuffers(symbol,rates_total) && tries<5)
        {
         //--- Zähler der CopyFromSymbolToBuffers() Funktionsaufrufe
         tries++;
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Füllen wir den Indikator-Puffer mit Preise aus                   |
//+------------------------------------------------------------------+
bool CopyFromSymbolToBuffers(string name,int total)
  {
//--- In die Array rates[] kopieren wir Preise Open, High, Low und Close
   MqlRates rates[];
//--- Zähler der Versuche
   int attempts=0;
//--- Wieviel kopiert
   int copied=0;
//--- Machen wir 25 Versuche, um die Zeitreihe für das gewünschten Symbol zu erhalten
   while(attempts<25 && (copied=CopyRates(name,_Period,0,bars,rates))<0)
     {
      Sleep(100);
      attempts++;
      if(messages) PrintFormat("%s CopyRates(%s) attempts=%d",__FUNCTION__,name,attemp
     }
//--- Wenn es gelang nicht eine ausreichende Anzahl von Balken zu kopieren
   if(copied!=bars)
     {
      //--- erstellen wir einen Nachrichtenstring
      string comm=StringFormat("Für Symbol %s konnte nur %d Balken von %d gefragten Ba
                               name,
                               copied,
                               bars
                               );
      //--- Zeigen wir einer Nachricht in einer Kommentar auf dem Haupt-Chart-Fenster
      Comment(comm);
      //--- Anzeige von Informationen
      if(messages) Print(comm);
      return(false);
     }
   else
     {
      //--- Einstellen wir Anzeige des Symbols
      PlotIndexSetString(0,PLOT_LABEL,name+" Open;"+name+" High;"+name+" Low;"+name+"
      IndicatorSetString(INDICATOR_SHORTNAME,"DRAW_BARS("+name+")");
     }
//--- Initialisieren wir den Puffer leere Werte
   ArrayInitialize(BarsBuffer1,0.0);
   ArrayInitialize(BarsBuffer2,0.0);
   ArrayInitialize(BarsBuffer3,0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
   ArrayInitialize(BarsBuffer4,0.0);
//--- Kopieren wir Preise in Puffer
   for(int i=0;i<copied;i++)
     {
 //--- Berechnen wir den entsprechenden Index für den Puffer
      int buffer_index=total-copied+i;
 //--- Schreiben wir die Preise in Puffer
      BarsBuffer1[buffer_index]=rates[i].open;
      BarsBuffer2[buffer_index]=rates[i].high;
      BarsBuffer3[buffer_index]=rates[i].low;
      BarsBuffer4[buffer_index]=rates[i].close;
     }
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt ein zufälliges Symbol aus dem Market Watch zurück           |
//+------------------------------------------------------------------+
string GetRandomSymbolName()
  {
//--- Anzahl der in dem "Market Watch" Fenster angezeigten Symbole
   int symbols=SymbolsTotal(true);
//--- Position des Symbols in der Liste - eine Zufallszahl zwischen 0 und symbols
   int number=MathRand()%symbols;
//--- Gibt den Namen des Symbols an der angegebenen Position zurück
   return SymbolName(number,true);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen von Balken                                   |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Balken zu erstellen
   string comm="";
//--- Block mit Änderungen der Balkenfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Linienfarbe
   comm=comm+"\r\n"+(string)colors[color_index];

//--- Block mit Änderungen der Balkenbreite
   number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
```

© 2000-2025, MetaQuotes Ltd.

```
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Schreiben wir den Namen des Symbols
   comm="\r\n"+symbol+comm;

//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_CANDLES

DRAW_CANDLES-Stil zeichnet Kerzen auf den Werten der vier Indikator-Puffer, die die Preise Open, High, Low und Close enthalten. Es erlaubt individuelle Indikatoren in Form von Kerzen, darunter ein in separates Grafikfenster und andere Finanzinstrumente, zu erstellen.

Kerzenfarbe kann durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren "wiederzubeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Der Indikator wird nur für die Balken, für die nicht leere Werte **aller** vier Indikator-Puffer angegeben werden, gezeichnet. Um anzugeben, welcher Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE:

```
//--- Der leere Wert (0) wird nicht beim Zeichnen berücksichtigt
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_CANDLES,PLOT_EMPTY_VALUE,0);
```

Füllen Sie die Indikator-Puffer immer explizit mit Werten aus; geben Sie den leeren Wert auf den Balken in Puffern ein, die übersprungen werden müssen.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_CANDLES ist 4. Alle Puffer sollen in dieser Reihenfolge erscheinen: Open, High, Low und Close. Keiner der Puffer darf nur leere Werte beinhalten, denn in diesem Fall wird nicht gezeichnet.

Für den Stil DRAW_CANDLES kann man eine bis drei Farben setzen, davon hängt das Aussehen von Kerzen ab. Wenn nur eine Farbe angegeben ist, werden alle Farben auf dem Chart diese Farbe haben.

```
//--- gleiche Kerzen, die die gleiche Farbe haben
#property indicator_label1  "One color candles"
#property indicator_type1   DRAW_CANDLES
//--- es wurde nur eine Farbe angegeben, deswegen haben alle Kerzen die gleiche Farbe
#property indicator_color1  clrGreen
```

Wenn man zwei Farben (durch Komma getrennt) angibt, dann werden der Umriss der Kerze mit der ersten Farbe und der Körper - mit der zweiten gezeichnet.

```
//--- die Farbe der Kerzen unterscheidet sich von der Farbe der Schatten
#property indicator_label1  "Two color candles"
#property indicator_type1   DRAW_CANDLES
//--- Schatten und Umriss - grün, Körper - weiss
#property indicator_color1  clrGreen,clrWhite
```

Damit bärische und bullische Kerzen unterschiedlich angezeigt werden, muss man drei Farben, getrennt durch Komma, angeben. In diesem Fall wird der Umriss der Kerze die erste Farbe, bullische Kerze - die zweite und bärische Kerze - die dritte Farbe haben.

```
//--- die Farbe der Kerzen unterscheidet sich von der Farbe der Schatten
#property indicator_label1  "One color candles"
#property indicator_type1   DRAW_CANDLES
//--- die Schatten und der Umriss sind grün, der Körper der bärischen Kerze ist weiß,
#property indicator_color1  clrGreen,clrWhite,clrRed
```

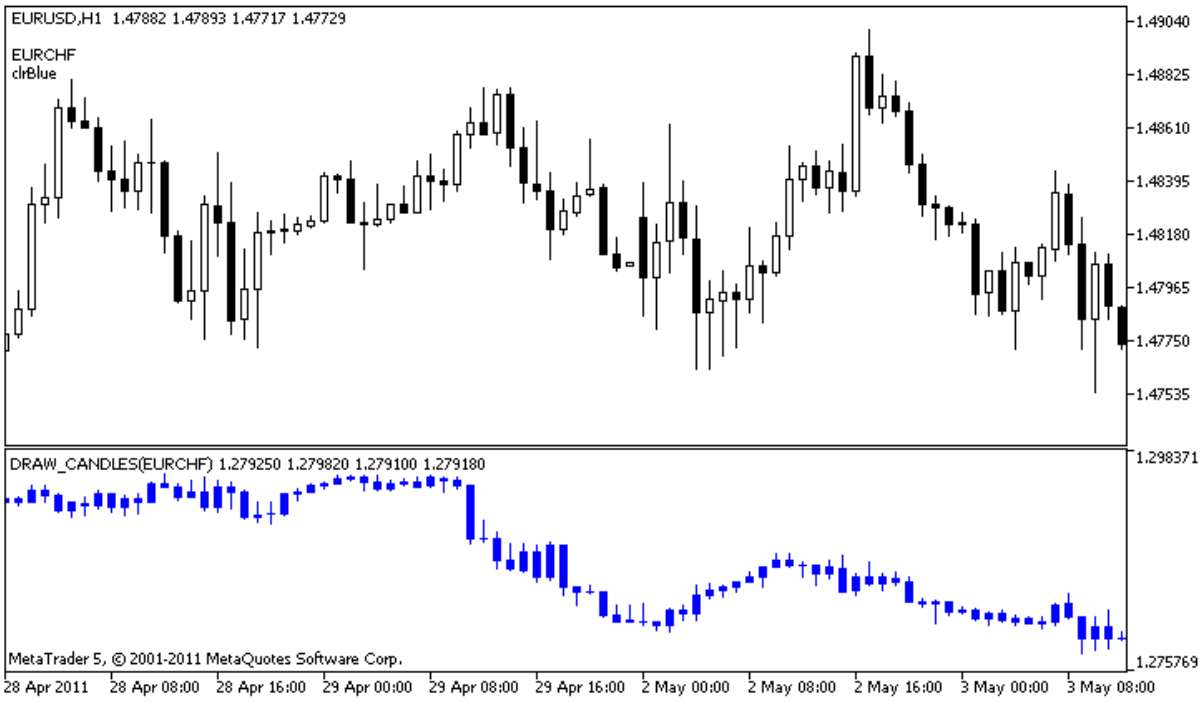
© 2000-2025, MetaQuotes Ltd.

Auf diese Weise kann man mithilfe des Stils DRAW_CANDLES eigene benutzerdefinierte Varianten der Kerzenfarben gestalten. Alle Farben können beim Laufen des Indikators mithilfe der Funktion PlotIndexSetInteger(Index_DRAW_CANDLES, PLOT_LINE_COLOR, modifier_nummer, Farbe) geändert werden, wobei:

- 0 - Farbe des Umrisses und der Schatten
- 1- Farbe des Körpers der bullischen Kerze
- 2 - Farbe des Körpers der bärischen Kerze

```
//--- setzen wir die Farbe für den Umriss und die Schatten
PlotIndexSetInteger(0,PLOT_LINE_COLOR,0,clrBlue);
//--- setzen wir die Farbe für den Körper der bullischen Kerze
PlotIndexSetInteger(0,PLOT_LINE_COLOR,1,clrGreen);
//--- setzen wir die Farbe für den Körper der bärischen Kerze
PlotIndexSetInteger(0,PLOT_LINE_COLOR,2,clrRed);
```

Ein Beispiel für einen Indikator, der Kerzen des Finanzinstrumenten in einem separaten Fenster zeichnet. Die Kerzenfarbe wird zufällig alle **N** Ticks verändert. Der Parameter N wird in externen Parameter des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).



Bitte beachten Sie, dass für **Plot1** die Farbe mit Compiler-Direktiven #property angegeben wird, und dann in der Funktion OnCalculate() wird die Farbe nach dem Zufallsprinzip aus einer vorbereiteten Liste ausgewählt.

```
//+------------------------------------------------------------------+
//|                                                DRAW_CANDLES.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
```

© 2000-2025, MetaQuotes Ltd.

```
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator zeigt DRAW_CANDLES."
#property description "Zeichnet in einem separaten Fenster verschiedenfarbige Kerzen d
#property description " "
#property description "Die Farbe und Größe der Kerzen und der Symbol werden nach dem Z
#property description "jede N Ticks verändert."

#property indicator_separate_window
#property indicator_buffers 4
#property indicator_plots   1
//--- plot Bars
#property indicator_label1  "DRAW_CANDLES1"
#property indicator_type1   DRAW_CANDLES
#property indicator_color1  clrGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1

//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks um Art zu ändern
input int      bars=500;         // Anzahl der Balken zu zeigen
input bool     messages=false;   // Ausgabe der Meldungen ins"Experts"-Journal
//--- Indicator-Puffer
double         Candle1Buffer1[];
double         Candle1Buffer2[];
double         Candle1Buffer3[];
double         Candle1Buffer4[];
//--- Symbolname
string symbol;
//--- Array, um die Farbe zu speichern
color colors[]={clrRed,clrBlue,clrGreen,clrPurple,clrBrown,clrIndianRed};
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- wenn es gibt zu wenig bars, Arbeit früher vollenden
   if(bars<50)
     {
      Comment("Geben Sie eine größere Anzahl von Bars! Arbeit des Indikators ist beend
      return(INIT_PARAMETERS_INCORRECT);
     }
//--- indicator buffers mapping
   SetIndexBuffer(0,Candle1Buffer1,INDICATOR_DATA);
   SetIndexBuffer(1,Candle1Buffer2,INDICATOR_DATA);
   SetIndexBuffer(2,Candle1Buffer3,INDICATOR_DATA);
   SetIndexBuffer(3,Candle1Buffer4,INDICATOR_DATA);
//--- Leerer Wert
```

© 2000-2025, MetaQuotes Ltd.

```
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//--- Name des Symbols, für das die Balken gezeichnet werden
   symbol=_Symbol;
//--- Einstellen wir Anzeige des Symbols
   PlotIndexSetString(0,PLOT_LABEL,symbol+" Open;"+symbol+" High;"+symbol+" Low;"+symb
   IndicatorSetString(INDICATOR_SHORTNAME,"DRAW_CANDLES("+symbol+")");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=INT_MAX-100;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
 //--- Wählen wir ein neues Symbol aus dem "Market Watch"
      symbol=GetRandomSymbolName();
      //--- Ansicht verändern
      ChangeLineAppearance();
 //--- Wählen wir ein neues Symbol aus dem "Market Watch"
      int tries=0;
      //--- Machen wir 5 Versuche, um den Puffer plot1 mit Preise aus symbol zu füller
      while(!CopyFromSymbolToBuffers(symbol,rates_total,0,
            Candle1Buffer1,Candle1Buffer2,Candle1Buffer3,Candle1Buffer4)
            && tries<5)
        {
         //--- Zähler der CopyFromSymbolToBuffers() Funktionsaufrufe
         tries++;
        }
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Füllt die angegebene Kerze                                       |
//+------------------------------------------------------------------+
bool CopyFromSymbolToBuffers(string name,
                             int total,
                             int plot_index,
                             double &buff1[],
                             double &buff2[],
                             double &buff3[],
                             double &buff4[]
                             )
  {
//--- In die Array rates[] kopieren wir Preise Open, High, Low und Close
   MqlRates rates[];
//--- Zähler der Versuche
   int attempts=0;
//--- Wieviel kopiert
   int copied=0;
//--- Machen wir 25 Versuche, um die Zeitreihe für das gewünschten Symbol zu erhalten
   while(attempts<25 && (copied=CopyRates(name,_Period,0,bars,rates))<0)
     {
      Sleep(100);
      attempts++;
      if(messages) PrintFormat("%s CopyRates(%s) attempts=%d",__FUNCTION__,name,attemp
     }
//--- Wenn es gelang nicht eine ausreichende Anzahl von Balken zu kopieren
   if(copied!=bars)
     {
      //--- erstellen wir einen Nachrichtenstring
      string comm=StringFormat("Für Symbol %s konnte nur %d Balken von %d gefragten Ba
                               name,
                               copied,
                               bars
                               );
      //--- Zeigen wir einer Nachricht in einer Kommentar auf dem Haupt-Chart-Fenster
      Comment(comm);
      //--- Anzeige von Informationen
      if(messages) Print(comm);
      return(false);
     }
   else
     {
      //--- Einstellen wir Anzeige des Symbols
      PlotIndexSetString(plot_index,PLOT_LABEL,name+" Open;"+name+" High;"+name+" Low;
     }
//--- Initialisieren wir den Puffer leere Werte
   ArrayInitialize(buff1,0.0);
   ArrayInitialize(buff2,0.0);
   ArrayInitialize(buff3,0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
   ArrayInitialize(buff4,0.0);
//--- Kopieren wir Preise in den Puffer bei jedem Tick
   for(int i=0;i<copied;i++)
     {
 //--- Berechnen wir den entsprechenden Index für den Puffer
      int buffer_index=total-copied+i;
 //--- Schreiben wir die Preise in Puffer
      buff1[buffer_index]=rates[i].open;
      buff2[buffer_index]=rates[i].high;
      buff3[buffer_index]=rates[i].low;
      buff4[buffer_index]=rates[i].close;
     }
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt ein zufälliges Symbol aus dem Market Watch zurück           |
//+------------------------------------------------------------------+
string GetRandomSymbolName()
  {
//--- Anzahl der in dem "Market Watch" Fenster angezeigten Symbole
   int symbols=SymbolsTotal(true);
//--- Position des Symbols in der Liste - eine Zufallszahl zwischen 0 und symbols
   int number=MathRand()%symbols;
//--- Gibt den Namen des Symbols an der angegebenen Position zurück
   return SymbolName(number,true);
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen von Balken                                   |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Balken zu erstellen
   string comm="";
//--- Block mit Änderungen der Balkenfarbe
   int number=MathRand(); // Erhalten wir Zufallszahl
//--- Teiler der Zahl entspricht der Größe des Arrays colors[]
   int size=ArraySize(colors);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um eine neue Farb
   int color_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_COLOR,colors[color_index]);
//--- Schreiben wir die Farbe
   comm=comm+"\r\n"+(string)colors[color_index];
//--- Schreiben wir den Namen des Symbols
   comm="\r\n"+symbol+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_LINE

Stil DRAW_COLOR_LINE ist eine Farbversion von DRAW_LINE, er zeichnet eine Linie von den Werten des Indikator-Puffers. Aber dieser Stil, wie in allen farbigen Stile, mit einem Namen **COLOR**, hat auch ein besonderer Indikator-Puffer, der den Index (Nummer) der Farbe aus dem angegebenen Farbarray erhielt. So kann die Farbe jedes Liniensegment eingestellt werden, wenn Sie die Farbindex der Linie auf diesem Balken angeben.

Die Breite, Stil und Farben um die Linie zu färben kann durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

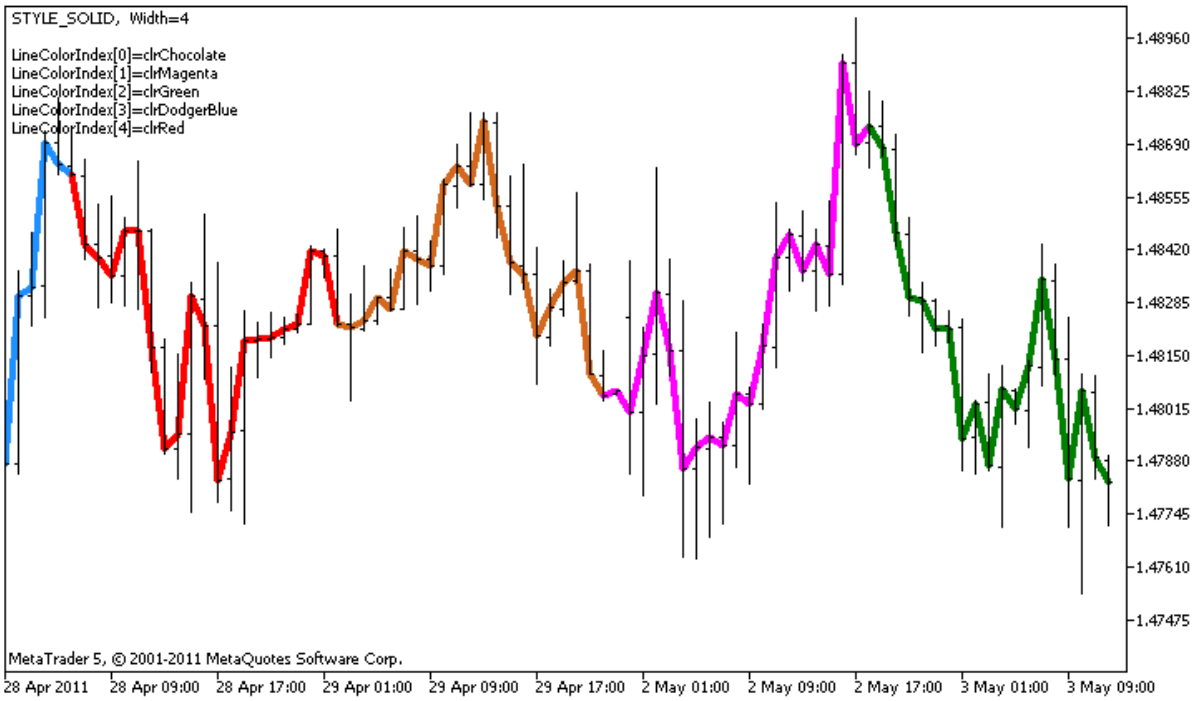
Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_LINE ist 2.

- einen Puffer für Speicherung die Werte des Indikators, auf denen die Linie gezeichnet wird;
- einen Puffer, um die Farbindex der Linie auf jedem Balken.

Farben können mit der Compiler-Direktive #property indicator_color1 durch Kommas angegeben werden. Die Anzahl der Farben darf maximal 64 sein.

```
//--- Definieren wir 5 Farben um Balken zu färben (sie sind im speziellen Array gespei
#property indicator_color1  clrRed,clrBlue,clrGreen,clrOrange,clrDeepPink // (Sie könn
```

Ein Beispiel für einen Indikator, der eine Linie auf die Close Priese der Balken zeichnet. Breite und Stil der Linie werden nach dem Zufallsprinzip jede N=5 Ticks geändert.



Farben der Liniensegmente werden nach dem Zufallsprinzip in der benutzerdefinierten Funktion ChangeColors() geändert.

```
//+------------------------------------------------------------------+
//| Ändert die Farbe der Linienteile                                 |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbeindex, in dem wir die Farbe s
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("LineColorIndex[%d]=%s \r\n",plot_color_ind,ColorToString
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
```

Das Beispiel zeigt die Besonderheit der Farbe-Versionen der Indikatoren - um die Farbe des Liniensegments zu ändern, brauchen Sie nicht die Werte im Puffer ColorLineColors[] (die Farbeindizes enthält) zu ändern. Es genügt, neue Farben in einem speziellen Array zu definieren. Dies ermöglicht Ihnen, schnell die Farbe für die grafische Konstruktion zu ändern, indem Sie nur eine kleine Array von Farben mit der PlotIndexSetInteger() Funktion ändern.

Bitte beachten Sie, dass für **plot1** mit dem DRAW_COLOR_LINE-Stil werden Eigenschaften mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden diese drei Eigenschaften nach dem Zufallsprinzip angegeben.

Die Parameter N und Länge (Länge der farbigen Segmente der Balken) sind in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                DRAW_COLOR_LINE.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator demonstriert DRAW_COLOR_LINE"
#property description "Zeichnet eine Linie mit farbigen Segmenten von 20 Balken auf Cl
#property description "Breite, Stil und Farbe der Liniensegmente werden zufällig"
#property description "jede N Ticks verändert

#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   1
//--- plot ColorLine
#property indicator_label1  "ColorLine"
#property indicator_type1   DRAW_COLOR_LINE
//--- Definieren wir 5 Farben um Balken zu färben (sie sind im speziellen Array gespei
#property indicator_color1  clrRed,clrBlue,clrGreen,clrOrange,clrDeepPink // (Sie könn
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      N=5;           // Anzahl der Ticks für Änderung
input int      Length=20;     // Länge der Farbsegmente in Balken
int            line_colors=5; // Anzahl der angegebenen Farben ist 5 - Sehen Sie #prop
//--- Buffer für das Rendering
double         ColorLineBuffer[];
//--- Puffer zu speichern Farben der Liniensegmente auf jedem Balken
double         ColorLineColors[];

//--- Array, um die Farbe zu speichern, enthält 7 Elemente
color colors[]={clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGolde
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung vom Array und Indikator-Puffer
   SetIndexBuffer(0,ColorLineBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,ColorLineColors,INDICATOR_COLOR_INDEX);
//--- Initialisierung des Zufallszahlengenerators
   MathSrand(GetTickCount());
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
 //--- Ändern wir die Farben der Liniensegmente
      ChangeColors(colors,5);
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Block mit Berechnung der Indikatorwerte
   for(int i=0;i<rates_total;i++)
     {
      //--- Schreiben wir den Wert des Indikators in den Puffer
      ColorLineBuffer[i]=close[i];
      //--- Nun definieren wir eine zufällige Farbeindex für diesen Balken
      int color_index=i%(5*Length);
      color_index=color_index/Length;
      //--- Auf diesem balken wird die Linie mit der Farbe Nummer color_index gezeichn
      ColorLineColors[i]=color_index;
     }

//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert die Farbe der Linienteile                                 |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
```

© 2000-2025, MetaQuotes Ltd.

```
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbeindex, in dem wir die Farbe s
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("LineColorIndex[%d]=%s \r\n",plot_color_ind,ColorToString
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der angezeigten Linie im Indikator           |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5; // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+" Width="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   int size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm=EnumToString(styles[style_index])+", "+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_SECTION

DRAW_COLOR_SECTION ist eine farbige version des Stils DRAW_SECTION, aber es erlaubt jede Sektion mit einer eigenen Farbe zu zeichnen. Stil DRAW_COLOR_SECTION, wie alle farbigen Stile mit **COLOR** in ihren Namen, hat ein besonderer Indikator-Puffer, der den Index (Nummer) der Farbe aus dem angegebenen Farbarray erhielt. So kann die Farbe jeder Sektion gesetzt werden, wenn Sie die Farbindex für die Balken, auf dem das Sektion ende ist, angeben.

Die Breite, Farbe und Stil der Sektionen können wie für DRAW_SECTION durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Die Sektionen werden aus einem nicht leeren Wert zu einem anderen leeren Wert des Indikator-Puffers gezeichnet, leere Werte werden ignoriert. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE. Zum Beispiel, wenn der Indikator wie Sektionen auf nicht leere Werte gezeichnet werden soll, müssen Sie einen Nullwert als leer angeben:

```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_COLOR_SECTION,PLOT_EMPTY_VALUE,0);
```

Füllen Sie immer alle Elemente des Indikator-Puffers mit den Werten, geben sie leere Werte für nicht-gezeichnete Elemente.

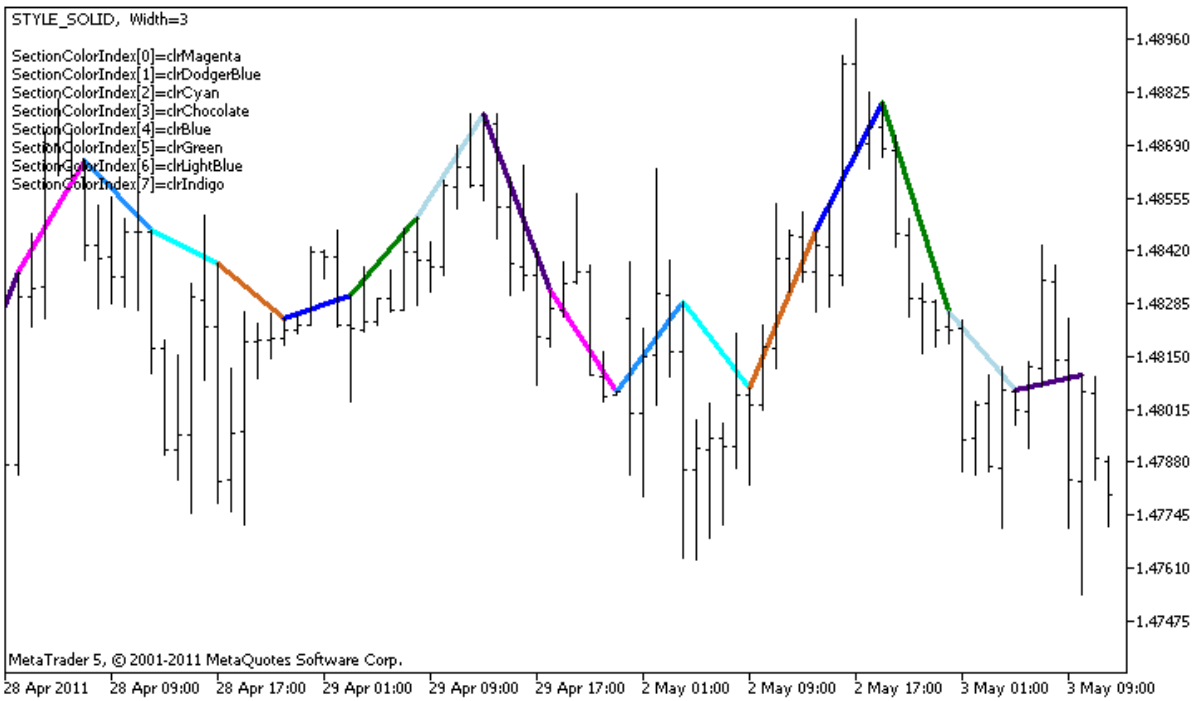
Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_SECTION ist 2.

- einen Puffer für Speicherung die Werte des Indikators, auf denen die Linie gezeichnet wird;
- ein Puffer, um die Farbindex des Segments (es its sinnvoll nur für Nicht-leere Werte) zu speichern.

Farben können mit der Compiler-Direktive #property indicator_color1 durch Kommas angegeben werden. Die Anzahl der Farben darf maximal 64 sein.

Ein Beispiel für einen Indikator, der farbige Sektionen von 5 Balken auf die High-Preise zeichnet. Farbe, Breite und Stil der Sektionen werden nach dem Zufallsprinzip jede **N** Ticks geändert.

© 2000-2025, MetaQuotes Ltd.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_COLOR_SECTION-Stil werden 8 Farben mithilfe Compiler-Direktive #property angegeben. Dann werden Farben zufällig aus Farbarray colors[] in der Funktion OnCalculate() angegeben.

Der Parameter N ist in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                              DRAW_COLOR_SECTION.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_COLOR_SECTION"
#property description "Zeichnet farbige Sektionen von bestimmten Anzahl der Balken"
#property description "Farbe, Breite und Stil der Sektionen werden zufällig"
#property description "jede N Ticks verändert"

#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   1
//--- plot ColorSection
#property indicator_label1  "ColorSection"
#property indicator_type1   DRAW_COLOR_SECTION
//--- Definieren wir 8 Farben um Segmente zu färben (sie sind im speziellen Array gesp
#property indicator_color1  clrRed,clrGold,clrMediumBlue,clrLime,clrMagenta,clrBrown,c
#property indicator_style1  STYLE_SOLID
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_width1  1
//--- Eingabeparameter
input int      N=5;                      // Anzahl der Balken zu ändern
input int      bars_in_section=5;        // Länge der Sektionen in Balken
//--- Hilfsvariable, um Ende der Sektionen zu berechnen
int            divider;
int            color_sections;
//--- Buffer für das Rendering
double         ColorSectionBuffer[];
//--- Puffer zu speichern Farben der Liniensegmente auf jefem Balken
double         ColorSectionColors[];
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrWhiteSmoke,clrCyan,clrMediumPurp
  };
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,ColorSectionBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,ColorSectionColors,INDICATOR_COLOR_INDEX);
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---- Anzahl der Farben um die Sektionen zu färben
   color_sections=8;   //  Sehen Sie Kommentar zur Eigenschaft #property indicator_col
//--- Überprüfen wir die Indikator-Parameter
   if(bars_in_section<=0)
     {
      PrintFormat("Ungültiger Wert für die Länge der Sektion=%d",bars_in_section);
      return(INIT_PARAMETERS_INCORRECT);
     }
   else divider=color_sections*bars_in_section;
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Ändern wir die Farben der Sektionen
      ChangeColors(colors,color_sections);
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Anzahl der Balken, mit dem die Berechnung des Indikators beginnt
   int start=0;
//--- Wenn der Indikator bevor berechnet wurde, dann setzen wir start bei dem vorherig
   if(prev_calculated>0) start=prev_calculated-1;
//--- alle Berechnungen der Indikatorwerte
   for(int i=start;i<rates_total;i++)
     {
      //--- Wenn der Nummer des Balkens ist teilbar durch Sektion_Länge, bedeutet dies
      if(i%bars_in_section==0)
        {
         //--- Ende der Segment ist auf den High-Preis dieses Balkens gesetzt
         ColorSectionBuffer[i]=high[i];
         //--- Rest der Division von Balkennummern durch Sektion_Länge*Anzahl_der_Farb
         int rest=i%divider;
         //Erhalten wir den Farbenummer =  von 0 bis Anzahl_der_Farben-1
         int color_indext=rest/bars_in_section;
         ColorSectionColors[i]=color_indext;
        }
      //--- Wenn der Rest der Division gleich bars ist,
      else
        {
 //--- Wenn etwas nicht passt, dann überspringen wir diesen Balken - setzen 0
         ColorSectionBuffer[i]=0;
        }
     }
//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Ändert die Farbe der Linienteile                                 |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbeindex, in dem wir die Farbe s
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("SectionColorIndex[%d]=%s \r\n",plot_color_ind,ColorToStr
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
 ;}
//+------------------------------------------------------------------+
//| Ändert das Aussehen der angezeigten Linie im Indikator           |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5; // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+" Width="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   int size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm=EnumToString(styles[style_index])+", "+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_HISTOGRAM

DRAW_COLOR_HISTOGRAM-Stil zeichnet ein Histogramm der Farbsäulen von Null bis den angegebenen Wert. Die Werte werden aus den Indikator-Puffer aufgenommen. Jede Säule kann eine eigene Farbe aus einer vorgegebenen Satz von Farben haben.

Die Breite, Farbe und Stil des Histogramms können wie für DRAW_HISTOGRAM durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, den Aussehens des Histogramms je nach der aktuellen Situation zu ändern.

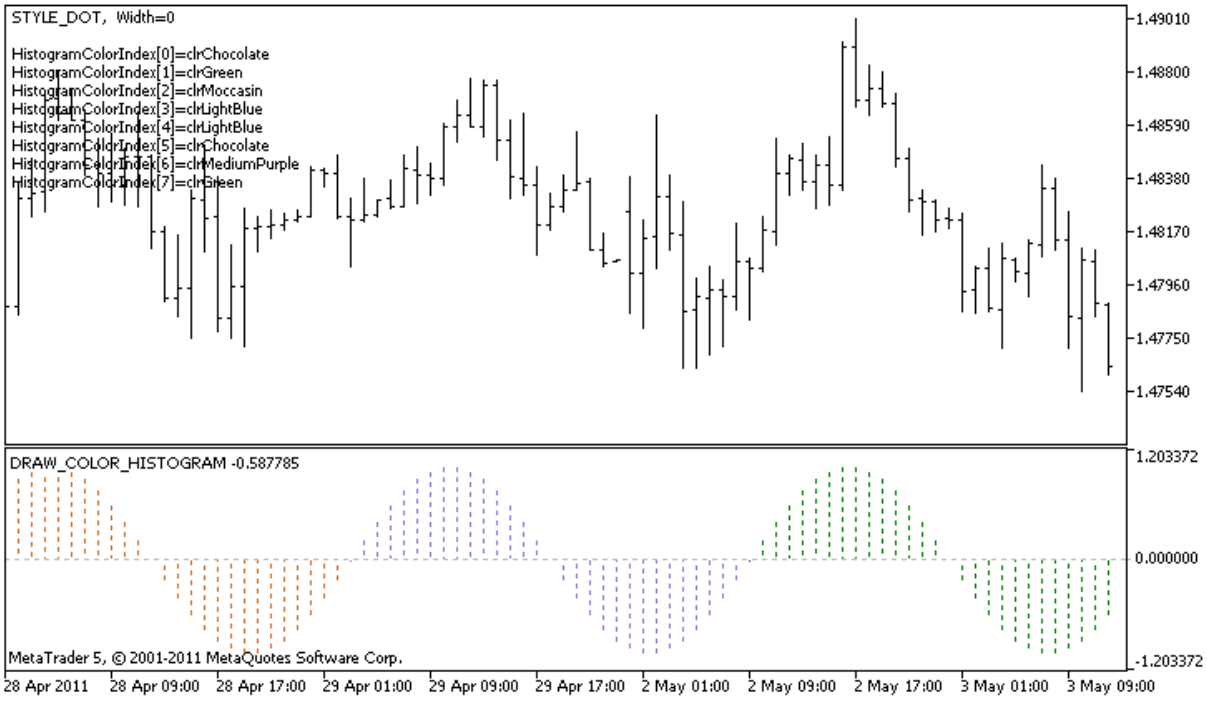
Da wird auf jeden Balken eine Säule von Null-Ebene gezeichnet, dann wird DRAW_COLOR_HISTOGRAM besser für Zeichnung in einem separaten Fenster verwendet. Meistens wird diese Art der graphischen Konstruktion an einen Oszillator Art von Indikatoren, z.B. Awesome Oscillator oder Market Facilitation Index verwendet. Für leere nicht gezeichnete Werte ist es genug Null-Werte anzugeben.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_HISTOGRAM ist 2.

- ein Puffer für die Speicherung von Nicht-Null vertikales Segment an jedem Balken, ist das zweite Ende des Segments immer an der Null-Linie des Indikators;
- ein Puffer, um die Farbindex des Segments (es its sinnvoll nur für Nicht-leere Werte) zu speichern.

Farben können mit der Compiler-Direktive #property indicator_color1 durch Kommas angegeben werden. Die Anzahl der Farben darf maximal 64 sein.

Ein Beispiel für einen Indikator, der eine Sinuskurve der angegebenen Farbe auf Funktion MathSin() zeichnet. Die Farbe, Breite und Stil **aller** Säule des Histogramms werden nach dem Zufallsprinzip jede **N** Ticks verändert. Parameter bars gibt den Zeitraum der Sinuskurve, d.h. in einer bestimmten Anzahl von Balken wird die Sinuskurve ihren Zyklus wiederholen.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_COLOR_HISTOGRAM-Stil ursprünglich 5 Farben mit Compiler-Direktiven #property angegeben werden, und dann, in der OnCalculate()-Funktion, werden

© 2000-2025, MetaQuotes Ltd.

die Farben nach dem Zufallsprinzip aus 14 Farben, die in Array colors[] gespeichert sind, ausgewählt. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                          DRAW_COLOR_HISTOGRAM.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_COLOR_HISTOGRAM"
#property description "Zeichnet die Sinuskurve als ein Histogram in einem separaten Fe
#property description "Farbe und Breite der Säulen werden zufällig"
#property description "jede N Ticks verändert"
#property description "Parameter bars setzt die Anzahl der Balken um die Sinuskurve zu

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- Eingabeparameter
input int      bars=30;          // Zeitraum der Sinuskurve in Balken
input int      N=5;              // Anzahl der Ticks um Histogram zu ändern
//--- plot Color_Histogram
#property indicator_label1  "Color_Histogram"
#property indicator_type1   DRAW_COLOR_HISTOGRAM
//--- Definieren wir 8 Farben um Segmente zu färben (sie sind im speziellen Array gesp
#property indicator_color1  clrRed,clrGreen,clrBlue,clrYellow,clrMagenta,clrCyan,clrMe
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Puffer mit Werte
double         Color_HistogramBuffer[];
//--- Puffer mit Farbindizes
double         Color_HistogramColors[];
//--- Faktor für 2Pi Winkel im Bogenmaß durch Multiplikation mit der Parameter bars
double         multiplier;
int            color_sections;
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrWhiteSmoke,clrCyan,clrMediumPurp
  };
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,Color_HistogramBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,Color_HistogramColors,INDICATOR_COLOR_INDEX);
//---- Anzahl der Farben um die Sinuskurve zu färben
   color_sections=8;   //  Sehen Sie Kommentar zur Eigenschaft #property indicator_co
//--- Berechnen wir den Multiplikator
   if(bars>1)multiplier=2.*M_PI/bars;
   else
     {
      PrintFormat("Geben Sie den Wert von bars=%d größer als 1",bars);
      //--- Vorzeitige Beendigung des Indikators
      return(INIT_PARAMETERS_INCORRECT);
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Ändern wir die Farben des Histogramms
      ChangeColors(colors,color_sections);
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Berechnen der Indikatorwerte
   int start=0;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wenn die Berechnung auf einem vorherigen Lauf von OnCalculate gemacht war
   if(prev_calculated>0) start=prev_calculated-1; // Setzen wir den Beginn der Berechn
//--- Füllen den Indikator-Puffer mit Werte
   for(int i=start;i<rates_total;i++)
     {
      //--- Wert
      Color_HistogramBuffer[i]=sin(i*multiplier);
      //--- Farbe
      int color_index=i%(bars*color_sections);
      color_index/=bars;
      Color_HistogramColors[i]=color_index;
     }
//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert die Farbe der Linienteile                                 |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbindex, in dem wir die Farbe s
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("HistogramColorIndex[%d]=%s \r\n",plot_color_ind,ColorToS
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der angezeigten Linie im Indikator           |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
```

© 2000-2025, MetaQuotes Ltd.

```
   string comm="";
//--- Block mit Änderungen der Linienbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5; // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+" Width="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   int size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm=EnumToString(styles[style_index])+", "+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_HISTOGRAM2

DRAW_COLOR_HISTOGRAM2-Stil zieht ein Histogramm der angegebenen Farbe - die vertikalen Segmente auf die Werte der zwei Indikator-Puffer. Aber im Gegensatz zu den einfarbigen DRAW_HISTOGRAM2, in diesem Stil zu jedem Histogrammsäule können Sie ihre eigene Farbe aus einer vordefinierten Gruppe angeben. Die Werte der Segmentende werden aus den Indikator-Puffer aufgenommen.

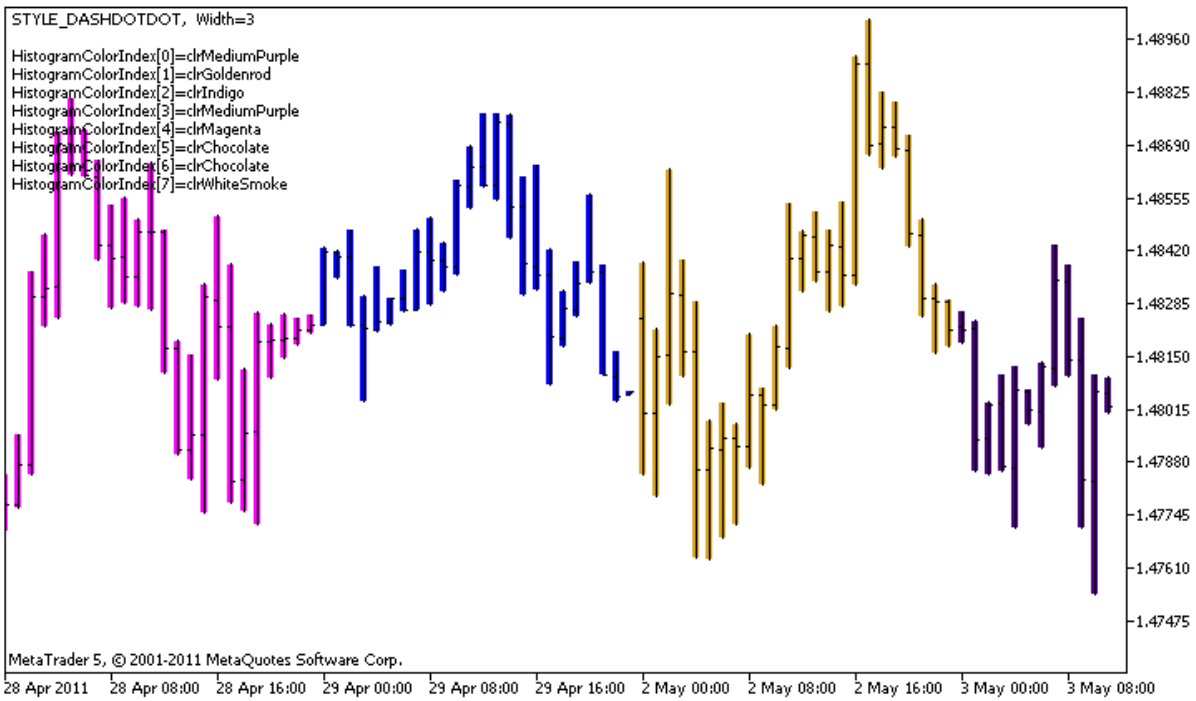
Breite, Stil und Farbe des Histogramms können wie für DRAW_HISTOGRAM2 durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, den Aussehens des Histogramms je nach der aktuellen Situation zu ändern.

DRAW_COLOR_HISTOGRAM2-Stil kann in einem separaten Grafikfenster und im Hauptfenster verwendet werden. Leere Werte werden nicht gezeichnet, alle Werte in Indikator-Puffern müssen explizit angegeben werden. Puffer werden nicht mit einem leeren Wert initialisiert.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_HISTOGRAM2 ist 3:

- zwei Puffer für Speicherung der oberen und unteren Enden des vertikalen Segments auf jeden Balken;
- ein Puffer, um die Farbindex des Segments (es its sinnvoll nur für Nicht-leere Werte) zu speichern.

Ein Beispiel für einen Indikator, der ein Histogram der angegebenen Farbe zwischen den Preisen von High und Low zeichnet. Für jeden Tag der Woche sind die Histogrammlinien mit ihrer Farbe gezeichnet. Die Farbe jeden Tages, Breite und Stil des Histogramms werden nach dem Zufallsprinzip jede **N** Ticks verändert.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_COLOR_HISTOGRAM2-Stil ursprünglich 5 Farben mit Compiler-Direktiven #property angegeben werden, und dann, in der OnCalculate()-Funktion, werden die Farben nach dem Zufallsprinzip aus 14 Farben, die in Array colors[] gespeichert sind, ausgewählt.

© 2000-2025, MetaQuotes Ltd.

Der Parameter N ist in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                        DRAW_COLOR_HISTOGRAM2.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_COLOR_HISTOGRAM2"
#property description "Zeichnet ein Segment auf jedem Balken zwischen Open und Close"
#property description "Farbe, Breite und Stil werden zufällig"
#property description "jede N Ticks verändert"

#property indicator_chart_window
#property indicator_buffers 3
#property indicator_plots   1
//--- plot ColorHistogram_2
#property indicator_label1  "ColorHistogram_2"
#property indicator_type1   DRAW_COLOR_HISTOGRAM2
//--- Definieren wir 5 Farben um Histogramm nach Wochentag zu färben (sie sind im spez
#property indicator_color1  clrRed,clrBlue,clrGreen,clrYellow,clrMagenta
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1

//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks um Histogram zu ändern
int            color_sections;
//--- Puffer mit Werte
double         ColorHistogram_2Buffer1[];
double         ColorHistogram_2Buffer2[];
//--- Puffer mit Farbindizes
double         ColorHistogram_2Colors[];
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrWhiteSmoke,clrCyan,clrMediumPurp
  };
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- indicator buffers mapping
   SetIndexBuffer(0,ColorHistogram_2Buffer1,INDICATOR_DATA);
   SetIndexBuffer(1,ColorHistogram_2Buffer2,INDICATOR_DATA);
   SetIndexBuffer(2,ColorHistogram_2Colors,INDICATOR_COLOR_INDEX);
//--- Leerer Wert setzen
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---- Anzahl der Farben um die Sinuskurve zu färben
   color_sections=8;   //  Sehen Sie Kommentar zur Eigenschaft #property indicator_col
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
 //--- Ändern wir die Farben des Histogramms
      ChangeColors(colors,color_sections);
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Berechnen der Indikatorwerte
   int start=0;
//--- Um den Wochentag durch den Öffnungszeiten jedes Balkens
   MqlDateTime dt;
//--- Wenn die Berechnung auf einem vorherigen Lauf von OnCalculate gemacht war
   if(prev_calculated>0) start=prev_calculated-1; // Setzen wir den Beginn der Berechn
//--- Füllen den Indikator-Puffer mit Werte
   for(int i=start;i<rates_total;i++)
     {
      TimeToStruct(time[i],dt);
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Wert
      ColorHistogram_2Buffer1[i]=high[i];
      ColorHistogram_2Buffer2[i]=low[i];
      //--- Angeben wir den Farbindex nach Wochentag
      int day=dt.day_of_week;
      ColorHistogram_2Colors[i]=day;
     }
//--- den Wert prev_calculated für den nächsten Anruf der Funktion zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert die Farbe der Linieteile                                  |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbindex, in dem wir die Farbe s
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("HistogramColorIndex[%d]=%s \r\n",plot_color_ind,ColorToS
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der angezeigten Linie im Indikator           |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5; // Breite ist von 0 bis 4
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+" Width="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   int size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm=EnumToString(styles[style_index])+", "+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_ARROW

Der DRAW_COLOR_ARROW-Stil zeichnet Pfeile (Zeichen aus dem Wingdings-Satz) der Farbe auf den Wert des Indikator-Puffers. Im Gegensatz zum Stil DRAW_ARROW, ist es möglich, die Farbe für jede Zeichen aus einer vordefinierten Gruppe von Farben, die durch die Eigenschaft indicator_color1 definiert werden, einzustellen.

Die Breite und die Farbe der Symbole können wie für DRAW_ARROW durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, den Aussehens eines Indikators zu ändern je nach der aktuellen Situation.

Der Symbolcode wird mit der PLOT_ARROW-Eigenschaft eingegeben.

```
//--- Den Szmbolcode aus Wingdings-Satz für PLOT_ARROW definieren
   PlotIndexSetInteger(0,PLOT_ARROW,code);
```

Der Standardwert ist PLOT_ARROW=159 (Kreis).

Jeder Pfeil ist eigentlich ein Zeichen, dass die Höhe und den Ankerpunkt hat, und kann einige wichtige Information über den Chart (z. B. der Schlusskurs am Balken) bedecken. Daher können Sie optional den vertikalen Verschiebung in Pixel, die nicht auf der Skala des Charts abhängig ist, angeben. Bei dieser Anzahl von Pixeln werden die Pfeile visuell vertikal verschoben, obwohl der Wert des Indikators gleich bleiben wird:

```
//--- Definieren wir vertikale Verschiebung der Pfeile in Pixel
   PlotIndexSetInteger(0,PLOT_ARROW_SHIFT,shift);
```

Ein negativer Wert von PLOT_ARROW_SHIFT bedeutet Verschiebung des Pfeils nach oben, ein positiver Wert verschiebt den Pfeil nach unten.

DRAW_COLOR_ARROW-Stil kann in einem separaten Grafikfenster und im Hauptfenster verwendet werden. Leere Werte werden nicht gezeichnet und nicht in den "Data Window" angezeigt, alle Werte in Indikator-Puffern müssen explizit angegeben werden. Puffer werden nicht mit einem leeren Wert initialisiert.

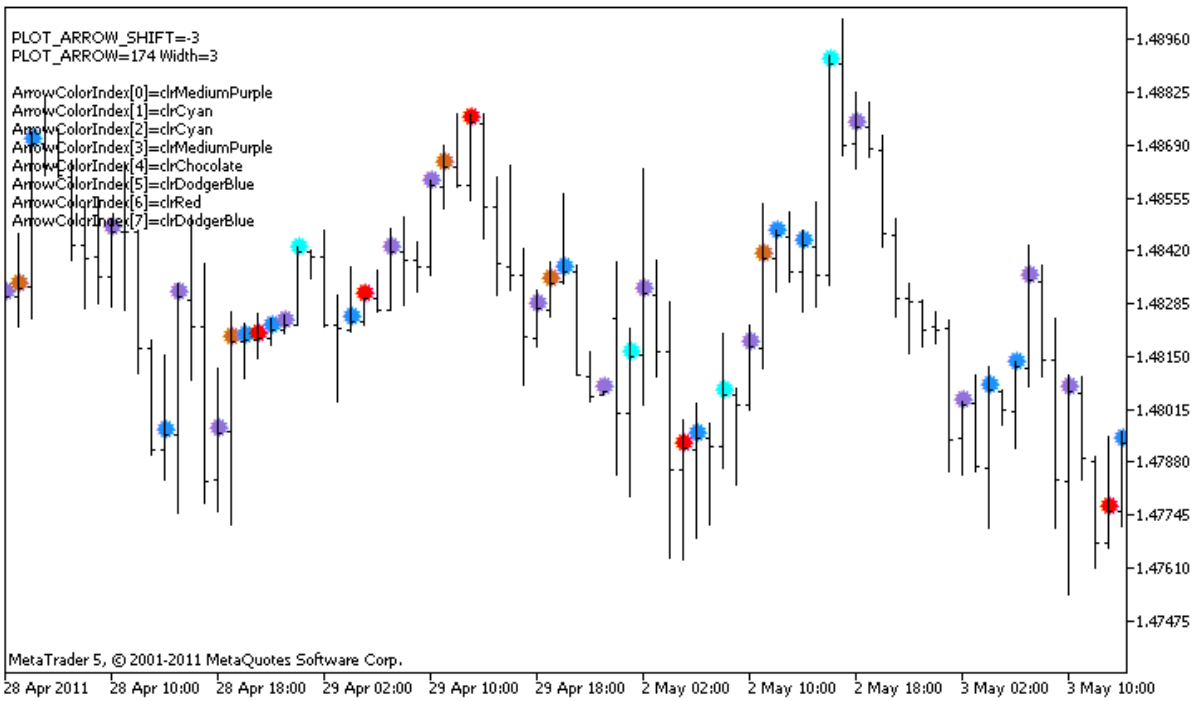
```
//--- Leerer Wert setzen
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_COLOR_ARROW,PLOT_EMPTY_VALUE,0);
```

Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_ARROW ist 4.

- einen Puffer, um den Wert der Preis, auf den das Symbol gezeichnet wird (plus Verschiebung in Pixel, die in PLOT_ARROW_SHIFT-Eigenschaft angegeben wird), zu speichern;
- einen Puffer, um die Farbindex des Pfeils (es its sinnvoll nur für Nicht-leere Werte) zu speichern.

Ein Beispiel des Indikators, der Pfeile auf jeder Balken, dessen Schlusspreis Close mehr als Schlusspreis des vorherigen Balkens ist, zeichnet. Die Breite, Verschiebung und Zeichencode **aller** Pfeile werden nach dem Zufallsprinzip jede **N** Ticks verändert. Die Symbolfarbe hängt von der Nummer des Balkens, auf der es gezeichnet wird, ab.

© 2000-2025, MetaQuotes Ltd.



In diesem Beispiel, für **plot1** mit dem DRAW_COLOR_ARROW-Stil werden Farbe und Größe mithilfe Compiler-Direktive #property angegeben, und dann in der Funktion OnCalculate() werden Eigenschaften nach dem Zufallsprinzip angegeben. Der Parameter N wird in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

Bitte beachten Sie, dass ursprünglich acht Farben mit Compiler-Direktiven #property angegeben werden, und dann, in der OnCalculate()-Funktion, wird die Farbe nach dem Zufallsprinzip aus 14 Farben, die in Array colors[] gespeichert sind, ausgewählt.

```
//+------------------------------------------------------------------+
//|                                                DRAW_COLOR_ARROW.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_COLOR_ARROW"
#property description "Auf dem Chart zeichnet es verschiedenfarbige Pfeile, die durch
#property description "Die Farbe, Größe, Verschiebung und Zeichencode des Pfeils werde
#property description "jede N Ticks verändert."
#property description "Parameter code gibt den Basiswert an: Code=159 (Kreis)"

#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   1
//--- plot ColorArrow
#property indicator_label1  "ColorArrow"
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_type1   DRAW_COLOR_ARROW
//--- Definieren wir 8 Farben um Histogramm nach Wochentag zu färben (sie sind im spez
#property indicator_color1  clrRed,clrBlue,clrSeaGreen,clrGold,clrDarkOrange,clrMagent
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1

//--- Eingabeparameter
input int       N=5;          // Anzahl der Ticks für Änderung
input ushort    code=159;     // Zeichencode zu zeichnen in DRAW_ARROW
int             color_sections;
//--- Indicator-Puffer
double          ColorArrowBuffer[];
//--- Puffer, um die Indexe der Farbe zu speichern
double          ColorArrowColors[];
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrWhiteSmoke,clrCyan,clrMediumPurp
  };
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,ColorArrowBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,ColorArrowColors,INDICATOR_COLOR_INDEX);
//--- Geben wir Zeichencode um in PLOT_ARROW zu zeichnen an
   PlotIndexSetInteger(0,PLOT_ARROW,code);
//--- Definieren wir vertikale Verschiebung der Pfeile in Pixel
   PlotIndexSetInteger(0,PLOT_ARROW_SHIFT,5);
//--- Setzen wir 0 als ein leerer Wert
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//---- Anzahl der Farben um die Sinuskurve zu färben
   color_sections=8;   //  Sehen Sie Kommentar zur Eigenschaft #property indicator_col
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um die Farbe, Größe, Verschiebung und Code des Pfeils zu ände
   ticks++;
//--- Wenn eine kritische Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Pfeile
      ChangeLineAppearance();
 //--- Ändern wir die Farben des Histogramms
      ChangeColors(colors,color_sections);
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Block mit Berechnung der Indikatorwerte
   int start=1;
   if(prev_calculated>0) start=prev_calculated-1;
//--- Berechnungzyklus
   for(int i=1;i<rates_total;i++)
     {
 //--- Wenn der aktuelle Close-Preis ist höher als der vorherige Close-price, dann set
      if(close[i]>close[i-1])
         ColorArrowBuffer[i]=close[i];
      //---Ansonsten geben einen leeren Wert an
      else
         ColorArrowBuffer[i]=0;
      //--- Pfeilfarbe
      int index=i%color_sections;
      ColorArrowColors[i]=index;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Ändert die Farbe der Linienteile                                 |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- erhalten wir eine Zufallszahl
       int number=MathRand();
       //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
       int i=number%size;
       //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
       PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                           PLOT_LINE_COLOR,      //  Property Identifier
                           plot_color_ind,       //  Farbindex, in dem wir die Farbe sc
                           cols[i]);             //  neue Farbe
       //--- schreiben wir die Farbe
       comm=comm+StringFormat("ArrowColorIndex[%d]=%s \r\n",plot_color_ind,ColorToStrin
       ChartSetString(0,CHART_COMMENT,comm);
      }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der angezeigten Linie im Indikator           |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Linie zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5; // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+" Width="+IntegerToString(width);

//--- Block mit Änderungen von Pfeilcode (PLOT_ARROW)
   number=MathRand();
//--- Erhalten wir den Rest aus Integer-Division, um den neuen Pfeilcode zu berechnen
   int code_add=number%20;
//--- Setzen wir einen neuen Zeichnencode als die Summe von code+code_add
   PlotIndexSetInteger(0,PLOT_ARROW,code+code_add);
//--- Schreiben wir den Zeichencode PLOT_ARROW
   comm="\r\n"+"PLOT_ARROW="+IntegerToString(code+code_add)+comm;

//--- Block mit vertikaler Verschiebung der Pfeile in Pixel
   number=MathRand();
//--- Erhalten wir die Verschiebung als den Rest der ganzzahligen Division
   int shift=20-number%41;
//--- Einstellen wir wine neue Verschiebung von
   PlotIndexSetInteger(0,PLOT_ARROW_SHIFT,shift);
//--- Screiben wir die Verschiebung PLOT_ARROW_SHIFT
   comm="\r\n"+"PLOT_ARROW_SHIFT="+IntegerToString(shift)+comm;
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_ZIGZAG

Stil DRAW_COLOR_ZIGZAG zeichnet Segmente in verschiedenen Farben auf die Werten der zwei Indikator-Puffer. Dieser Stil ist eine farbige Version des Stils DRAW_ZIGZAG und ermöglicht Ihnen, die Farbe für jedes Segment aus einer vorgegebenen Satz von Farben anzugeben. Die Segmente werden vom dem Wert im ersten Puffer bis dem Wert im zweiten Indikator-Puffer gezeichnet. Keiner der Puffer darf nicht nur leere Werte erhalten, da in diesem Fall nichts gezeichnet wird.

Die Breite, Farbe und Stil der Linie können wie für DRAW_ZIGZAG durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Die Segmente werden vom nicht leeren Wert eines Puffers zum nicht leeren Wert anderes Indikator-Puffers gezeichnet. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE:

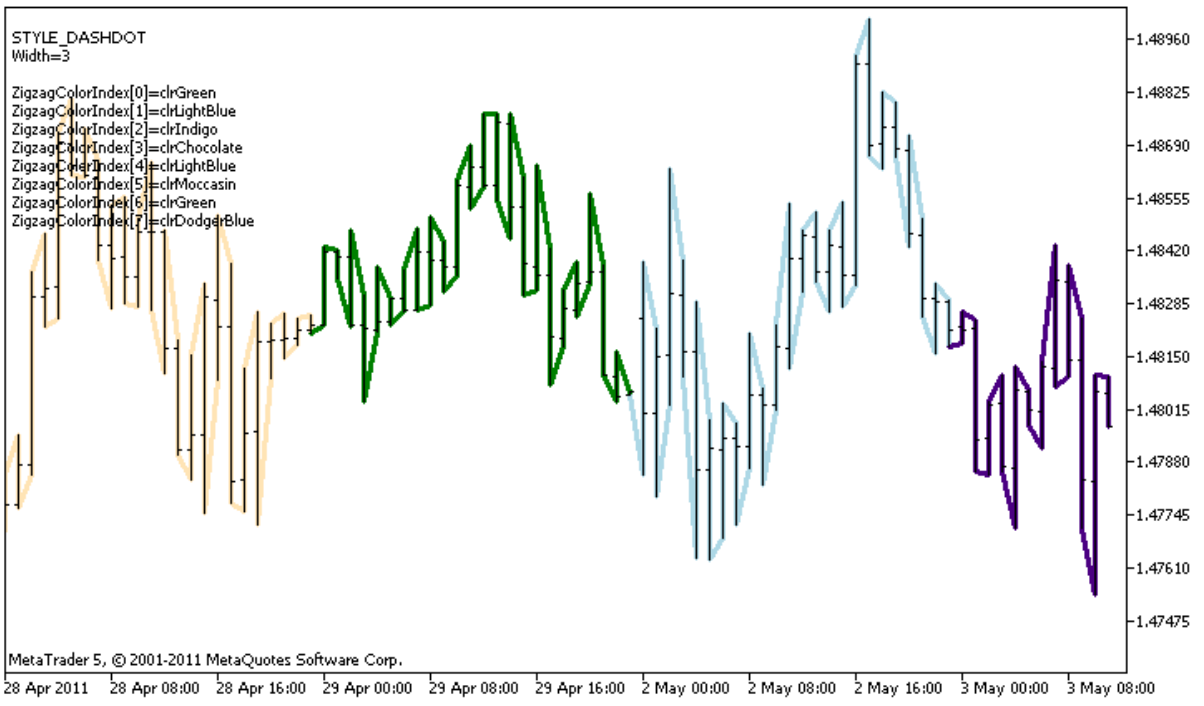
```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_COLOR_ZIGZAG,PLOT_EMPTY_VALUE,0);
```

Immer füllen Sie die Indikator-Puffer mit Werten expliziert aus, geben Sie den leeren Wert in Puffer der Balken zu überspringen.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_ZIGZAG ist 3:

- zwei Puffer zum Speichern der Werte von Ende der Segmente von Zigiag;
- ein Puffer, um die Farbindex des Segments (es its sinnvoll nur für Nicht-leere Werte) zu speichern.

Ein Beispiel für einen Indikator, der eine Säge auf die High und Low Pries zeichnet. Farbe, Breite und Stil der Linien von Zigzag werden nach dem Zufallsprinzip jede **N** Ticks geändert.



© 2000-2025, MetaQuotes Ltd.

Bitte beachten Sie, dass für **plot1** mit dem DRAW_COLOR_ZIGZAG-Stil ursprünglich acht Farben mit Compiler-Direktiven #property angegeben werden, und dann, in der OnCalculate()-Funktion, wird die Farbe nach dem Zufallsprinzip aus 14 Farben, die in Array colors[] gespeichert sind, ausgewählt.

Der Parameter N ist in externen Parametern des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

```
//+------------------------------------------------------------------+
//|                                                DRAW_COLOR_ZIGZAG.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_COLOR_ZIGZAG"
#property description "Zeichnet eine gebrochene Linie von farbigen Segmenten, die Farb
#property description "Farbe, Breite und Stil der Segmente variieren zufällig"
#property description " jede N Ticks"

#property indicator_chart_window
#property indicator_buffers 3
#property indicator_plots   1
//--- plot Color_Zigzag
#property indicator_label1  "Color_Zigzag"
#property indicator_type1   DRAW_COLOR_ZIGZAG
//--- Definieren wir 8 Farben um Segmente zu färben (sie sind im speziellen Array gesp
#property indicator_color1  clrRed,clrBlue,clrGreen,clrYellow,clrMagenta,clrCyan,clrLi
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks für Änderung
int            color_sections;
//--- Puffer mit den Werten der Segmentenden
double         Color_ZigzagBuffer1[];
double         Color_ZigzagBuffer2[];
//--- Puffer mit den Farbindizes der Segmentenden
double         Color_ZigzagColors[];
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrWhiteSmoke,clrCyan,clrMediumPurp
  };
//--- Array für Speicherung der Linienstile
ENUM_LINE_STYLE styles[]={STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,Color_ZigzagBuffer1,INDICATOR_DATA);
   SetIndexBuffer(1,Color_ZigzagBuffer2,INDICATOR_DATA);
   SetIndexBuffer(2,Color_ZigzagColors,INDICATOR_COLOR_INDEX);
//---- Anzahl der Farben um Zigzag zu färben
   color_sections=8;   //  Sehen Sie Kommentar zur Eigenschaft #property indicator_col
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite der Linie zu ändern
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
      //--- Ändern wir die Farben der Sektionen
      ChangeColors(colors,color_sections);
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }

//--- Die Zeitstruktur ist benötigt, um den Wochentag jedes Balkens zu erhalten
   MqlDateTime dt;

//--- Startposition der Berechnungen
   int start=0;
//--- Wenn der Indikator wurde auf dem vorherigen Tick berechnet, dann beginnen wir Be
   if(prev_calculated!=0) start=prev_calculated-1;
//--- Berechnungzyklus
   for(int i=start;i<rates_total;i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- Schreiben die Balkenöffnungszeit in die Struktur
       TimeToStruct(time[i],dt);

       //--- Wenn die balkennummer gerade ist
       if(i%2==0)
         {
          //--- Shreiben wir High in den ersten Puffer und Low in den Zweiten Puffer
          Color_ZigzagBuffer1[i]=high[i];
          Color_ZigzagBuffer2[i]=low[i];
          //--- Farbe des Segments
          Color_ZigzagColors[i]=dt.day_of_year%color_sections;
         }
       //--- Balkennummer ist ungerade
       else
         {
          //--- Füllen den Balken in der umgekehrten Reihenfolge
          Color_ZigzagBuffer1[i]=low[i];
          Color_ZigzagBuffer2[i]=high[i];
          //--- Farbe des Segments
          Color_ZigzagColors[i]=dt.day_of_year%color_sections;
         }
      }
//--- return value of prev_calculated for next call
   return(rates_total);
 ;}
//+------------------------------------------------------------------+
//| Ändert die Farbe der Segmente des Zick-Zacks                     |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbindex, in dem wir die Farbe sc
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
```

© 2000-2025, MetaQuotes Ltd.

```
      comm=comm+StringFormat("ZigzagColorIndex[%d]=%s \r\n",plot_color_ind,ColorToStri
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Segmente in Zigzag                       |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften von Color_ZigZag zu erstellen
   string comm="";
//--- Block mit Änderungen der Linienbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Block mit Änderungen des Linienstils
   number=MathRand();
//--- Teiler der Zahl entspricht der Größe des Arrays styles
   int size=ArraySize(styles);
//--- Erhalten wir den Index als den Rest der ganzzahligen Division, um ein neuer Stil
   int style_index=number%size;
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_COLOR an
   PlotIndexSetInteger(0,PLOT_LINE_STYLE,styles[style_index]);
//--- Schreiben wir den Linienstil
   comm="\r\n"+EnumToString(styles[style_index])+""+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_BARS

Stil DRAW_COLOR_BARS zieht Balken auf die Werte der vier Indikator-Puffer, die Preise Open, High, Low und Close erhalten. Dieser Stil ist eine erweiterte Version des Stils DRAW_BARS und ermöglicht Ihnen, die Farbe für jeden Balken aus einer vorgegebenen Satz von Farben anzugeben. Es erlaubt individuelle Indikatoren in Form von Balken, darunter ein in separates Grafikfenster und andere Finanzinstrumente, zu erstellen.

Balkenfarbe kann durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Der Indikator ist nur für die Balken, fur die nicht leer Werte **aller** vier Indikator-Puffern angegeben wird, gezeichnet. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE:

```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_COLOR_BARS,PLOT_EMPTY_VALUE,0);
```

Immer füllen Sie die Indikator-Puffer mit Werten expliziert aus, geben Sie den leeren Wert in Puffer der Balken zu überspringen.

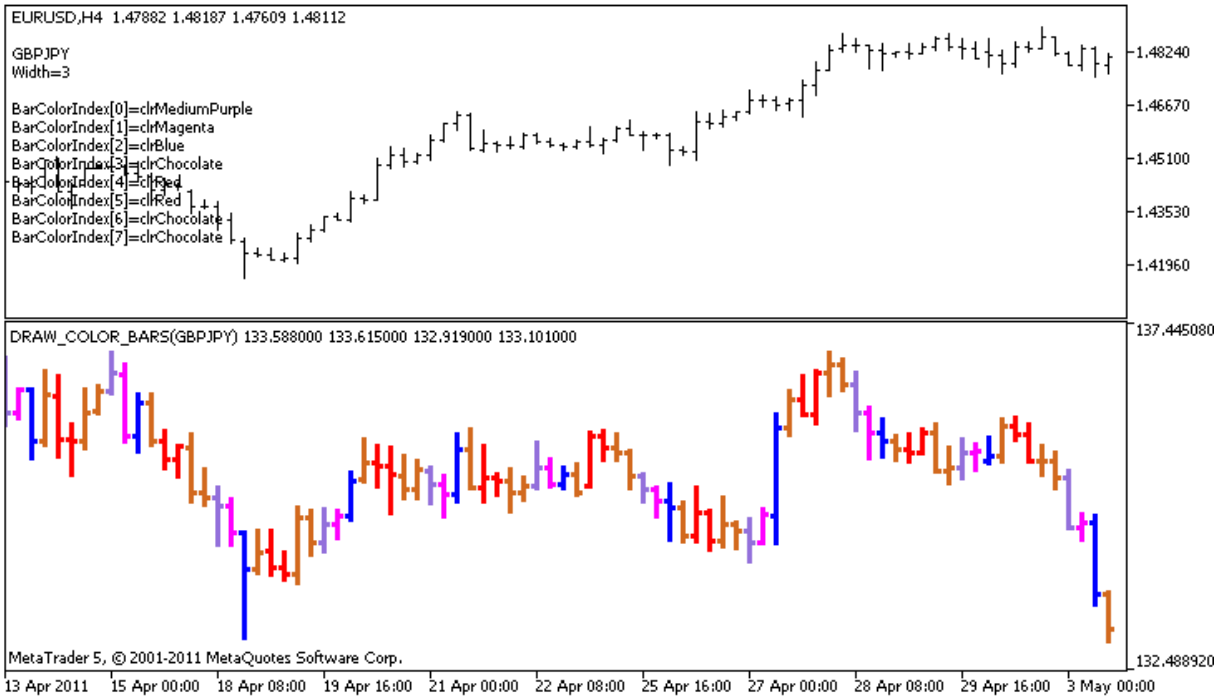
Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_BARS ist 5:

- vier Puffer zum Speichern der Werte von Open, High, Low und Close;
- einen Puffer, um die Farbindex des Balkens (es its sinnvoll nur für gezeichnete Balken) zu speichern.

Alle Puffer sollen eine nach der anderen in dieser Reihenfolge gehen: Open, High, Low, Close und Farbpuffer. Keiner der Preis-Puffer darf nicht nur leere Werte haben, da in diesem Fall nichts gezeichnet wird.

Ein Beispiel für einen Indikator, der Balken des Finanzinstrumenten in einem separaten Fenster zeichnet. Die Balkenfarbe wird zufällig alle **N** Ticks verändert. Der Parameter N wird in externen Parameter des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

© 2000-2025, MetaQuotes Ltd.



Bitte beachten Sie, dass für **plot1** mit dem DRAW_COLOR_BARS-Stil ursprünglich acht Farben mit Compiler-Direktiven #property angegeben werden, und dann, in der OnCalculate()-Funktion, wird die Farbe nach dem Zufallsprinzip aus 14 Farben, die in Array colors[] gespeichert sind, ausgewählt.

```
//+------------------------------------------------------------------+
//|                                                DRAW_COLOR_BARS.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator demonstriert DRAW_COLOR_BARS"
#property description "Zeichnet in einem separaten Fenster verschiedenfarbige Balken c
#property description "Die Farbe und Größe der Balken und der Symbol werden nach dem Z
#property description "jede N Ticks verändert

#property indicator_separate_window
#property indicator_buffers 5
#property indicator_plots   1
//--- plot ColorBars
#property indicator_label1  "ColorBars"
#property indicator_type1   DRAW_COLOR_BARS
//--- Definieren wir 8 Farben um Balken zu färben (sie sind im speziellen Array gespei
#property indicator_color1  clrRed,clrBlue,clrGreen,clrYellow,clrMagenta,clrCyan,clrLi
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks um Art zu ändern
```

© 2000-2025, MetaQuotes Ltd.

```
input int      bars=500;         // Anzahl der Balken zu zeigen
input bool     messages=false;   // Ausgabe der Meldungen ins"Experts"-Journal
//--- Indicator-Puffer
double         ColorBarsBuffer1[];
double         ColorBarsBuffer2[];
double         ColorBarsBuffer3[];
double         ColorBarsBuffer4[];
double         ColorBarsColors[];
//--- Symbolname
string symbol;
int    bars_colors;
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrMagenta,clrCyan,clrMediumPurple
  };
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,ColorBarsBuffer1,INDICATOR_DATA);
   SetIndexBuffer(1,ColorBarsBuffer2,INDICATOR_DATA);
   SetIndexBuffer(2,ColorBarsBuffer3,INDICATOR_DATA);
   SetIndexBuffer(3,ColorBarsBuffer4,INDICATOR_DATA);
   SetIndexBuffer(4,ColorBarsColors,INDICATOR_COLOR_INDEX);
//---- Anzahl der Farben um Balken zu färben
   bars_colors=8;   //  Sehen Sie. Kommentar zur Eigenschaft #property indicator_colo
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=0;
//--- Zählen wir Ticks, um den Stil, die Farbe und Breite des Balkens zu ändern
```

© 2000-2025, MetaQuotes Ltd.

```
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
 //--- Wählen wir ein neues Symbol aus dem "Market Watch"
      symbol=GetRandomSymbolName();
      //--- Ändern wir die Eigenschaften der Linie
      ChangeLineAppearance();
 //--- Ändern wir die Farben der Balken
      ChangeColors(colors,bars_colors);
      int tries=0;
      //--- Machen wir 5 Versuche, um den Puffer mit Preise aus symbol zu füllen
      while(!CopyFromSymbolToBuffers(symbol,rates_total,bars_colors) && tries<5)
        {
         //--- Zähler der CopyFromSymbolToBuffers() Funktionsaufrufe
         tries++;
       ;}
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Füllen wir den Indikator-Puffer mit Preise aus                   |
//+------------------------------------------------------------------+
bool CopyFromSymbolToBuffers(string name,int total,int bar_colors)
  {
//--- In die Array rates[] kopieren wir Preise Open, High, Low und Close
   MqlRates rates[];
//--- Zähler der Versuche
   int attempts=0;
//--- Wie viel kopiert
   int copied=0;
//--- Machen wir 25 Versuche, um die Zeitreihe für das gewünschten Symbol zu erhalten
   while(attempts<25 && (copied=CopyRates(name,_Period,0,bars,rates))<0)
     {
      Sleep(100);
      attempts++;
      if(messages) PrintFormat("%s CopyRates(%s) attempts=%d",__FUNCTION__,name,attemp
     }
//--- Wenn es gelang nicht eine ausreichende Anzahl von Balken zu kopieren
   if(copied!=bars)
     {
      //--- erstellen wir einen Nachrichtenstring
      string comm=StringFormat("Für Symbol %s konnte nur %d Balken von %d gefragten Ba
                               name,
                               copied,
                               bars
```

© 2000-2025, MetaQuotes Ltd.

```
                                  );
      //--- Zeigen wir einer Nachricht in einer Kommentar auf dem Haupt-Chart-Fenster
      Comment(comm);
      //--- Anzeige von Informationen
      if(messages) Print(comm);
      return(false);
     }
   else
     {
      //--- Einstellen wir Anzeige des Symbols
      PlotIndexSetString(0,PLOT_LABEL,name+" Open;"+name+" High;"+name+" Low;"+name+"
      IndicatorSetString(INDICATOR_SHORTNAME,"DRAW_COLOR_BARS("+name+")");
     }
//--- Initialisieren wir den Puffer leere Werte
   ArrayInitialize(ColorBarsBuffer1,0.0);
   ArrayInitialize(ColorBarsBuffer2,0.0);
   ArrayInitialize(ColorBarsBuffer3,0.0);
   ArrayInitialize(ColorBarsBuffer4,0.0);

//--- Kopieren wir Preise in Puffer
   for(int i=0;i<copied;i++)
     {
 //--- Berechnen wir den entsprechenden Index für den Puffer
      int buffer_index=total-copied+i;
 //--- Schreiben wir die Preise in Puffer
      ColorBarsBuffer1[buffer_index]=rates[i].open;
      ColorBarsBuffer2[buffer_index]=rates[i].high;
      ColorBarsBuffer3[buffer_index]=rates[i].low;
      ColorBarsBuffer4[buffer_index]=rates[i].close;
      //---
      ColorBarsColors[buffer_index]=i%bar_colors;
     }
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt ein zufälliges Symbol aus dem Market Watch zurück           |
//+------------------------------------------------------------------+
string GetRandomSymbolName()
  {
//--- Anzahl der in dem "Market Watch" Fenster angezeigten Symbole
   int symbols=SymbolsTotal(true);
//--- Position des Symbols in der Liste - eine Zufallszahl zwischen 0 und symbols
   int number=MathRand()%symbols;
//--- Gibt den Namen des Symbols an der angegebenen Position zurück
   return SymbolName(number,true);
  }
//+------------------------------------------------------------------+
//| Ändert die Farbe der Segmente des ZigZags                        |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbindex, in dem wir die Farbe sc
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("BarColorIndex[%d]=%s \r\n",plot_color_ind,ColorToString
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen von Balken                                   |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Balken zu erstellen
   string comm="";

//--- Block mit Änderungen der Balkenbreite
   int number=MathRand();
//--- Erhalten wir die Breite als den Rest der ganzzahligen Division
   int width=number%5;   // Breite ist von 0 bis 4
//--- Geben wir die Farbe als Eigenschaft PLOT_LINE_WIDTH
   PlotIndexSetInteger(0,PLOT_LINE_WIDTH,width);
//--- Schreiben wir die Linienbreite
   comm=comm+"\r\nWidth="+IntegerToString(width);

//--- Schreiben wir den Namen des Symbols
   comm="\r\n"+symbol+comm;

//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

DRAW_COLOR_CANDLES

DRAW_COLOR_CANDLES-Stil, als DRAW_CANDLES zeichnet Kerzen auf den Werten der vier Indikator-Puffer, die die Preise Open, High, Low und Close enthalten. Aber außerdem, erlaubt es Ihnen, eine Farbe für jede Kerzen aus einer gegebenen Satz anzugeben. Zu diesem Zweck hat der Stil einen speziellen Farbpuffer, der Farbindizes für jeden Balken speichert. Es erlaubt individuelle Indikatoren in Form von Kerzen, darunter ein in separates Grafikfenster und andere Finanzinstrumente, zu erstellen.

Die Anzahl der Farben um die Kerzen zu färben kann durch Compiler-Direktiven oder dynamisch mit der PlotIndexSetInteger()-Funktion angegeben werden. Dynamische Veränderungen in den Eigenschaften der graphischen Konstruktion ermöglichen es Ihnen, die Indikatoren zu "wiederbeleben" so, dass sie ihr Aussehen ändern je nach der aktuellen Situation.

Der Indikator ist nur für die Balken, fur die nicht leer Werte der vier Indikator-Puffern angegeben wird, gezeichnet. Um anzugeben, welchen Wert als "leer" betrachtet werden soll, setzen Sie diesen Wert in der Eigenschaft PLOT_EMPTY_VALUE:

```
//--- Der Leerer Wert (0) wird nicht in der Zeichnung teilnehmen
   PlotIndexSetDouble(Index_der_Darstellung_DRAW_COLOR_CANDLES,PLOT_EMPTY_VALUE,0);
```

Immer füllen Sie die Indikator-Puffer mit Werten expliziert aus, geben Sie den leeren Wert in Puffer der Balken zu überspringen.

Die erforderliche Anzahl der Puffer für den Bau von DRAW_COLOR_CANDLES ist 5:

- vier Puffer zum Speichern der Werte von Open, High, Low und Close;
- einen Puffer, um die Farbindex der Kerze (es its sinnvoll nur für gezeichnete Kerzen) zu speichern.

Alle Puffer sollen eine nach der anderen in dieser Reihenfolge gehen: Open, High, Low, Close und Farbpuffer. Keiner der Preis-Puffer darf nicht nur leere Werte haben, da in diesem Fall nichts gezeichnet wird.

Ein Beispiel für einen Indikator, der Kerzen des Finanzinstrumenten in einem separaten Fenster zeichnet. Die Kerzenfarbe wird zufällig alle **N** Ticks verändert. Der Parameter N wird in externen Parameter des Indikators angegeben, damit kann es manuell angegeben werden (die Registerkarte "Einstellungen" im Indikatoreigenschaften-Fenster).

© 2000-2025, MetaQuotes Ltd.



Bitte beachten Sie, dass für **Plot1** die Farbe mit Compiler-Direktiven #property angegeben wird, und dann in der Funktion OnCalculate() wird die Farbe nach dem Zufallsprinzip aus einer vorbereiteten Liste ausgewählt.

```
//+------------------------------------------------------------------+
//|                                              DRAW_COLOR_CANDLES.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"

#property description "Der Indikator zeigt DRAW_COLOR_CANDLES."
#property description "Zeichnet in einem separaten Fenster verschiedenfarbige Kerzen c
#property description " "
#property description "Die Farbe und Größe der Kerzen und der Symbol werden nach dem Z
#property description "jede N Ticks verändert."

#property indicator_separate_window
#property indicator_buffers 5
#property indicator_plots   1
//--- plot ColorCandles
#property indicator_label1  "ColorCandles"
#property indicator_type1   DRAW_COLOR_CANDLES
//--- Definieren wir 8 Farben um Kerzen zu färben (sie sind im speziellen Array gespei
#property indicator_color1  clrRed,clrBlue,clrGreen,clrYellow,clrMagenta,clrCyan,clrLi
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Eingabeparameter
input int      N=5;              // Anzahl der Ticks um Art zu ändern
input int      bars=500;         // Anzahl der Kerzen zu zeigen
input bool     messages=false;   // Ausgabe der Meldungen ins"Experts"-Journal
//--- Indicator-Puffer
double         ColorCandlesBuffer1[];
double         ColorCandlesBuffer2[];
double         ColorCandlesBuffer3[];
double         ColorCandlesBuffer4[];
double         ColorCandlesColors[];
int            candles_colors;
//--- Symbolname
string symbol;
//--- Array, um die Farbe zu speichern, enthält 14 Elemente
color colors[]=
  {
   clrRed,clrBlue,clrGreen,clrChocolate,clrMagenta,clrDodgerBlue,clrGoldenrod,
   clrIndigo,clrLightBlue,clrAliceBlue,clrMoccasin,clrMagenta,clrCyan,clrMediumPurple
  };
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- wenn es gibt zu wenig bars, Arbeit früher vollenden
   if(bars<50)
     {
      Comment("Geben Sie eine größere Anzahl von Bars! Arbeit des Indikators ist beend
      return(INIT_PARAMETERS_INCORRECT);
     }
//--- indicator buffers mapping
   SetIndexBuffer(0,ColorCandlesBuffer1,INDICATOR_DATA);
   SetIndexBuffer(1,ColorCandlesBuffer2,INDICATOR_DATA);
   SetIndexBuffer(2,ColorCandlesBuffer3,INDICATOR_DATA);
   SetIndexBuffer(3,ColorCandlesBuffer4,INDICATOR_DATA);
   SetIndexBuffer(4,ColorCandlesColors,INDICATOR_COLOR_INDEX);
//--- Leerer Wert
   PlotIndexSetDouble(0,PLOT_EMPTY_VALUE,0);
//--- Name des Symbols, für das die Balken gezeichnet werden
   symbol=_Symbol;
//--- Einstellen wir Anzeige des Symbols
   PlotIndexSetString(0,PLOT_LABEL,symbol+" Open;"+symbol+" High;"+symbol+" Low;"+symb
   IndicatorSetString(INDICATOR_SHORTNAME,"DRAW_COLOR_CANDLES("+symbol+")");
//---- Anzahl der Farben um Kerzen zu färben
   candles_colors=8;      //  Sehen Sie. Kommentar zur Eigenschaft #property indicator_
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int ticks=INT_MAX-100;
//--- Zählen wir Ticks, um den Stil und die zu ändern
   ticks++;
//--- Wenn eine ausreichende Anzahl von Ticks angesammelt hat
   if(ticks>=N)
     {
 //--- Wählen wir ein neues Symbol aus dem "Market Watch"
      symbol=GetRandomSymbolName();
      //--- Ansicht verändern
      ChangeLineAppearance();
 //--- Ändern wir die Farben der Balken
      ChangeColors(colors,candles_colors);

      int tries=0;
      //--- Machen wir 5 Versuche, um den Puffer plot1 mit Preise aus symbol zu füller
      while(!CopyFromSymbolToBuffers(symbol,rates_total,0,
            ColorCandlesBuffer1,ColorCandlesBuffer2,ColorCandlesBuffer3,
            ColorCandlesBuffer4,ColorCandlesColors,candles_colors)
            && tries<5)
        {
         //--- Zähler der CopyFromSymbolToBuffers() Funktionsaufrufe
         tries++;
       ;}
      //--- Setzen wir den Zähler der Ticks auf Null
      ticks=0;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Füllt die angegebene Kerze                                       |
//+------------------------------------------------------------------+
bool CopyFromSymbolToBuffers(string name,
                             int total,
                             int plot_index,
                             double &buff1[],
```

© 2000-2025, MetaQuotes Ltd.

```
                              double &buff2[],
                              double &buff3[],
                              double &buff4[],
                              double &col_buffer[],
                              int    cndl_colors
                              )
  {
//--- In die Array rates[] kopieren wir Preise Open, High, Low und Close
   MqlRates rates[];
//--- Zähler der Versuche
   int attempts=0;
//--- Wieviel kopiert
   int copied=0;
//--- Machen wir 25 Versuche, um die Zeitreihe für das gewünschten Symbol zu erhalten
   while(attempts<25 && (copied=CopyRates(name,_Period,0,bars,rates))<0)
     {
      Sleep(100);
      attempts++;
      if(messages) PrintFormat("%s CopyRates(%s) attempts=%d",__FUNCTION__,name,attemp
     }
//--- Wenn es gelang nicht eine ausreichende Anzahl von Balken zu kopieren
   if(copied!=bars)
     {
      //--- erstellen wir einen Nachrichtenstring
      string comm=StringFormat("Für Symbol %s konnte nur %d Balken von %d gefragten Ba
                               name,
                               copied,
                               bars
                               );
      //--- Zeigen wir einer Nachricht in einer Kommentar auf dem Haupt-Chart-Fenster
      Comment(comm);
      //--- Anzeige von Informationen
      if(messages) Print(comm);
      return(false);
     }
   else
     {
      //--- Einstellen wir Anzeige des Symbols
      PlotIndexSetString(plot_index,PLOT_LABEL,name+" Open;"+name+" High;"+name+" Low;
      IndicatorSetString(INDICATOR_SHORTNAME,"DRAW_COLOR_CANDLES("+symbol+")");
     }
//--- Initialisieren wir den Puffer leere Werte
   ArrayInitialize(buff1,0.0);
   ArrayInitialize(buff2,0.0);
   ArrayInitialize(buff3,0.0);
   ArrayInitialize(buff4,0.0);
//--- Kopieren wir Preise in den Puffer bei jedem Tick
   for(int i=0;i<copied;i++)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
 //--- Berechnen wir den entsprechenden Index für den Puffer
      int buffer_index=total-copied+i;
 //--- Schreiben wir die Preise in Puffer
      buff1[buffer_index]=rates[i].open;
      buff2[buffer_index]=rates[i].high;
      buff3[buffer_index]=rates[i].low;
      buff4[buffer_index]=rates[i].close;
      //--- Geben wir die Kerzefarbe an
      int color_index=i%cndl_colors;
      col_buffer[buffer_index]=color_index;
     }
   return(true);
  }
//+------------------------------------------------------------------+
//| Gibt ein zufälliges Symbol aus dem Market Watch zurück           |
//+------------------------------------------------------------------+
string GetRandomSymbolName()
  {
//--- Anzahl der in dem "Market Watch" Fenster angezeigten Symbole
   int symbols=SymbolsTotal(true);
//--- Position des Symbols in der Liste - eine Zufallszahl zwischen 0 und symbols
   int number=MathRand()%symbols;
//--- Gibt den Namen des Symbols an der angegebenen Position zurück
   return SymbolName(number,true);
  }
//+------------------------------------------------------------------+
//| Ändert die Farbe der Segmente der Kerzen                         |
//+------------------------------------------------------------------+
void  ChangeColors(color  &cols[],int plot_colors)
  {
//--- Anzahl der Farben
   int size=ArraySize(cols);
//---
   string comm=ChartGetString(0,CHART_COMMENT)+"\r\n\r\n";

//--- Für jede Farbeindex definieren wir eine neue Farbe zufällig
   for(int plot_color_ind=0;plot_color_ind<plot_colors;plot_color_ind++)
     {
      //--- erhalten wir eine Zufallszahl
      int number=MathRand();
      //--- Erhalten wir den Index in Array col[] als den Rest der ganzzahligen Divisi
      int i=number%size;
      //--- Geben wir die Farbe für jeden Index als Eigenschaft PLOT_LINE_COLOR an
      PlotIndexSetInteger(0,                    //  Nummer der Grafikstile
                          PLOT_LINE_COLOR,      //  Property Identifier
                          plot_color_ind,       //  Farbeindex, in dem wir die Farbe s
                          cols[i]);             //  neue Farbe
      //--- schreiben wir die Farbe
      comm=comm+StringFormat("CandleColorIndex[%d]=%s \r\n",plot_color_ind,ColorToStri
```

© 2000-2025, MetaQuotes Ltd.

```
      ChartSetString(0,CHART_COMMENT,comm);
     }
//---
  }
//+------------------------------------------------------------------+
//| Ändert das Aussehen der Kerzen                                   |
//+------------------------------------------------------------------+
void ChangeLineAppearance()
  {
//--- String um Informationen über die Eigenschaften der Kerzen zu erstellen
   string comm="";
//--- Schreiben wir den Namen des Symbols
   comm="\r\n"+symbol+comm;
//--- Zeigen wir Informationen auf dem Chart durch den Kommentar
   Comment(comm);
  }
```

© 2000-2025, MetaQuotes Ltd.

Zusammenhang zwischen Eigenschaften des Indikators und entsprechenden Funktionen

Jeder Benutzeranzeiger hat viele Eigenschaften, einige von ihnen sind obligatorisch und befinden sich immer am Anfang der Beschreibung. Das sind Eigenschaften:

- Angeben des Fensters für die Darstellung des Anzeigers - indicator_separate_window oder indicator_chart_window;
- Anzahl derIndikatorpuffer - indicator_buffers;
- Anzahl der graphischen Darstellungen des Anzeigers - indicator_plots.

Aber es gibt aich andere Eigenschaften, die sowohl durch Befehlsanweisungen des Preprozessors, als auch durch die Funktionen für Erzeugung des Benutzeranzeigers vorgegeben werden können. In der Tabelle werden diese Eigenschaften und ihnen entsprechende Funktionen beschrieben.

Befehlsanweisungen für Eigenschaften des Subfensters des Anzeigers	**Funktionen des Typs IndicatorSet...()**	**Beschreibung der eingestellten Eigenschaft des Subfensters**
indicator_height	IndicatorSetInteger (INDICATOR_INDICATOR_HEIGHT, nHeight)	Der Wert der festen Höhe des Subfensters
indicator_minimum	IndicatorSetDouble (INDICATOR_MINIMUM, dMaxValue)	Minimaler Wert der vertikalen Achse
indicator_maximum	IndicatorSetDouble (INDICATOR_MAXIMUM, dMinValue)	Maximaler Wet der vertikalen Achse
indicator_levelN	IndicatorSetDouble (INDICATOR_LEVELVALUE, N-1, nLevelValue)	Wert der vertikalen Achse für N Level
keine Befehlsanweisung des Preprozessors	IndicatorSetString (INDICATOR_LEVELTEXT, N-1, sLevelName)	Name des dargestellten Levels
indicator_levelcolor	IndicatorSetInteger (INDICATOR_LEVELCOLOR, N-1, nLevelColor)	Farbe für die Darstellung des N Level
indicator_levelwidth	IndicatorSetInteger (INDICATOR_LEVELWIDTH, N-1, nLevelWidth)	Breite der Linie für die Darstellung des n Levels
indicator_levelstyle	IndicatorSetInteger (INDICATOR_LEVELSTYLE, N-1, nLevelStyle)	Stil der Linie für die Darstellung des N Levels
Befehlsanweisungen für graphische Darstellungen	**Funktionen des Typs PlotIndexSet...()**	**Beschreibung der eingestellten Eigenschaft für graphische Darstellung**

© 2000-2025, MetaQuotes Ltd.

Befehlsanweisungen für Eigenschaften des Subfensters des Anzeigers	Funktionen des Typs IndicatorSet...()	Beschreibung der eingestellten Eigenschaft des Subfensters
indicator_labelN	PlotIndexSetString(N-1,PLOT_LABEL,sLabel)	Kurze Benennung für die graphische Darstellung mit Nummer N. Gezeigt im Fenster DataWindow und im pop-up tooltip beim bei Cursor-Beswegung darauf
indicator_colorN	PlotIndexSetInteger(N-1, PLOT_LINE_COLOR, nColor)	Liniefarbe für graphische Darstellung mit Nummer N
indicator_styleN	PlotIndexSetInteger(N-1, PLOT_LINE_STYLE, nType)	Liniestil für für graphische Darstellung mit Nummer N
indicator_typeN	PlotIndexSetInteger(N-1, PLOT_DRAW_TYPE, nType)	Linietyp für graphische Darstellung mit Nummer N
indicator_widthN	PlotIndexSetInteger(N-1, PLOT_LINE_WIDTH, nWidth)	Breite der Linie für graphische Darstellung mit Nummer N
Allgemeine Eigenschaften des Anzeigers	**Funktionen des Typs IndicatorSet...()**	**Beschreibung**
keine Befehlsanweisung des Preprozessors	IndicatorSetString (INDICATOR_SHORTNAME, sShortName)	Stellt passenden kurzen Namen des Anzeigers, der in der Liste der Anzeiger dargestellt werden wird (im Terminal wird durch **Ctrl+I** aufgerufen)).
keine Befehlsanweisung des Preprozessors	IndicatorSetInteger (INDICATOR_DIGITS, nDigits)	Stellt die angeforderte Genauigkeit für Darstelung der Anzeigerwerte - Anzahl der Dezimalzeichen
keine Befehlsanweisung des Preprozessors	IndicatorSetInteger (INDICATOR_LEVELS, nLevels)	Stellt die Anzahl der Levels im Fenster des Anzeigers ein
indicator_applied_price	Keine Funktion, Eigenschaft wird nur durch die Anweisung des Prepozessors eingestellt)	Default-Preistyp, der für Berechnung der Anzeigerwerte vewendet wird. Angegeben nur bei der Notwendigkeit, die Funktion OnCalculate()des ersten Typs zu verwenden. Wert der Eigenschaft kann auch vom Dialog der Anzeigereigenschaften in "Parameter" - "Anwenden zu" vorgegeben wereden.

Es muss bemerkt werden, dass Numerierung der Levels und graphischer Darstellungen in Termini des Preprozessors mit eins anfaengt, während die Numerierung derselben Eigenschaften bei der

© 2000-2025, MetaQuotes Ltd.

Verwendung der Funktionen mit Null anfaengt, d.h. der angegebene Wert muss um 1 kleiner sein als für #property.

Es gibt einige Befehlsanweisungen, für denen keine entsprechenden Funktionen existieren:

Befehlsanweisung	Beschreibung
indicator_chart_window	Anzeiger wird im Hauptfenster dargestellt
indicator_separate_window	Anzeiger wird im selbststaendigen Subfenster dargestellt
indicator_buffers	Gibt die Anzahl der erforderlichen Anzeigerbuffer an
indicator_plots	Gibt die Anzahl der graphischen Darstellungen im Anzeiger an

© 2000-2025, MetaQuotes Ltd.

SetIndexBuffer

Verbindet den angegebenen Indikatorpuffer mit einem eindimensionalen dynamischen Feld des Typs double.

```
bool  SetIndexBuffer(
   int                   index,          // Index des Puffers
   double                buffer[],       // Feld
   ENUM_INDEXBUFFER_TYPE data_type       // was wird aufbewahren
   );
```

Parameter

index

[in] Nummer des Indikatorpuffers. Numerierung faengt mit 0 an. Nummer muss kleiner sein als der Wert, der in #property indicator_buffers erklärt wurde.

buffer[]

[in] Feld, das im Programm des Benutzerindikators erklärt wird.

data_type

[in] Typ der Daten, die im Indikatorfeld aufbewahren werden. Default-Wert ist INDICATOR_DATA (Werte des berechneten Indikators). Kann auch den Wert INDICATOR_COLOR_INDEX annehmen, dann wird der vorliegende Puffer für Aufbewahrung der Farbenindexe für den vorherigen Indikatorpuffer. Man kann bis 64 Farben in der Zeile #property indicator_colorN vorgeben. Wert INDICATOR_CALCULATIONS bedeutet dass der vorliegende Puffer für Zwischenberechnungen des Indikators verwendet wird und für Zeichnen nicht bestimmt ist.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false.

Hinweis

Nach der Bindung wird das dynamische Feld *buffer[]* dieselbe Indizierung haben, wie in normalen Feldern, auch wenn für das gebundene Feld Indizierung wie in Timeserien vorher eingestellt wird. Wenn es notwendig ist, die Folge des Zugangs zu Elementen des Indikatorfeldes zu verändern, muss man die Funktion ArraySetAsSeries() nach der Bindung des Feldes durch die Funktion SetIndexBuffer() anwenden. Dabei muss man behalten, dass die Größe der dynamischen Felder, die als Indikatorpuffer durch die Funktion SetIndexBuffer() zugeordnet werden, nicht verändert werden muss. Für Indikatorpuffer werden alle Operationen der Größenveränderung vom ausführenden Untersystem des Terminals durchgeführt.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                TestCopyBuffer1.mq5 |
//|                        Copyright 2009, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot MA
#property indicator_label1  "MA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input bool                AsSeries=true;
input int                 period=15;
input ENUM_MA_METHOD      smootMode=MODE_EMA;
input ENUM_APPLIED_PRICE  price=PRICE_CLOSE;
input int                 shift=0;
//--- indicator buffers
double                    MABuffer[];
int                       ma_handle;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   if(AsSeries) ArraySetAsSeries(MABuffer,true);
   Print("Anzeigerpuffer ist eine Zeitreihe = ",ArrayGetAsSeries(MABuffer));
   SetIndexBuffer(0,MABuffer,INDICATOR_DATA);
   Print("Anzeigerpuffer nach SetIndexBuffer() ist eine Zeitreihe = ",
         ArrayGetAsSeries(MABuffer));

//--- verändern wir die Folge des Zugangs zu Elementen des Anzeigerpuffers
   ArraySetAsSeries(MABuffer,AsSeries);

   IndicatorSetString(INDICATOR_SHORTNAME,"MA("+period+")"+AsSeries);
//---
   ma_handle=iMA(Symbol(),0,period,shift,smootMode,price);
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- kopieren wir die Werte des gleitenden Mittelwertes in Puffer MABuffer
   int copied=CopyBuffer(ma_handle,0,0,rates_total,MABuffer);

   Print("MABuffer[0] = ",MABuffer[0]);// abhängig vom Wert AsSeries
                                       // bekommen wir entweder den aeltesten Wert
                                       // oder für die laufende unbeendete Bar

//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

Sehen Sie auch

Eigenschaften der Benutzerindikatoren , Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

IndicatorSetDouble

Gibt den Wert der entsprechenden Indikatoreigenschaft. Indikatoreigenschaft muss des Typs double sein. Es gibt 2 Varianten der Funktion.

Aufruf mit dem Identifikator der Eigenschaft

```
bool  IndicatorSetDouble(
   int     prop_id,           // Identifikator
   double  prop_value         // der eingestellte Wert
   );
```

Aufruf mit dem Identifikator und Modifikator der Eigenschaft

```
bool  IndicatorSetDouble(
   int     prop_id,           // Identifikator
   int     prop_modifier,     // Modifikator
   double  prop_value         // der eingestellte Wert
   )
```

Parameter

prop_id

[in] Identifikator der Indikatoreigenschaft. Wert kann einer der Enumerationswerte ENUM_CUSTOMIND_PROPERTY_DOUBLE sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Nur Leveleigenschaften fordern Modifikator an. Nummerierung der Stände beginnt bei 0, d.h. um eine Eigenschaft dem zweiten Stand anzugeben, setzen sie 1 (1 weniger als bei der Verwendung von Compiler-Direktiven).

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Bei der erfolgreichen Durchführung gibt true zurück, anderenfalls false.

Hinweis

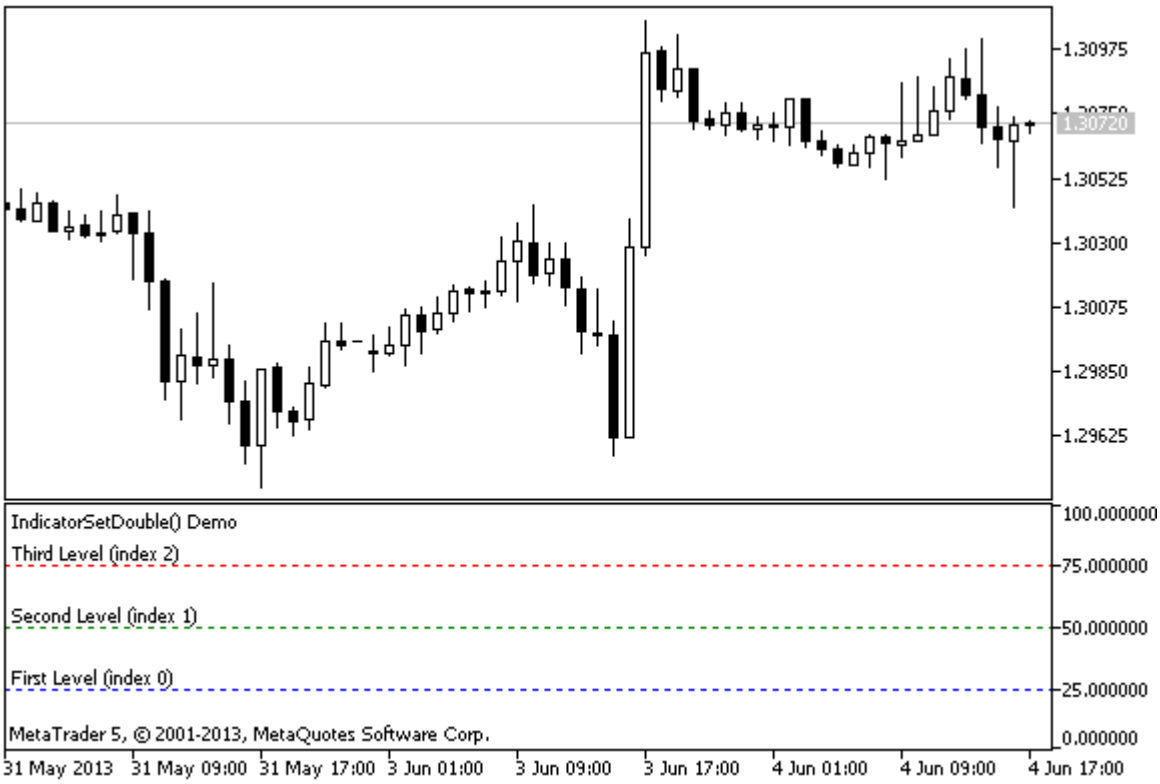
Nummerierung der Eigenschaften (Modifikatoren) durch Direktive #property beginnt bei 1 (eins), während die Funktionen verwenden Nummerierung bei 0 (Null). Bei die falsche Eingabe einer Nummer des Stands, kann Anzeige des Indikators verschieden sein.

Können Sie beispielsweise den Wert des ersten Stands für den Indikator in einem separaten Fenster auf zwei Methoden angeben:

- property indicator_level**1** 50 - 1 wird für Angabe des Stands verwendet,
- IndicatorSetDouble(INDICATOR_LEVELVALUE, **0**, 50) - 0 wird für Angabe des Stands verwendet.

Beispiel: umwendender Indikator, der den maximalen und minimalen Wert des Indikatorfensters und die Werte der Stände, an denen die horizontalen Linien liegen, ändert.

© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

```
//--- die maximalen und minimalen Werte für den Indikatorfenster setzen
#property indicator_minimum  0
#property indicator_maximum  100
//--- Anzeige der drei horizontalen Stände in einem separaten Indikatorfenster angeben
#property indicator_level1 25
#property indicator_level2 50
#property indicator_level3 75
//--- Breite der Stände angeben
#property indicator_levelwidth 1
//--- Stil der horizontalen Stände angeben
#property indicator_levelstyle STYLE_DOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Beschreibung der horizontalen Stände angeben
   IndicatorSetString(INDICATOR_LEVELTEXT,0,"First Level (index 0)");
   IndicatorSetString(INDICATOR_LEVELTEXT,1,"Second Level (index 1)");
   IndicatorSetString(INDICATOR_LEVELTEXT,2,"Third Level (index 2)");
//--- Kurzname des Indikators angeben
   IndicatorSetString(INDICATOR_SHORTNAME,"IndicatorSetDouble() Demo");
//--- Individuelle Farbe für die St'nde angeben
   IndicatorSetInteger(INDICATOR_LEVELCOLOR,0,clrBlue);
   IndicatorSetInteger(INDICATOR_LEVELCOLOR,1,clrGreen);
   IndicatorSetInteger(INDICATOR_LEVELCOLOR,2,clrRed);
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int tick_counter=0;
   static double level1=25,level2=50,level3=75;
   static double max=100,min=0, shift=100;
//--- Ticks berechnen
   tick_counter++;
//--- Umwälzen auf jedem zehnten Tick
   if(tick_counter%10==0)
     {
      //--- umwenden Zeichen für Standwerte
      level1=-level1;
      level2=-level2;
      level3=-level3;
      //--- umwenden Zeichen für maximale und minimale Werte
      max-=shift;
      min-=shift;
      //--- umwenden Wert von Verschiebung
      shift=-shift;
      //--- neue Werte der Stände angeben
```

© 2000-2025, MetaQuotes Ltd.

```
      IndicatorSetDouble(INDICATOR_LEVELVALUE,0,level1);
      IndicatorSetDouble(INDICATOR_LEVELVALUE,1,level2);
      IndicatorSetDouble(INDICATOR_LEVELVALUE,2,level3);
      //--- Neue maximale und minimale Werte für Indikatorfenster angeben
      Print("Set up max = ",max,",   min = ",min);
      IndicatorSetDouble(INDICATOR_MAXIMUM,max);
      IndicatorSetDouble(INDICATOR_MINIMUM,min);
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Sehen Sie auch

Stile der Indikator in den Beispielen, Zusammenhang zwischen Eigenschaften des Indikators und entsprechenden Funktionen, Zeichnungsstile

© 2000-2025, MetaQuotes Ltd.

IndicatorSetInteger

Gibt den Wert der entsprechenden Eigenschaft des Indikators vor. Eigenschaft des Indikators muss des Typ int oder color sein.Es gibt 2 Varianten der Funktion.

Aufruf mit Andeutung der Eigenschaft des Identifikatoren.

```
bool  IndicatorSetInteger(
   int  prop_id,           // Identifikator
   int  prop_value         // der eingestellte Wert
   );
```

Aufruf mit Andeutung des Identifikatoren und Modifikatoren der Eigenschaft

```
bool  IndicatorSetInteger(
   int  prop_id,           // Identifikator
   int  prop_modifier,     // Modifikator
   int  prop_value         // der eingestellte Wert
   )
```

Parameter

prop_id

[in] Identifikator der Indikatoreigenschaft. Wert kann einer der Enumerationswerte ENUM_CUSTOMIND_PROPERTY_INTEGER sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Nur Leveleigenschaften erfordern einen Modifikator.

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false.

Hinweis

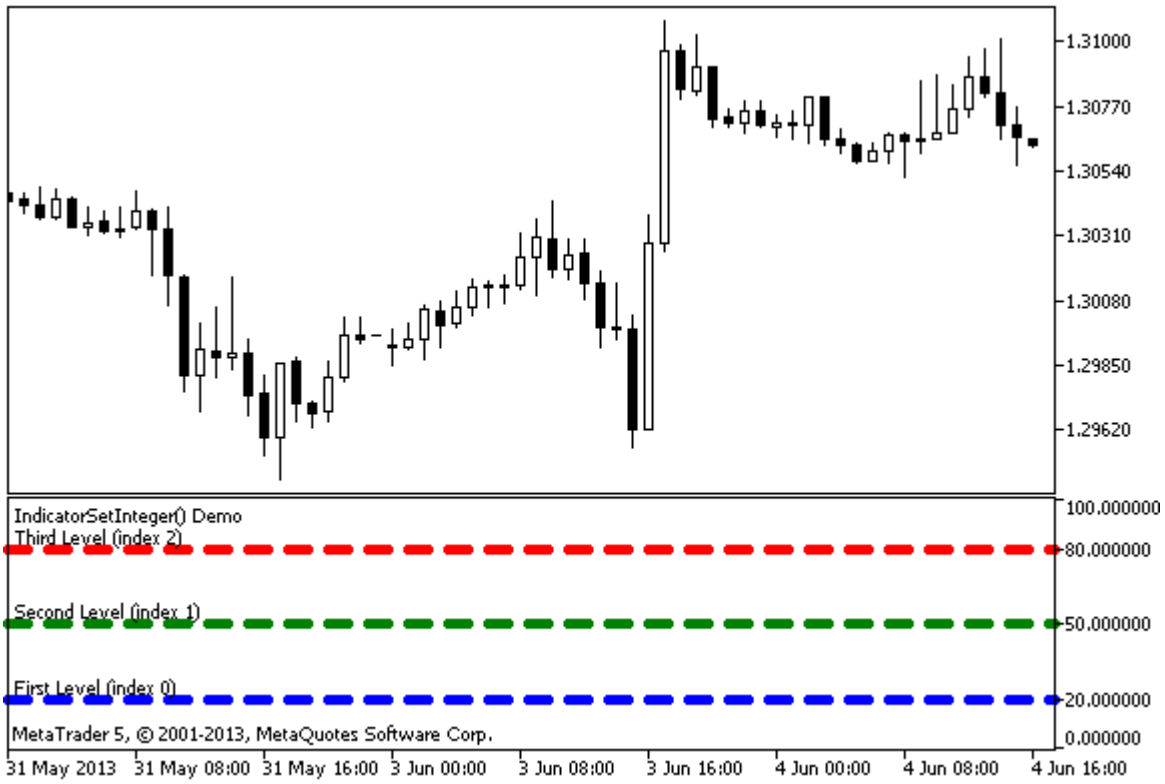
Nummerierung der Eigenschaften (Modifikatoren) durch Direktive #property beginnt bei 1 (eins), während die Funktion verwendet Nummerierung bei 0 (Null). Bei die falsche Eingabe einer Nummer des Stands, kann Anzeige des Indikators verschieden sein.

Zum Beispiel, um die Dicke der ersten horizontalen Ebene anzugeben, verwenden Sie Null-Index:

- IndicatorSetInteger(INDICATOR_LEVELWIDTH, **0**, 5) - 0 wird für Angabe des Linienbreite für ersten Stand verwendet.

Beispiel: der Indikator setzt die Farbe, den Stil und die Breite der horizontalen Linien des Indikators.

© 2000-2025, MetaQuotes Ltd.



© 2000-2025, MetaQuotes Ltd.

```
#property indicator_separate_window
#property indicator_minimum 0
#property indicator_maximum 100
//--- Anzeige der horizontalen Stände des Indikators in einem separaten Fenster
#property indicator_level1 20
#property indicator_level2 50
#property indicator_level3 80
//--- Breite der horizontalen Linien angeben
#property indicator_levelwidth 5
//--- Farbe der horizontalen Linien angeben=
#property indicator_levelcolor clrAliceBlue
//--- Stil der horizontalen Linien angeben
#property indicator_levelstyle STYLE_DOT
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Beschreibung der horizontalen Linien angeben
   IndicatorSetString(INDICATOR_LEVELTEXT,0,"First Level (index 0)");
   IndicatorSetString(INDICATOR_LEVELTEXT,1,"Second Level (index 1)");
   IndicatorSetString(INDICATOR_LEVELTEXT,2,"Third Level (index 2)");
//--- Kurzname für Indikator angeben
   IndicatorSetString(INDICATOR_SHORTNAME,"IndicatorSetInteger() Demo");
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   static int tick_counter=0;
//--- Ticks berechnen
   tick_counter++;
//--- Farben der horizontalen Linien abhängig vom Tick-Zähler
   ChangeLevelColor(0,tick_counter,3,6,10); // drei letzte Parameter ändern die Farbe
   ChangeLevelColor(1,tick_counter,3,6,8);
   ChangeLevelColor(2,tick_counter,4,7,9);
//--- Stil der horizontalen Linien ändern
   ChangeLevelStyle(0,tick_counter);
   ChangeLevelStyle(1,tick_counter+5);
   ChangeLevelStyle(2,tick_counter+15);
//--- Linienbreite wie der Rest der ganzzahligen Division der Anzahl von Ticks bei 5
   int width=tick_counter%5;
//--- Durch alle horizontalen Ebenen gehen und angeben
   for(int l=0;l<3;l++)
      IndicatorSetInteger(INDICATOR_LEVELWIDTH,l,width+1);
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Wir geben die Farbe der horizontale Linie im separaten Indikatorfenster an  |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void ChangeLevelColor(int level,      // Nummer der horizontale Linie
                      int tick_number,// die Dividende, um einen Rest der Division zu
                      int f_trigger,  // erster Teiler von Farbschalter
                      int s_trigger,  // zweiter Teiler von Farbschalter
                      int t_trigger)  // dritter Teiler von Farbschalter
  {
   static color colors[3]={clrRed,clrBlue,clrGreen};
//--- Farbindex aus Array colors[]
   int index=-1;
//--- Berechnung der Farbnummer in Array colors[] um die horizontale Linie zu malen
   if(tick_number%f_trigger==0)
      index=0;   // wenn die Zahl tick_number ist teilbar durch f_trigger
   if(tick_number%s_trigger==0)
      index=1;   // wenn die Zahl tick_number ist teilbar durch s_trigger
   if(tick_number%t_trigger==0)
      index=2;   // wenn die Zahl tick_number ist teilbar durch t_trigger
//--- wenn die Farbe definiert ist, setzen wir sie
   if(index!=-1)
      IndicatorSetInteger(INDICATOR_LEVELCOLOR,level,colors[index]);
//---
  }
//+------------------------------------------------------------------+
//| Stil der horizontale Linie im separaten Indikatorfenster angeben |
//+------------------------------------------------------------------+
void ChangeLevelStyle(int level,     // Nummer der horizontale Linie
                      int tick_number// Zahl um Rest der Division zu erhalten)
                      )
  {
//--- Array um Stile zu speichern
   static ENUM_LINE_STYLE styles[5]=
     {STYLE_SOLID,STYLE_DASH,STYLE_DOT,STYLE_DASHDOT,STYLE_DASHDOTDOT};
//--- Stilindex aus Array styles[]
   int index=-1;
//--- Berechnung der Nummer aus Array styles[] um den Stil der horizontalen Linie zu s
   if(tick_number%50==0)
      index=5;   // wenn die Zahl tick_number ist durch 50 teilbar, dann Stil STYLE_DA
   if(tick_number%40==0)
      index=4;   // ... Stil STYLE_DASHDOT
   if(tick_number%30==0)
      index=3;   // ... STYLE_DOT
   if(tick_number%20==0)
      index=2;   // ... STYLE_DASH
   if(tick_number%10==0)
      index=1;   // ... STYLE_SOLID
//--- wenn Stil definiert ist, setzen wir ihn
   if(index!=-1)
      IndicatorSetInteger(INDICATOR_LEVELSTYLE,level,styles[index]);
  }
```

Sehen Sie auch

Eigenschaften der Benutzerobjekte, Programmeigenschaften (#property), Zeichnungsstile

© 2000-2025, MetaQuotes Ltd.

IndicatorSetString

Gibt den Wert der entsprechenden Eigenschaft des Indikators vor. Eigenschaft des Indikators muss des Typs string sein. Es gibt 2 Varianten der Funktion.

Aufruf mit Andeutung des Identifikatoren der Eigenschaft.

```
bool  IndicatorSetString(
   int     prop_id,           // Identifikator
   string  prop_value         // der eingestellte Wert
   );
```

Aufruf mit Andeutung des Identifikatoren und Modifikatoren der Eigenschaft.

```
bool  IndicatorSetString(
   int     prop_id,           // Identifikator
   int     prop_modifier,     // Modifikator
   string  prop_value         // der eingestellte Wert
   )
```

Parameter

prop_id

[in] Identifikator der Eigenschaft des Indikators. Wert kann einer der Enumerationswerte ENUM_CUSTOMIND_PROPERTY_STRING sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Nur Leveleigenschaften erfordern einen Modifikator.

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false.

Hinweis

Nummerierung der Eigenschaften (Modifikatoren) durch Direktive #property beginnt bei 1 (eins), während die Funktion verwendet Nummerierung bei 0 (Null). Bei die falsche Eingabe einer Nummer des Stands, kann Anzeige des Indikators verschieden sein.

Zum Beispiel, um die Beschreibung der ersten horizontalen Ebene anzugeben, verwenden Sie Null-Index:

- IndicatorSetString(INDICATOR_LEVELTEXT, **0**, "First Level") - 0 wird für Angabe des Textbeschreibung für ersten Stand verwendet.

Beispiel: der Indikator setzt Texte für horizontale Linien des Indikators.

© 2000-2025, MetaQuotes Ltd.




```
#property indicator_separate_window
#property indicator_minimum 0
#property indicator_maximum 100
//--- Anzeige der Indikatorebene in einem separaten Indikatorfenster angeben
#property indicator_level1 30
#property indicator_level2 50
#property indicator_level3 70
//--- Farbe der horizontale Linien angeben
#property indicator_levelcolor clrRed
//--- Stil der horizontale Linien angeben
#property indicator_levelstyle STYLE_SOLID
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Beschreibungen der horizontale Linien angeben
   IndicatorSetString(INDICATOR_LEVELTEXT,0,"First Level (index 0)");
   IndicatorSetString(INDICATOR_LEVELTEXT,1,"Second Level (index 1)");
   IndicatorSetString(INDICATOR_LEVELTEXT,2,"Third Level (index 2)");
//--- Kurzname des Indikators angeben
   IndicatorSetString(INDICATOR_SHORTNAME,"IndicatorSetString() Demo");
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---

//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

Sehen Sie auch

Eigenschaften der Benutzerobjekte, Programmeigenschaften (#property)

© 2000-2025, MetaQuotes Ltd.

PlotIndexSetDouble

Gibt den Wert der entsprechenden Eigenschaft der entsprechenden Indikatorlinie vor. Eigenschaft des Indikators muss des Typs double sein.

```
bool  PlotIndexSetDouble(
   int     plot_index,     // Index des graphischen Stils
   int     prop_id,        // Identifikator der Eigenschaft
   double  prop_value      // der eingestellte Wert
   );
```

Parameter

plot_index

[in] Index der graphischen Darstellung

prop_id

[in] Identifikator der Indikatoreigenschaft. Wert kann einer der Enumerationswerte ENUM_PLOT_PROPERTY_DOUBLE sein.

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Bei der erfolgreichen Durchführung gibt true zurück, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

PlotIndexSetInteger

Gibt den Wert der entsprechenden Eigenschaft des Indikators vor. Eigenschaft des Indikators muss des Typs int, char, bool oder color sein. Es gibt 2 Varianten der Funktion.

Aufruf mit Andeutung des Identifikatoren der Eigenschaft.

```
bool  PlotIndexSetInteger(
   int  plot_index,         // Index des graphischen Stils
   int  prop_id,            // Identifikator der Eigenschaft
   int  prop_value          // der eingestellte Wert
   );
```

Aufruf mit Andeutung des Identifikatoren und Modifikatoren der Eigenschaft.

```
bool  PlotIndexSetInteger(
   int  plot_index,         // Index des graphischen Stils
   int  prop_id,            // Identifikator der Eigenschaft
   int  prop_modifier,      // Modifikator der Eigenschaft
   int  prop_value          // der eingestellte Wert
   )
```

Parameter

plot_index

[in] Index der graphischen Darstellung

prop_id

[in] Identifikator der Indikatoreigenschaft. Wert kann einer der Enumerationswerte ENUM_PLOT_PROPERTY_INTEGER sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Nur Eigenschaften der Indexe erfordern einen Modifikator.

prop_value

[in] Wert der Eigenschaft.

Rueclgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false.

Beispiel: Indikator, der eine dreifarbige Linie zeichnet. Farbschema verändert sich wird jede 5 Ticks.

© 2000-2025, MetaQuotes Ltd.




```
#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   1
//---- plot ColorLine
#property indicator_label1  "ColorLine"
#property indicator_type1   DRAW_COLOR_LINE
#property indicator_color1  clrRed,clrGreen,clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  3
//--- indicator buffers
double         ColorLineBuffer[];
double         ColorBuffer[];
int            MA_handle;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,ColorLineBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,ColorBuffer,INDICATOR_COLOR_INDEX);
//--- get MA handle
   MA_handle=iMA(Symbol(),0,10,0,MODE_EMA,PRICE_CLOSE);
//---
  }
//+------------------------------------------------------------------+
//| get color index                                                  |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int getIndexOfColor(int i)
  {
   int j=i%300;
   if(j<100) return(0);// first index
   if(j<200) return(1);// second index
   return(2); // third index
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---
   static int ticks=0,modified=0;
   int limit;
//--- first calculation or number of bars was changed
   if(prev_calculated==0)
     {
      //--- copy values of MA into indicator buffer ColorLineBuffer
      int copied=CopyBuffer(MA_handle,0,0,rates_total,ColorLineBuffer);
      if(copied<=0) return(0);// copying failed - throw away
      //--- now set line color for every bar
      for(int i=0;i<rates_total;i++)
         ColorBuffer[i]=getIndexOfColor(i);
     }
   else
     {
      //--- copy values of MA into indicator buffer ColorLineBuffer
      int copied=CopyBuffer(MA_handle,0,0,rates_total,ColorLineBuffer);
      if(copied<=0) return(0);

      ticks++;// ticks counting
      if(ticks>=5)//it's time to change color scheme
        {
         ticks=0; // reset counter
         modified++; // counter of color changes
         if(modified>=3)modified=0;// reset counter
         ResetLastError();
         switch(modified)
           {
```

© 2000-2025, MetaQuotes Ltd.

```
            case 0:// first color scheme
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,0,clrRed);
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,1,clrBlue);
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,2,clrGreen);
               Print("Color scheme "+modified);
               break;
            case 1:// second color scheme
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,0,clrYellow);
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,1,clrPink);
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,2,clrLightSlateGray);
               Print("Color scheme "+modified);
               break;
            default:// third color scheme
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,0,clrLightGoldenrod);
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,1,clrOrchid);
               PlotIndexSetInteger(0,PLOT_LINE_COLOR,2,clrLimeGreen);
               Print("Color scheme "+modified);
           }
        }
      else
        {
         //--- set start position
         limit=prev_calculated-1;
         //--- now we set line color for every bar
         for(int i=limit;i<rates_total;i++)
            ColorBuffer[i]=getIndexOfColor(i);
        }
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

PlotIndexSetString

Gibt den Wert der entsprechenden Eigenschaft der entsprechenden Indikatorlinie zurück. Indikatoreigenschaft muss des Typs string sein.

```
bool  PlotIndexSetString(
   int     plot_index,     // Index des graphischen Stils
   int     prop_id,        // Identifikator der Eigenschaft
   string  prop_value      // der eingestellte Wert
   );
```

Parameter

plot_index

[in] Index der graphischen Darstellung

prop_id

[in] Identifikator der Indikatoreigenschaft. Wert kann einer der Enumerationswerte ENUM_PLOT_PROPERTY_STRING sein.

prop_value

[in] Eigenschaftswert.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

PlotIndexGetInteger

Gibt den Wert der entsprechenden Eigenschaft der entsprechenden Indikatorlinie zurück. Eigenschaft muss der Typen int, color, bool oder char sein. Es gibt 2 Varianten der Funktion.

Aufruf mit Andeutung des Identifikatoren der Eigenschaft.

```
int  PlotIndexGetInteger(
   int  plot_index,         // Index des graphischen Stils
   int  prop_id,            // Identifikator der Eigenschaft
   );
```

Aufruf mit Andeutung des Identifikatoren und Modifikatoren der Eigenschaft .

```
int  PlotIndexGetInteger(
   int  plot_index,         // Index des graphischen Stils
   int  prop_id,            // Identifikator der Eigenschaft
   int  prop_modifier       // Modifikator der Eigenschaft
   )
```

Parameter

plot_index

[in] Index der graphischen Darstellung

prop_id

[in] Identifikator der Indikatoreigenschaft. Wert kann einer der Enumerationswerte ENUM_PLOT_PROPERTY_INTEGER sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Nur Eigenschaften der Farbeneindexe erfordert einen Modifikator.

Hinweis

Funktion ist für Extrahieren der Zeichneneinstellungen der entsprechenden Indikatorlinie bestimmt. Funktion arbeitet mit der Funktion PlotIndexSetInteger für Kopieren der Indikatoreigenschaften von einer Linie in die andere.

Beispiel: Indikator, der Kerzen in Farben zeichnet abhängig von dem Wochentag. Farben für jeden Tag werden programmweise vorgegeben.

© 2000-2025, MetaQuotes Ltd.




```
#property indicator_separate_window
#property indicator_buffers 5
#property indicator_plots   1
//---- plot ColorCandles
#property indicator_label1  "ColorCandles"
#property indicator_type1   DRAW_COLOR_CANDLES
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- indicator buffers
double          OpenBuffer[];
double          HighBuffer[];
double          LowBuffer[];
double          CloseBuffer[];
double          ColorCandlesColors[];
color           ColorOfDay[6]={CLR_NONE,clrMediumSlateBlue,
                               clrDarkGoldenrod,clrForestGreen,clrBlueViolet,clrRed};
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,OpenBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,HighBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,LowBuffer,INDICATOR_DATA);
   SetIndexBuffer(3,CloseBuffer,INDICATOR_DATA);
   SetIndexBuffer(4,ColorCandlesColors,INDICATOR_COLOR_INDEX);
//--- set number of colors in color buffer
```

© 2000-2025, MetaQuotes Ltd.

```
   PlotIndexSetInteger(0,PLOT_COLOR_INDEXES,6);
//--- set colors for color buffer
   for(int i=1;i<6;i++)
      PlotIndexSetInteger(0,PLOT_LINE_COLOR,i,ColorOfDay[i]);
//--- set accuracy
   IndicatorSetInteger(INDICATOR_DIGITS,_Digits);
   printf("We have %u colors of days",PlotIndexGetInteger(0,PLOT_COLOR_INDEXES));
//---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//---
   int i;
   MqlDateTime t;
//----
   if(prev_calculated==0) i=0;
   else i=prev_calculated-1;
//----
   while(i<rates_total)
     {
      OpenBuffer[i]=open[i];
      HighBuffer[i]=high[i];
      LowBuffer[i]=low[i];
      CloseBuffer[i]=close[i];
      //--- set color for every candle
      TimeToStruct(time[i],t);
      ColorCandlesColors[i]=t.day_of_week;
      //---
      i++;
     }
//--- return value of prev_calculated for next call
   return(rates_total);
  }
```

© 2000-2025, MetaQuotes Ltd.

Graphische Objekte

Gruppe der Funktionen, die für Arbeit mit graphischen Objekten, die zum jeden angegebenen Chart gehören, bestimmt ist. Diese Funktionen können nicht in Anzeigern verwendet werden.

Die Funktionen, die die Eigenschaften von grafischen Objekten eingeben, und die Operationen der Erstellung ObjectCreate() und Bewegen ObjectMove() von Objekte auf dem Chart sind tatsächliche verwendet, um Befehle dem Chart zu senden. Der erfolgreiche Erfüllung dieser Funktionen wird der Befehl in die allgemeine Warteschlange der Chartereignisse platziert. Visuelle Veränderung der Eigenschaften von grafischen Objekten wird während der Verarbeitung der Ereigniswarteschlange von diesem Chart geändert.

Aus diesem Grund sollte man nicht erwarten, dass die graphische Objekten sofort nach dem Aufruf dieser Funktionen visuell aktualisiert wird. Im Allgemeinen werden die graphische Objekten durch das Terminal automatisch nach Änderung-Ereignisse aktualisiert - die Ankunft der neuen Quote, Änderung von Chartfenstergröße, etc.

Um graphischen Objekten zu aktualisieren, verwenden Sie ChartRedraw().

Funktion	Massnahme
ObjectCreate	Erzeugt das Objekt des vorgegebenen Typs auf dem vorgegebenen Chart
ObjectName	Gibt den Namen des Objekts des entsprechenden Typs im angegebenen Chart zurück (im angegebenen Fenster des Charts)
ObjectDelete	Entfernt das Objekt mit dem angegebenen Namen vom angegebenen Chart (vom angegebenen Subfenster des Charts)
ObjectsDeleteAll	Entfernt alle Objekte des angegebenen Typs vom angegebenen Chart (vom angegebenen Subfenster des Charts)
ObjectFind	Sucht Objekt mit dem angegebenen Identifikator nach dem Namen
ObjectGetTimeByValue	Gibt Zeitwert für den angegebenen Preiswert des Objekts zurück
ObjectGetValueByTime	Gibt Preiswert des Objekts für die angegebene Zeit zurück
ObjectMove	Verändert die Koordinaten des angegebenen Bezugspunktes des Objekts.
ObjectsTotal	Gibt die Anzahl der Objekte des angegebenen Typs im angegebenen Chart (angegebenen Subfenster des Charts) zurück.
ObjectGetDouble	Gibt den Wert des Typs double der entsprechenden Eigenschaft des Objekts zurück.
ObjectGetInteger	Gibt den ganzzahligen Wert der entsprechenden Eigenschaft des Objekts zurück.
ObjectGetString	Gibt den Wert des Typs string der entsprechenden Eigenschaft des Objekts
ObjectSetDouble	Stellt den wert der entsprechenden Eigenschaft des Objekts ein

© 2000-2025, MetaQuotes Ltd.

Funktion	Massnahme
ObjectSetInteger	Stellt den wert der entsprechenden Eigenschaft des Objekts ein
ObjectSetString	Stellt den wert der entsprechenden Eigenschaft des Objekts ein
TextSetFont	Setzt die Schrift für die Textausgabe durch Zeichnenmethoden (die Standardschriftart ist Arial 20)
TextOut	Gibt den Text in einem benutzerdefinierten Array (Puffer), der für die Erstellung von grafischen Ressource benutzt wird, aus
TextGetSize	Gibt die Breite und Höhe der Zeile mit den aktuellen Schrifteinstellungen zurück

Jedes graphische Objekt muss einen Namen haben, der unikal innerhalb eines Charts, sein muss einschliesslich seine Subfenster. Veranderung des Namens des graphischen Objekts formiert zwei Ereignisse: das erste Ereignis ist die Entfernung des Objekts mit dem alten Namen, das zweite Ereignis ist die Erzeugung des graphischen Objekts mit dem neuen Namen.

Nach Erzeugung des Objekt oder Modifikation der Objekteigenschaften ist es empfehlenswert, die Funktion ChartRedraw() aufzurufen, die die Anweisung gibt, den Chart (und alle darauf sichtbaren Objekte) zu zeichnen.

© 2000-2025, MetaQuotes Ltd.

ObjectCreate

Erzeugt ein Objekt mit dem angegebenen Namen, Typ und ursprünglichen Koordinaten im angegebenen Subfenster des Charts. Bei der Erzeugung kann man bis zu 30 Koordinaten angeben.

```
bool  ObjectCreate(
   long          chart_id,      // Identifikator des Charts
   string        name,          // Objektname
   ENUM_OBJECT   type,          // Objekttyp
   int           sub_window,    // Index des Fensters
   datetime      time1,         // Zeit des ersten Ankerpunktes
   double        price1,        // Preis des ersten Ankerpunktes
   ...
   datetime      timeN=0,       // Zeit des N-Ankerpunktes
   double        priceN=0,      // Preis des N-Ankerpunktes
   ...
   datetime      time30=0,      // Zeit des 30. Ankerpunktes
   double        price30=0      // Preis des 30. Ankerpunktes
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname. Name muss unikal im Rahmen eines Charts und seiner Subfenster sein.

type

[in] Objekttyp. Wert kann der einer der Enumerationswerte ENUM_OBJECT sein.

sub_window

[in] Nummer des Subfensters des Charts. Das angegebene Subfenster muss existieren, anderenfalls gibt die Funktion false zurück. .

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

timeN=0

[in] Zeitkoordinate des N-Ankerpunktes.

priceN=0

[in] Preiskoordinate des N-Ankerpunktes.

time30=0

[in] Zeikoordinate des 30. Ankerpunktes.

price30=0

[in] Preiskoordinate des 30. Ankerpunktes.

© 2000-2025, MetaQuotes Ltd.

Rückgabewert

Gibt true zurück, wenn ein Befehl zur Warteschlange des angegebenen Charts erfolgreich hinzugefügt wurde, andernfalls false. Wenn das Objekt bereits früher erstellt wurde, wird versucht, seine Koordinaten zu ändern.

Hinweis

Beim Aufruf von ObjectCreate() wird immer ein asynchroner Aufruf verwendet, deswegen gibt die Funktion nur das Ergebnis des Hinzufügens eines Befehls zur Warteschlange des Charts zurück. In diesem Fall bedeutet true nur, dass der Befehl zur Warteschlange erfolgreich hinzugefügt wurde, das Ergebnis der Ausführung ist unbekannt.

Für die Überprüfung des Ergebnisses der Ausführung kann man die Funktion ObjectFind() oder andere Funktionen verwenden, die die Eigenschaften des Objektes abrufen, z.B. vom Typ ObjectGetXXX. Dabei muss man beachten, dass solche Funktionen am Ende der Warteschlange hinzugefügt werden und auf das Ergebnis der Ausführung warten (weil sie synchrone Aufrufe sind). D.h. sie können viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Der Name des grafischen Objekts darf nicht länger als 63 Symbole sein.

Die Nummerierung der Subfenster des Chart (wenn es auf dem Chart Subfenster mit Indikatoren gibt) fängt mit 1 an. Hauptfenster eines Charts gibt es immer und hat Index 0.

Die große Anzahl von Ankerpunkten (bis zu 30) ist für die weitere Verwendung vorgesehen. Die Einschränkung auf 30 Ankerpunkte ist damit verbunden, dass die Anzahl der Parameter beim Funktionsaufruf nicht mehr als 64 betragen darf.

Bei der Umbenennung des graphischen Objekts werden gleichzeitig zwei Ereignisse formiert, die im Experten oder im Indikator durch die Funktion OnChartEvent() verarbeitet werden können:

- Entfernung eines Objektes mit dem alten Namen;
- Erzeugung eines graphischen Objektes mit dem neuen Namen.

Für die Erzeugung jedes der Objekttypen ist es erforderlich, eine bestimmte Anzahl der Bezugspunkte anzugeben:

Identifikator	Beschreibung	Bezugspunkte
OBJ_VLINE	Senkrechte Linie	Ein Ankerpunkt. Tatsächlich wird nur die Koordinate der Zeitachse entlang verwendet.
OBJ_HLINE	Horizontale Linie	Ein Ankerpunkt. Tatsächlich wird nur die Koordinate der Preisachse entlang verwendet.
OBJ_TREND	Trendlinie	Zwei Ankerpunkte.
OBJ_TRENDBYANGLE	Trendlinie nach Winkel	Zwei Ankerpunkte.
OBJ_CYCLES	Zykluslinien	Zwei Ankerpunkte.

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Bezugspunkte
OBJ_ARROWED_LINE	Objekt "Linie mit einem Pfeil"	Zwei Ankerpunkte.
OBJ_CHANNEL	Abstandsgleicher Kanal	Drei Ankerpunkte.
OBJ_STDDEVCHANNEL	Kanal der Standardabweichung	Zwei Ankerpunkte.
OBJ_REGRESSION	Kanal der linearen Regression	Zwei Ankerpunkte.
OBJ_PITCHFORK	Andrews' Pitchfork	Drei Ankerpunkte.
OBJ_GANNLINE	Gannlinie	Zwei Ankerpunkte.
OBJ_GANNFAN	Gannfan	Zwei Ankerpunkte.
OBJ_GANNGRID	Ganngrid	Zwei Ankerpunkte.
OBJ_FIBO	Fibonacci Retracement	Zwei Ankerpunkte.
OBJ_FIBOTIMES	Zeitzonen Fibonacci	Zwei Ankerpunkte.
OBJ_FIBOFAN	Fibonacci Fan	Zwei Ankerpunkte.
OBJ_FIBOARC	Fibonacci Arcs	Zwei Ankerpunkte.
OBJ_FIBOCHANNEL	Kanal Fibonacci	Drei Ankerpunkte.
OBJ_EXPANSION	Erweiterung Fibonacci	Drei Ankerpunkte.
OBJ_ELLIOTWAVE5	Elliott Motive Wave	Fünf Ankerpunkte.
OBJ_ELLIOTWAVE3	Elliott Correction Wave	Drei Ankerpunkte.
OBJ_RECTANGLE	Rechteck	Zwei Ankerpunkte.
OBJ_TRIANGLE	Dreieck	Drei Ankerpunkte.
OBJ_ELLIPSE	Ellipse	Drei Ankerpunkte.
OBJ_ARROW_THUMB_UP	Zeichen "Gut" (Daumen oben)	Ein Ankerpunkt.
OBJ_ARROW_THUMB_DOWN	Zeichen "Schlecht" (Daumen unten)	Ein Ankerpunkt.
OBJ_ARROW_UP	Zeichen "Weiche oben"	Ein Ankerpunkt.
OBJ_ARROW_DOWN	Zeichen "Weich unten"	Ein Ankerpunkt.
OBJ_ARROW_STOP	Zeichen "Stop"	Ein Ankerpunkt.
OBJ_ARROW_CHECK	Zeichen "Kontrollmarke"	Ein Ankerpunkt.
OBJ_ARROW_LEFT_PRICE	linke Preismarke	Ein Ankerpunkt.
OBJ_ARROW_RIGHT_PRICE	Rechte Preismarke	Ein Ankerpunkt.

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Bezugspunkte
OBJ_ARROW_BUY	Zeichen "Buy"	Ein Ankerpunkt.
OBJ_ARROW_SELL	Zeichen "Sell"	Ein Ankerpunkt.
OBJ_ARROW	Objekt "Weiche"	Ein Ankerpunkt.
OBJ_TEXT	Objekt "Text"	Ein Ankerpunkt.
OBJ_LABEL	Objekt "Textmarke"	Die Position wird durch Eigenschaften OBJPROP_XDISTANCE und OBJPROP_YDISTANCE angegeben.
OBJ_BUTTON	Objekt "Schaltfläche"	Die Position wird durch Eigenschaften OBJPROP_XDISTANCE und OBJPROP_YDISTANCE angegeben.
OBJ_CHART	Objekt "Chart"	Die Position wird durch Eigenschaften OBJPROP_XDISTANCE und OBJPROP_YDISTANCE angegeben.
OBJ_BITMAP	Objekt "Bild"	Ein Ankerpunkt.
OBJ_BITMAP_LABEL	Objekt "graphische Marke"	Die Position wird durch Eigenschaften OBJPROP_XDISTANCE und OBJPROP_YDISTANCE angegeben.
OBJ_EDIT	Objekt "Eingabefeld"	Die Position wird durch Eigenschaften OBJPROP_XDISTANCE und OBJPROP_YDISTANCE angegeben.
OBJ_EVENT	Objekt "Ereignis", das einem Ereignis in der wirtschaftlichen Kalender entspricht	Ein Ankerpunkt. Tatsächlich wird nur die Koordinate der Zeitachse entlang verwendet.
OBJ_RECTANGLE_LABEL	Objekt "Rechteckige Marke" um eigene grafische Benutzeroberfläche zu entwerten.	Die Position wird durch Eigenschaften OBJPROP_XDISTANCE und OBJPROP_YDISTANCE angegeben.

© 2000-2025, MetaQuotes Ltd.

ObjectName

Gibt den Namen des entsprechenden Objekts im angegebenen Chart, angegebenen Subfenster des angegebenen Chart des angegebenen Typs zurück.

```
string  ObjectName(
   long   chart_id,           // Identifikator des Charts
   int    pos,                // Nummer in der Liste der Objekte
   int    sub_window=-1,      // Nummer des Fensters
   int    type=-1             // Typ des Objekts
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

pos

[in] Ordnungsnummer des Objekts laut dem angegebenen Filter nach Nummer des Subfensters und Typ.

sub_window=-1

[in] Nummer des Subfensters des Charts, -1 bedeutet alle Subfenster des Charts einschliesslich Hauptfenster.

type=-1

[in] Typ des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT sein. -1 bedeutet alle Typen.

Rückgabewert

Name des Objekts im Erfolgsfall.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Bei Umbenennung des graphischen Objekts werden gleichzeitig zwei Ereignisse gebildet, die im Expert oder im Anzeiger durch die Funktion OnChartEvent() verarbeitet werden können:

- Ereigniss der Entfernung des Objekts mit dem alten Namen;
- Ereignis der Erzeugung des graphischen Objekts mit dem neuen Namen.

© 2000-2025, MetaQuotes Ltd.

ObjectDelete

Entfernt das Objekt mit dem angegebenen Namen des angegebenen Charts.

```
bool  ObjectDelete(
   long    chart_id,      // Identifikator des Charts
   string  name           // Name des Objekts
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Name des entfernten Objekts.

Rückgabewert

Gibt true zurück, wenn ein Befehl zur Warteschlange des angegebenen Charts erfolgreich hinzugefügt wurde, andernfalls false.

Hinweis

Beim Aufruf von ObjectDelete() wird immer ein asynchroner Aufruf verwendet, deswegen gibt die Funktion nur das Ergebnis des Hinzufügens eines Befehls zur Warteschlange des Charts zurück. In diesem Fall bedeutet true nur, dass der Befehl zur Warteschlange erfolgreich hinzugefügt wurde, das Ergebnis der Ausführung ist unbekannt.

Für die Überprüfung des Ergebnisses der Ausführung kann man die Funktion ObjectFind() oder andere Funktionen verwenden, die die Eigenschaften des Objektes abrufen, z.B. vom Typ ObjectGetXXX. Dabei muss man beachten, dass solche Funktionen am Ende der Warteschlange hinzugefügt werden und auf das Ergebnis der Ausführung warten (weil sie synchrone Aufrufe sind). D.h. sie können viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Bei der Umbenennung des graphischen Objekts werden gleichzeitig zwei Ereignisse gebildet, die im Expert oder im Anzeiger durch die Funktion OnChartEvent() verarbeitet werden können:

- Ereigniss der Entfernung des Objekts mit dem alten Namen;
- Ereignis der Erzeugung des graphischen Objekts mit dem neuen Namen.

© 2000-2025, MetaQuotes Ltd.

ObjectsDeleteAll

Löscht alle Objekte des angegebenen Typs im angegebenen Chart, angegebenen Unterfenster des angegebenen Charts.

```
int  ObjectsDeleteAll(
   long  chart_id,           // Identifikator des Charts
   int   sub_window=-1,      // Index des Fensters
   int   type=-1             // Typ des Objekts für Entfernung
   );
```

Löscht alle Objekte des angegebenen Typs mit dem angegebenen Präfix im Objektnamen im Unterfenster des Charts.

```
int  ObjectsDeleteAll(
   long           chart_id,   // Chart ID
   const string     prefix,   // Präfix im Objektnamen
   int        sub_window=-1,  // Fensterindex
   int       object_type=-1   // Objektyp
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

prefix

[in] Das Präfix in der Objektenamen die Sie löschen möchten. Das Präfix kann als 'name' oder 'name*' angegeben werden - beide Optionen sind gleich. Wenn das Präfix ist ein leerer String, werden alle Objekte mit einem beliebigen Namen gelöscht werden.

sub_window=-1

[in] Nummer des Subfensters des Charts. 0 bedeutet das Hauptfenster des Charts, -1 bedeutet alle Subfenster des Charts, einschliesslich Hauptfenster.

type=-1

[in] Typ des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT sein. -1 bedeutet all Typen.

Rückgabewert

Gibt die Anzahl der entfernten Objekte zurück. Für die Erhaltung der weiteren Information über den Fehler muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

© 2000-2025, MetaQuotes Ltd.

ObjectFind

Sucht ein Objekt mit dem angegebenen Namen auf dem Chart mit dem angegebenen Identifikatoren.

```
int  ObjectFind(
   long    chart_id,      // Identifikator des Charts
   string  name           // Name des Objekts
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Name de gesuchten Objekts.

Rückgabewert

Im Erfolgsfall gibt die Funktion Nummer des Subfensters zurück (0bedeutet das Hauptfenster des Chart), in dem sich das gefundene Objekt befindet. Wenn das Objekt nicht gefunden ist, gibt die Funktion eine negative Zahl zurück. Für die Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Bei der Umbenennung des graphischen Objekts werden gleichzeitig zwei Ereignisse gebildet, die im Expert oder Anzeiger durch die Funktion OnChartEvent() verarbeitet werden können:

- Entfernung des Objekts mit dem alten Namen;
- Erzeugung des graphischen Objekts mit dem Namen.

© 2000-2025, MetaQuotes Ltd.

ObjectGetTimeByValue

Gibt Zeitwert für den angegebenen Preiswert des angegebenen Objekts.

```
datetime  ObjectGetTimeByValue(
   long    chart_id,      // Identifikator des Charts
   string  name,          // Name des Objekts
   double  value,         // Preis
   int     line_id        // linie
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Name des Objekts.

value

[in] Preiswert.

line_id

[in] Identifikator der Linie.

Rückgabewert

Zeitwert für den angegebenen Preiswert des angegebenen Objekts.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Da das Objekt in einer Preiskoordinate mehrere Werte haben kann, muss man Nummer der Linie angeben. Diese Funktion ist nur für folgende Objekte anwendbar:

- Trendlinie (OBJ_TREND)
- Trendlinie nach der Ecke (OBJ_TRENDBYANGLE)
- Gann Linie (OBJ_GANNLINE)
- Abstandsgleicher Kanal (OBJ_CHANNEL) - 2 Linien
- Kanal auf der linealen Regression (OBJ_REGRESSION) - 3 Linien
- Kanal der Standardabweichung (OBJ_STDDEVCHANNEL) - 3 Linien
- Linie mit der Weiche (OBJ_ARROWED_LINE)

Sehen Sie auch

Objekttypen

© 2000-2025, MetaQuotes Ltd.

ObjectGetValueByTime

Gibt den Preizwert für die angegebene Zeit des angegebenen Objekts.

```
double  ObjectGetValueByTime(
   long      chart_id,     // Identifikator des Charts
   string    name,         // Objektname
   datetime  time,         // Zeit
   int       line_id       // Linie
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

time

[in] Zeitwert.

line_id

[in] Identifikator der Linie.

Rückgabewert

Preiswert für die angegebene Zeit des angegebenen Objekts.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Da das Objekt in einer Preiskoordinate mehrere Werte haben kann, muss man die Nummer der Linie angeben. Diese Funktion ist nur für folgende Objekte anwendbar:

- Trendlinie (OBJ_TREND)
- Trendlinie nach der Ecke (OBJ_TRENDBYANGLE)
- Gann Linie (OBJ_GANNLINE)
- Abstandsgleicher Kanal (OBJ_CHANNEL) - 2 Linien
- Kanal auf der linealen Regression (OBJ_REGRESSION) - 3 Linien
- Kanal der Standardabweichnug (OBJ_STDDEVCHANNEL) - 3 Linien
- Linie mit der Weiche (OBJ_ARROWED_LINE)

Sehen Sie auch

Objekttypen

© 2000-2025, MetaQuotes Ltd.

ObjectMove

Verändert Koordinaten des angegebenen Bezugspunktes des Objekts.

```
bool  ObjectMove(
   long      chart_id,         // Identifikator des Charts
   string    name,             // Objektname
   int       point_index,      // Nummer des Bezugspunktes
   datetime  time,             // Zeit
   double    price             // Preis
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

point_index

[in] Nummer des Bezugspunktes. Anzahl der Bezugspunkte hängt vom Objekttyp ab.

time

[in] Zeitkoordinate des angegebenen Bezugspunktes.

price

[in] Preiskoordinate des angegebenen Bezugspunktes.

Rückgabewert

Gibt true zurück, wenn ein Befehl zur Warteschlange des angegebenen Charts erfolgreich hinzugefügt wurde, andernfalls false.

Hinweis

Beim Aufruf von ObjectMove() wird immer ein asynchroner Aufruf verwendet, deswegen gibt die Funktion nur das Ergebnis des Hinzufügens eines Befehls zur Warteschlange zurück. In diesem Fall bedeutet true nur, dass der Befehl zur Warteschlange erfolgreich hinzugefügt wurde, das Ergebnis der Ausführung ist unbekannt.

Für die Überprüfung des Ergebnisses der Ausführung kann man eine Funktion verwenden, die die Eigenschaften des Objektes abruft, z.B. vom Typ ObjectGetXXX. Dabei muss man beachten, dass solche Funktionen am Ende der Warteschlange hinzugefügt werden und auf das Ergebnis der Ausführung warten (weil sie synchrone Aufrufe sind). D.h. sie können viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

© 2000-2025, MetaQuotes Ltd.

ObjectsTotal

Gibt die Anzahl der Objekte im angegebenen Chart, angegebenen Subfenster des angegebenen Charts des angegebenen Typs zurück.

```
int  ObjectsTotal(
   long  chart_id,           // Identifikator des Charts
   int   sub_window=-1,      // Index des Fensters
   int   type=-1             // Objekttyp
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

sub_window=-1

[in] Nummer des Subfensters des Charts. 0 bedeutet Hauptfenster des Charts, -1 bedeutet alle Subfenster des Charts, einschliesslich Hauptfenster.

type=-1

[in] Objekttyp. Wert kann einer der Enumerationswerte ENUM_OBJECT sein. -1 bedeutet alle Typen.

Rückgabewert

Anzahl der Objekte.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

© 2000-2025, MetaQuotes Ltd.

ObjectSetDouble

Gibt den Wert der entsprechenden Eigenschaft des Objekts vor. Eigenschaft des Objekts muss des Typs double sein. Es gibt 2 Varianten der Funktion.

Einstellung des Wertes der Eigenschaft ohne Modifikator

```
bool  ObjectSetDouble(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Name
   ENUM_OBJECT_PROPERTY_DOUBLE   prop_id,           // Eigenschaft
   double                        prop_value         // Wert
   );
```

Einstellung des Wertes der Eigenschaft mit Andeutung des Modifikatoren

```
bool  ObjectSetDouble(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Name
   ENUM_OBJECT_PROPERTY_DOUBLE   prop_id,           // Eigenschaft
   int                           prop_modifier,     // Modifikator
   double                        prop_value         // Wert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

prop_id

[in] Identifikator der Eigenschaft des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT_PROPERTY_DOUBLE sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Die meisten Eigenschaften erfordern keinen Modifikatoren.

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Die Funktion gibt true zurück nur wenn der Befehl zum Ändern der Eigenschaften von Objekten zum Chart erfolgreich gesendet wurde. Ansonsten gibt es false zurück. Für die Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Die Funktion verwendet einen asynchronen Aufruf, d.h. dass die Funktion nicht auf die Ausführung des zur Warteschlange des Charts erfolgreich hinzugefügten Befehls wartet, sondern direkt die Kontrolle zurückgibt.

Für die Überprüfung des Ergebnisses der Ausführung in einem anderen Chart kann man eine Funktion verwenden, die die angegebene Eigenschaft des Objektes abruft. Dabei sollte man beachten, dass solche Funktion am Ende der Warteschlange von Befehlen eines anderen Charts hinzugefügt werden und auf das Ergebnis der Ausführung warten. D.h. sie können viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Beispiel der Erzeugung des Objekts Fibonacci und der Hinzufuegung des neuen Levels darin

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Hilfsfelder
   double high[],low[],price1,price2;
   datetime time[],time1,time2;
//--- kopieren wir Eroeffnungspreise - 100 letzte Bars reicht es
   int copied=CopyHigh(Symbol(),0,0,100,high);
   if(copied<=0)
     {
      Print("Nicht gelungen, Werte der Preisserie zu kopieren High");
      return;
     }
//--- kopieren wir Schlusspreise - 100 letzte Bars reicht es
   copied=CopyLow(Symbol(),0,0,100,low);
   if(copied<=0)
     {
      Print("Misslugen, die Werte der Preisserie Low" zu kopieren);
      return;
     }
//--- kopieren wir Eroeffnungszeit für 100 letzte Bars
   copied=CopyTime(Symbol(),0,0,100,time);
   if(copied<=0)
     {
      Print("Misslungen, die Werte der Preisserie Time zu kopieren");
      return;
     }
//--- organisieren wir Zugang zu kopierten Daten als zu einer Preisserie -  umgekehrt
   ArraySetAsSeries(high,true);
   ArraySetAsSeries(low,true);
   ArraySetAsSeries(time,true);

//--- Koordinaten des ersten Bezugspunktes des Fibo-Objektes
   price1=high[70];
   time1=time[70];
//--- Koordinaten des ersten Bezugspunktes des Fiboobjektes
```

© 2000-2025, MetaQuotes Ltd.

```
   price2=low[50];
   time2=time[50];

//--- nun muessen wir ein Fibo-Objekt selbst erzeugen
   bool created=ObjectCreate(0,"Fibo",OBJ_FIBO,0,time1,price1,time2,price2);
   if(created) // wenn das Objekt erfolgreich erzeugt wurde
     {
      //--- geben wir die Farbe der Fibo-Levels vor
      ObjectSetInteger(0,"Fibo",OBJPROP_LEVELCOLOR,Blue);
      //--- uebrigens, wieviel Fibo-Levels haben wir ?
      int levels=ObjectGetInteger(0,"Fibo",OBJPROP_LEVELS);
      Print("Fibo levels before = ",levels);
      //---geben wir in Journal=> Levelnummer:Werte Beschreibung_des Levels
      for(int i=0;i<levels;i++)
        {
         Print(i,":",ObjectGetDouble(0,"Fibo",OBJPROP_LEVELVALUE,i),
               "  ",ObjectGetString(0,"Fibo",OBJPROP_LEVELTEXT,i));
        }
      //--- versuchen wir, die Anzahl der Levels um 1 zu vergroessern
      bool modified=ObjectSetInteger(0,"Fibo",OBJPROP_LEVELS,levels+1);
      if(!modified) // Misslungen, die Anzahl der Levels zu verändern
        {
         Print("Misslungen die Anzahl der Levels Fibo zu verändern, Fehler ",GetLastEr
        }
      //--- wir melden nur
      Print("Fibo levels after = ",ObjectGetInteger(0,"Fibo",OBJPROP_LEVELS));
      //--- geben wir den Wert für den nur erzeugten Level vor
      bool added=ObjectSetDouble(0,"Fibo",OBJPROP_LEVELVALUE,levels,133);
      if(added) // gelungen, Wert für Level vorzugeben
        {
         Print("Gelungen, noch einen Level Fibo einzustellen");
         //--- nicht vergessen, Levelbeschreibung vorzugeben
         ObjectSetString(0,"Fibo",OBJPROP_LEVELTEXT,levels,"my level");
         ChartRedraw(0);
         //--- bekommen wir aktuelle Werte der Levelsanzahl in Fibo-Objekt
         levels=ObjectGetInteger(0,"Fibo",OBJPROP_LEVELS);
         Print("Fibo levels after adding = ",levels);
         //--- geben wir nochmals alle Levels aus - nur um uns zu versichern
         for(int i=0;i<levels;i++)
           {
            Print(i,":",ObjectGetDouble(0,"Fibo",OBJPROP_LEVELVALUE,i),
                  "  ",ObjectGetString(0,"Fibo",OBJPROP_LEVELTEXT,i));
           }
        }
      else // Misserfolg beim Versuch Levelsanzahl in einem Fibo-Objekt zu vergroesser
        {
         Print("Misslungen, noch einen Level Fibo einzustellen. Fehler ",GetLastError
        }
     }
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

Sehen Sie auch

Objekttypen, Objekteigenschaften

© 2000-2025, MetaQuotes Ltd.

ObjectSetInteger

Gibt den Wert der entsprechenden Eigenschaft des Objekts vor.Eigenschaft des Objekts muss der Typen datetime, int, color, bool oder char sein. Es gibt 2 Varianten der Funktion.

Einstellung des Wertes der Eigenschaft ohne Modifikator

```
bool  ObjectSetInteger(
   long                              chart_id,          // Identifikator des Charts
   string                            name,              // Name
   ENUM_OBJECT_PROPERTY_INTEGER      prop_id,           // Eigenschaft
   long                              prop_value         // Wert
   );
```

Einstellung des Wertes der Eigenschaft mit Andeutung des Modifikatoren

```
bool  ObjectSetInteger(
   long                              chart_id,          // Identifikator des Dagramms
   string                            name,              // Name
   ENUM_OBJECT_PROPERTY_INTEGER      prop_id,           // Eigenschaft
   int                               prop_modifier,     // Modifikator
   long                              prop_value         // Wert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

prop_id

[in] Identifikator der Eigenschaft des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT_PROPERTY_INTEGER sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Die meisten Eigenschaften erfordern keinen Modifikator.

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Die Funktion gibt true zurück nur wenn der Befehl zum Ändern der Eigenschaften von Objekten zum Chart erfolgreich gesendet wurde. Ansonsten gibt es false zurück. Für die Erhaltung derweiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Die Funktion verwendet einen asynchronen Aufruf, d.h. dass die Funktion nicht auf die Ausführung des zur Warteschlange des Charts erfolgreich hinzugefügten Befehls wartet, sondern direkt die Kontrolle zurückgibt.

Für die Überprüfung des Ergebnisses der Ausführung in einem anderen Chart kann man eine Funktion verwenden, die die angegebene Eigenschaft des Objektes abruft. Dabei sollte man beachten, dass solche Funktion am Ende der Warteschlange von Befehlen eines anderen Charts hinzugefügt werden und auf das Ergebnis der Ausführung warten. D.h. sie können viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Beispiel der Erstellung einer Web-Farbtabelle

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                Table of Web Colors|
//|                        Copyright 2011, MetaQuotes Software Corp |
//|                                        https://www.metaquotes.net |
//+------------------------------------------------------------------+
#define X_SIZE 140      // Breite des Objekts OBJ_EDIT
#define Y_SIZE 33       // Breite des Objekts OBJ_EDIT
//+------------------------------------------------------------------+
//| Ein Array von Web-Farben                                         |
//+------------------------------------------------------------------+
color ExtClr[140]=
  {
   clrAliceBlue,clrAntiqueWhite,clrAqua,clrAquamarine,clrAzure,clrBeige,clrBisque,clrB
   clrBlue,clrBlueViolet,clrBrown,clrBurlyWood,clrCadetBlue,clrChartreuse,clrChocolate
   clrCornsilk,clrCrimson,clrCyan,clrDarkBlue,clrDarkCyan,clrDarkGoldenrod,clrDarkGray
   clrDarkMagenta,clrDarkOliveGreen,clrDarkOrange,clrDarkOrchid,clrDarkRed,clrDarkSalm
   clrDarkSlateBlue,clrDarkSlateGray,clrDarkTurquoise,clrDarkViolet,clrDeepPink,clrDee
   clrDodgerBlue,clrFireBrick,clrFloralWhite,clrForestGreen,clrFuchsia,clrGainsboro,cl
   clrGoldenrod,clrGray,clrGreen,clrGreenYellow,clrHoneydew,clrHotPink,clrIndianRed,cl
   clrLavender,clrLavenderBlush,clrLawnGreen,clrLemonChiffon,clrLightBlue,clrLightCora
   clrLightGoldenrod,clrLightGreen,clrLightGray,clrLightPink,clrLightSalmon,clrLightSe
   clrLightSlateGray,clrLightSteelBlue,clrLightYellow,clrLime,clrLimeGreen,clrLinen,cl
   clrMediumAquamarine,clrMediumBlue,clrMediumOrchid,clrMediumPurple,clrMediumSeaGreen
   clrMediumSpringGreen,clrMediumTurquoise,clrMediumVioletRed,clrMidnightBlue,clrMintC
   clrNavajoWhite,clrNavy,clrOldLace,clrOlive,clrOliveDrab,clrOrange,clrOrangeRed,clrO
   clrPaleGreen,clrPaleTurquoise,clrPaleVioletRed,clrPapayaWhip,clrPeachPuff,clrPeru,c
   clrPurple,clrRed,clrRosyBrown,clrRoyalBlue,clrSaddleBrown,clrSalmon,clrSandyBrown,c
   clrSienna,clrSilver,clrSkyBlue,clrSlateBlue,clrSlateGray,clrSnow,clrSpringGreen,clr
   clrThistle,clrTomato,clrTurquoise,clrViolet,clrWheat,clrWhite,clrWhiteSmoke,clrYell
  };
//+------------------------------------------------------------------+
//| Erstellung und Initialisierung des Objekts OBJ_EDIT              |
//+------------------------------------------------------------------+
void CreateColorBox(int x,int y,color c)
  {
//--- Wir generieren einen Namen für das neue Objekt auf den Farbnamen
   string name="ColorBox_"+(string)x+"_"+(string)y;
//--- Wir erstellen ein neues Objekt OBJ_EDIT
   if(!ObjectCreate(0,name,OBJ_EDIT,0,0,0))
     {
      Print("Konnte nicht erstellen '",name,"'");
      return;
     }
//--- Wir definieren die Koordinate des Ankerpunktes, Breite und Höhe in Pixel
   ObjectSetInteger(0,name,OBJPROP_XDISTANCE,x*X_SIZE);
   ObjectSetInteger(0,name,OBJPROP_YDISTANCE,y*Y_SIZE);
   ObjectSetInteger(0,name,OBJPROP_XSIZE,X_SIZE);
   ObjectSetInteger(0,name,OBJPROP_YSIZE,Y_SIZE);
//--- Wir geben die Textfarbe für das Objekt ein
   if(clrBlack==c) ObjectSetInteger(0,name,OBJPROP_COLOR,clrWhite);
   else            ObjectSetInteger(0,name,OBJPROP_COLOR,clrBlack);
//--- Wir geben die Hintergrundfarbe ein
   ObjectSetInteger(0,name,OBJPROP_BGCOLOR,c);
//--- Wir setzen den Text des Objekts OBJ_EDIT entsprechend der Hintergrundfarbe
   ObjectSetString(0,name,OBJPROP_TEXT,(string)c);
  }
//+------------------------------------------------------------------+
//| Die Funktion starter Skript                                      |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Wir erstellen eine Tabelle der Farbblöcke 7x20
   for(uint i=0;i<140;i++)
      CreateColorBox(i%7,i/7,ExtClr[i]);
  }
```

Sehen Sie auch

Objekttypen, Objekteigenschaften

© 2000-2025, MetaQuotes Ltd.

ObjectSetString

Gibt den Wert der entsprechenden Eigenschaftdes Objekts. Eigenschaft muss des Typs string sein. Es gibt 2 Varianten der Funktion.

Einstellung des Wertes der Eigenschaft ohne Modifikator

```
bool  ObjectSetString(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Name
   ENUM_OBJECT_PROPERTY_STRING   prop_id,           // Eigenschaft
   string                        prop_value         // Wert
   );
```

Einstellung des Wertes der Eigenschaft mit Andeutung von Modifikator

```
bool  ObjectSetString(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Name
   ENUM_OBJECT_PROPERTY_STRING   prop_id,           // Eigenschaft
   int                           prop_modifier,     // Modifikator
   string                        prop_value         // Wert
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

prop_id

[in] Identifikator der Eigenschaft des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT_PROPERTY_STRING sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Die meisten Eigenscgaften erfordern keinen Modifikator. Bedeutet die Standnummer in Fibonacci Werkzeuge und in Andrew's Pitchfork. Nummerierung beginnt bei Null.

prop_value

[in] Wert der Eigenschaft.

Rückgabewert

Die Funktion gibt true zurück nur wenn der Befehl zum Ändern der Eigenschaften von Objekten zum Chart erfolgreich gesendet wurde. Ansonsten gibt es false zurück. Für die Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Die Funktion verwendet einen asynchronen Aufruf, d.h. dass die Funktion nicht auf die Ausführung des zur Warteschlange des Charts erfolgreich hinzugefügten Befehls wartet, sondern direkt die Kontrolle zurückgibt.

Für die Überprüfung des Ergebnisses der Ausführung in einem anderen Chart kann man eine Funktion verwenden, die die angegebene Eigenschaft des Objektes abruft. Dabei sollte man beachten, dass solche Funktion am Ende der Warteschlange von Befehlen eines anderen Charts hinzugefügt werden und auf das Ergebnis der Ausführung warten. D.h. sie können viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Bei der Umbenennung des graphischen Objekts werden gleichzeitig zwei Ereignisse gebildet, die durch die Funktion OnChartEvent() im Expert oder Anzeiger verarbeitet werden können:

- Entfernung des Objekts mit dem alten Namen;
- Erzeugung des graphischen Objekts mit dem neuen Namen.

© 2000-2025, MetaQuotes Ltd.

ObjectGetDouble

Gibt den Wert der entsprechenden Eigenschaft des Objekt zurücks. Eigenschaft des Objekts muss des Typs double sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft sofort zurück.

```
double  ObjectGetDouble(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Objektname
   ENUM_OBJECT_PROPERTY_DOUBLE   prop_id,           // Identifikator der Eigenschaft
   int                           prop_modifier=0    // Modifikator der Eigenschaft v
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wird. Im Erfolgsfall wird die Funktion in eine Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  ObjectGetDouble(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Objektname
   ENUM_OBJECT_PROPERTY_DOUBLE   prop_id,           // Identifikator der Eigenschaft
   int                           prop_modifier,     // Modifikator der Eigenschaft
   double&                       double_var         // hier erhalten wir den Wert de
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

prop_id

[in] Identifikator der Eigenschaft des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT_PROPERTY_DOUBLE sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Für die erste Variante des Aufrufs ist der Default-Wert des Modifikators 0. Die meisten Eigenschaften erfordern keinen Modifikator. Bedeutet die Standnummer in Fibonacci Werkzeuge und in Andrew's Pitchfork. Nummerierung beginnt bei Null.

double_var

[out] Variable des Typs double, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs double für die erste Variante des Aufrufes.

für die zweite Variante des Aufrufs gibt true zurück, wenn diese Eigenschaft unterstuetzt wird und der Wert in die Variable double_var gesetzt wurde, anderenfalls gibt false zurück. Für Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

© 2000-2025, MetaQuotes Ltd.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

© 2000-2025, MetaQuotes Ltd.

ObjectGetInteger

Gibt den Wert der entsprechenden Eigenschaft des Objekts zurück. Eigenschaft des Objekts muss der Typen datetime, int, color, bool oder char sein. Es gibt 2 Varianten der Funktion.

1. Gibt den Wert der Eigenschaft direkt zurück.

```
long  ObjectGetInteger(
   long     chart_id,          // Identifikator des Charts
   string   name,              // Objektname
   int      prop_id,           // Identifikator der Eigenschaft
   int      prop_modifier=0    // Modifikator der Eigenschaft wenn erforderlich
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wird. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  ObjectGetInteger(
   long     chart_id,          // Identifikator des Charts
   string   name,              // Objektname
   int      prop_id,           // Identifikator der Eifgenschaft
   int      prop_modifier,     // Modifikator der Eigenschaft
   long&    long_var           // hier erhalten wir den Wert der Eigenschaft
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

prop_id

[in] Identifikator der Eigenschaft des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT_PROPERTY_INTEGER sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Für die erste Variante ist der Default-Wert des Modifikators 0. Die meisten Eigenschaften erfordern keinen Modifikator. Bedeutet die Standnummer in Fibonacci Werkzeuge und in Andrew's Pitchfork. Nummerierung beginnt bei Null.

long_var

[out] Variable des Typs long, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs long für die erste Variante des Aufrufs.

für die zweite Variante des Aufrufs gibt true zurück, wenn diese Eigenschaft unterstuetzt wird und der Wert in die Variable long_var gesetzt wurde, anderenfalls gibt false zurück. Für Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

© 2000-2025, MetaQuotes Ltd.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

© 2000-2025, MetaQuotes Ltd.

ObjectGetString

Gibt den Wert der entsprechenden Eigenschaft des Objekts zurück. Eigenschaft des Objekts muss des Typs string sein. Es gibt 2 Varianten der Funktion.

1.Gibt den wert der Eigenschaft sofort zurück.

```
string  ObjectGetString(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Objektname
   ENUM_OBJECT_PROPERTY_STRING   prop_id,           // Identifikator der Eigenschaft
   int                           prop_modifier=0    // Modifikator der Eigenschaft,
   );
```

2. Gibt true oder false zurück abhängig davon, ob die Funktion erfolgreich durchgeführt wurde. Im Erfolgsfall wird der Wert der Eigenschaft in die Empfangsvariable gesetzt, die durch Referenz vom letzten Parameter übertragen wird.

```
bool  ObjectGetString(
   long                          chart_id,          // Identifikator des Charts
   string                        name,              // Objektname
   ENUM_OBJECT_PROPERTY_STRING   prop_id,           // Identifikator der Eigenschaft
   int                           prop_modifier,     // Modifikator der Eigenschaft
   string&                       string_var         // hier erhalten wir den Wert de
   );
```

Parameter

chart_id

[in] Identifikator des Charts. 0 bedeutet den laufenden Chart.

name

[in] Objektname.

prop_id

[in] Identifikator der Eigenschaft des Objekts. Wert kann einer der Enumerationswerte ENUM_OBJECT_PROPERTY_STRING sein.

prop_modifier

[in] Modifikator der angegebenen Eigenschaft. Für die erste Default-Variante ist der Wert des Modifikators gleich 0. Die meisten Eigenschaften erfordern keinen Modifikator. It denotes the number of the level in Fibonacci tools and in the graphical object Andrew's pitchfork. The numeration of levels starts from zero.

string_var

[out] Variable des Typs string, die den Wert der angeforderten Eigenschaft annimmt.

Rückgabewert

Wert des Typs string für die erste Variante des Aufrufes.

© 2000-2025, MetaQuotes Ltd.

für die zweite Variante des Aufrufs gibt true zurück, wenn diese Eigenschaft unterstuetzt wird und der Wert in die Variable string_var gesetzt wurde, anderenfalls gibt false zurück. Für Erhaltung der weiteren Information über den Fehler, muss die Funktion GetLastError() aufgerufen werden.

Hinweis

Die Funktion verwendet einen synchronen Aufruf, d.h. dass die Funktion auf die Ausführung aller Befehle wartet, die vor deren Aufruf zur Warteschlange des Charts hinzugefügt wurden, deswegen kann die Funktion viel Zeit in Anspruch nehmen. Dies muss man beachten, wenn man mit vielen Objekten im Chart arbeitet.

Bei der Umbenennung des graphischen Objekts werden gleichzeitig zwei Ereignisse gebildet, die im Expert oder Anzeiger durch die Funktion OnChartEvent() verarbeitet können:

- Entfernung des Objekts mit dem alten Namen;
- Erzeugung des graphischen Objekts mit dem neuen Namen.

© 2000-2025, MetaQuotes Ltd.

TextSetFont

Setzt die Schrift für die Textausgabe durch Zeichnenmethoden und dann gibt das Ergebnis der Erfolg dieser Operation zurück. Die Standardschriftart ist Arial -120 (12 pt).

```
bool  TextSetFont(
   const string  name,            // Schriftname und Pfad zur Schriftdatei auf der Fes
   int           size,            // Schriftgröße
   uint          flags,           // Kombination von Flags
   int           orientation=0    // Winkel des Textes
   );
```

Optionen

name

[in] Schriftname und Pfad zur Schriftdatei auf der Festplatte.

size

[in] Die Schriftgröße kann als positive und negative Werte eingegeben werden. Bei positiven Werten ist die Größe des angegebenen Textes nicht auf Schriftgrößeneinstellungen im Betriebssystem abhängen. Bei negativen Werten wird der Wert in Zehntelpunkt angegeben, und Textgröße wird von den Systemeinstellungen ("Standardmaßstab" oder "großen Maßstab") abhängen. Weitere Informationen über die Unterschiede in den Betriebsarten finden Sie in der Note.

flags

[in] Kombination der Flaggen, die den Stil der Schriftart beschreiben.

orientation

[in] Horizontaler Winkel des Textes zu der Achse X, Maßeinheit ist 0,1 Grad. Das heißt *orientation=450* bedeutet Neigung von 45 Grad.

Rückgabewert

Gibt true zurück, wenn die aktuelle Schrift erfolgreich gesetzt wird, ansonsten false. Mögliche Fehlercodes:

- ERR_INVALID_PARAMETER(4003) - *name* ist NULL oder "" (leere Zeile),
- ERR_INTERNAL_ERROR(4001) - Betriebssystemfehler (zum Beispiel der Versuch, eine nicht vorhandene Schriftart zu erstellen).

Hinweis

Wenn "::" im Schriftname verwendet wird, dann wird die Schriftart aus der EX5-Ressource heruntergeladen. Wenn der Schriftname *name* Mit der Erweiterung angegeben ist, wird die Schriftart aus einer Datei geladen, in diesem Fall - wenn der Pfad mit "\" oder "/" beginnt, wird die Datei relativ zu dem Verzeichnis MQL5 gesucht, ansonsten wird sie relativ zu dem Pfad der EX5-Datei, die die Funktion TextSetFont() anruft hat, gesucht.

Die Schriftgröße wird als positive oder negative Werte angegeben, definiert das Zeichen die Abhängigkeit der Textgröße von den Betriebssystemeinstellungen (Schriftskala).

- Wenn die Größe durch eine positive Zahl angegeben ist, so während der Anzeige der logischen Schriftart als eine physikalische Schriftart, wird die Größe in der physikalischen Maßeinheit des Gerätes (Pixel) umgewandelt, und diese Größe entspricht der Höhe der Zeichenzellen aus

© 2000-2025, MetaQuotes Ltd.

verfügbaren Schriftarten. Es ist nicht empfohlen in den Fällen, wenn die gemeinsame Nutzung der durch Funktion TextOut() ausgegebenen Texten und durch das Graphikobjekt OBJ_LABEL ("Text-Label") angezeigten Texten auf einem Chart erwartet ist.

- Wenn die Größe durch eine negative Zahl angegeben ist, so ist die angegebene Größe in Zehntelverknüpfungspunkt angegeben (Wert -350 bedeutet 35 logischen Punkte) und wird dividiert durch 10, und dann wird der resultierende Wert in die physikalischen Geräteeinheiten (Pixel) umgewandelt und entspricht den absoluten Wert der Symbolhöhe von verfügbaren Schriftarten. Um den Text auf dem Bildschirm mit der gleichen Größe wie im Objekt OBJ_LABEL zu erhalten, nehmen Sie die Schriftgröße, die in Objekt-Eigenschaften angegeben ist, und multiplizieren Sie sie mit -10.

Die Flaggen können als eine Kombination von Stil-Flaggen mit eine der Flaggen, die die Dicke der Schrift angeben, verwendet werden. Namen der Flaggen sind unten verfügbar.

Flaggen, um Schriftstil anzugeben

Flagge	Beschreibung
FONT_ITALIC	Kursive
FONT_UNDERLINE	Unterstreichen
FONT_STRIKEOUT	Durchstreichen

Flaggen, um die Dicke der Schriftart anzugeben

Flagge
FW_DONTCARE
FW_THIN
FW_EXTRALIGHT
FW_ULTRALIGHT
FW_LIGHT
FW_NORMAL
FW_REGULAR
FW_MEDIUM
FW_SEMIBOLD
FW_DEMIBOLD
FW_BOLD
FW_EXTRABOLD
FW_ULTRABOLD
FW_HEAVY
FW_BLACK

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Ressourcen, ResourceCreate(), ResourceSave(), TextOut()

© 2000-2025, MetaQuotes Ltd.

TextOut

Gibt den Text in einem benutzerdefinierten Array (Puffer) aus und dann gibt das Ergebnis der Erfolg dieser Operation zurück. Dieses Array wird für die Erstellung von der grafischen Ressource verwendet.

```
bool  TextOut(
   const string       text,          // Textausgabe
   int                x,             // X Koordinate
   int                y,             // Y Koordinate
   uint               anchor,        // Bindungsmethode
   uint               &data[],       // Puffer für Ausgabe
   uint               width,         // Pufferbreite in Punkte
   uint               height,        // Pufferhöhe in Punkte
   uint               color,         // Textfarbe
   ENUM_COLOR_FORMAT  color_format   // Farbformat für Ausgabe
   );
```

Optionen

text

[in] Der Text, der in den Puffer geschrieben wird. Nur einzeilige Textausgabe verfügbar ist.

x

[in] X Koordinate des Ankerpunktes des ausgegebenen Textes.

y

[in] Y Koordinate des Ankerpunktes des ausgegebenen Textes.

anchor

[In] Der Wert aus der Gruppe von neun vordefinierten Methoden der Position des Ankerpunktes des angezeigten Textes. Wird durch eine Kombination von zwei Flaggen gesetzt - die Flagge der horizontalen Textausrichtung und die Flagge der vertikalen Textausrichtung. Namen der Flaggen sind in Bemerkung verfügbar.

data[]

[in] Der Puffer, in dem der Text ausgegeben wird. Dieser Puffer wird für die Erstellung von der grafischen Ressource verwendet.

width

[in] Die Breite des Puffers in Punkte (Pixels).

height

[in] Die Höhe des Puffers in Punkte (Pixels).

color

[in] Textfarbe.

color_format

[in] Farbformat wird durch einen Wert aus der Enumeration ENUM_COLOR_FORMAT angegeben.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Hinweis

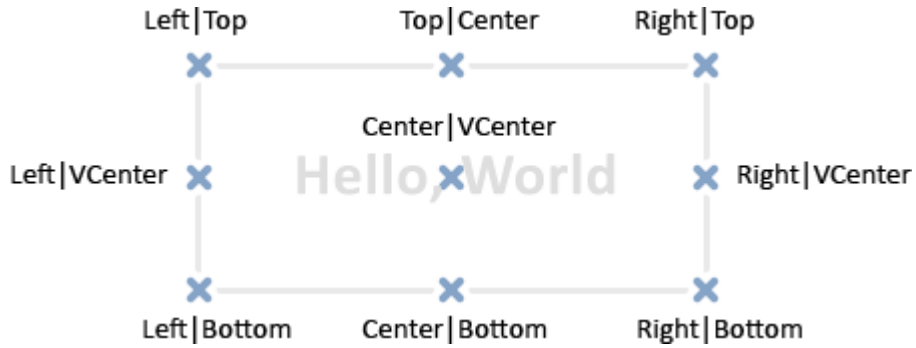
Die Bindungsmethode, die durch Parameter *anchor* angegeben wird, ist eine Kombination von zwei Flaggen von der vertikalen und horizontalen Textausrichtung. Flagge der horizontalen Textausrichtung:

- TA_LEFT - Ankerpunkt auf der linken Seite des Begrenzungsrahmens
- TA_CENTER - Ankerpunkt ist horizontal in der Mitte des Begrenzungsrahmens
- TA_RIGHT - Ankerpunkt auf der rechten Seite des Begrenzungsrahmens

Flagge der vertikalen Textausrichtung:

- TA_TOP - Ankerpunkt auf der oberen Seite des Begrenzungsrahmens
- TA_VCENTER - Ankerpunkt ist vertikal in der Mitte des Begrenzungsrahmens
- TA_BOTTOM - Ankerpunkt auf der unteren Seite des Begrenzungsrahmens

Mögliche Kombinationen von Flaggen und die Bindungsmethoden, die durch die Flaggen gesetzt werden, sind im Bild angezeigt.



Beispiel:

```
//--- Die Breite und Höhe der Leinwand (Leinwand, auf der wir zeichnen)
#define IMG_WIDTH  200
#define IMG_HEIGHT 200
//--- bevor das Skript zu starten, zeigen wir ein Fenster mit Parameter
#property script_show_inputs
//--- Möglichkeit Farbe anzugeben
input ENUM_COLOR_FORMAT clr_format=COLOR_FORMAT_XRGB_NOALPHA;
//--- Array (Puffer) zu zeichnen
uint ExtImg[IMG_WIDTH*IMG_HEIGHT];
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- wir erstellen das Objekt OBJ_BITMAP_LABEL zu zeichnen
   ObjectCreate(0,"CLOCK",OBJ_BITMAP_LABEL,0,0,0);
//--- wir geben den Namen der Ressource für Zeichnung im Objekt CLOCK an
   ObjectSetString(0,"CLOCK",OBJPROP_BMPFILE,"::IMG");

//--- Hilfsvariable
   double a;            // Winkel des Pfeils
```

© 2000-2025, MetaQuotes Ltd.

```
   uint    nm=2700;       // Minutenzähler
   uint    nh=2700*12;    // Stundenzähler
   uint    w,h;           // Variable für Größe der Textzeilen
   int     x,y;           // Variable für Berechnung der aktuellen Ankerpunktekoordinate

//--- Wir drehen die Uhr in einer Endlosschleife, da das Skript nicht aufgehört wird
   while(!IsStopped())
     {
      //--- Inhalt des Arrays von Uhr-Zeichnung Puffer löschen
      ArrayFill(ExtImg,0,IMG_WIDTH*IMG_HEIGHT,0);
      //--- Schriftart für Zahlen auf dem Zifferblatt angeben
      TextSetFont("Arial",-200,FW_EXTRABOLD,0);
      //--- Zifferblatt zeichnen
      for(int i=1;i<=12;i++)
        {
         //--- Größe der aktuellen Stunde auf dem Zifferblatt erhalten
         TextGetSize(string(i),w,h);
         //--- Koordinaten der aktuellen Stunde auf dem Zifferblatt berechnen
         a=-((i*300)%3600*M_PI)/1800.0;
         x=IMG_WIDTH/2-int(sin(a)*80+0.5+w/2);
         y=IMG_HEIGHT/2-int(cos(a)*80+0.5+h/2);
         //--- diese Stunde auf dem Zifferblatt in Puffer ExtImg[] ausgeben
         TextOut(string(i),x,y,TA_LEFT|TA_TOP,ExtImg,IMG_WIDTH,IMG_HEIGHT,0xFFFFFFFF,c
        }
      //--- Jetzt geben wir eine Schriftart für den Minutenzeiger an
      TextSetFont("Arial",-200,FW_EXTRABOLD,-int(nm%3600));
      //--- Größe des Minutenzeigers erhalten
      TextGetSize("----->",w,h);
      //--- Koordinaten des Minutenzeigers auf dem Zifferblatt berechnen
      a=-(nm%3600*M_PI)/1800.0;
      x=IMG_WIDTH/2-int(sin(a)*h/2+0.5);
      y=IMG_HEIGHT/2-int(cos(a)*h/2+0.5);
      //--- Minutenzeiger auf dem Zifferblatt in Puffer ExtImg[] ausgeben
      TextOut("----->",x,y,TA_LEFT|TA_TOP,ExtImg,IMG_WIDTH,IMG_HEIGHT,0xFFFFFFFF,clr_f

      //--- Jetzt geben wir eine Schriftart für den Stundenzeiger an
      TextSetFont("Arial",-200,FW_EXTRABOLD,-int(nh/12%3600));
      TextGetSize("==>",w,h);
      //--- Koordinaten des Stundenzeigers auf dem Zifferblatt berechnen
      a=-(nh/12%3600*M_PI)/1800.0;
      x=IMG_WIDTH/2-int(sin(a)*h/2+0.5);
      y=IMG_HEIGHT/2-int(cos(a)*h/2+0.5);
      //--- Stundenzeiger auf dem Zifferblatt in Puffer ExtImg[] ausgeben
      TextOut("==>",x,y,TA_LEFT|TA_TOP,ExtImg,IMG_WIDTH,IMG_HEIGHT,0xFFFFFFFF,clr_forn

      //--- Grafikressource aktualisieren
      ResourceCreate("::IMG",ExtImg,IMG_WIDTH,IMG_HEIGHT,0,0,IMG_WIDTH,clr_format);
      //--- Zwangsaktualisierung des Charts
```

© 2000-2025, MetaQuotes Ltd.

```
      ChartRedraw();

      //--- Die Erhöhung der Stunden- und Minutenzähler
      nm+=60;
      nh+=60;
      //--- Machen wir eine kurze Pause zwischen den Bildern
      Sleep(10);
     }
//--- Objekt CLOCK nach dem Ausschaltung des Skripts löschen
   ObjectDelete(0,"CLOCK");
//---
  }
```

Sehen Sie auch

Ressourcen, ResourceCreate(), ResourceSave(), TextGetSize(), TextSetFont()

© 2000-2025, MetaQuotes Ltd.

TextGetSize

Gibt die Breite und Höhe der Zeile mit den aktuellen Schrifteinstellungen zurück.

```
bool  TextGetSize(
   const string       text,          // Textzeile
   uint&               width,        // Pufferbreite in Punkte
   uint&               height        // Pufferhöhe in Punkte
   );
```

Optionen

text

[in] Die Zeile, für deren wir die Höhe und Breite erhalten.

width

[out] Die Eingangsparameter für die Breite.

height

[out] Die Eingangsparameter für die Höhe.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Mögliche Fehlercodes:

- ERR_INTERNAL_ERROR(4001) - in dem Fall eines Betriebssystemfehlers.

Sehen Sie auch

Ressourcen, ResourceCreate(), ResourceSave(), TextSetFont(), TextOut()

© 2000-2025, MetaQuotes Ltd.

Funktionen für die Arbeit mit technischen Indikatoren

Alle Funktionen wie iMA, iAC, iMACD, iIchimoku usw. erzeugen im globalen cache des Kundenterminlas eine Kopie des entsprechenden Kundenindikators. Wenn eine Kopie des Indikators mit solchen Parametern schon existiert, wird eine neue Kopie nicht erzeugt, sondern Counter der Verweise an die vorgegebene Kopie steigt.

Diese Funktionen geben handle der entsprechenden Kopie des Indikators zurück. Im weiteren können Sie durch Verwendung dieses Handles Daten bekommen, die vom entsprechenden Indikator berechnet werden. Daten des entsprechenden Puffers (technische Anzeiger haben berechnete Daten in ihren Innenpuffern, deren Anzahl abhängig vom Anzeiger von 1 bis 5 sein kann) können in mql5-Programm durch die Funktion CopyBuffer() kopiert werden.

Man kann nicht Daten des Indikators sofort nah seiner Erzeugung verwenden, denn man braucht einige Zeit für Berechnung der Indikatorwerte, darum ist es am besten Handles der Indikatoren in OnInit() zu erzeugen. Funktion iCustom() erzeugt entsprechenden Benutzerindikator und bei der erfolgreichen Erzeugung gibt sein handle zurück. Benutzerindikatoren können bis 512 Indikatorbuffer haben, den Inhalt durch die Funktion CopyBuffer() erhalten werden kann mit der Verwendung von handle.

Es gibt eine universelle Methode, durch die Funktion durch die Funktion IndicatorCreate() jeden technischen Indikator zu erzeugen. Als Eingabeparameter bekommt diese Funktion:

- Symbolname;
- Timeframe;
- Typ des erzeugten Indikators;
- Anzahl der Eingabeparameter des Indikators;
- Feld des Typs MqlParam, mit allen erforderlichen Eingabeparametern.

für die Befreiung des Speichers des Computers wird die Funktion IndicatorRelease() verwendet, der handle dieses Indikators übertragen wird.

Bemerkung. Wiederholter Aufruf der Funktion des Indikators mit denselben Parametern führt nicht zu vielfachen Steigerung des Counters der Verweise, Counter wird nur einmal um 1 vergroessert. Es ist aber empfehlenswert, handles der Indikatoren in der Funktion OnInit() oder im Konstruktor der Klasse zu bekommen, und dann diese handles in anderen Funktionen zu verwenden.

Alle Indikatoren haben mindestens 2 Parameter - Symbol und Periode. Wert des Symbols NULL bedeutet das laufende Instrument, Wert der Periode 0 bedeutet das laufende Timeframe.

Funktion	Gibt handle des Indikators zurück:
iAC	Accelerator Oscillator
iAD	Accumulation/Distribution
iADX	Average Directional Index
iADXWilder	Average Directional Index by Welles Wilder
iAlligator	Alligator
iAMA	Adaptive Moving Average
iAO	Awesome Oscillator

© 2000-2025, MetaQuotes Ltd.

Funktion	Gibt handle des Indikators zurück:
iATR	Average True Range
iBearsPower	Bears Power
iBands	Bollinger Bands®
iBullsPower	Bulls Power
iCCI	Commodity Channel Index
iChaikin	Chaikin Oscillator
iCustom	Benutzerindikator
iDEMA	Double Exponential Moving Average
iDeMarker	DeMarker
iEnvelopes	Envelopes
iForce	Force Index
iFractals	Fractals
iFrAMA	Fractal Adaptive Moving Average
iGator	Gator Oscillator
iIchimoku	Ichimoku Kinko Hyo
iBWMFI	Market Facilitation Index by Bill Williams
iMomentum	Momentum
iMFI	Money Flow Index
iMA	Moving Average
iOsMA	Moving Average of Oscillator (MACD histogram)
iMACD	Moving Averages Convergence-Divergence
iOBV	On Balance Volume
iSAR	Parabolic Stop And Reverse System
iRSI	Relative Strength Index
iRVI	Relative Vigor Index
iStdDev	Standard Deviation
iStochastic	Stochastic Oscillator
iTEMA	Triple Exponential Moving Average
iTriX	Triple Exponential Moving Averages Oscillator
iWPR	Williams' Percent Range

© 2000-2025, MetaQuotes Ltd.

Funktion	Gibt handle des Indikators zurück:
iVIDyA	Variable Index Dynamic Average
iVolumes	Volumes

© 2000-2025, MetaQuotes Ltd.

iAC

Erzeugt im globalen cache des Client-Terminals den Indikator Accelerator Oscillator und gibt sein Handle zurück. Hat nur einen Puffer.

```
int  iAC(
   string           symbol,     // Name des Symbols
   ENUM_TIMEFRAMES  period      // Periode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, auf dessen Daten Indikator berechnet wird. NULL bedeutet den laufenden Chart.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Speicher des Computers vom nicht mehr verwendeten Indikator zu befreien, gibt es die Funktion IndicatorRelease(), der Handle dieses Indikators übertragen wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                     Demo_iAC.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iAC."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- iAC bauen
#property indicator_label1  "iAC"
#property indicator_type1   DRAW_COLOR_HISTOGRAM
#property indicator_color1  clrGreen, clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iAC,               // iAC verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iAC;          // Funktionstyp
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double iACBuffer[];
double iACColors[];
//--- Eine Variable um Handle des Indikators iAC zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Accelerator Oscillator gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,iACBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,iACColors,INDICATOR_COLOR_INDEX);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iAC)
      handle=iAC(name,period);
   else
      handle=IndicatorCreate(name,period,IND_AC);
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- darüber schreiben und Nummer des Fehlers anzeigen
       PrintFormat("Ein Handle des Indikators iAC für das Paar %s/%s konnte nicht erste
                   name,
                   EnumToString(period),
                   GetLastError());
       //--- Arbeit des Indikators ist früher geendet
       return(INIT_FAILED);
      }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Accelerator Oscillator berechnet war, zei
   short_name=StringFormat("iAC(%s/%s)",name,EnumToString(period));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iAC zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iACBuffer größer als die Anzahl der Werte in iAC auf symbol/per
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
    ;}
//--- Arrays iACBuffer und iACColors mit den Werten aus dem Indikator Accelerator Osci
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffer(iACBuffer,iACColors,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Accelerator Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
 ;}
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iAC ausfüllen                 |
//+------------------------------------------------------------------+
bool FillArraysFromBuffer(double &values[],        // Indikator-Puffer mit Werten von
                          double &color_indexes[], // Farbpuffer (um Farbindex zu spei
                          int ind_handle,          // Handle des Indikators iAC
                          int amount               // Anzahl der Werte, die kopiert we
                          )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iACBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAC kopiert werden, Fehlercode
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Nun die Farbeindizes kopieren
   if(CopyBuffer(ind_handle,1,0,amount,color_indexes)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Die Farbwerte konnte nicht aus dem Indikator iAC kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iAD

Gibt das Handle des Indikators Accumulation/Distribution zurück. Hat nur einen Puffer.

```
int  iAD(
   string               symbol,             // Name des Symbols
   ENUM_TIMEFRAMES      period,             // Periode
   ENUM_APPLIED_VOLUME  applied_volume      // Volumentyp für Berechnung
   );
```

Parameter

symbol

[in] Symbolname des Instruments, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

applied_volume

[in] Das verwendete Volumen. Kann einer der Werte ENUM_APPLIED_VOLUME sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                      Demo_iAD.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iAD."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot iAD
#property indicator_label1  "iAD"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iAD,               // iAD verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iAD;          // Funktionstyp
input ENUM_APPLIED_VOLUME   volumes;                // verwendetes Volumen
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double          iADBuffer[];
//--- Eine Variable um Handle des Indikators iAD zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Accumulation/Distribution gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iADBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iAD)
      handle=iAD(name,period,volumes);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
```

© 2000-2025, MetaQuotes Ltd.

```
      MqlParam pars[1];
      pars[0].type=TYPE_INT;
      pars[0].integer_value=volumes;
      handle=IndicatorCreate(name,period,IND_AD,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iAD für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren Accumulation/Distribution berechnet wa
   short_name=StringFormat("iAD(%s/%s)",name,EnumToString(period));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iAD zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- wenn Array iADBuffer größer als die Anzahl der Werte in iAD auf symbol/per
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iADBuffer mit den Werten aus dem Indikator Accumulation/Distribution a
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArrayFromBuffer(iADBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Accumulation/Distribution speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iAD ausfüllen                 |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],   // Indikator-Puffer der Linie von Accumul
                         int ind_handle,          // Handle des Indikators iAD
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iADBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAD kopiert werden, Fehlercode
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iADX

Gibt das Handle des Indikators Average Directional Movement Index zurück.

```
int  iADX(
   string           symbol,          // Name des Symbols
   ENUM_TIMEFRAMES  period,          // Periode
   int              adx_period       // Glättungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, auf dessen Daten Indikator berechnet wird. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

adx_period

[in] Periode für die Berechnung des Indexes.

Hinweis

Nummer der Puffer: 0 - MAIN_LINE, 1 - PLUSDI_LINE, 2 - MINUSDI_LINE.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                    Demo_iADX.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iADX."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 3
#property indicator_plots   3
```

© 2000-2025, MetaQuotes Ltd.

```
//--- plot ADX
#property indicator_label1  "ADX"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- plot DI_plus
#property indicator_label2  "DI_plus"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrYellowGreen
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- plot DI_minus
#property indicator_label3  "DI_minus"
#property indicator_type3   DRAW_LINE
#property indicator_color3  clrWheat
#property indicator_style3  STYLE_SOLID
#property indicator_width3  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iADX,              // iADX verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iADX;         // Typ der Funktion
input int                   adx_period=14;          // Berechnungszeitraum
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double         ADXBuffer[];
double         DI_plusBuffer[];
double         DI_minusBuffer[];
//--- Eine Variable um Handle des Indikators iADX zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Average Directional Movement Index gespeichert
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zu den Indikator-Puffern
```

© 2000-2025, MetaQuotes Ltd.

```
   SetIndexBuffer(0,ADXBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,DI_plusBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,DI_minusBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iADX)
      handle=iADX(name,period,adx_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[1];
      pars[0].type=TYPE_INT;
      pars[0].integer_value=adx_period;
      handle=IndicatorCreate(name,period,IND_ADX,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iADX für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Average Directional Movement Index berech
   short_name=StringFormat("iADX(%s/%s period=%d)",name,EnumToString(period),adx_perio
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iADX zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array ADXBuffer größer als die Anzahl der Werte in iADX auf symbol/pe
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Average Directional Movement Index au
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(ADXBuffer,DI_plusBuffer,DI_minusBuffer,handle,values_to_c
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Average Directional Movement Index speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iADX ausfüllen                |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool FillArraysFromBuffers(double &adx_values[],      // Indikator-Puffer der Linie vo
                           double &DIplus_values[],   // Indikator-Puffer für DI+
                           double &DIminus_values[],  // Indikator-Puffer für DI-
                           int ind_handle,            // Handle des Indikators iADXWil
                           int amount                 // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays ADXBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,0,amount,adx_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iADX kopiert werden, Fehlercod
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays DI_plusBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfül
   if(CopyBuffer(ind_handle,1,0,amount,DIplus_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iADX kopiert werden, Fehlercod
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays DI_plusBuffer mit Werten auf Indikator-Puffer mit Index 2 ausfül
   if(CopyBuffer(ind_handle,2,0,amount,DIminus_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iADX kopiert werden, Fehlercod
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iADXWilder

Gibt das Handle des Indikators Average Directional Movement Index by Welles Wilder zurück.

```
int  iADXWilder(
   string           symbol,          // Symbolname
   ENUM_TIMEFRAMES  period,          // Periode
   int              adx_period       // Glättungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instruments, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

adx_period

[in] Periode für Berechnung des Indexes.

Hinweis

Puffernummern: 0 - MAIN_LINE, 1 - PLUSDI_LINE, 2 - MINUSDI_LINE.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  iADXWilder.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iADXWilder."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 3
#property indicator_plots   3
```

© 2000-2025, MetaQuotes Ltd.

```
//--- plot ADX
#property indicator_label1  "ADX"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- plot DI_plus
#property indicator_label2  "DI_plus"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrYellowGreen
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- plot DI_minus
#property indicator_label3  "DI_minus"
#property indicator_type3   DRAW_LINE
#property indicator_color3  clrWheat
#property indicator_style3  STYLE_SOLID
#property indicator_width3  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iADXWilder,         // iADXWilder verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iADXWilder;   // Funktionstyp
input int                   adx_period=14;          // Berechnungszeitraum
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double          ADXBuffer[];
double          DI_plusBuffer[];
double          DI_minusBuffer[];
//--- Eine Variable um Handle des Indikators iADXWilder zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Average Directional Movement Index by Welles W
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zu den Indikator-Puffern
```

© 2000-2025, MetaQuotes Ltd.

```
   SetIndexBuffer(0,ADXBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,DI_plusBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,DI_minusBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iADXWilder)
      handle=iADXWilder(name,period,adx_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[1];
      pars[0].type=TYPE_INT;
      pars[0].integer_value=adx_period;
      handle=IndicatorCreate(name,period,IND_ADXW,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iADXWilder für das Paar %s/%s konnte nicht er
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Average Directional Movement Index by Wel
   short_name=StringFormat("iADXWilder(%s/%s period=%d)",name,EnumToString(period),adx
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iADXWilder zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array ADXBuffer größer als die Anzahl der Werte in iADXWilder auf sym
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Average Directional Movement Index by
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(ADXBuffer,DI_plusBuffer,DI_minusBuffer,handle,values_to_c
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Average Directional Movement Index speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iADXWilder ausfüllen          |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool FillArraysFromBuffers(double &adx_values[],      // Indikator-Puffer der Linie vo
                           double &DIplus_values[],   // Indikator-Puffer für DI+
                           double &DIminus_values[],  // Indikator-Puffer für DI-
                           int ind_handle,            // Handle des Indikators iADXWil
                           int amount                 // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays ADXBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,0,amount,adx_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iADXWilder kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays DI_plusBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfül
   if(CopyBuffer(ind_handle,1,0,amount,DIplus_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iADXWilder kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays DI_plusBuffer mit Werten auf Indikator-Puffer mit Index 2 ausfül
   if(CopyBuffer(ind_handle,2,0,amount,DIminus_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iADXWilder kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iAlligator

Gibt das Handle des Indikators Alligator zurück.

```
int  iAlligator(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  jaw_period,        // Periode für Berechnung der Kiefer
   int                  jaw_shift,         // horizontale Verschiebung der Kiefer
   int                  teeth_period,      // Periode für die Berechnung der Zähne
   int                  teeth_shift,       // horizontale Verschiebung der Zähne
   int                  lips_period,       // Periode für Berechnung der Lippen
   int                  lips_shift,        // horizontale Verschiebung der Lippen
   ENUM_MA_METHOD       ma_method,         // Glaettungstyp
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

jaw_period

[in] Mittelungsperiode der blauen Linie (Kiefer des Alligators).

jaw_shift

[in] Verschiebung der blauen Linie in Bezug auf das Preischart.

teeth_period

[in] Mittelungsperiode der roten Linie (Zähne des Alligators).

teeth_shift

[in] Verschiebung der roten Linie in Bezug auf das Preischart.

lips_period

[in] Mittelungsperiode der grünen Linie (Lippen des Alligators).

lips_shift

[in] Verschiebung der grünen Linie in Bezug auf das Preischart.

ma_method

[in] Mittelungsmethode. Kann einer der Enumerationswerte ENUM_MA_METHOD sein.

applied_price

[in] Der verwendete Preis. Kann einer der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein

© 2000-2025, MetaQuotes Ltd.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - GATORJAW_LINE, 1 - GATORTEETH_LINE, 2 - GATORLIPS_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iAlligator.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iAlligator."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Alligator.

#property indicator_chart_window
#property indicator_buffers 3
#property indicator_plots   3
//--- plot Jaws
#property indicator_label1  "Jaws"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- plot Teeth
#property indicator_label2  "Teeth"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- plot Lips
#property indicator_label3  "Lips"
#property indicator_type3   DRAW_LINE
#property indicator_color3  clrLime
#property indicator_style3  STYLE_SOLID
#property indicator_width3  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iAlligator,         // iAlligator verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iAlligator;   // Funktionstyp
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
input int                   jaw_period=13;          // Zeitraum für die Kiefer-Linie
input int                   jaw_shift=8;            // Verschiebung der Kiefer-Linie
input int                   teeth_period=8;         // Zeitraum für die Zähne-Linie
input int                   teeth_shift=5;          // Verschiebung der Zähne-Linie
input int                   lips_period=5;          // Zeitraum für die Lippen-Linie
input int                   lips_shift=3;           // Verschiebung der Lippen-Linie
input ENUM_MA_METHOD        MA_method=MODE_SMMA;    // Methode der Mittelung der Alliga
input ENUM_APPLIED_PRICE    applied_price=PRICE_MEDIAN;// Typ der Preise aus der der Al
//--- Indicator-Puffer
double          JawsBuffer[];
double          TeethBuffer[];
double          LipsBuffer[];
//--- Eine Variable um Handle des Indikators iAlligator zu speichern
int     handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Alligator gespeichert wird
int     bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zu den Indikator-Puffern
   SetIndexBuffer(0,JawsBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,TeethBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,LipsBuffer,INDICATOR_DATA);
//--- Verschiebung jede Linie definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,jaw_shift);
   PlotIndexSetInteger(1,PLOT_SHIFT,teeth_shift);
   PlotIndexSetInteger(2,PLOT_SHIFT,lips_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
```

© 2000-2025, MetaQuotes Ltd.

```
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iAlligator)
      handle=iAlligator(name,period,jaw_period,jaw_shift,teeth_period,
                        teeth_shift,lips_period,lips_shift,MA_method,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[8];
     //--- Perioden und Verschiebungen der Alligator-Linien
      pars[0].type=TYPE_INT;
      pars[0].integer_value=jaw_period;
      pars[1].type=TYPE_INT;
      pars[1].integer_value=jaw_shift;
      pars[2].type=TYPE_INT;
      pars[2].integer_value=teeth_period;
      pars[3].type=TYPE_INT;
      pars[3].integer_value=teeth_shift;
      pars[4].type=TYPE_INT;
      pars[4].integer_value=lips_period;
      pars[5].type=TYPE_INT;
      pars[5].integer_value=lips_shift;
//--- Art der Glättung
      pars[6].type=TYPE_INT;
      pars[6].integer_value=MA_method;
//--- Art des Preises
      pars[7].type=TYPE_INT;
      pars[7].integer_value=applied_price;
//--- Handle erstellen
      handle=IndicatorCreate(name,period,IND_ALLIGATOR,8,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iAlligator für das Paar %s/%s konnte nicht er
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren Alligator berechnet war, anzeigen
   short_name=StringFormat("iAlligator(%s/%s, %d,%d,%d,%d,%d,%d)",name,EnumToString(pe
                           jaw_period,jaw_shift,teeth_period,teeth_shift,lips_period,l
```

© 2000-2025, MetaQuotes Ltd.

```
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iAlligator zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array JawsBuffer größer als die Anzahl der Werte in iAlligator auf sy
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator Alligator ausfüllen
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(JawsBuffer,jaw_shift,TeethBuffer,teeth_shift,LipsBuffer,l
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
```

© 2000-2025, MetaQuotes Ltd.

```
                              values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Alligator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iAlligator ausfüllen          |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &jaws_buffer[],  // Indikator-Puffer für die Kiefer-
                           int j_shift,            // Verschiebung der Kiefer-Linie
                           double &teeth_buffer[], // Indikator-Puffer für die Zähne-I
                           int t_shift,            // Verschiebung der Zähne-Linie
                           double &lips_buffer[],  // Indikator-Puffer für die Lippen-
                           int l_shift,            // Verschiebung der Lippen-Linie
                           int ind_handle,         // Handle von iAlligator
                           int amount              // Anzahl der Werte, die kopiert we
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays JawsBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,-j_shift,amount,jaws_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAlligator kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays TeethBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfülle
   if(CopyBuffer(ind_handle,1,-t_shift,amount,teeth_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAlligator kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays LipsBuffer mit Werten auf Indikator-Puffer mit Index 2 ausfüller
   if(CopyBuffer(ind_handle,2,-l_shift,amount,lips_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAlligator kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iAMA

Gibt das Handle des Indikators Adaptive Moving Average zurück. Hat nur einen Puffer.

```
int  iAMA(
   string               symbol,              // Symbolname
   ENUM_TIMEFRAMES      period,              // Periode
   int                  ama_period,          // Periode AMA
   int                  fast_ma_period,      // Schnelle MA Periode
   int                  slow_ma_period,      // Langsame MA Periode
   int                  ama_shift,           // horizontale Verschiebung des Indikators
   ENUM_APPLIED_PRICE   applied_price        // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ama_period

[in] Periode der Berechnung, wenn Effektivitätskoeffizient berechnet wird.

fast_ma_period

[in] Schnelle Periode für Berechnung des Mittelungskoeffizient für schnellen Markt.

slow_ma_period

[in] Lanfsame Periode für Berechnung des Mittelungskoeffizient beim Fehlen des Trends .

ama_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iAMA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iAMA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator iAMA."

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot iAMA
#property indicator_label1  "iAMA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iAMA,              // iAMA verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iAMA;          // Funktionstyp
input string                symbol=" ";              // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;   // Zeitrahmen
input int                   ama_period=15;           // Berechnungszeitraum
input int                   fast_ma_period=2;        // Schnelle MA Periode
input int                   slow_ma_period=30;       // Langsame MA Periode
input int                   ama_shift=0;             // horizontale Verschiebung des Ir
input ENUM_APPLIED_PRICE    applied_price;           // Preistyp
//--- Indicator-Puffer
double         iAMABuffer[];
//--- Eine Variable um Handle des Indikators iAMA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Adaptive Moving Average gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iAMABuffer,INDICATOR_DATA);
//--- Verschiebung definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ama_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iAMA)
      handle=iAMA(name,period,ama_period,fast_ma_period,slow_ma_period,ama_shift,appli
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[5];
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ama_period;
      pars[1].type=TYPE_INT;
      pars[1].integer_value=fast_ma_period;
      pars[2].type=TYPE_INT;
      pars[2].integer_value=slow_ma_period;
      pars[3].type=TYPE_INT;
      pars[3].integer_value=ama_shift;
      //--- Preistyp
      pars[4].type=TYPE_INT;
      pars[4].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_AMA,5,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iAMA für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Adaptive Moving Average berechnet war, ze
```

© 2000-2025, MetaQuotes Ltd.

```
   short_name=StringFormat("iAMA(%s/%s,%d,%d,%d,d)",name,EnumToString(period),ama_peri
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iAlligator zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iADBuffer größer als die Anzahl der Werte in iAD auf symbol/per
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator Alligator ausfüllen
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArrayFromBuffer(iAMABuffer,ama_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
```

© 2000-2025, MetaQuotes Ltd.

```
                              short_name,
                              values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Accelerator Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iAMA ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &ama_buffer[],  // Indikator-Puffer der Linie von AMA
                         int a_shift,           // Verschiebung der Linie AMA
                         int ind_handle,        // Handle von iAMA
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iAMABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,-a_shift,amount,ama_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAlligator kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iAO

Gibt das Handle des Indikators Awesome Oscillator zurück. Hat nur einen Ruffer.

```
int  iAO(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                    Demo_iAO.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iAO."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- iAO bauen
#property indicator_label1  "iAO"
#property indicator_type1   DRAW_COLOR_HISTOGRAM
#property indicator_color1  clrGreen,clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iAO,               // iAO verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iAO;          // Funktionstyp
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double          iAOBuffer[];
double          iAOColors[];
//--- Eine Variable um Handle des Indikators iAO zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Awesome Oscillator gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,iAOBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,iAOColors,INDICATOR_COLOR_INDEX);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iAO)
      handle=iAO(name,period);
   else
      handle=IndicatorCreate(name,period,IND_AO);
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iAO für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Awesome Oscillator berechnet war, zeigen
   short_name=StringFormat("iAO(%s/%s)",name,EnumToString(period));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iAO zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iAOBuffer größer als die Anzahl der Werte in iAO auf symbol/per
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Arrays iAOBuffer und iAOColors mit den Werten aus dem Indikator Awesome Oscillat
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffer(iAOBuffer,iAOColors,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Accelerator Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iAO ausfüllen                 |
//+------------------------------------------------------------------+
bool FillArraysFromBuffer(double &values[],        // Indikator-Puffer mit Werten von
                          double &color_indexes[], // Farbpuffer (um Farbindex zu spei
                          int ind_handle,          // Handle von iAO
                          int amount               // Anzahl der Werte, die kopiert we
                          )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iAOBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iAO kopiert werden, Fehlercode
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Nun die Farbeindizes kopieren
   if(CopyBuffer(ind_handle,1,0,amount,color_indexes)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Die Farbwerte konnte nicht aus dem Indikator iAO kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iATR

Gibt das Handle des Indikators Average True Range zurück. Hat nur einen Puffer.

```
int  iATR(
   string           symbol,        // Symbolname
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Glättungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instruments, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Glaettungsperiode für Berechnung des Indikators.

Ruecjgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iATR.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iATR."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- plot iATR
#property indicator_label1  "iATR"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iATR,// iATR verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input int                  atr_period=14;          // Berechnungszeitraum
input Creation             type=Call_iATR;          // Funktionstyp
input string               symbol=" ";             // Symbol
input ENUM_TIMEFRAMES      period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double         iATRBuffer[];
//--- Eine Variable um Handle des Indikators iAC zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Average True Range gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iATRBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iATR)
      handle=iATR(name,period,atr_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
```

© 2000-2025, MetaQuotes Ltd.

```
      MqlParam pars[1];
      pars[0].type=TYPE_INT;
      pars[0].integer_value=atr_period;
      handle=IndicatorCreate(name,period,IND_ATR,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iATR für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Average True Range berechnet war, zeigen
   short_name=StringFormat("iATR(%s/%s, period=%d)",name,EnumToString(period),atr_peri
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iATR zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- wenn Array iATRBuffer größer als die Anzahl der Werte in iATR auf symbol/p
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iATRBuffer mit den Werten aus dem Indikator Average True Range ausfül
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iATRBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Accelerator Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iATR ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von ATR
                         int ind_handle,    // Handle von iATR
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iATRBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iATR kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iBearsPower

Gibt das Handle des Indikators Bears Power zurück. Hat nur einen Puffer.

```
int  iBearsPower(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Glättungsperiode für Berechnung des Indikators.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iBearsPower.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iBearsPower."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iBearsPower bauen
#property indicator_label1  "iBearsPower"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrSilver
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iBearsPower,       // iBearsPower verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iBearsPower;  // Funktionstyp
input int                   ma_period=13;           // MA Periode
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double          iBearsPowerBuffer[];
//--- Eine Variable um Handle des Indikators iBearsPower zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Bears Power gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iBearsPowerBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iBearsPower)
      handle=iBearsPower(name,period,ma_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
```

© 2000-2025, MetaQuotes Ltd.

```
      MqlParam pars[1];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      handle=IndicatorCreate(name,period,IND_BEARS,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iBearsPower für das Paar %s/%s konnte nicht e
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Bears Power berechnet war, zeigen
   short_name=StringFormat("iBearsPower(%s/%s, period=%d)",name,EnumToString(period),r
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iBearsPower zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- wenn Array iATRBuffer größer als die Anzahl der Werte in iBearsPower auf s
       //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
       if(calculated>rates_total) values_to_copy=rates_total;
       else                       values_to_copy=calculated;
      }
    else
      {
       //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
       //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
       values_to_copy=(rates_total-prev_calculated)+1;
      }
//--- Das Array mit den Werten aus dem Indikator Bears Power ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
    if(!FillArrayFromBuffer(iBearsPowerBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
    string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                             TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                             short_name,
                             values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
    Comment(comm);
//--- die Anzahl der Werte im Indikator Bears Power speichern
    bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
    return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iBearsPower ausfüllen         |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Bears Po
                         int ind_handle,    // Handle von iBearsPower
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
    ResetLastError();
//--- Teil des Arrays iBearsPowerBuffer mit Werten auf Indikator-Puffer mit Index 0 au
    if(CopyBuffer(ind_handle,0,0,amount,values)<0)
      {
       //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
       PrintFormat("Daten konnte nicht aus dem Indikator iBearsPower kopiert werden, Fe
       //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
       return(false);
      }
//--- Alles gelang
    return(true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iBands

Gibt das Handle des Indikators Bollinger Bands® zurück.

```
int  iBands(
   string               symbol,           // Symbolname
   ENUM_TIMEFRAMES      period,           // Periode
   int                  bands_period,     // Periode für Berechnung der Mittellinie
   int                  bands_shift,      // horizontale Verschiebung des Indikators
   double               deviation,        //Anzahl der Standardabweichungen
   ENUM_APPLIED_PRICE   applied_price     // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

bands_period

[in] Mittelungsperiode der Grundlinie des Indikators.

bands_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

deviation

[in] Abweichung von der Grundlinie.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - BASE_LINE, 1 - UPPER_BAND, 2 - LOWER_BAND

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  Demo_iBands.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iBands."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_chart_window
#property indicator_buffers 3
#property indicator_plots   3
//--- Upper bauen
#property indicator_label1  "Upper"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrMediumSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Lower bauen
#property indicator_label2  "Lower"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrMediumSeaGreen
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- Middle bauen
#property indicator_label3  "Middle"
#property indicator_type3   DRAW_LINE
#property indicator_color3  clrMediumSeaGreen
#property indicator_style3  STYLE_SOLID
#property indicator_width3  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iBands,             // iBands verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation             type=Call_iBands;          // Funktionstyp
input int                  bands_period=20;           // MA Periode
input int                  bands_shift=0;             // Verschiebung
input double               deviation=2.0;             // Anzahl der Standardabweichung
input ENUM_APPLIED_PRICE   applied_price=PRICE_CLOSE; // Preistyp
input string               symbol=" ";                // Symbol
input ENUM_TIMEFRAMES      period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         UpperBuffer[];
double         LowerBuffer[];
```

© 2000-2025, MetaQuotes Ltd.

```
double         MiddleBuffer[];
//--- Eine Variable um Handle des Indikators iBands zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Bollinger Bands gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,UpperBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,LowerBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,MiddleBuffer,INDICATOR_DATA);
//--- Verschiebung jede Linie definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,bands_shift);
   PlotIndexSetInteger(1,PLOT_SHIFT,bands_shift);
   PlotIndexSetInteger(2,PLOT_SHIFT,bands_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iBands)
      handle=iBands(name,period,bands_period,bands_shift,deviation,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=bands_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=bands_shift;
      //--- Anzahl der Standardabweichungen
      pars[2].type=TYPE_DOUBLE;
      pars[2].double_value=deviation;
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_BANDS,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iBands für das Paar %s/%s konnte nicht erstel
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Bollinger Bands berechnet war, zeigen
   short_name=StringFormat("iBands(%s/%s, %d,%d,%G,%s)",name,EnumToString(period),
                           bands_period,bands_shift,deviation,EnumToString(applied_pri
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iBands zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
    ;}
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      //--- wenn die Größe der Indikatorsarrays ist größer als die Anzahl der Werte in
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Bollinger Bands ausfüllen
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(MiddleBuffer,UpperBuffer,LowerBuffer,bands_shift,handle,v
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Bollinger Bands speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iBands ausfüllen              |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &base_values[],     // Indikator-Puffer der Mittell
                           double &upper_values[],    // Indicator-Puffer der Obergren
                           double &lower_values[],    // Indicator-Puffer der Untergre
                           int shift,                 // Verschiebung
                           int ind_handle,            // Handle von iBands
                           int amount                 // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays MiddleBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüll
   if(CopyBuffer(ind_handle,0,-shift,amount,base_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iBands kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Teil des Arrays UpperBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfülle
   if(CopyBuffer(ind_handle,1,-shift,amount,upper_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iBands kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays LowerBuffer mit Werten auf Indikator-Puffer mit Index 2 ausfülle
   if(CopyBuffer(ind_handle,2,-shift,amount,lower_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iBands kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iBullsPower

Gibt das Handle des Indikators Bulls Power zurück. Hat nur einen Puffer.

```
int  iBullsPower(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period          // Mittelungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Indikators.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iBullsPower.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iBullsPower."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iBullsPower bauen
#property indicator_label1  "iBullsPower"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrSilver
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iBullsPower,       // iBullsPower verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iBullsPower;  // Funktionstyp
input int                   ma_period=13;           // MA Periode
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
//--- Indicator-Puffer
double          iBullsPowerBuffer[];
//--- Eine Variable um Handle des Indikators iBullsPower zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Bulls Power gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iBullsPowerBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iBullsPower)
      handle=iBullsPower(name,period,ma_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
```

© 2000-2025, MetaQuotes Ltd.

```
      MqlParam pars[1];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      handle=IndicatorCreate(name,period,IND_BULLS,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iBullsPower für das Paar %s/%s konnte nicht e
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Bulls Power berechnet war, zeigen
   short_name=StringFormat("iBullsPower(%s/%s, period=%d)",name,EnumToString(period),r
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iBullsPower zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      //--- wenn Array iBullsPowerBuffer größer als die Anzahl der Werte in iBullsPowe
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iBullsPowerBuffer mit den Werten aus dem Indikator Bulls Power ausfül
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iBullsPowerBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Bulls Power speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iBullsPower ausfüllen         |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Bulls Po
                         int ind_handle,    // Handle von iBullsPower
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iBullsPowerBuffer mit Werten auf Indikator-Puffer mit Index 0 au
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iBullsPower kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

iCCI

Gibt das Handle des Indikators Commodity Channel Index zurück. Hat nur einen Puffer.

```
int  iCCI(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname de Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Indikators.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iCCI.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iCCI."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_plots   1
//--- iCCI bauen
#property indicator_label1  "iCCI"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Horizontale Ebenen im Indikatorsfenster
#property indicator_level1  -100.0
#property indicator_level2  100.0
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iCCI,              // iCCI verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iCCI;               // Funktionstyp
input int                   ma_period=14;                 // MA Periode
input ENUM_APPLIED_PRICE    applied_price=PRICE_TYPICAL;  // Preistyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double         iCCIBuffer[];
//--- Eine Variable um Handle des Indikators iCCI zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Commodity Channel Index gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iCCIBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iCCI)
      handle=iCCI(name,period,ma_period,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[2];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Preistyp
      pars[1].type=TYPE_INT;
      pars[1].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_CCI,2,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iCCI für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Bollinger Bands® berechnet war, zeigen
   short_name=StringFormat("iCCI(%s/%s, %d, %s)",name,EnumToString(period),
                           ma_period,EnumToString(applied_price));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Anzahl der Werte des Indikators iCCI zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iATRBuffer größer als die Anzahl der Werte in iCCI auf symbol/p
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iCCIBuffer mit den Werten aus dem Indikator Commodity Channel Index au
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iCCIBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Commodity Channel Index speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iCCI ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Commodit
                         int ind_handle,    // Handle von iCCI
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Teil des Arrays iCCIBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iCCI kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iChaikin

Gibt das Handle des Indikators Chaikin Oscillator zurück. Hat nur einen Puffer.

```
int  iChaikin(
   string               symbol,             // Symbolname
   ENUM_TIMEFRAMES      period,             // Periode
   int                  fast_ma_period,     // schnelle Periode
   int                  slow_ma_period,     // langsame Periode
   ENUM_MA_METHOD       ma_method,          // Glättungstyp
   ENUM_APPLIED_VOLUME  applied_volume      // das verwendete Volumen
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

fast_ma_period

[in] Schnelle Mittelungsperiode für Berechnung des Indikators.

slow_ma_period

[in] langsame Mittelungsperiode für Berechnung des Indikators .

ma_method

[in] Glaettungstyp .Kann eine der Glättungkonstanten von ENUM_MA_METHOD sein.

applied_volume

[in] Das verwendete Volumen. Kann eine der Enumerationskonstanten ENUM_APPLIED_VOLUME sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                 Demo_iChaikin.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iChaikin."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iChaikin bauen
#property indicator_label1  "iChaikin"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iChaikin,           // iChaikin verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iChaikin;           // Funktionstyp
input int                   fast_ma_period=3;             // Schnelle Zeitraum
input int                   slow_ma_period=10;            // Langsame Zeitraum
input ENUM_MA_METHOD        ma_method=MODE_EMA;           // Art der Glättung
input ENUM_APPLIED_VOLUME   applied_volume=VOLUME_TICK;   // Volumentyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double         iChaikinBuffer[];
//--- Eine Variable um Handle des Indikators iChaikin zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Chaikin Oscillator gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iChaikinBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
```

© 2000-2025, MetaQuotes Ltd.

```
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iChaikin)
      handle=iChaikin(name,period,fast_ma_period,slow_ma_period,ma_method,applied_volu
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- Schnelle Zeitraum
      pars[0].type=TYPE_INT;
      pars[0].integer_value=fast_ma_period;
      //--- Langsame Periode
      pars[1].type=TYPE_INT;
      pars[1].integer_value=slow_ma_period;
      //--- Art der Glättung
      pars[2].type=TYPE_INT;
      pars[2].integer_value=ma_method;
      //--- Volumentyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_volume;
      handle=IndicatorCreate(name,period,IND_CHAIKIN,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iChaikin für das Paar %s/%s konnte nicht erst
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Chaikin Oscillator berechnet war, zeigen
   short_name=StringFormat("iChaikin(%s/%s, %d, %d, %s, %s)",name,EnumToString(period)
                           fast_ma_period,slow_ma_period,
                           EnumToString(ma_method),EnumToString(applied_volume));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iChaikin zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
    ;}
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iChaikinBuffer größer als die Anzahl der Werte in iChaikin auf
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iChaikinBuffer mit den Werten aus dem Indikator Chaikin Oscillator aus
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iChaikinBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die Anzahl der Werte im Indikator Chaikin Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
 ;}
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iChaikin ausfüllen            |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Chaikin
                         int ind_handle,    // Handle von iChaikin
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iChaikinBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfü
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iChaikin kopiert werden, Fehle
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iCustom

Gibt das Handle des angegebenen Benutzerindikators zurück.

```
int  iCustom(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period,     // Periode
   string           name        // Ordner/Name_des Benutzerindikators
   ...                          // Liste der Eingabeparameter des Indikators
   );
```

Prameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

name

[in] Nutzerdefinierter Indikatorname. Wenn der Backslash '\' vor dem Namen des nutzerdefinierten Indikators angezeigt wird, wird die EX5-Indikator-Datei relativ zum MQL5-Stammordner durchsucht. Bei einem Aufruf von iCustom(Symbol(), Periode(), "\FirstIndicator"...) wird daher der Indikator MQL5\FirstIndicator.ex5 geladen. Wenn die Datei unter diesem Pfad nicht gefunden wird, wird der Fehler 4802 (ERR_INDICATOR_CANNOT_CREATE) ausgeworfen.

Beginnt der Pfad nicht mit '\', wird der Indikator wie folgt gesucht und geladen:

- Zuerst wird die EX5-Datei des Indikators in dem Ordner gesucht, in dem sich die EX5-Datei des aufrufenden Programms befindet. Liegt zum Beispiel CrossMA.EX5 EA in MQL5\Experts\MyExperts und sie enthält den Aufruf iCustom(Symbol(), Periode(), "SecondIndicator"...), wird in diesem Fall der Indikator hier MQL5\Experts\MyExperts\SecondIndicator.ex5 gesucht.
- Wenn der Indikator nicht im gleichen Verzeichnis gefunden wird, erfolgt die Suche relativ zum Stammverzeichnis des Indikators MQL5\Indicators. Mit anderen Worten, es wird die Datei MQL5\Indicators\SecondIndicator.ex5 gesucht. Wenn der Indikator immer noch nicht gefunden wurde, gibt die Funktion INVALID_HANDLE zurück und der Fehler 4802 (ERR_INDICATOR_CANNOT_CREATE) wird ausgeworfen.

Wenn der Pfad zum Indikator auf ein Unterverzeichnis (z.B. MyIndicators\ThirdIndicator) zeigt, wird zunächst die Suche im aufrufenden Programmordner (der EA befindet sich in MQL5\Experts\MyExperts) in MQL5\Experts\MyExperts\MyIndicators\ThirdIndicator.ex5 durchgeführt. Wenn dies nicht erfolgreich ist, wird die Datei hier MQL5\Indicators\MyIndicators\ThirdIndicator.ex5 gesucht. Achten Sie darauf, den doppelten umgekehrten Schrägstrich '\\' als Trennzeichen im Pfad zu verwenden, z.B. iCustom(Symbol(), Punkt(), "MeineIndikatoren\\DritterIndikator"...)

...

© 2000-2025, MetaQuotes Ltd.

[in] input-Parameter eines Benutzerindikators, die durch Kommas getrennt sind. Typ und Parameterfolge müssen entsprechen. Wenn Parameter nicht angegeben werden, werden Default-Werte verwendet.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück.

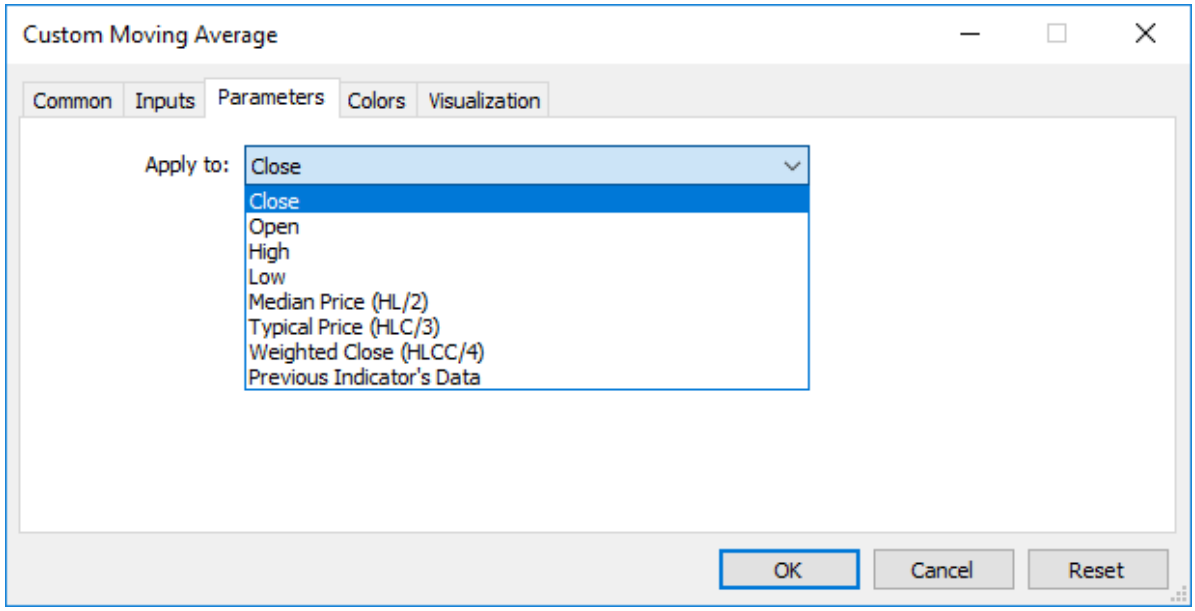
Hinweis

Benutzeranzeiger muss kompiliert werden (Datei mit Verbreitung EX5) und sich im Verzeichnis *MQL5/Indicators* des Client-Terminals oder in einem eingebettetenen Verzeichnis befinden.

Die für Testenerforderliche Indikatoren werden automatisch aus Aufruf der Funktionen iCustom() bestimmt, wenn der entsprechende Parameter von der Konstantzeile vorgegeben ist. Für andere Fälle (Verwendung der Funktion IndicatorCreate() oder Verwendung einer nicht Konstantzeile im Parameter, der den Namen des Indikator vorgibt) ist diese Eigenschaft #property tester_indicator erforderlich :

```
#property tester_indicator "indicator_name.ex5"
```

Wenn die erste Aufrufform im Indikator verwendet wird, kann beim Ablauf des Benutzerindikators im Registerblatt "Parameters" Daten für seine Berechnung angegeben werden. Wenn der Parameter "Apply to" explizit nicht ausgewählt ist, wird die Berechnung mit den Werten "Close" durchgeführt.



Beim Aufruf eines Benutzerindikators aus dem mql5-Programm kann Parameter Applied_Price oder handle eines anderen Parameters am letzten nach aller vom Benutzerindikator vorausgesehenen Eingabevariablen übertragen werden.

Sehen Sie auch

Programmeigenschaften, Zugang zu Zeitreihen und Indikatoren, IndicatorCreate(), IndicatorRelease()

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---- plot Label1
#property indicator_label1  "Label1"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- input parameters
input int MA_Period=21;
input int MA_Shift=0;
input ENUM_MA_METHOD MA_Method=MODE_SMA;
//--- indicator buffers
double         Label1Buffer[];
//--- handle des Benutzerindikators Custom Moving Average.mq5
int MA_handle;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,Label1Buffer,INDICATOR_DATA);
   ResetLastError();
   MA_handle=iCustom(NULL,0,"Examples\\Custom Moving Average",
                     MA_Period,
                     MA_Shift,
                     MA_Method,
                     PRICE_CLOSE // Schlusspreise verwenden
                     );
   Print("MA_handle = ",MA_handle,"  error = ",GetLastError());
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const int &spread[])
  {
//--- Werte des Indikators Custom Moving Average in den Indikator-Puffer kopieren
   int copy=CopyBuffer(MA_handle,0,0,rates_total,Label1Buffer);
   Print("copy = ",copy,"    rates_total = ",rates_total);
//--- beim erfolglosen Versuch geben wir das aus
   if(copy<=0)
      Print("Erfolgloser Versuch, Indikatorwerte Custom Moving Average zu erhalten");
//--- return value of prev_calculated for next call
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

iDEMA

Gibt das Handle des Indikators Double Exponential Moving Average zurück. Hat nur einen Puffer.

```
int  iDEMA(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   int                  ma_shift,          // horizontale Verschiebung des Indikators
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Periode(Anzahl der Bars) für Berechnung des Indikators.

ma_shift

[in] Verschibung des Indikators in Bezug auf das Preischart.

applied_price

[in] der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  Demo_iDEMA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iDEMA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iDEMA bauen
#property indicator_label1  "iDEMA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iDEMA,              // iDEMA verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iDEMA;           // Funktionstyp
input int                   ma_period=14;              // MA Periode
input int                   ma_shift=0;                // Verschiebung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         iDEMABuffer[];
//--- Eine Variable um Handle des Indikators iDEMA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Double Exponential Moving Average gespeichert
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iDEMABuffer,INDICATOR_DATA);
   //--- Verschiebung eingeben
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
```

© 2000-2025, MetaQuotes Ltd.

```
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iDEMA)
      handle=iDEMA(name,period,ma_period,ma_shift,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[3];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_shift;
      //--- Preistyp
      pars[2].type=TYPE_INT;
      pars[2].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_DEMA,3,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iDEMA für das Paar %s/%s konnte nicht erstell
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Double Exponential Moving Average berechn
   short_name=StringFormat("iDEMA(%s/%s, %d, %d, %s)",name,EnumToString(period),
                           ma_period,ma_shift,EnumToString(applied_price));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iDEMA zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iDEMABuffer größer als die Anzahl der Werte in iDEMA auf symbol
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iDEMABuffer mit den Werten aus dem Indikator Double Exponential Moving
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iDEMABuffer,ma_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Double Exponential Moving Average speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iDEMA ausfüllen               |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Double I
                         int shift,         // Verschiebung
                         int ind_handle,    // Handle von iDEMA
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iDEMABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfülle
   if(CopyBuffer(ind_handle,0,-shift,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iDEMA kopiert werden, Fehlercc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iDeMarker

Gibt das Handle des Indikators DeMarker zurück. Hat nur einen Puffer.

```
int  iDeMarker(
   string           symbol,        // Symbolname
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelungswert
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mitelungsperiode für Berechnung des Indikators.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iDeMarker.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iDeMarker."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iDeMarker bauen
#property indicator_label1  "iDeMarker"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Horizontale Ebenen im Indikatorsfenster
#property indicator_level1  0.3
#property indicator_level2  0.7
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iDeMarker,         // iDeMarker verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iDeMarker;       // Funktionstyp
input int                   ma_period=14;              // MA Periode
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double          iDeMarkerBuffer[];
//--- Eine Variable um Handle des Indikators iDeMarker zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator DeMarker gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iDeMarkerBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iDeMarker)
      handle=iDeMarker(name,period,ma_period);
```

© 2000-2025, MetaQuotes Ltd.

```
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[1];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      handle=IndicatorCreate(name,period,IND_DEMARKER,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iDeMarker für das Paar %s/%s konnte nicht ers
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren DeMarker berechnet war, anzeigen
   short_name=StringFormat("iDeMarker(%s/%s, period=%d)",name,EnumToString(period),ma_
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iDeMarker zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iDeMarkerBuffer größer als die Anzahl der Werte in iDeMarker au
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iDeMarkerBuffer mit den Werten aus dem Indikator DeMarker ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iDeMarkerBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator DeMarker speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iDeMarker ausfüllen           |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von DeMarker
                         int ind_handle,    // Handle von iDeMarker
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iDeMarkerBuffer mit Werten auf Indikator-Puffer mit Index 0 aus
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iDeMarker kopiert werden, Fehl
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
```

© 2000-2025, MetaQuotes Ltd.

```
    return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iEnvelopes

Gibt das Handle des Indikators Envelopes zurück.

```
int  iEnvelopes(
   string               symbol,           // Symbolname
   ENUM_TIMEFRAMES      period,           // Periode
   int                  ma_period,        // Periode für Berechnung der Mittelungslini
   int                  ma_shift,         // horizontale Verschiebung des Indikators
   ENUM_MA_METHOD       ma_method,        // Mittelungstyp
   ENUM_APPLIED_PRICE   applied_price,    // Preistyp oder handle
   double               deviation         // Abweichung der Grenzen von der Mittellini
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES, sein 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode der Hauptlinie des Indikators.

ma_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

ma_method

[in] Glättungsmethode. Kann einer der Enumerationswerte ENUM_MA_METHOD sein.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

deviation

[in] Abweichung von der Grundlinie in Prozenten.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - UPPER_LINE, 1 - LOWER_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                Demo_iEnvelopes.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iEnvelopes."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   2
//--- Upper bauen
#property indicator_label1  "Upper"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Lower bauen
#property indicator_label2  "Lower"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iEnvelopes,         // iEnvelopes verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iEnvelopes;      // Funktionstyp
input int                   ma_period=14;              // MA Periode
input int                   ma_shift=0;                // Verschiebung
input ENUM_MA_METHOD        ma_method=MODE_SMA;        // Art der Glättung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input double                deviation=0.1;             // Abweichung von Grenzen aus MA
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         UpperBuffer[];
double         LowerBuffer[];
//--- Eine Variable um Handle des Indikators iEnvelopes zu speichern
```

© 2000-2025, MetaQuotes Ltd.

```
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Envelopes gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,UpperBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,LowerBuffer,INDICATOR_DATA);
//--- Verschiebung jede Linie definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
   PlotIndexSetInteger(1,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iEnvelopes)
      handle=iEnvelopes(name,period,ma_period,ma_shift,ma_method,applied_price,deviati
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[5];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_shift;
      //--- Art der Glättung
      pars[2].type=TYPE_INT;
      pars[2].integer_value=ma_method;
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      //--- Preistyp
```

© 2000-2025, MetaQuotes Ltd.

```
      pars[4].type=TYPE_DOUBLE;
      pars[4].double_value=deviation;
      handle=IndicatorCreate(name,period,IND_ENVELOPES,5,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iEnvelopes für das Paar %s/%s konnte nicht er
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren Envelopes berechnet war, anzeigen
   short_name=StringFormat("iEnvelopes(%s/%s, %d, %d, %s,%s, %G)",name,EnumToString(pe
   ma_period,ma_shift,EnumToString(ma_method),EnumToString(applied_price),deviation);
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iEnvelopes zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- wenn Array UpperBuffer größer als die Anzahl der Werte in iEnvelopes auf s
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Arrays UpperBuffer und LowerBuffer mit den Werten aus dem Indikator Envelopes au
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArraysFromBuffers(UpperBuffer,LowerBuffer,ma_shift,handle,values_to_copy))
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Envelopes speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iEnvelopesator ausfüllen      |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &upper_values[],    // Indicator-Puffer der Obergren
                           double &lower_values[],    // Indicator-Puffer der Untergre
                           int shift,                 // Verschiebung
                           int ind_handle,            // Handle des Indikators iEnvelo
                           int amount                 // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays UpperBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfülle
   if(CopyBuffer(ind_handle,0,-shift,amount,upper_values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iEnvelopes kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Teil des Arrays LowerBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfülle
   if(CopyBuffer(ind_handle,1,-shift,amount,lower_values)<0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iEnvelopes kopiert werden, Feh
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iForce

Gibt das Handle des Indikators Force Index zurück. Hat nur einen Puffer.

```
int  iForce(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   ENUM_MA_METHOD       ma_method,         // Glaettungstyp
   ENUM_APPLIED_VOLUME  applied_volume     // Volumentyp für Berechnung
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Indikators.

ma_method

[in] Glaettungsmethode. Kann einer der Enumerationswerte ENUM_MA_METHOD sein.

applied_volume

[in] Das verwendete Volumen. Kann einer der Enumerationswerte ENUM_APPLIED_VOLUME sein.

Ruecjgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  Demo_iForce.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iForce."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iForce bauen
#property indicator_label1  "iForce"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iForce,             // iForce verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iForce;             // Funktionstyp
input int                   ma_period=13;                 // MA Periode
input ENUM_MA_METHOD        ma_method=MODE_SMA;           // Art der Glättung
input ENUM_APPLIED_VOLUME   applied_volume=VOLUME_TICK;   // Volumentyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double          iForceBuffer[];
//--- Eine Variable um Handle des Indikators Art der Glättung zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Force gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iForceBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
       name=_Symbol;
      }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iForce)
      handle=iForce(name,period,ma_period,ma_method,applied_volume);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[3];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Art der Glättung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_method;
      //--- Volumentyp
      pars[2].type=TYPE_INT;
      pars[2].integer_value=applied_volume;
      //--- Preistyp
      handle=IndicatorCreate(name,period,IND_FORCE,3,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iForce für das Paar %s/%s konnte nicht erstel
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren Force berechnet war, anzeigen
   short_name=StringFormat("iForce(%s/%s, %d, %s, %s)",name,EnumToString(period),
                           ma_period,EnumToString(ma_method),EnumToString(applied_volu
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iForce zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iForceBuffer größer als die Anzahl der Werte in iForce auf symb
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iForceBuffer mit den Werten aus dem Indikator Force ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iForceBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Force speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iForce ausfüllen              |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Force In
```

© 2000-2025, MetaQuotes Ltd.

```
                          int ind_handle,     // Handle des Indikators iForce
                          int amount          // Anzahl der Werte, die kopiert werden
                          )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iForceBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüll
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iForce kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iFractals

Gibt das Handle des Indikators Fractals zurück.

```
int  iFractals(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein , 0 bedeutet das laufende Timeframe.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - UPPER_LINE, 1 - LOWER_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iFractals.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iFractals."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
#property indicator_chart_window
#property indicator_buffers 2
#property indicator_plots   2
//--- FractalUp bauen
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_label1  "FractalUp"
#property indicator_type1   DRAW_ARROW
#property indicator_color1  clrBlue
//--- FractalDown bauen
#property indicator_label2  "FractalDown"
#property indicator_type2   DRAW_ARROW
#property indicator_color2  clrRed
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iFractals,          // iFractals verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iFractals;          // Funktionstyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double          FractalUpBuffer[];
double          FractalDownBuffer[];
//--- Eine Variable um Handle des Indikators iFractals zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//---  Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Fractals gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,FractalUpBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,FractalDownBuffer,INDICATOR_DATA);
//--- Codes der Symbole aus Reihe von Wingdings für PLOT_ARROW Eigenschaften definiere
   PlotIndexSetInteger(0,PLOT_ARROW,217); // Pfeil nach oben
   PlotIndexSetInteger(1,PLOT_ARROW,218); // Pfeil nach unten
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iFractals)
      handle=iFractals(name,period);
   else
      handle=IndicatorCreate(name,period,IND_FRACTALS);
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iFractals für das Paar %s/%s konnte nicht ers
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren Fractals berechnet war, anzeigen
   short_name=StringFormat("iFractals(%s/%s)",name,EnumToString(period));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
 ;}
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iFractals zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
```

© 2000-2025, MetaQuotes Ltd.

```
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array FractalUpBuffer größer als die Anzahl der Werte in iFractals au
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
    ;}
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Arrays FractalUpBuffer und FractalDownBuffer mit den Werten aus dem Indikator Fr
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArraysFromBuffers(FractalUpBuffer,FractalDownBuffer,handle,values_to_copy))
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Fractals speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iFractals ausfüllen           |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &up_arrows[],        // Indicator-Puffer der Pfeile
                           double &down_arrows[],      // Indicator-Puffer der Pfeile
                           int ind_handle,             // Handle des Indikators iFract
                           int amount                  // Anzahl der Werte, die kopier
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays FractalUpBuffer mit Werten auf Indikator-Puffer mit Index 0 aus
   if(CopyBuffer(ind_handle,0,0,amount,up_arrows)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iFractals ins Array FractalUpB
                  GetLastError());
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Teil des Arrays FractalDownBuffer mit Werten auf Indikator-Puffer mit Index 1 au
   if(CopyBuffer(ind_handle,1,0,amount,down_arrows)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iFractals ins Array FractalDow
                  GetLastError());
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iFrAMA

Gibt das Handle des Indikators Fractal Adaptive Moving Average zurück. Hat nur einen Puffer.

```
int  iFrAMA(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   int                  ma_shift,          // horizontale Verschiebung des Indikators
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert kann eine der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Periode(Anzahl der Bars) für Berechnung des Indikators.

ma_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

applied_price

[in] Der verwendete Preis. Kann einer der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                 Demo_iFrAMA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iFrAMA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iFrAMA bauen
#property indicator_label1  "iFrAMA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iFrAMA,            // iFrAMA verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iFrAMA;             // Funktionstyp
input int                   ma_period=14;                 // MA Periode
input int                   ma_shift=0;                   // Verschiebung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE;    // Preistyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double         iFrAMABuffer[];
//--- Eine Variable um Handle des Indikators iFrAMA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Fractal Adaptive Moving Average gespeichert wi
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iFrAMABuffer,INDICATOR_DATA);
//--- Verschiebung definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
```

© 2000-2025, MetaQuotes Ltd.

```
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iFrAMA)
      handle=iFrAMA(name,period,ma_period,ma_shift,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[3];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_shift;
      //--- Preistyp
      pars[2].type=TYPE_INT;
      pars[2].integer_value=applied_price;
      //--- Preistyp
      handle=IndicatorCreate(name,period,IND_FRAMA,3,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iFrAMA für das Paar %s/%s konnte nicht erstel
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren iFrAMA berechnet war, anzeigen
   short_name=StringFormat("iFrAMA(%s/%s, %d, %d, %s)",name,EnumToString(period),
                           ma_period,ma_shift,EnumToString(applied_price));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
 ;}
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
```

© 2000-2025, MetaQuotes Ltd.

```
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iFrAMA zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iFrAMABuffer größer als die Anzahl der Werte in iFrAMA auf symb
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iFrAMABuffer mit den Werten aus dem Indikator Fractal Adaptive Moving
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iFrAMABuffer,ma_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Fractal Adaptive Moving Average speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Indikator-Puffer aus dem Indikator iFrAMA ausfüllen              |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Fractal
                         int shift,         // Verschiebung
                         int ind_handle,    // Handle des Indikators iFrAMA
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iFrAMABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfül
   if(CopyBuffer(ind_handle,0,-shift,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iFrAMA kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iGator

Gibt das Handle des Indikators Gator zurück. Oscillator zeigt den Unterschied zwischen der blauen und der roten Linien des Alligators (oberer Chart) und Unterschied zwischen der roten und der grünen Linien (unterer Chart).

```
int  iGator(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  jaw_period,        // Periode für Berechnung der Kiefer
   int                  jaw_shift,         // horizontale Verschiebung der Kiefer
   int                  teeth_period,      // Periode für Berechnung der Zähne
   int                  teeth_shift,       // horizontale Verschiebung der Kiefer
   int                  lips_period,       // Periode für Berechnung der Zähne
   int                  lips_shift,        // horizontale Verschiebung der Lippen
   ENUM_MA_METHOD       ma_method,         // Glättungstyp
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

jaw_period

[in] Mittelungsperiode der blauen Linie (Kiefer des Alligators).

jaw_shift

[in] Verschiebung der blauen Linie des Alligators in Bezug auf das Preischart. Nicht direkt verbunden mit der visuellen Verschiebung des Histogramms des Indikators.

teeth_period

[in] Mittelungsperiode der roten Linie (Zähne des Alligators).

teeth_shift

[in] Verschiebung der roten Linie des Alligators in Bezug auf das Preischart. Nicht direkt verbunden mit der visuellen Verschiebung des Histogramms des Indikators.

lips_period

[in] Mittelungsperiode der grünen Linie (Lippen des Alligators).

lips_shift

[in] Verschiebung der gruenen Linie des Alligators in Bezug auf das Preischart. Nicht direkt verbunden mit der visuellen Verschiebung des Histogramms des Indikators.

ma_method

[in] Mittelungsmethode. Kann einer der Werte ENUM_MA_METHOD sein.

© 2000-2025, MetaQuotes Ltd.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - UPPER_HISTOGRAM, 1- Farbenpuffer des oberen Histogramms, 2 - LOWER_HISTOGRAM, 3- Farbenuffer des unteren Histogramms.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  Demo_iGator.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iGator."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Gator Osci

#property indicator_separate_window
#property indicator_buffers 4
#property indicator_plots   2
//--- GatorUp bauen
#property indicator_label1  "GatorUp"
#property indicator_type1   DRAW_COLOR_HISTOGRAM
#property indicator_color1  clrGreen, clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- GatorDown bauen
#property indicator_label2  "GatorDown"
#property indicator_type2   DRAW_COLOR_HISTOGRAM
#property indicator_color2  clrGreen, clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   Call_iGator,            // iGator verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iGator;       // Funktionstyp
input string                symbol=" ";             // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;  // Zeitrahmen
input int                   jaw_period=13;          // Zeitraum für die Kiefer-Linie
input int                   jaw_shift=8;            // Verschiebung der Kiefer-Linie
input int                   teeth_period=8;         // Zeitraum für die Zähne-Linie
input int                   teeth_shift=5;          // Verschiebung der Zähne-Linie
input int                   lips_period=5;          // Zeitraum für die Lippen-Linie
input int                   lips_shift=3;           // Verschiebung der Lippen-Linie
input ENUM_MA_METHOD        MA_method=MODE_SMMA;    // Methode der Mittelung der Alliga
input ENUM_APPLIED_PRICE    applied_price=PRICE_MEDIAN;// Typ der Preise aus der der Al
//--- Indicator-Puffer
double          GatorUpBuffer[];
double          GatorUpColors[];
double          GatorDownBuffer[];
double          GatorDownColors[];
//--- Eine Variable um Handle des Indikators iGator zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- Verschiebung für die obere und untere Histogramme
int shift;
//--- die Anzahl der Werte im Indikator Gator Oscillator gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,GatorUpBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,GatorUpColors,INDICATOR_COLOR_INDEX);
   SetIndexBuffer(2,GatorDownBuffer,INDICATOR_DATA);
   SetIndexBuffer(3,GatorDownColors,INDICATOR_COLOR_INDEX);
/*
  Alle Verschiebungen, die in den Parametern angegeben werden, gehören zum Indikator A
  Daher produzieren diese Verschiebungen keine Verschiebung des Indikators Gator, und
  auf dessen Werte die Indikatorwerte von Gator Oscillator konstruiert sind!
*/
//--- Berechnung der Verschiebung für den oberen und unteren Histogramm, die die Abwei
   shift=MathMin(jaw_shift,teeth_shift);
   PlotIndexSetInteger(0,PLOT_SHIFT,shift);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Trotz der Tatsache, dass der Indikator zwei Histogramme hat, die gleiche Verschi
   PlotIndexSetInteger(1,PLOT_SHIFT,shift);

//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iGator)
      handle=iGator(name,period,jaw_period,jaw_shift,teeth_period,teeth_shift,
                    lips_period,lips_shift,MA_method,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[8];
      //--- Perioden und Verschiebungen der Alligator-Linien
      pars[0].type=TYPE_INT;
      pars[0].integer_value=jaw_period;
      pars[1].type=TYPE_INT;
      pars[1].integer_value=jaw_shift;
      pars[2].type=TYPE_INT;
      pars[2].integer_value=teeth_period;
      pars[3].type=TYPE_INT;
      pars[3].integer_value=teeth_shift;
      pars[4].type=TYPE_INT;
      pars[4].integer_value=lips_period;
      pars[5].type=TYPE_INT;
      pars[5].integer_value=lips_shift;
      //--- Art der Glättung
      pars[6].type=TYPE_INT;
      pars[6].integer_value=MA_method;
      //--- Preistyp
      pars[7].type=TYPE_INT;
      pars[7].integer_value=applied_price;
      //--- Handle erstellen
      handle=IndicatorCreate(name,period,IND_GATOR,8,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iGator für das Paar %s/%s konnte nicht erstel
```

© 2000-2025, MetaQuotes Ltd.

```
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Gator Oscillator berechnet war, zeigen
   short_name=StringFormat("iGator(%s/%s, %d, %d ,%d, %d, %d, %d)",name,EnumToString(p
                           jaw_period,jaw_shift,teeth_period,teeth_shift,lips_period,l
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iGator zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array GatorUpBuffer größer als die Anzahl der Werte in iGator auf syn
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
```

© 2000-2025, MetaQuotes Ltd.

```
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator Gator Oscillator ausfüllen
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(GatorUpBuffer,GatorUpColors,GatorDownBuffer,GatorDownColo
      shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Gator Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iGator ausfüllen              |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &ups_buffer[],         // Indikator-Puffer für die o
                           double &up_color_buffer[],    // Indikator-Puffer für die F
                           double &downs_buffer[],       // Indikator-Puffer für die u
                           double &downs_color_buffer[], // Indikator-Puffer für die F
                           int u_shift,                  // Verschiebung für die obere
                           int ind_handle,               // Handle des Indikators iGat
                           int amount                    // Anzahl der Werte, die kopi
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays GatorUpBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfül
   if(CopyBuffer(ind_handle,0,-u_shift,amount,ups_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iGator kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays GatorUpColors mit Werten auf Indikator-Puffer mit Index 1 ausfül
   if(CopyBuffer(ind_handle,1,-u_shift,amount,up_color_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iGator kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Teil des Arrays GatorDownBuffer mit Werten auf Indikator-Puffer mit Index 2 aus
   if(CopyBuffer(ind_handle,2,-u_shift,amount,downs_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iGator kopiert werden, Fehler
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays GatorDownColors mit Werten auf Indikator-Puffer mit Index 3 aus
   if(CopyBuffer(ind_handle,3,-u_shift,amount,downs_color_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iGator kopiert werden, Fehler
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iIchimoku

Gibt das Handle des Indikators Ichimoku Kinko Hyo zurück.

```
int  iIchimoku(
   string           symbol,           // Symbolname
   ENUM_TIMEFRAMES  period,           // Periode
   int              tenkan_sen,       // Periode Tenkan-sen
   int              kijun_sen,        // Periode Kijun-sen
   int              senkou_span_b     // Periode Senkou Span B
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL das laufende Symbol.

period

[in] Wert er Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

tenkan_sen

[in] Mittelungsperiode Tenkan Sen.

kijun_sen

[in] Mittelungsperiode Kijun Sen.

senkou_span_b

[in] Mittelunfsperiode Senkou Span B.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - TENKANSEN_LINE, 1 - KIJUNSEN_LINE, 2 - SENKOUSPANA_LINE, 3 - SENKOUSPANB_LINE, 4 - CHIKOUSPAN_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iIchimoku.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Indikator-Puffer für den technischen Indikator iIchimoku."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Ichimoku Kinko Hyo."

#property indicator_chart_window
#property indicator_buffers 5
#property indicator_plots   4
//--- Tenkan_sen konstruieren
#property indicator_label1  "Tenkan_sen"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Kijun_sen konstruieren
#property indicator_label2  "Kijun_sen"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrBlue
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- Senkou_Span konstruieren
#property indicator_label3  "Senkou Span A;Senkou Span B" // Zwei Felder werden in Dat
#property indicator_type3   DRAW_FILLING
#property indicator_color3  clrSandyBrown, clrThistle
#property indicator_style3  STYLE_SOLID
#property indicator_width3  1
//--- Chinkou_Span konstruieren
#property indicator_label4  "Chinkou_Span"
#property indicator_type4   DRAW_LINE
#property indicator_color4  clrLime
#property indicator_style4  STYLE_SOLID
#property indicator_width4  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iIchimoku,         // iIchimoku verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation             type=Call_iIchimoku;       // Funktionstyp
input int                  tenkan_sen=9;              // Periode von Tenkan-sen
input int                  kijun_sen=26;              // Periode von Kijun-sen
input int                  senkou_span_b=52;          // Periode von Senkou Span B
input string               symbol=" ";                // Symbol
input ENUM_TIMEFRAMES      period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
```

© 2000-2025, MetaQuotes Ltd.

```
double         Tenkan_sen_Buffer[];
double         Kijun_sen_Buffer[];
double         Senkou_Span_A_Buffer[];
double         Senkou_Span_B_Buffer[];
double         Chinkou_Span_Buffer[];
//--- Eine Variable um Handle des Indikators iIchimoku zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Ichimoku Kinko Hyo gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,Tenkan_sen_Buffer,INDICATOR_DATA);
   SetIndexBuffer(1,Kijun_sen_Buffer,INDICATOR_DATA);
   SetIndexBuffer(2,Senkou_Span_A_Buffer,INDICATOR_DATA);
   SetIndexBuffer(3,Senkou_Span_B_Buffer,INDICATOR_DATA);
   SetIndexBuffer(4,Chinkou_Span_Buffer,INDICATOR_DATA);
//--- Die Verschiebung für Kanal Senkou Span auf kijun_sen Balken nach Zukunft definie
   PlotIndexSetInteger(2,PLOT_SHIFT,kijun_sen);
//--- Für Linie Chinkou Span, es ist nicht erforderlich, die Verschiebung anzugeben, d
//--- in iIchimoku mit Verschiebung gespeichert
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iIchimoku)
      handle=iIchimoku(name,period,tenkan_sen,kijun_sen,senkou_span_b);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[3];
      //--- Perioden und Verschiebungen der Alligator-Linien
      pars[0].type=TYPE_INT;
      pars[0].integer_value=tenkan_sen;
```

© 2000-2025, MetaQuotes Ltd.

```
      pars[1].type=TYPE_INT;
      pars[1].integer_value=kijun_sen;
      pars[2].type=TYPE_INT;
      pars[2].integer_value=senkou_span_b;
      //--- Handle erstellen
      handle=IndicatorCreate(name,period,IND_ICHIMOKU,3,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iBWMFI für das Paar %s/%s konnte nicht erstel
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
    ;}
//--- Das Paar, Symbol/Zeitrahmen, auf deren Ichimoku Kinko Hyo berechnet war, zeigen
   short_name=StringFormat("iIchimoku(%s/%s, %d, %d ,%d)",name,EnumToString(period),
                           tenkan_sen,kijun_sen,senkou_span_b);
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iIchimoku zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
```

© 2000-2025, MetaQuotes Ltd.

```
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array Tenkan_sen_Buffer größer als die Anzahl der Werte in iIchimoku
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator Ichimoku Kinko Hyo ausfüllen
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(Tenkan_sen_Buffer,Kijun_sen_Buffer,Senkou_Span_A_Buffer,S
      kijun_sen,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Ichimoku Kinko Hyo speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iIchimoku ausfüllen           |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &tenkan_sen_buffer[],     // Indikator-Puffer für di
                           double &kijun_sen_buffer[],      // Indikator-Puffer für di
                           double &senkou_span_A_buffer[],  // Indikator-Puffer für di
                           double &senkou_span_B_buffer[],  // Indikator-Puffer für di
                           double &chinkou_span_buffer[],   // Indikator-Puffer für Ch
                           int senkou_span_shift,           // Verschiebung der Linie
                           int ind_handle,                  // Handle des Indikators i
                           int amount                       // Anzahl der Werte, die k
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays Tenkan_sen_Buffer mit Werten auf Indikator-Puffer mit Index 0 au
   if(CopyBuffer(ind_handle,0,0,amount,tenkan_sen_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("1.Daten konnte nicht aus dem Indikator iIchimoku kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays Kijun_sen_Buffer mit Werten auf Indikator-Puffer mit Index 1 aus
   if(CopyBuffer(ind_handle,1,0,amount,kijun_sen_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("2.Daten konnte nicht aus dem Indikator iIchimoku kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays Chinkou_Span_Buffer mit Werten auf Indikator-Puffer mit Index 2
//--- Mit senkou_span_shift>0 Linie wird die Linie zu senkou_span_shift Balken nach Zu
   if(CopyBuffer(ind_handle,2,-senkou_span_shift,amount,senkou_span_A_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("3.Daten konnte nicht aus dem Indikator iIchimoku kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays Senkou_Span_A_Buffer mit Werten auf Indikator-Puffer mit Index 3
//--- Mit senkou_span_shift>0 Linie wird die Linie zu senkou_span_shift Balken nach Zu
   if(CopyBuffer(ind_handle,3,-senkou_span_shift,amount,senkou_span_B_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("4.Daten konnte nicht aus dem Indikator iIchimoku kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays Senkou_Span_B_Buffer mit Werten auf Indikator-Puffer mit Index 4
//--- Beim Kopieren der Linie Chinkou Span, es ist nicht erforderlich, die Verschiebun
//--- in iIchimoku mit Verschiebung gespeichert
   if(CopyBuffer(ind_handle,4,0,amount,chinkou_span_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("5.Daten konnte nicht aus dem Indikator iIchimoku kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
 ;}
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iBWMFI

Gibt das Handle des Indikators Market Facilitation Index zurück. Hat nur einen Puffer.

```
int  iBWMFI(
   string               symbol,              // Symbolname
   ENUM_TIMEFRAMES      period,              // Periode
   ENUM_APPLIED_VOLUME  applied_volume       // Volumentyp für Berechnung
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

applied_volume

[in] Das verwendete Volumen. Kann einer der Enumerationswerte ENUM_APPLIED_VOLUME sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                 Demo_iBWMFI.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iBWMFI."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- iBWMFI bauen
#property indicator_label1  "iBWMFI"
#property indicator_type1   DRAW_COLOR_HISTOGRAM
#property indicator_color1  clrLime,clrSaddleBrown,clrBlue,clrPink
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iBWMFI,            // iBWMFI verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iBWMFI;          // Funktionstyp
input ENUM_APPLIED_VOLUME   applied_volume=VOLUME_TICK;// Volumentyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double          iBWMFIBuffer[];
double          iBWMFIColors[];
//--- Eine Variable um Handle des Indikators iBWMFI zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Market Facilitation Index von Bill Williams ge
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,iBWMFIBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,iBWMFIColors,INDICATOR_COLOR_INDEX);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iBWMFI)
      handle=iBWMFI(name,period,applied_volume);
   else
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- die Struktur mit Werten von Indikatorparametern ausfüllen
       MqlParam pars[1];
       //--- Volumentyp
       pars[0].type=TYPE_INT;
       pars[0].integer_value=applied_volume;
       handle=IndicatorCreate(name,period,IND_BWMFI,1,pars);
      }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
      {
       //--- darüber schreiben und Nummer des Fehlers anzeigen
       PrintFormat("Handle des Indikators iBWMFI für das Paar %s/%s konnte nicht erstel
                   name,
                   EnumToString(period),
                   GetLastError());
       //--- Arbeit des Indikators ist früher geendet
       return(INIT_FAILED);
      }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Market Facilitation Index von Bill Willia
   short_name=StringFormat("iBWMFI(%s/%s, %s)",name,EnumToString(period),
                           EnumToString(applied_volume));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iBWMFI zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
      {
       PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
       return(0);
      }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array Tenkan_sen_Buffer größer als die Anzahl der Werte in iBWMFI auf
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator Market Facilitation Index von Bill W
//--- Wenn FillArraysFromBuffer false zurückgegeben hat, dann die Daten nicht bereit s
   if(!FillArraysFromBuffers(iBWMFIBuffer,iBWMFIColors,handle,values_to_copy)) return
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Market Facilitation Index von Bill Williams sp
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iBWMFI ausfüllen              |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &values[],    // Indikator-Puffer der Histogrammswer
                           double &colors[],    // Indikator-Puffer der Histogrammsfar
                           int ind_handle,      // Handle von iBWMFI
                           int amount           // Anzahl der Werte, die kopiert werde
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iBWMFIBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüll
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iBWMFI kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Teil des Arrays iBWMFIColors mit Werten auf Indikator-Puffer mit Index 1 ausfüll
   if(CopyBuffer(ind_handle,1,0,amount,colors)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iBWMFI kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iMomentum

Gibt das Handle des Indikators Momentum zurück. Hat nur einen Buffer.

```
int  iMomentum(
   string                symbol,             // Symbolname
   ENUM_TIMEFRAMES       period,             // Periode
   int                   mom_period,         // Glättungsperiode
   ENUM_APPLIED_PRICE    applied_price       // Preistyp oder handle
   );
```

Parameter

symbol

[in] symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

mom_period

[in] Periode(Anzahl der Bars) für Berechnung der Veränderung des Preises.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE sein oder Handle eines anderen Indikators.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iMomentum.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iMomentum."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Momentum."

#property indicator_separate_window
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_buffers 1
#property indicator_plots   1
//--- plot iMomentum
#property indicator_label1  "iMomentum"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrDodgerBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iMomentum,          // iMomentum verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iMomentum;       // Funktionstyp
input int                   mom_period=14;             // Periode von Momentum
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double          iMomentumBuffer[];
//--- Eine Variable um Handle des Indikators iMomentum zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Momentum gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iMomentumBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iMomentum)
      handle=iMomentum(name,period,mom_period,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[2];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=mom_period;
      //--- Preistyp
      pars[1].type=TYPE_INT;
      pars[1].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_MOMENTUM,2,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iMomentum für das Paar %s/%s konnte nicht ers
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar von Symbol/Zeitrahmen, auf deren Momentum berechnet war, anzeigen
   short_name=StringFormat("iMomentum(%s/%s, %d, %s)",name,EnumToString(period),
                           mom_period, EnumToString(applied_price));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iMomentum zu kopieren
```

© 2000-2025, MetaQuotes Ltd.

```
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iMomentumBuffer größer als die Anzahl der Werte in iMomentum au
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iMomentumBuffer mit den Werten aus dem Indikator Momentum ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iMomentumBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Momentum speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iMomentum ausfüllen           |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Momentur
                         int ind_handle,    // Handle des Indikators iMomentum
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iMomentumBuffer mit Werten auf Indikator-Puffer mit Index 0 aus
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iMomentum kopiert werden, Fehl
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iMFI

Berechnung Money Flow Index.

```
int  iMFI(
   string               symbol,             // Symbolname
   ENUM_TIMEFRAMES      period,             // Periode
   int                  ma_period,          // Glättungperiode
   ENUM_APPLIED_VOLUME  applied_volume      // Volumentyp für Berechnung
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Periode(Anzahl der Bars) für Berechnung des Indikators.

applied_volume

[in] Das verwendete Volumen. Kann einer der Enumerationswerte ENUM_APPLIED_VOLUME sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iMFI.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iMFI."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Money Flow Index."

#property indicator_separate_window
#property indicator_buffers 1
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_plots   1
//--- iMFI bauen
#property indicator_label1  "iMFI"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrDodgerBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Horizontale Ebenen im Indikatorsfenster
#property indicator_level1  20
#property indicator_level2  80
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iMFI,              // iMFI verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iMFI;               // Funktionstyp
input int                   ma_period=14;                 // Periode
input ENUM_APPLIED_VOLUME   applied_volume=VOLUME_TICK;   // Volumentyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double         iMFIBuffer[];
//--- Eine Variable um Handle des Indikators iMFI zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Money Flow Index gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iMFIBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iMFI)
      handle=iMFI(name,period,ma_period,applied_volume);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[2];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Volumentyp
      pars[1].type=TYPE_INT;
      pars[1].integer_value=applied_volume;
      handle=IndicatorCreate(name,period,IND_MFI,2,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iMFI für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Money Flow Index berechnet war, zeigen
   short_name=StringFormat("iMFI(%s/%s, %d, %s)",name,EnumToString(period),
                           ma_period, EnumToString(applied_volume));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Anzahl der Werte des Indikators iMFI zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iMFIBuffer größer als die Anzahl der Werte in iMFI auf symbol/p
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iMFIBuffer mit den Werten aus dem Indikator Money Flow Index ausfüller
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iMFIBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Money Flow Index speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iMFI ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],  // Indikator-Puffer der Werte von Money Fl
                         int ind_handle,    // Handle des Indikators iMFI
                         int amount         // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
```

© 2000-2025, MetaQuotes Ltd.

```
   ResetLastError();
//--- Teil des Arrays iMFIBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iMFI kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iMA

Gibt das Handle des Indikators von Mooving Average zurück. Hat nur einen Puffer.

```
int  iMA(
   string                symbol,            // Symbolname
   ENUM_TIMEFRAMES       period,            // Periode
   int                   ma_period,         // Mittelungsperiode
   int                   ma_shift,          // horizontale Verschiebung des Indikators
   ENUM_MA_METHOD        ma_method,         // Glaettungstyp
   ENUM_APPLIED_PRICE    applied_price      // Preistyp oder handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Glättungsmittelwertes.

ma_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

ma_method

[in] Mittelungsmethode. Kann einer der Werte ENUM_MA_METHOD sein.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                     Demo_iMA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iMA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Moving Ave

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iMA bauen
#property indicator_label1  "iMA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iMA,               // iMA verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iMA;               // Funktionstyp
input int                   ma_period=10;                // MA Periode
input int                   ma_shift=0;                  // Verschiebung
input ENUM_MA_METHOD        ma_method=MODE_SMA;          // Art der Glättung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE;   // Preistyp
input string                symbol=" ";                  // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;       // Zeitrahmen
//--- Indicator-Puffer
double         iMABuffer[];
//--- Eine Variable um Handle des Indikators iMA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Moving Average gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iMABuffer,INDICATOR_DATA);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Verschiebung definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iMA)
      handle=iMA(name,period,ma_period,ma_shift,ma_method,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_shift;
      //--- Art der Glättung
      pars[2].type=TYPE_INT;
      pars[2].integer_value=ma_method;
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_MA,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iMA für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Moving Average berechnet war, zeigen
   short_name=StringFormat("iMA(%s/%s, %d, %d, %s, %s)",name,EnumToString(period),
                           ma_period, ma_shift,EnumToString(ma_method),EnumToString(ap
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iMA zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iMABuffer größer als die Anzahl der Werte in iMA auf symbol/per
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array iMABuffer mit den Werten aus dem Indikator Moving Average ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iMABuffer,ma_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Moving Average speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iMA ausfüllen                 |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &values[],   // Indikator-Puffer der Werte von Moving
                         int shift,          // Verschiebung
                         int ind_handle,     // Handle des Indikators iMA
                         int amount          // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iMABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,-shift,amount,values)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iMA kopiert werden, Fehlercode
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iOsMA

Gibt das Handle des Indikators Moving Average of Oscillator zurück. Oscillator OsMA zeigt Unterschied zwischen den Werten MACD und seiner Signallinie. Hat nur einen Puffer.

```
int  iOsMA(
   string               symbol,              // Symbolname
   ENUM_TIMEFRAMES      period,              // Periode
   int                  fast_ema_period,     // Periode von Fast Moving Average
   int                  slow_ema_period,     // Periode von Slow Moving Average
   int                  signal_period,       // Mittelungsperiode der Differenz
   ENUM_APPLIED_PRICE   applied_price        // Preistyp oder handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

fast_ema_period

[in] Mittelungsperiode für Berechnung von Fast Moving Average.

slow_ema_period

[in] Mittelungsperiode für Berechnung von Slow Moving Average.

signal_period

[in] Mittelungsperiode für Berechnung von Signallinie.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

In einigen Systemen wird dieser Oscillator Histogramm MACD genannt.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  Demo_iOsMA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iOsMA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Moving Ave

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iOsMA bauen
#property indicator_label1  "iOsMA"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrSilver
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iOsMA,              // iOsMA verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iOsMA;           // Funktionstyp
input int                   fast_ema_period=12;        // Periode der schnellen MA
input int                   slow_ema_period=26;        // Periode der langsamen MA
input int                   signal_period=9;           // Mittelungsperiode der Unterso
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         iOsMABuffer[];
//--- Eine Variable um Handle des Indikators iAMA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Moving Average of Oscillator gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iOsMABuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iOsMA)
      handle=iOsMA(name,period,fast_ema_period,slow_ema_period,signal_period,applied_p
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- Schnelle Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=fast_ema_period;
      //--- Langsame Periode
      pars[1].type=TYPE_INT;
      pars[1].integer_value=slow_ema_period;
//--- Mittelungsperiode der Unterschied zwischen der schnellen und langsamen MA
      pars[2].type=TYPE_INT;
      pars[2].integer_value=signal_period;
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_OSMA,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iOsMA für das Paar %s/%s konnte nicht erstell
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Moving Average of Oscillator berechnet wa
   short_name=StringFormat("iOsMA(%s/%s,%d,%d,%d,%s)",name,EnumToString(period),
```

© 2000-2025, MetaQuotes Ltd.

```
                              fast_ema_period,slow_ema_period,signal_period,EnumToString
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iOsMA zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iOsMABuffer größer als die Anzahl der Werte in iOsMA auf symbol
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iOsMA ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iOsMABuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
```

© 2000-2025, MetaQuotes Ltd.

```
                              short_name,
                              values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Moving Average of Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iOsMA ausfüllen               |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &ama_buffer[],  // Indikator-Puffer der Werte von OsMA
                         int ind_handle,        // Handle des Indikators iOsMA
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iOsMABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfülle
   if(CopyBuffer(ind_handle,0,0,amount,ama_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iOsMA kopiert werden, Fehlercc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iMACD

Gibt das Handle des Indikators Moving Averages Convergence/Divergence zurück. In den Systemen, wo OsMA als Histogramm MACD bezeichnet wird, wird dieser Indikator als zwei Linien dargestellt. Im Client-Terminal wird Konvergenz/Divergenz von Moving Averages als Histogramm dargestellt.

```
int  iMACD(
   string               symbol,              // Symbolname
   ENUM_TIMEFRAMES      period,              // Periode
   int                  fast_ema_period,     // Periode für Berechnung von Fast average
   int                  slow_ema_period,     // Periode für Berechnung von Slow average
   int                  signal_period,       // Periode für Mittelung ihrer Differenz
   ENUM_APPLIED_PRICE   applied_price        // Preistyp oder handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

fast_ema_period

[in] Mittelungsperiode für Berechnung von Fast Moving Average.

slow_ema_period

[in] Mittelungsperiode für Berechnung von Slow Moving Average.

signal_period

[in] Mittelunfsperiode für Berechnung von Signallinie.

applied_price

[in] Der verwendete Preis. Kann einer der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - MAIN_LINE, 1 - SIGNAL_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iMACD.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iMACD."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator MACD."

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   2
//--- MACD bauen
#property indicator_label1  "MACD"
#property indicator_type1   DRAW_HISTOGRAM
#property indicator_color1  clrSilver
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Signal bauen
#property indicator_label2  "Signal"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrRed
#property indicator_style2  STYLE_DOT
#property indicator_width2  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iMACD,              // iMACD verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iMACD;           // Funktionstyp
input int                   fast_ema_period=12;        // Periode der schnellen MA
input int                   slow_ema_period=26;        // Periode der langsamen MA
input int                   signal_period=9;           // Mittelungsperiode der Untersc
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         MACDBuffer[];
double         SignalBuffer[];
//--- Eine Variable um Handle des Indikators iMACD zu speichern
int    handle;
//--- Variable für Speicherung
```

© 2000-2025, MetaQuotes Ltd.

```
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Moving Averages Convergence/Divergence gespeic
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,MACDBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,SignalBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iMACD)
      handle=iMACD(name,period,fast_ema_period,slow_ema_period,signal_period,applied_p
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- Schnelle Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=fast_ema_period;
      //--- Langsame Periode
      pars[1].type=TYPE_INT;
      pars[1].integer_value=slow_ema_period;
//--- Mittelungsperiode der Unterschied zwischen der schnellen und langsamen MA
      pars[2].type=TYPE_INT;
      pars[2].integer_value=signal_period;
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_MACD,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("Handle des Indikators iMACD für das Paar %s/%s konnte nicht erstell
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Moving Averages Convergence/Divergence be
   short_name=StringFormat("iMACD(%s/%s,%d,%d,%d,%s)",name,EnumToString(period),
                           fast_ema_period,slow_ema_period,signal_period,EnumToString
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iMACD zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array MACDBuffer größer als die Anzahl der Werte in iMACD auf symbol/
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iMACD ausfüllen
//--- Wenn FillArraysFromBuffers false zurückgegeben hat, dann die Daten nicht bereit
   if(!FillArraysFromBuffers(MACDBuffer,SignalBuffer,handle,values_to_copy)) return(0)
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Moving Averages Convergence/Divergence speiche
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
 ;}
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iMACD ausfüllen               |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &macd_buffer[],    // Indikator-Puffer der Werte vor
                           double &signal_buffer[],  // Indicator-Puffer von MACD Sign
                           int ind_handle,           // Handle des Indikators iMACD
                           int amount                // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iMACDBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfülle
   if(CopyBuffer(ind_handle,0,0,amount,macd_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iMACD kopiert werden, Fehlercc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }

//--- Teil des Arrays SignalBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfüll
   if(CopyBuffer(ind_handle,1,0,amount,signal_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iMACD kopiert werden, Fehlercc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iOBV

Gibt das Handle des Indikators On Balance Volume zurück. Hat nur einen Puffer.

```
int  iOBV(
   string                symbol,              // Symbolname
   ENUM_TIMEFRAMES       period,              // Periode
   ENUM_APPLIED_VOLUME   applied_volume       // Volumentyp für Berechnung
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

applied_volume

[in] The volume used. Can be any of the ENUM_APPLIED_VOLUME values.

Rückgabewert

Gibt handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                    Demo_iOBV.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iOBV."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//---  iOBV
#property indicator_label1  "iOBV"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iOBV ,             // iOBV verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iOBV;               // Funktionstyp
input ENUM_APPLIED_VOLUME   applied_volume=VOLUME_TICK;   // Volumentyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double          iOBVBuffer[];
//--- Eine Variable um Handle des Indikators iOBV zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator On Balance Volume gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iOBVBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iOBV)
      handle=iOBV(name,period,applied_volume);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
```

© 2000-2025, MetaQuotes Ltd.

```
      MqlParam pars[1];
      //--- Volumentyp
      pars[0].type=TYPE_INT;
      pars[0].integer_value=applied_volume;
      handle=IndicatorCreate(name,period,IND_OBV,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iOBV für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren On Balance Volume berechnet war, zeigen
   short_name=StringFormat("iOBV(%s/%s, %s)",name,EnumToString(period),
                           EnumToString(applied_volume));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iOBV zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
```

© 2000-2025, MetaQuotes Ltd.

```
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iOBVBuffer größer als die Anzahl der Werte in iOBV auf symbol/p
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iOBV ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iOBVBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator On Balance Volume speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iOBV ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &obv_buffer[],  // Indikator-Puffer der Werte von OBV
                         int ind_handle,        // Handle des Indikators iOBV
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iOBVBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,obv_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iOBV kopiert werden, Fehlercod
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iSAR

Gibt das Handle des Indikators Parabolic Stop and Reverse system zurück. Hat nur einen Puffer.

```
int  iSAR(
   string           symbol,      // Symbolname
   ENUM_TIMEFRAMES  period,      // Periode
   double           step,        // Schritt von Geschwindigkeitsinkerement - Beschleur
   double           maximum      // Maximales Stoplevel
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

step

[in] Inkrement von Stoplevel, gewöhnlich 0.02.

maximum

[in] Maximales Stoplevel, gewöhnlich 0.2.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iSAR.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iSAR."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Parabolic

#property indicator_chart_window
#property indicator_buffers 1
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_plots   1
//--- iSAR bauen
#property indicator_label1  "iSAR"
#property indicator_type1   DRAW_ARROW
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iSAR,              // iSAR verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iSAR;              // Funktionstyp
input double                step=0.02;                   // Schritt - die Beschleunigu
input double                maximum=0.2;                 // Maximalwert des Schritts
input string                symbol=" ";                  // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;       // Zeitrahmen
//--- Indicator-Puffer
double         iSARBuffer[];
//--- Eine Variable um Handle des Indikators iSAR zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Parabolic SAR gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iSARBuffer,INDICATOR_DATA);
//--- Code des Symbols aus Reihe von Wingdings für PLOT_ARROW Eigenschaft definieren
   PlotIndexSetInteger(0,PLOT_ARROW,159);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
```

© 2000-2025, MetaQuotes Ltd.

```
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iSAR)
      handle=iSAR(name,period,step,maximum);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[2];
      //--- Schrittwert
      pars[0].type=TYPE_DOUBLE;
      pars[0].double_value=step;
      //--- ser Grenzwert des Schritts, der in den Berechnungen verwendet werden kann
      pars[1].type=TYPE_DOUBLE;
      pars[1].double_value=maximum;
      handle=IndicatorCreate(name,period,IND_SAR,2,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iSAR für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Parabolic SAR berechnet war, zeigen
   short_name=StringFormat("iSAR(%s/%s, %G, %G)",name,EnumToString(period),
                           step,maximum);
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
 ;}
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Anzahl der Werte des Indikators iSAR zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iSARBufferer größer als die Anzahl der Werte in iSAR auf symbol
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iSAR ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iSARBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Parabolic SAR speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iSAR ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &sar_buffer[],  // Indikator-Puffer der Werte von Para
                         int ind_handle,        // Handle des Indikators iSAR
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Teil des Arrays iSARBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,sar_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iSAR kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iRSI

Gibt das Handle des Indikators Relative Strength Index zurück. Hat nur einen Puffer.

```
int  iRSI(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes,dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Indexes.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iRSI.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iRSI."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Relative Strength In

#property indicator_separate_window
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_buffers 1
#property indicator_plots   1
//--- iRSI bauen
#property indicator_label1  "iRSI"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrDodgerBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Limits, um die Werte im Indikatorsfenster anzuzeigen
#property indicator_maximum 100
#property indicator_minimum 0
//--- Horizontale Ebenen im Indikatorsfenster
#property indicator_level1  70.0
#property indicator_level2  30.0
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iRSI,              // iRSI verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation             type=Call_iRSI;               // Funktionstyp
input int                  ma_period=14;                 // MA Periode
input ENUM_APPLIED_PRICE   applied_price=PRICE_CLOSE;    // Preistyp
input string               symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES      period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double         iRSIBuffer[];
//--- Eine Variable um Handle des Indikators iRSI zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Relative Strength Index gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iRSIBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
```

© 2000-2025, MetaQuotes Ltd.

```
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iRSI)
      handle=iRSI(name,period,ma_period,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[2];
      //--- MA Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- ser Grenzwert des Schritts, der in den Berechnungen verwendet werden kann
      pars[1].type=TYPE_INT;
      pars[1].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_RSI,2,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iRSI für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Relative Strength Index berechnet war, ze
   short_name=StringFormat("iRSI(%s/%s, %d, %d)",name,EnumToString(period),
                           ma_period,applied_price);
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
```

© 2000-2025, MetaQuotes Ltd.

```
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iRSI zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iRSIBuffer größer als die Anzahl der Werte in iRSI auf symbol/p
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
    ;}
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator iRSI ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iRSIBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Relative Strength Index speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iRSI ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &rsi_buffer[],  // Indikator-Puffer der Werte von Rela
                         int ind_handle,        // Handle des Indikators iSAR
```

© 2000-2025, MetaQuotes Ltd.

```
                           int amount              // Anzahl der Werte, die kopiert werde
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iRSIBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,rsi_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iRSI kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iRVI

Gibt das Handle des Indikators Relative Vigor Index zurück.

```
int  iRVI(
   string           symbol,        // Symbolname
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, essen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Indexes.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - MAIN_LINE, 1 - SIGNAL_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                    Demo_iRVI.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iRVI."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Relative Vigor Index

#property indicator_separate_window
#property indicator_buffers 2
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_plots   2
//--- RVI bauen
#property indicator_label1  "RVI"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Signal bauen
#property indicator_label2  "Signal"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iRVI,              // iRVI verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iRVI;               // Funktionstyp
input int                   ma_period=10;                 // Zeitraum für die Berechnur
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double          RVIBuffer[];
double          SignalBuffer[];
//--- Eine Variable um Handle des Indikators iRVI zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Relative Vigor Index gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,RVIBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,SignalBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
```

© 2000-2025, MetaQuotes Ltd.

```
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iRVI)
      handle=iRVI(name,period,ma_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[1];
      //--- Zeitraum für die Berechnung
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      handle=IndicatorCreate(name,period,IND_RVI,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iRVI für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Relative Vigor Index berechnet war, zeige
   short_name=StringFormat("iRSI(%s/%s, %d, %d)",name,EnumToString(period),ma_period);
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- Anzahl der Werte des Indikators iRVI zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array RVIBuffer größer als die Anzahl der Werte in iRVI auf symbol/pe
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iRVI ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(RVIBuffer,SignalBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Relative Vigor Index speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iRVI ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &rvi_buffer[],     // Indikator-Puffer der Werte von I
                         double &signal_buffer[],  // Indikator-Puffer der Signallinie
                         int ind_handle,           // Handle des Indikators iRVI
                         int amount                // Anzahl der Werte, die kopiert we
                         )
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iRVIBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüller
   if(CopyBuffer(ind_handle,0,0,amount,rvi_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iRVI kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Teil des Arrays SignalBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfül
   if(CopyBuffer(ind_handle,1,0,amount,signal_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iRVI kopiert werden, Fehlercoc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iStdDev

Gibt das Handle des Indikators Standard Deviation zurück. Hat nur einen Puffer.

```
int  iStdDev(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   int                  ma_shift,          // horizontale Verschiebung des Indikators
   ENUM_MA_METHOD       ma_method,         // Glättungstyp
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet ads laufende Timeframe.

ma_period

[in] Mittelungsperiode für Berechnung des Indikators.

ma_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

ma_method

[in] Mittelungsmethode. Kann einer der Werte ENUM_MA_METHOD sein.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iStdDev.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iStdDev."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Standard I

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iStdDev bauen
#property indicator_label1  "iStdDev"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrMediumSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iStdDev,           // iStdDev verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iStdDev;         // Funktionstyp
input int                   ma_period=20;              // Mittelungszeitraum
input int                   ma_shift=0;                // Verschiebung
input ENUM_MA_METHOD        ma_method=MODE_SMA;        // Art der Glättung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         iStdDevBuffer[];
//--- Eine Variable um Handle des Indikators iStdDev zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Standard Deviation gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iStdDevBuffer,INDICATOR_DATA);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Verschiebung definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iStdDev)
      handle=iStdDev(name,period,ma_period,ma_shift,ma_method,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_shift;
      //--- Art der Glättung
      pars[2].type=TYPE_INT;
      pars[2].integer_value=ma_method;
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_STDDEV,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iStdDev für das Paar %s/%s konnte nicht erste
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Standard Deviation berechnet war, zeigen
   short_name=StringFormat("iStdDev(%s/%s, %d, %d, %s, %s)",name,EnumToString(period),
                           ma_period,ma_shift,EnumToString(ma_method),EnumToString(app
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iStdDev zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iStdDevBuffer größer als die Anzahl der Werte in iStdDev auf sy
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Standard Deviation ausfüllen
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iStdDevBuffer,ma_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Standard Deviation speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iStdDev ausfüllen             |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &std_buffer[],  // Indikator-Puffer der Linie von Star
                         int std_shift,         // Verschiebung der Linie Standard Dev
                         int ind_handle,        // Handle des Indikators iStdDev
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iStdDevBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfül
   if(CopyBuffer(ind_handle,0,-std_shift,amount,std_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iStdDev kopiert werden, Fehler
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iStochastic

Gibt das Handle des Indikators Stochastic Oscillator zurück.

```
int  iStochastic(
   string           symbol,          // Symbolname
   ENUM_TIMEFRAMES  period,          // Periode
   int              Kperiod,         // K-Periode (Anzahl der Bars für Berechnung)
   int              Dperiod,         // D-Periode (Periode der ersten Glättung)
   int              slowing,         // abschliessende Glättung
   ENUM_MA_METHOD   ma_method,       // Glättungstyp
   ENUM_STO_PRICE   price_field      // Berechnungsmethode der Stochastic
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES, 0 bedeutet das laufende Timeframe.

Kperiod

[in] K-Periode(Anzahl der Bars) für Berechnung der Linie %K.

Dperiod

[in] Mittelungsperiode für Berechnung der Linie %D.

slowing

[in] Wert von Verlangsamung.

ma_method

[in] Mittelungsmethode. Kann einer der Enumerationswerte ENUM_MA_METHOD sein.

price_field

[in] Parameter der Preiswahl für Berechnung. Kann einer Werte ENUM_STO_PRICE sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Hinweis

Puffernummern: 0 - MAIN_LINE, 1 - SIGNAL_LINE.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                Demo_iStochastic.mq5 |
```

© 2000-2025, MetaQuotes Ltd.

```
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iStochastic."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Stochastic

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   2
//--- Stochastic bauen
#property indicator_label1  "Stochastic"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrLightSeaGreen
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- Signal bauen
#property indicator_label2  "Signal"
#property indicator_type2   DRAW_LINE
#property indicator_color2  clrRed
#property indicator_style2  STYLE_SOLID
#property indicator_width2  1
//--- die Grenzwerte des Indikators definieren
#property indicator_minimum 0
#property indicator_maximum 100
//--- Horizontale Ebenen im Indikatorsfenster
#property indicator_level1  -100.0
#property indicator_level2  100.0
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iStochastic,       // iStochastic verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iStochastic;     // Funktionstyp
input int                   Kperiod=5;                 // K-Periode (Anzahl der Balken
input int                   Dperiod=3;                 // D-Periode (Periode der primär
input int                   slowing=3;                 // Periode der endgültigen Glätt
input ENUM_MA_METHOD        ma_method=MODE_SMA;        // Art der Glättung
input ENUM_STO_PRICE        price_field=STO_LOWHIGH;   // Methode zur Berechnung von St
```

© 2000-2025, MetaQuotes Ltd.

```
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double          StochasticBuffer[];
double          SignalBuffer[];
//--- Eine Variable um Handle des Indikators iStochastic zu speichern
int     handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Stochastic Oscillator gespeichert wird
int     bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Arrays zu den Indikator-Puffern
   SetIndexBuffer(0,StochasticBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,SignalBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iStochastic)
      handle=iStochastic(name,period,Kperiod,Dperiod,slowing,ma_method,price_field);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[5];
      //--- K-Periode für die Berechnung
      pars[0].type=TYPE_INT;
      pars[0].integer_value=Kperiod;
      //--- D-Periode für primäre Glättung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=Dperiod;
      //--- K-Periode für finale Glättung
      pars[2].type=TYPE_INT;
      pars[2].integer_value=slowing;
      //--- Art der Glättung
```

© 2000-2025, MetaQuotes Ltd.

```
      pars[3].type=TYPE_INT;
      pars[3].integer_value=ma_method;
      //--- Methode zur Berechnung von Stochastic
      pars[4].type=TYPE_INT;
      pars[4].integer_value=price_field;
      handle=IndicatorCreate(name,period,IND_STOCHASTIC,5,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iStochastic für das Paar %s/%s konnte nicht e
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Stochastic Oscillator berechnet war, zeig
   short_name=StringFormat("iStochastic(%s/%s, %d, %d, %d, %s, %s)",name,EnumToString
                           Kperiod,Dperiod,slowing,EnumToString(ma_method),EnumToStrin
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iStochastic zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
```

© 2000-2025, MetaQuotes Ltd.

```
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array StochasticBuffer größer als die Anzahl der Werte in iStochastic
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iStochastic ausfüllen
//--- Wenn FillArraysFromBuffers false zurückgegeben hat, dann die Daten nicht bereit
   if(!FillArraysFromBuffers(StochasticBuffer,SignalBuffer,handle,values_to_copy)) ret
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Stochastic Oscillator speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iStochastic ausfüllen         |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &main_buffer[],    // Indikator-Puffer der Werte vor
                           double &signal_buffer[],  // Indikator-Puffer der Signallin
                           int ind_handle,           // Handle des Indikators iStochas
                           int amount                // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays StochasticBuffer mit Werten auf Indikator-Puffer mit Index 0 aus
   if(CopyBuffer(ind_handle,MAIN_LINE,0,amount,main_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iStochastic kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Teil des Arrays SignalBuffer mit Werten auf Indikator-Puffer mit Index 1 ausfül
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CopyBuffer(ind_handle,SIGNAL_LINE,0,amount,signal_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iStochastic kopiert werden, Fe
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iTEMA

Gibt das Handle des Indikators Triple Exponential Moving Average zurück. Hat nur einen Puffer.

```
int  iTEMA(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   int                  ma_shift,          // horizontale Verschiebung des Indikators
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

ma_period

[in] Periode(Anzahl der Bars) für Berechnung des Indikators.

ma_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

applied_price

[in] Der verwendete Preis. Kann einer der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iTEMA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iTEMA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Triple Exp

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iTEMA bauen
#property indicator_label1  "iTEMA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iTEMA,              // iTEMA verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iTEMA;           // Funktionstyp
input int                   ma_period=14;              // Mittelungszeitraum
input int                   ma_shift=0;                // Verschiebung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         iTEMABuffer[];
//--- Eine Variable um Handle des Indikators iTEMA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Triple Exponential Moving Average gespeichert
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iTEMABuffer,INDICATOR_DATA);
//--- Verschiebung definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
```

© 2000-2025, MetaQuotes Ltd.

```
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iTEMA)
      handle=iTEMA(name,period,ma_period,ma_shift,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[3];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Verschiebung
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ma_shift;
      //--- Preistyp
      pars[2].type=TYPE_INT;
      pars[2].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_TEMA,3,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iTEMA für das Paar %s/%s konnte nicht erstell
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Triple Exponential Moving Average berechn
   short_name=StringFormat("iTEMA(%s/%s, %d, %d, %s)",name,EnumToString(period),
                           ma_period,ma_shift,EnumToString(applied_price));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
```

© 2000-2025, MetaQuotes Ltd.

```
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iTEMA zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iTEMABuffer größer als die Anzahl der Werte in iTEMA auf symbol
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Triple Exponential Moving Average aus
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iTEMABuffer,ma_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Triple Exponential Moving Average speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Indikator-Puffer aus dem Indikator iTEMA ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &tema_buffer[], // Indikator-Puffer der Werte von Trip
                         int t_shift,           // Verschiebung der Linie
                         int ind_handle,        // Handle des Indikators iTEMA
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iTEMABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfülle
   if(CopyBuffer(ind_handle,0,-t_shift,amount,tema_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iTEMA kopiert werden, Fehlerco
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iTriX

Gibt das Handle des Indikators Triple Exponential Moving Averages Oscillator zurück. Hat nur einen Puffer.

```
int  iTriX(
   string               symbol,            // Symbolname
   ENUM_TIMEFRAMES      period,            // Periode
   int                  ma_period,         // Mittelungsperiode
   ENUM_APPLIED_PRICE   applied_price      // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Aufzählunswerte ENUM_TIMEFRAMES sein, sein 0 bedeutet das laufende Timeframe.

ma_period

[in] Periode(Anzahl der Bars) für Berechnung des Indikators.

applied_price

[in] Der verwendete Preis. Kann eine der Preiskonstanten ENUM_APPLIED_PRICE oder Handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                  Demo_iTriX.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iTriX."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_buffers 1
#property indicator_plots   1
//--- iTriX bauen
#property indicator_label1  "iTriX"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrRed
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iTriX,             // iTriX verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation             type=Call_iTriX;           // Funktionstyp
input int                  ma_period=14;              // Periode
input ENUM_APPLIED_PRICE   applied_price=PRICE_CLOSE; // Preistyp
input string               symbol=" ";                // Symbol
input ENUM_TIMEFRAMES      period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         iTriXBuffer[];
//--- Eine Variable um Handle des Indikators iTriX zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Triple Exponential Moving Averages Oscillator
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iTriXBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iTriX)
      handle=iTriX(name,period,ma_period,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[2];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=ma_period;
      //--- Preistyp
      pars[1].type=TYPE_INT;
      pars[1].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_TRIX,2,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iTriX für das Paar %s/%s konnte nicht erstell
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Triple Exponential Moving Averages Oscil
   short_name=StringFormat("iTriX(%s/%s, %d, %s)",name,EnumToString(period),
                           ma_period,EnumToString(applied_price));
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iTriX zu kopieren
```

© 2000-2025, MetaQuotes Ltd.

```
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iTriXBuffer größer als die Anzahl der Werte in iTriX auf symbol
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Triple Exponential Moving Averages Os
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iTriXBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Triple Exponential Moving Averages Oscillator
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iTriX ausfüllen               |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &trix_buffer[], // Indikator-Puffer der Werte von Trip
                         int ind_handle,        // Handle des Indikators iTriX
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iTriXBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfülle
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CopyBuffer(ind_handle,0,0,amount,trix_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iTriX kopiert werden, Fehlerco
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iWPR

Gibt das Handle des Indikators Larry Williams' Percent Range zurück. Hat nur einen Puffer.

```
int  iWPR(
   string           symbol,          // Symbolname
   ENUM_TIMEFRAMES  period,          // Periode
   int              calc_period      // Mittelungsperiode
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

calc_period

[in] Periode(Anzahl der Bars) für Berechnung des Indikators.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                   Demo_iWPR.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iWPR."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iWPR bauen
#property indicator_label1  "iWPR"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrCyan
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//--- die Grenzwerte des Indikators definieren
#property indicator_minimum -100
#property indicator_maximum 0
//--- Horizontale Ebenen im Indikatorsfenster
#property indicator_level1  -20.0
#property indicator_level2  -80.0
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iWPR,              // iWPR verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iWPR;             // Funktionstyp
input int                   calc_period=14;             // Periode
input string                symbol=" ";                 // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;      // Zeitrahmen
//--- Indicator-Puffer
double          iWPRBuffer[];
//--- Eine Variable um Handle des Indikators iWPR zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Larry Williams' Percent Range gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iWPRBuffer,INDICATOR_DATA);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Erstellen ein Handel des Indikators
   if(type==Call_iWPR)
      handle=iWPR(name,period,calc_period);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[1];
      //--- Periode
      pars[0].type=TYPE_INT;
      pars[0].integer_value=calc_period;
      handle=IndicatorCreate(name,period,IND_WPR,1,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iWPR für das Paar %s/%s konnte nicht erstellt
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Larry Williams' Percent Range berechnet w
   short_name=StringFormat("iWPR(%s/%s, %d)",name,EnumToString(period),calc_period);
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iWPR zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iWPRBuffer größer als die Anzahl der Werte in iWPR auf symbol/p
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Larry Williams' Percent Range ausfüll
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iWPRBuffer,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Larry Williams' Percent Range speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iWPR ausfüllen                |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &wpr_buffer[], // Indikator-Puffer der Werte von Willi
                         int ind_handle,       // Handle des Indikators iWPR
                         int amount            // Anzahl der Werte, die kopiert werden
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iWPRBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüllen
   if(CopyBuffer(ind_handle,0,0,amount,wpr_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iWPR kopiert werden, Fehlercod
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iVIDyA

Gibt das Handle des Indikators Variable Index Dynamic Average zurück. Hat nur einen Puffer.

```
int  iVIDyA(
   string               symbol,           // Symbolname
   ENUM_TIMEFRAMES      period,           // Periode
   int                  cmo_period,       // Periode Chande Momentum
   int                  ema_period,       // EMA Glättungsperiode
   int                  ma_shift,         // horizontale Verschiebung des Indikators
   ENUM_APPLIED_PRICE   applied_price     // Preistyp oder Handle
   );
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

cmo_period

[in] Periode(Anzahl der Bars) für Berechnung Chande Momentum Oscillator.

ema_period

[in] Periode(Anzahl der Bars) EMA für Berechnung des Glättungfaktors.

ma_shift

[in] Verschiebung des Indikators in Bezug auf das Preischart.

applied_price

[in] Der verwendete Preis. Kann eine der Enumerationskonstanten ENUM_APPLIED_PRICE oder handle eines anderen Indikators sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                 Demo_iVIDyA.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
```

© 2000-2025, MetaQuotes Ltd.

```
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iVIDyA."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty
#property description "Alle andere Parameter sind wie im normalen Indikator Variable I

#property indicator_chart_window
#property indicator_buffers 1
#property indicator_plots   1
//--- iVIDyA bauen
#property indicator_label1  "iVIDyA"
#property indicator_type1   DRAW_LINE
#property indicator_color1  clrBlue
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iVIDyA,             // iVIDyA verwenden
   Call_IndicatorCreate     // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iVIDyA;          // Funktionstyp
input int                   cmo_period=15;             // Periode Chande Momentum
input int                   ema_period=12;             // Zeitraum von Glättungsfaktor
input int                   ma_shift=0;                // Verschiebung
input ENUM_APPLIED_PRICE    applied_price=PRICE_CLOSE; // Preistyp
input string                symbol=" ";                // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;     // Zeitrahmen
//--- Indicator-Puffer
double         iVIDyABuffer[];
//--- Eine Variable um Handle des Indikators iVIDyA zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Variable Index Dynamic Average gespeichert wi
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zum Indikator-Puffer
   SetIndexBuffer(0,iVIDyABuffer,INDICATOR_DATA);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Verschiebung definieren
   PlotIndexSetInteger(0,PLOT_SHIFT,ma_shift);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iVIDyA)
      handle=iVIDyA(name,period,cmo_period,ema_period,ma_shift,applied_price);
   else
     {
      //--- die Struktur mit Werten von Indikatorparametern ausfüllen
      MqlParam pars[4];
      //--- Periode Chande Momentum
      pars[0].type=TYPE_INT;
      pars[0].integer_value=cmo_period;
      //--- Zeitraum von Glättungsfaktor
      pars[1].type=TYPE_INT;
      pars[1].integer_value=ema_period;
      //--- Verschiebung
      pars[2].type=TYPE_INT;
      pars[2].integer_value=ma_shift;
      //--- Preistyp
      pars[3].type=TYPE_INT;
      pars[3].integer_value=applied_price;
      handle=IndicatorCreate(name,period,IND_VIDYA,4,pars);
     }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iVIDyA für das Paar %s/%s konnte nicht erstel
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     }
//--- Das Paar, Symbol/Zeitrahmen, auf deren Triple Exponential Moving Average berechn
   short_name=StringFormat("iVIDyA(%s/%s, %d, %d, %d, %s)",name,EnumToString(period),
                           cmo_period,ema_period,ma_shift,EnumToString(applied_price))
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iVIDyA zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iVIDyABuffer größer als die Anzahl der Werte in iVIDyA auf symb
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Das Array mit den Werten aus dem Indikator Variable Index Dynamic Average ausfül
//--- Wenn FillArrayFromBuffer false zurückgegeben hat, dann die Daten nicht bereit si
   if(!FillArrayFromBuffer(iVIDyABuffer,ma_shift,handle,values_to_copy)) return(0);
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Variable Index Dynamic Average speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iVIDyA ausfüllen              |
//+------------------------------------------------------------------+
bool FillArrayFromBuffer(double &vidya_buffer[],// Indikator-Puffer der Werte von Vari
                         int v_shift,           // Verschiebung der Linie
                         int ind_handle,        // Handle des Indikators iVIDyA
                         int amount             // Anzahl der Werte, die kopiert werde
                         )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iVIDyABuffer mit Werten auf Indikator-Puffer mit Index 0 ausfüll
   if(CopyBuffer(ind_handle,0,-v_shift,amount,vidya_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iVIDyA kopiert werden, Fehlerc
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

iVolumes

Gibt das Handle des Indikators der Volumen zurück. Hat nur einen Puffer.

```
int  iVolumes(
   string                symbol,              // Symbolname
   ENUM_TIMEFRAMES       period,              // Periode
   ENUM_APPLIED_VOLUME   applied_volume       // Volumentyp
   )
```

Parameter

symbol

[in] Symbolname des Instrumentes, dessen Daten für Berechnung des Indikators verwendet werden. NULL bedeutet das laufende Symbol.

period

[in] Wert der Periode kann einer der Enumerationswerte ENUM_TIMEFRAMES sein, 0 bedeutet das laufende Timeframe.

applied_volume

[in] Das verwendete Volumen. Kann einer der Enumerationswerte ENUM_APPLIED_VOLUME sein.

Rückgabewert

Gibt das Handle des angegebenen technischen Indikators zurück, beim Misserfolg gibt INVALID_HANDLE zurück. Um Computerspeicher aus nicht genutzten Indikatoren zu befreien, wird die Funktion IndicatorRelease() verwendet, zu deren Indikatorhandle passiert wird.

Beispiel:

```
//+------------------------------------------------------------------+
//|                                                 Demo_iVolumes.mq5 |
//|                        Copyright 2011, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Der Indikator zeigt, wie Daten erhalten werden sollen"
#property description "Indikator-Puffer für den technischen Indikator iVolumes."
#property description "Symbol und Zeitrahmen, in denen der Indikator berechnet wird,"
#property description "sind durch die Parameter symbol und period angegeben."
#property description "Erstellungsweise des Handles ist durch den 'type' Parameter (Ty

#property indicator_separate_window
#property indicator_buffers 2
#property indicator_plots   1
//--- iVolumes bauen
#property indicator_label1  "iVolumes"
#property indicator_type1   DRAW_COLOR_HISTOGRAM
#property indicator_color1  clrGreen, clrRed
```

© 2000-2025, MetaQuotes Ltd.

```
#property indicator_style1  STYLE_SOLID
#property indicator_width1  1
//+------------------------------------------------------------------+
//| Enumeration der Erstellungsweisen des Handles                    |
//+------------------------------------------------------------------+
enum Creation
  {
   Call_iVolumes,          // iVolumes verwenden
   Call_IndicatorCreate    // IndicatorCreate verwenden
  };
//--- Eingabeparameter
input Creation              type=Call_iVolumes;           // Funktionstyp
input ENUM_APPLIED_VOLUME   applied_volume=VOLUME_TICK;   // Volumentyp
input string                symbol=" ";                   // Symbol
input ENUM_TIMEFRAMES       period=PERIOD_CURRENT;        // Zeitrahmen
//--- Indicator-Puffer
double          iVolumesBuffer[];
double          iVolumesColors[];
//--- Eine Variable um Handle des Indikators iVolumes zu speichern
int    handle;
//--- Variable für Speicherung
string name=symbol;
//--- Name des Indikators auf dem Chart
string short_name;
//--- die Anzahl der Werte im Indikator Volumes gespeichert wird
int    bars_calculated=0;
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Bindung von Array zu den Indikator-Puffern
   SetIndexBuffer(0,iVolumesBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,iVolumesColors,INDICATOR_COLOR_INDEX);
//--- das Symbol, auf das der Indikator gebaut wird, definieren
   name=symbol;
//--- Leerzeichen aus dem linken und rechten löschen
   StringTrimRight(name);
   StringTrimLeft(name);
//--- Wenn nach diesem die Länge des String name ist Null
   if(StringLen(name)==0)
     {
      //--- das Symbol aus dem Chart, auf dem der Indikator läuft, nehmen
      name=_Symbol;
     }
//--- Erstellen ein Handel des Indikators
   if(type==Call_iVolumes)
      handle=iVolumes(name,period,applied_volume);
   else
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- die Struktur mit Werten von Indikatorparametern ausfüllen
       MqlParam pars[1];
       //--- Preistyp
       pars[0].type=TYPE_INT;
       pars[0].integer_value=applied_volume;
       handle=IndicatorCreate(name,period,IND_VOLUMES,1,pars);
      }
//--- Wenn Handle konnte nicht erstellt werden
   if(handle==INVALID_HANDLE)
     {
      //--- darüber schreiben und Nummer des Fehlers anzeigen
      PrintFormat("Handle des Indikators iVolumes für das Paar %s/%s konnte nicht erst
                  name,
                  EnumToString(period),
                  GetLastError());
      //--- Arbeit des Indikators ist früher geendet
      return(INIT_FAILED);
     ;}
//--- Das Paar von Symbol/Zeitrahmen, auf deren Volumes berechnet war, anzeigen
   short_name=StringFormat("iVolumes(%s/%s, %s)",name,EnumToString(period),EnumToStrin
   IndicatorSetString(INDICATOR_SHORTNAME,short_name);
//--- normale Initialisierung des Indikators
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
//--- Anzahl der Werte des Indikators iVolumes zu kopieren
   int values_to_copy;
//--- Anzahl der berechneten Werte im Indikator finden
   int calculated=BarsCalculated(handle);
   if(calculated<=0)
     {
      PrintFormat("BarsCalculated() hat %d zurückgegeben, Fehlercode ist %d",calculate
      return(0);
     }
//--- wenn dies der erste Start der Berechnung des Indikators ist oder die Anzahl der
```

© 2000-2025, MetaQuotes Ltd.

```
//--- oder wenn Sie brauchen den Indikator für zwei oder mehr Balken (was bedeutet, da
   if(prev_calculated==0 || calculated!=bars_calculated || rates_total>prev_calculated
     {
      //--- wenn Array iVolumesBuffer größer als die Anzahl der Werte in iVolumes auf
      //--- sonst kopieren wir weniger als Größe der Indikator-Puffer
      if(calculated>rates_total) values_to_copy=rates_total;
      else                       values_to_copy=calculated;
     }
   else
     {
      //--- dann ist unser Indikator nicht das erste Mal berechnet, und seit dem letzt
      //--- nicht mehr als ein Balken für die Berechnung hinzugefügt war
      values_to_copy=(rates_total-prev_calculated)+1;
     }
//--- Die Arrays mit den Werten aus dem Indikator iVolumes ausfüllen
//--- Wenn FillArraysFromBuffers false zurückgegeben hat, dann die Daten nicht bereit
   if(!FillArraysFromBuffers(iVolumesBuffer,iVolumesColors,handle,values_to_copy)) ret
//--- Nachricht bilden
   string comm=StringFormat("%s ==> Aktualisierte Werte im Indikator %s: %d",
                            TimeToString(TimeCurrent(),TIME_DATE|TIME_SECONDS),
                            short_name,
                            values_to_copy);
//--- Hilfsnachrichtung auf dem Chart anzeigen
   Comment(comm);
//--- die Anzahl der Werte im Indikator Volumes speichern
   bars_calculated=calculated;
//--- den Wert prev_calculated für den nächsten Anruf zurückgeben
   return(rates_total);
  }
//+------------------------------------------------------------------+
//| Indikator-Puffer aus dem Indikator iVolumes ausfüllen            |
//+------------------------------------------------------------------+
bool FillArraysFromBuffers(double &volume_buffer[],   // Indikator-Puffer der Werte vo
                           double &color_buffer[],    // Farbpuffer des Indikators
                           int ind_handle,            // Handle des Indikators iVolume
                           int amount                 // Anzahl der Werte, die kopiert
                           )
  {
//--- Fehlercode rücksetzen
   ResetLastError();
//--- Teil des Arrays iVolumesBuffer mit Werten auf Indikator-Puffer mit Index 0 ausfü
   if(CopyBuffer(ind_handle,0,0,amount,volume_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iVolumes kopiert werden, Fehle
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Teil des Arrays iVolumesColors mit Werten auf Indikator-Puffer mit Index 1 ausfü
```

© 2000-2025, MetaQuotes Ltd.

```
   if(CopyBuffer(ind_handle,1,0,amount,color_buffer)<0)
     {
      //--- wenn die Kopie fehlschlägt, Fehlercode anzeigen
      PrintFormat("Daten konnte nicht aus dem Indikator iVolumes kopiert werden, Fehle
      //--- Beenden mit Null-Ergebnis - dies bedeutet, dass der Indikator nicht berech
      return(false);
     }
//--- Alles gelang
   return(true);
  }
//+------------------------------------------------------------------+
//| Indicator deinitialization function                              |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
   if(handle!=INVALID_HANDLE)
      IndicatorRelease(handle);
//--- Das Chart nach der Löschung des Indikators leeren
   Comment("");
  }
```

© 2000-2025, MetaQuotes Ltd.

Arbeit mit Ergebnisse der Optimierung

Funktionen für die Organisation benutzerdefinierte Verarbeitung der Ergebnisse der Optimierung im Strategie-Tester. Sie können während der Optimierung in Test-Agenten sowie lokal in Expert Advisors und Skripte aufgerufen werden.

Wenn Sie einen Expert Advisor im Strategie-Tester laufen, können Sie Ihre eigenen Daten-Array auf den einfachen Typen oder einfachen Strukturen (sie enthalten keine Strings, Objekte der Klasse oder Objekte den dynamischen Arrays) erstellen. Dieser Datensatz kann mit Hilfe der Funktion FrameAdd() in einer speziellen Struktur, die ein Frame genannt wird, gespeichert werden. Bei der Optimierung eines Expert Advisor kann jeder Agent eine Reihe von Frames in das Terminal senden. Alle empfangenen Frames werden in der Datei *.MQD im Ordner terminal_directory/MQL5/Files/Tester genannt als Expert Advisor geschrieben. Sie werden in der Reihenfolge wie sie von den Agenten empfangen werden geschrieben. Empfang eines Frames im Client-Terminal dem Testagent erstellt ein Ereignis TesterPass.

Frames können im Speicher des Computers und in einer Datei mit dem angegebenen Namen gespeichert werden. Die MQL5 Sprache setzt keine Einschränkungen für die Anzahl der Frames.

Speicher- und Festplattenplatzbeschränkungen im MQL5 Cloud Network

Für Optimierungen, die im MQL5 Cloud Network ausgeführt werden, gilt folgende Einschränkung: Der Expert Advisor darf nicht mehr als 4 GB an Informationen auf die Festplatte schreiben oder mehr als 4 GB an RAM verwenden. Wird diese Grenze überschritten, kann der Netzwerk-Agent die Berechnung nicht korrekt abschließen, und Sie erhalten kein Ergebnis. Allerdings wird Ihnen die gesamte für die Berechnungen aufgewendete Zeit in Rechnung gestellt.

Wenn Sie Informationen aus jedem Optimierungsdurchlauf benötigen, senden Sie Frames ohne auf die Festplatte zu schreiben. Um die Verwendung von Dateioperationen in Expert Advisors während der Berechnungen im MQL5 Cloud Network zu vermeiden, können Sie die folgende Prüfung verwenden:

```
int handle=INVALID_HANDLE;
bool file_operations_allowed=true;
if(MQLInfoInteger(MQL_OPTIMIZATION) || MQLInfoInteger(MQL_FORWARD))
   file_operations_allowed=false;

if(file_operations_allowed)
  {
   ...
   handle=FileOpen(...);
   ...
  }
```

Funktion	Aktion
FrameFirst	Verschiebt einen Zeiger der Lesung des Frames auf den Anfang und rücksetzt den zuvor angegebenen Filter

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
FrameFilter	Setzt den Filter der Lesung des Frames und verschiebt den Zeiger auf den Anfang
FrameNext	Liest einen Frame und verschiebt den Zeiger auf das nächste
FrameInputs	Empfängt Input-Parameter, auf die der Frame gebildet ist
FrameAdd	Fügt einen Frame mit Daten hinzu
ParameterGetRange	Empfängt die Information über den Bereich der Werte und den Schritt der Veränderung für input-Variable bei der Optimierung eines Expert Advisors in Strategie-Tester
ParameterSetRange	Erstellt Regeln für die Verwendung von input-Variable bei der Optimierung eines Expert Advisors in Strategie-Tester: Wert, Schritt der Veränderung, Anfangs-und Endwerte

Sehen Sie auch

Teststatistik, Information über das ausgeführte MQL5-Programm MQL5 Cloud limitations_IT

© 2000-2025, MetaQuotes Ltd.

FrameFirst

Verschiebt einen Zeiger der Lesung des Frames auf den Anfang und rücksetzt den angegebenen Filter.

```
bool  FrameFirst();
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

FrameFilter

Setzt den Filter der Lesung des Frames und verschiebt den Zeiger auf den Anfang.

```
bool  FrameFilter(
   const string  name,          // Öffentlicher Name/Label
   long          id             // Öffentliche ID
   );
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Wenn ein leerer String als ersten Parameter übergeben wird, der Filter funktioniert nur mit einem numerischen Parameter, dh nur Frames mit der angegebenen id angezeigt werden. Wenn der Wert des zweiten Parameters ist ULONG_MAX, nur Text Filter arbeitet.

Aufruf von FrameFilter("", ULONG_MAX) ist mit dem Aufruf von FrameFirst() gleichbedeutend, das entspricht der Nutzung keiner Filter.

© 2000-2025, MetaQuotes Ltd.

FrameNext

Liest einen Frame und verschiebt den Zeiger auf das nächste. Es gibt zwei Varianten der Funktion.

1. Aufruf, um einen numerischen Wert zu erhalten

```
bool  FrameNext(
   ulong&   pass,      // Nummer des Durchgangs in der Optimierung, auf dem der Frame
   string&  name,      // Öffentlicher Name/Label
   long&    id,        // Öffentliche ID
   double&  value      // Wert
   );
```

2. Aufruf, um alle Daten des Frames zu erhalten

```
bool  FrameNext(
   ulong&   pass,      // Nummer des Durchgangs in der Optimierung, auf dem der Frame
   string&  name,      // Öffentlicher Name/Label
   long&    id,        // Öffentliche ID
   double&  value,     // Wert
   void&    data[]     // Array eines beliebigen Typs
   );
```

Parameter

pass

[out] Nummer des Durchgangs in der Optimierung im Strategie-Tester.

name

[out] Der Name des Bezeichners.

id

[out] Der Wert des Bezeichners.

value

[out] Ein einzelner numerischer Wert.

data

[out] Ein Array von beliebigen Typs.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

In der zweiten Version des Aufrufs, müssen Sie die empfangenen Daten im Array *data[]* korrekt verarbeiten.

© 2000-2025, MetaQuotes Ltd.

FrameInputs

Empfängt Input-Parameter, auf die der Frame mit dem angegebenen Durchgangsnummer gebildet ist.

```
bool  FrameInputs(
   ulong    pass,                // Nummer des Durchgangs in der Optimierung
   string&  parameters[],        // Array von String wie "parameterN=valueN"
   uint&    parameters_count     // Gesamtzahl der Parameter
   );
```

Parameter

pass

[in] Nummer des Durchgangs in der Optimierung im Strategie-Tester.

parameters

[out] Ein String-Array mit der Beschreibung der Namen und Parameterwerten.

parameters_count

[out] Die Anzahl der Elemente im Array *parameters[]*.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Nach Erhalt der Anzahl der Strings *parameters_count* im Array *parameters[]*, können Sie eine Schleife organisieren, um durch alle Datensätze zu durchlaufen. Diese helfen Ihnen dabei, die Werte der Eingabeparameter eines Expert Advisors für die angegebene Nummer des Durchgangs zu erhalten.

© 2000-2025, MetaQuotes Ltd.

FrameAdd

Fügt einen Frame mit Daten hinzu. Es gibt zwei Varianten der Funktion.

1. Hinzufügen von Daten aus einer Datei

```
bool  FrameAdd(
   const string  name,        // Öffentlicher Name/Label
   long          id,          // Öffentliche ID
   double        value,       // Wert
   const string  filename     // Name der Datendatei
   );
```

2. Hinzufügen von Daten aus einem Array eines beliebigen Typs

```
bool  FrameAdd(
   const string  name,        // Öffentlicher Name/Label
   long          id,          // Öffentliche ID
   double        value,       // Wert
   const void&   data[]       // Array eines beliebigen Typs
   );
```

Parameter

name

[in] Öffentliches Label des Frames. Es kann für Filterung in der FrameFilter()-Funktion verwendet werden.

id

[in] Ein öffentlicher Identifikator des Frames. Es kann für Filterung in der FrameFilter()-Funktion verwendet werden.

value

[in] Ein numerischer Wert für Schreibung im Frame. Es wird verwendet, um eine einzelnes Ergebnis des Durchlaufs wie in der OnTester()-Funktion zu übertragen.

filename

[in] Der Name der Datei, die Daten für Hinzufügung in den Frame enthält. Die Datei muss im Ordner MQL5/Files sein.

data

[in] Ein Array des beliebigen Typs für Schreibung in den Frame. Wird als Referenz übergeben.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

ParameterGetRange

Empfängt die Information über den Bereich der Werte und den Schritt der Veränderung für input-Variable bei der Optimierung eines Expert Advisors in Strategie-Tester. Es gibt zwei Varianten der Funktion.

1. Empfang von Information für den input-Parameter des Integer-Typs

```
bool  ParameterGetRange(
   const string  name,           // Name des (input-Variable) Parameters
   bool&         enable,         // Optimierung des Parameters ist erlaubt
   long&         value,          // Wert des Parameters
   long&         start,          // Anfangswert
   long&         step,           // Schritt der Veränderung
   long&         stop            // Endwert
   );
```

2. Empfang von Information für den input-Parameter des tatsächlichen Typs

```
bool  ParameterGetRange(
   const string  name,           // Name des (input-Variable) Parameters
   bool&         enable,         // Optimierung des Parameters ist erlaubt
   double&       value,          // Wert des Parameters
   double&       start,          // Anfangswert
   double&       step,           // Schritt der Veränderung
   double&       stop            // Endwert
   );
```

Parameter

name

[in] Identifikator der input-Variable. Diese Variablen sind externe Parameter des Programms, deren Werte man beim Start auf einem Chart oder während eines einzelnen Testens angeben kann.

enable

[out] Merkmal, dass dieser Parameter für Durchsuchen der Werte bei der Optimierung in Strategie-Tester verwendet werden kann.

value

[out] Wert des Parameters.

start

[out] Anfangswert des Parameters bei der Optimierung.

step

[out] Schritt der Veränderung des Parameters beim Durchsuchen seiner Werte.

stop

[out] Endwert des Parameters bei der Optimierung.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, verwenden Sie die Funktion GetLastError().

Hinweis

Die Funktion kann nur aus den Handlern OnTesterInit(), OnTesterPass() und OnTesterDeinit() aufgerufen werden. Sie ist für den Erhalt von Wert und Veränderungsbereich der Eingabeparameter des Expert Advisors bei der Optimierung in Strategie-Tester bestimmt.

Beim Aufruf in OnTesterInit() kann die erhaltene Information für Neudefinierung der Regeln für Durchsuchen jeder input-Variable mit Hilfe der Funktion ParameterSetRange() verwendet werden. So kann man die neuen Werte Start, Stop, Schritt einstellen und sogar diesen Parameter von der Optimierung unabhängig von den Einstellungen in Strategie-Tester völlig ausschließen. Dies ermöglicht es Ihnen eigene Skripten zu erstellen, um den Raum von Eingabeparametern bei der Optimierung zu verwalten, das heißt von der Optimierung einige Parameter je nach den Werten der wichtigsten Parameter des Expert Advisors auszuschließen.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#property description "Expert Advisor demonstriert die Funktion ParameterGetRange()."
#property description "Man muss in Strategie-Tester im Optimierung-Modus starten"
//--- input parameters
input int                 Input1=1;
input double              Input2=2.0;
input bool                Input3=false;
input ENUM_DAY_OF_WEEK    Input4=SUNDAY;

//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Expert Advisor ist nur für die Arbeit in Strategie-Tester bestimmt
   if(!MQL5InfoInteger(MQL5_OPTIMIZATION))
     {
      MessageBox("Man muss in Strategie-Tester im Optimierung-Modus starten!");
      //--- beenden wir die Arbeit des Expert Advisors vorfristig und entfernen wir au
      return(INIT_FAILED);
     }
//--- erfolgreicher Abschluss der Initialisierung
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| TesterInit function                                              |
//+------------------------------------------------------------------+
void OnTesterInit()
  {
//--- Beispiel für den input-Parameter des Typs long
   string name="Input1";
   bool enable;
   long par1,par1_start,par1_step,par1_stop;
   ParameterGetRange(name,enable,par1,par1_start,par1_step,par1_stop);
   Print("Der erste Parameter");
   PrintFormat("%s=%d  enable=%s  from %d to %d with step=%d",
               name,par1,(string)enable,par1_start,par1_stop,par1_step);
//--- Beispiel für den input-Parameter des Typs double
   name="Input2";
   double par2,par2_start,par2_step,par2_stop;
   ParameterGetRange(name,enable,par2,par2_start,par2_step,par2_stop);
   Print("Der zweite Parameter");
   PrintFormat("%s=%G  enable=%s  from %G to %G with step=%G",
               name,par2,(string)enable,par2_start,par2_stop,par2_step);

//--- Beispiel für den input-Parameter des Typs bool
   name="Input3";
   long par3,par3_start,par3_step,par3_stop;
   ParameterGetRange(name,enable,par3,par3_start,par3_step,par3_stop);
   Print("Der dritte Parameter");
   PrintFormat("%s=%s  enable=%s  from %s to %s",
               name,(string)par3,(string)enable,
               (string)par3_start,(string)par3_stop);
//--- Beispiel für den input-Parameter des Enumerationstyps
   name="Input4";
   long par4,par4_start,par4_step,par4_stop;
   ParameterGetRange(name,enable,par4,par4_start,par4_step,par4_stop);
   Print("Der vierte Parameter");
   PrintFormat("%s=%s  enable=%s  from %s to %s",
               name,EnumToString((ENUM_DAY_OF_WEEK)par4),(string)enable,
               EnumToString((ENUM_DAY_OF_WEEK)par4_start),
               EnumToString((ENUM_DAY_OF_WEEK)par4_stop));
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| TesterDeinit function                                            |
//+------------------------------------------------------------------+
void OnTesterDeinit()
  {
//--- diese Meldung wird nach dem Abschluss der Optimierung ausgegeben
   Print(__FUNCTION__," Optimisation completed");
  }
```

© 2000-2025, MetaQuotes Ltd.

ParameterSetRange

Erstellt Regeln für die Verwendung von input-Variable bei der Optimierung eines Expert Advisors in Strategie-Tester: Wert, Schritt der Veränderung, Anfangs-und Endwerte. Es gibt zwei Varianten der Funktion.

1. Einstellung von Werten für den input-Parameter des Integer-Typs

```
bool  ParameterSetRange(
   const string  name,          // Name des (input-Variable) Parameters
   bool          enable,        // Optimierung des Parameters erlauben
   long          value,         // Wert des Parameters
   long          start,         // Anfangswert
   long          step,          // Schritt der Veränderung
   long          stop           // Endwert
   );
```

2. Einstellung von Werten für den input-Parameter des tatsächlichen Typs

```
bool  ParameterSetRange(
   const string  name,          // Name des (input-Variable) Parameters
   bool          enable,        // Optimierung des Parameters erlauben
   double        value,         // Wert des Parameters
   double        start,         // Anfangswert
   double        step,          // Schritt der Veränderung
   double        stop           // Endwert
   );
```

Parameter

name

[in] Identifikator der input oder sinput Variable. Diese Variablen sind externe Parameter des Programms, deren Werte man beim Start angeben kann.

enable

[In] Aktivieren Sie diesen Parameter für Durchsuchen der Werte bei der Optimierung in Strategie-Tester.

value

[in] Wert des Parameters.

start

[in] Anfangswert des Parameters bei der Optimierung.

step

[in] Schritt der Veränderung des Parameters beim Durchsuchen seiner Werte.

stop

[in] Endwert des Parameters bei der Optimierung.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, verwenden Sie die Funktion GetLastError().

Hinweis

Die Funktion kann nur aus dem Handler OnTesterInit() beim Start der Optimierung im Strategie-Tester aufgerufen werden. Sie ist für die Einstellung von Bereich und Schritt der Veränderung des Parameters bestimmt, und erlaubt diesen Parameter von der Optimierung, unabhängig von den Einstellungen in Strategie-Tester, völlig auszuschließen. Auch erlaubt sie in der Optimierung sogar die mit dem Modifikator sinput erklärten Variablen zu verwenden.

Die Funktion ParameterSetRange() ermöglicht es Ihnen eigene Skripten der Optimierung des Expert-Advisors in Strategie-Tester je nach den Werten seinen wichtigsten Parameter zu erstellen, das heißt von der Optimierung die notwendigen Eingabeparameter ein- oder auszuschließen, und auch die notwendigen Bereich und Schritt der Veränderung einzustellen.

© 2000-2025, MetaQuotes Ltd.

Arbeit mit Eregnissen

Funktionen für die Arbeit mit Benutzerereignissen und Ereignissen des Timers . Ausser dieser Funktionen gibt es auch Sonderfunktionen für die Verarbeitung Vorwegparameter.

Funktion	Massnahme
EventSetMillisecondTimer	Startet den Generator der hochauflösenden Ereignissetimer mit einer Periode von weniger als 1 Sekunde für den aktuellen Chart
EventSetTimer	Läuft Generierer der Timerereignisse mit dem angegebenen Abruf für den laufenden Chart ab
EventKillTimer	Setzt auf dem laufenden Chart die Zeitgebergenerierung von Ereignissen still
EventChartCustom	Generiert das Benutzerereignis für den angegebenen Chart.

Sehen Sie auch

Typen der Chartereignisse

© 2000-2025, MetaQuotes Ltd.

EventSetMillisecondTimer

Informiert den Terminal, dass für diesen Expert oder Indikator Timer Ereignisse mit Periodizität kleiner als eine Sekunde generiert werden sollten.

```
bool  EventSetMillisecondTimer(
   int  milliseconds      // Die Anzahl der Millisekunden
   );
```

Optionen

milliseconds

[in] Die Anzahl der Millisekunden, die die Häufigkeit der Timer-Ereignisse definiert.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Fehlercode zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Diese Funktion ist für die diejenigen Fälle verwendet, wenn Sie hochauflösende Timer brauchen, d.h. wenn Sie Timer-Ereignisse häufiger als einmal pro Sekunde zu empfangen brauchen. Wenn Sie einen einfachen Timer mit Periode von mehr als 1 Sekunde brauchen, dann verwenden Sie EventSetTimer ().

Im Allgemeinen erhöht das kurze Intervall Testierungszeit, weil es erhöht die Anzahl der Anrufe von Timerereignis-Handler. In Echtzeitmodus werden Timer-Ereignisse nicht mehr als 1 Mal in 10-16 Millisekunden aufgrund von Hardware-Einschränkungen generiert.

Normalerweise muss diese Funktion von die Funktion OnInit() oder in Klassenkonstruktor aufgerufen werden. Um Ereignisse aus der Timer zu behandeln, muss ein Experten oder Indikator die Funktion OnTimer() haben.

Jeder Expert Advisor und jeder Indikator arbeitet mit seinem Timer und empfängt Ereignisse nur von ihm. Nach Abschluss des MQL5-Programs wird der Timer zerstört, wenn es erstellt wurde, aber wurde nicht ausschalten durch die Funktion EventKillTimer().

Für ein Programm kann nicht mehr als ein Timer gestartet werden. Jedes MQL5-Programm und jeder Chart hat seine eigenen Ereignis-Warteschlange, wo alle neu eingehenden Ereignisse geraten. Ist die Warteschlange bereits hat ein Timer-Ereignis oder dieses Ereignis ist in Bearbeitungslage, wird das neue Timer-Ereignis in Warteschlange vom MQL5-Programm nicht platziert.

© 2000-2025, MetaQuotes Ltd.

EventSetTimer

Gibt dem Client-Terminal an, dass Ereignisse für diesen Expert oder Indikator von Timer mit der angegebener Periodizitaet generiert werden muessen.

```
bool  EventSetTimer(
   int  seconds      // Anzahl der Sekunden
   );
```

Parameter

seconds

[in] Anzahl der Sekunden, die Periodizität des Vorkommens der Ereignisse vom Timer bestimmt.

Rückgabewert

Im Erfolgsfall gibt true zurück, anderenfalls false. Für die Erhaltung des Kodes des Fehlers muss die Funktuon GetLastError() aufgerufen werden.

Hinweis

Normalerweise muss diese Funktion aus der Funktion OnInit() oder im Konstruktor der Klasse aufgerufen werden. Für Verarbeitung der Ereignisse vom Timer muss der Expert oder der Indikator die Funktion OnTimer() haben.

Jeder Expert Advisor und jeder Indikator arbeitet nur mit seinem Timer und bekommt Ereignisse nur von ihm. Bei der Beendung des mql5-Programms wird Timer zwangsläufig vernichtet, falls er erzeugt wurde aber nicht durch die Funktion EventKillTimer() ausgeschaltet wurde.

Für jedes Programm kann nicht mehr als ein Timer ablaufen lassen. Jedes MQL5-Programm und jeder Chart hat seinen eigenen Warteschlange von Ereignisse, in der alle neu eingehenden Ereignisse hinzugefügt werden. Wenn ein Timer-Ereignis in der Warteschlange ist oder behandelt wird, wird die neue Timer-Ereignis nicht in der Warteschlange des MQL5-Programms platziert.

© 2000-2025, MetaQuotes Ltd.

EventKillTimer

Gibt dem Client-Terminal an, dass die Generierung der Ereignisse vom Timer für den Expert oder Indikator gestoppt werden muss.

```
void  EventKillTimer();
```

Rückgabewert

Keinen Rückgabewert.

Hinweis

Normalerweise muss diese Funktion aus der Funktion OnDeinit() aufgerufen werden, falls die Funktion EventSentTimer() in der Funktion OnInit() aufgerufen wurde. Oder sie muss vom Destruktor der Klasse aufgerufen werden, wenn die Funktion EventSetTimer() im Konstruktor dieser Klasse aufgerufen wird.

Jeder Expert und jeder Indikator arbeitet nur mit seine Timer und erhaelt Ereignisse nur von ihm. Bei der Beendung des mql5-Programms wird Timer zwangslaeufig vernichtet, falls er erzeugt wurde aber nicht durch die Funktion EventKillTimer() ausgeschaltet wurde.

© 2000-2025, MetaQuotes Ltd.

EventChartCustom

Generiert ein Benutzerereignis für den angegebenen Chart.

```
bool  EventChartCustom(
   long    chart_id,            // Identifikator des Charts-Ereignisrezipient
   ushort  custom_event_id,     // Identifikator des Ereignisses
   long    lparam,              // Parameter des Typs long
   double  dparam,              // Parameter des Typs double
   string  sparam               // Zeilenparameter des Ereignisses
   );
```

Parameter

chart_id

[in] Identifikatoren des Charts. 0 bedeutet den laufenden Chart.

custom_event_id

[in] Identifikator des Benutzerereignisses. Dieser Identifikator wird automatisch zum Wert CHARTEVENT_CUSTOM hinzugefügt und zum ganzzahligen Typ reduziert.

lparam

[in] Ereignisparameter des Typs long, der der Funktion OnChartEvent übertragen wird.

dparam

[in] Ereignisparameter des Typs double, der der Funktion OnChartEvent übertragen wird.

sparam

[in] Parameter des Ereignisses des Typs string, der der Funktion OnChartEvent übertragen wird. Wenn die Zeile länger als 63 Symbole, wird die Zeile reduziert.

Rückgabewert

Gibt true zurück, wenn ein benutzerdefiniertes Ereignis wurde erfolgreich in der Warteschlange von Ereignissen des Charts-Ereignisrezipient, platziert. Im Falle eines Fehlers, wird false zurückgegeben. Verwenden Sie GetLastError(), um einen Fehlercode zu bekommen.

Hinweis

Expert oder Indikator, angehängt zum angegebenen Chart, verarbeitet dieses Ereignis durch die Funktion OnChartEvent(int event_id, long& lparam, double& dparam, string& sparam).

Für jede Art von Ereignis, Eingabeparameter der Funktion OnChartEvent() haben bestimmte Werte, die nötig sind, um das Ereignis zu behandeln.Die folgende Tabelle listet die Ereignisse und Werte, die durch Parameter übergeben werden.

Ereignis	Wert des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
Ereignis des Keyboard Klickens	CHARTEVENT_KEYDOWN	Kode der betätigten Taste	Die Anzahl der Tastenanschläge, die generiert	Der String-Wert eines Bit-Maske, die den Status

© 2000-2025, MetaQuotes Ltd.

Ereignis	Wert des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
			werden, während diese Taste in gedrücktem Zustand war	der Tastatur-Tasten beschreibt
Ereignisse der Maus-Bewegung und Mausklicks (wenn die Eigenschaft CHART_EVENT_MOUSE_MOVE=true ist für den Chart eingegeben)	CHARTEVENT_MOUSE_MOVE	X-Koordinate	Y-Koordinate	Der String-Wert einer Bit-Maske, der den Status der Maustasten beschreibt
Ereignis der Erzeugung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_CREATE=true ist für den Chart eingegeben)	CHARTEVENT_OBJECT_CREATE	–	–	Name des erzeugten graphischen Objektes
Ereignis der Veränderung der Eigenschaften des Objektes durch Dialog der Eigenschaften	CHARTEVENT_OBJECT_CHANGE	–	–	Name des veränderten graphischen Objektes
Ereignis der Löschung des graphischen Objektes (wenn die Eigenschaft CHART_EVENT_OBJECT_DELETE=true ist für den Chart eingegeben)	CHARTEVENT_OBJECT_DELETE	–	–	Name des entfernten graphischen Objektes
Ereignis des Mausklicks auf der graphischen Darstellung	CHARTEVENT_CLICK	X-Koordinate	Y-Koordinate	–

© 2000-2025, MetaQuotes Ltd.

Ereignis	Wert des Parameters id	Wert des Parameters lparam	Wert des Parameters dparam	Wert des Parameters sparam
Ereignis des Mausklicks auf dem graphischen Objekt	CHARTEVENT_OBJECT_CLICK	X-Koordinate	Y-Koordinate	Name des graphischen Objektes, auf dem das Ereignis geschehen ist
Ereignis der Verschiebung des graphischen Objektes per Mausklick	CHARTEVENT_OBJECT_DRAG	—	—	Name des verschobenen graphischen Objektes
Ereignis der Beendung der Textbearbeitung im Eingabefeld des graphischen Objektes "Eingabefeld"	CHARTEVENT_OBJECT_ENDEDIT	—	—	Name des graphischen Objektes "Eingabefeld", in dem Editieren des Textes beendet wurde
Ereignis der Chart-Veränderung	CHARTEVENT_CHART_CHANGE	—	—	—
Benutzerereignis mit Nummer N	CHARTEVENT_CUSTOM+N	Wert, vorgegeben durch die Funktion EventChartCustom()	Wert, vorgegeben durch die Funktion EventChartCustom()	Wert, vorgegeben durch die Funktion EventChartCustom()

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                            ButtonClickExpert.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "2009, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"

string buttonID="Button";
string labelID="Info";
int broadcastEventID=5000;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- Erzeugen wir einen Button für übertragung der Benutzerereignisse
   ObjectCreate(0,buttonID,OBJ_BUTTON,0,100,100);
   ObjectSetInteger(0,buttonID,OBJPROP_COLOR,clrWhite);
   ObjectSetInteger(0,buttonID,OBJPROP_BGCOLOR,clrGray);
   ObjectSetInteger(0,buttonID,OBJPROP_XDISTANCE,100);
   ObjectSetInteger(0,buttonID,OBJPROP_YDISTANCE,100);
   ObjectSetInteger(0,buttonID,OBJPROP_XSIZE,200);
   ObjectSetInteger(0,buttonID,OBJPROP_YSIZE,50);
   ObjectSetString(0,buttonID,OBJPROP_FONT,"Arial");
   ObjectSetString(0,buttonID,OBJPROP_TEXT,"Button");
   ObjectSetInteger(0,buttonID,OBJPROP_FONTSIZE,10);
   ObjectSetInteger(0,buttonID,OBJPROP_SELECTABLE,0);

//--- Erzeugen wir eine Marke für Ausgabe der Information
   ObjectCreate(0,labelID,OBJ_LABEL,0,100,100);
   ObjectSetInteger(0,labelID,OBJPROP_COLOR,clrRed);
   ObjectSetInteger(0,labelID,OBJPROP_XDISTANCE,100);
   ObjectSetInteger(0,labelID,OBJPROP_YDISTANCE,50);
   ObjectSetString(0,labelID,OBJPROP_FONT,"Trebuchet MS");
   ObjectSetString(0,labelID,OBJPROP_TEXT,"Keine Information");
   ObjectSetInteger(0,labelID,OBJPROP_FONTSIZE,20);
   ObjectSetInteger(0,labelID,OBJPROP_SELECTABLE,0);

//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   ObjectDelete(0,buttonID);
   ObjectDelete(0,labelID);
  }
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//---

  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnChartEvent(const int id,
                  const long &lparam,
                  const double &dparam,
                  const string &sparam)
  {
//--- Pruefen wir das Ereignis durch Drücken des Mausbuttons
   if(id==CHARTEVENT_OBJECT_CLICK)
     {
      string clickedChartObject=sparam;
      //--- wenn Sie drücken das Objekt mit dem Namen buttonID
      if(clickedChartObject==buttonID)
        {
         //--- Buttonstatus - gedrückt oder nicht
         bool selected=ObjectGetInteger(0,buttonID,OBJPROP_STATE);
         //--- geben wir in log Debugging-Nachricht aus
         Print("Knopf gedrückt = ",selected);
         int customEventID; // Nummer des Benutzerereignisses für Senden
         string message;    // Nachricht für Senden im Ereignis
         //--- wenn Knopf gedrückt ist
         if(selected)
           {
            message="Knopf gedrückt";
            customEventID=CHARTEVENT_CUSTOM+1;
           }
         else // Knopf nicht gedrückt
           {
            message="Knopf nicht gedrückt";
            customEventID=CHARTEVENT_CUSTOM+999;
           }
         //--- senden wir Benutzerereignis "unserem" Chart
         EventChartCustom(0,customEventID-CHARTEVENT_CUSTOM,0,0,message);
         //--- senden wir eine Nachricht an alle offenen Charts
         BroadcastEvent(ChartID(),0,"Broadcast Message");
         //--- Debugging-Nachricht
         Print("Ereignis mit ID ist gesendet = ",customEventID);
        }
      ChartRedraw();// zeichnen wir alle Objekte auf dem Chart erneut
     }

//--- pruefen wir, ob das Ereignis zu Benutzerereignissen gehört
   if(id>CHARTEVENT_CUSTOM)
     {
      if(id==broadcastEventID)
        {
         Print("Erhalten wir Broadcast-Nachricht vom Chart mit id = "+lparam);
        }
      else
        {
         //--- lesen wir eine Textnachricht im Ereignis
         string info=sparam;
         Print("Das Benutzerereignis mit ID wird verarbeitet = ",id);
         //--- geben wir eine Nachricht in der Marke aus
         ObjectSetString(0,labelID,OBJPROP_TEXT,sparam);
         ChartRedraw();// zeichnen wir alle Objekte auf dem Chart zwangsläufig erneut
        }
     }
  }
//+------------------------------------------------------------------+
//| Broadcast-Nachricht allen offenen Charts senden                  |
//+------------------------------------------------------------------+
void BroadcastEvent(long lparam,double dparam,string sparam)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   int eventID=broadcastEventID-CHARTEVENT_CUSTOM;
   long currChart=ChartFirst();
   int i=0;
   while(i<CHARTS_MAX)           // Sie haben bestimmt nicht mehr als CHARTS_MAX offene
     {
      EventChartCustom(currChart,eventID,lparam,dparam,sparam);
      currChart=ChartNext(currChart); // auf Grund des vorigen Charts erhalten wir ein
      if(currChart==-1) break;         // das Ende der Chartliste erreicht
      i++;                             // Nicht vergessen, Counter zu erhoehen
     }
  }
//+------------------------------------------------------------------+
```

Sehen Sie auch

Ereignisse des Client-Terminals, Ereignisbearbeiter

© 2000-2025, MetaQuotes Ltd.

Arbeit mit OpenCL

Programme auf OpenCLsind entwickelt, um Berechnungen auf Grafikkarten mit Unterstützung für OpenCL 1.1 oder höher auszuführen. Moderne Grafikkarten enthalten hunderte von kleinen, spezialisierten Prozessoren, die gleichzeitig einfache mathematische Operationen auf eingehenden Datenströme ausführen können. Die Sprache OpenCL übernimmt die Organisation der Parallel-Computing und ermöglicht eine hohe Geschwindigkeit für eine bestimmte Klasse von Aufgaben.

Bei einigen Grafikkarten ist der Modus der Arbeit mit double-Zahlen deaktiviert, das zum Kompilierung-Fehler 5105 führen kann. Um Unterstützung von double-Zahlen zu aktivieren, fügen Sie eine Direktive #pragma OPENCL EXTENSION cl_khr_fp64 : enable in den Text des OpenCL-Programms hinzu. Aber wenn die Grafikkarte den double-Typ nicht unterstützt, die Einbeziehung dieser Direktive wird nicht helfen.

Es ist zu empfehlen, den OpenCL-Code in separaten CL-Dateien zu schreiben, die man später mithilfe von Ressourcenvariablen einem MQL5-Programm hinzufügen kann.

Behandlung von Fehlern in OpenCL-Programmen

Um Informationen über den letzten Fehler in einem OpenCL-Programm zu erhalten, verwenden Sie die Funktionen CLGetInfoInteger und CLGetInfoString, die es ermöglichen, den Fehlercode und die Textbeschreibung zu erhalten.

Der letzte Fehler von: Um den letzten OpenCL-Fehler zu erhalten, rufen Sie CLGetInfoInteger auf, wobei der Parameter *handle* ignoriert wird (kann auf Null gesetzt werden). Fehlerbeschreibung: https://registry.khronos.org/OpenCL/specs/3.0-unified/html/OpenCL_API.html#CL_SUCCESS.

Bei einem unbekannten Fehlercode wird die Zeichenkette "unbekannter OpenCL-Fehler N" zurückgegeben, wobei N der Fehlercode ist. Beispiel:

```
//--- der erste Parameter „handle“ wird ignoriert, wenn man den letzten Fehlercode abr
int code = (int)CLGetInfoInteger(0,CL_LAST_ERROR);
```

Textbeschreibung des OpenCL-Fehlers: Um den letzten OpenCL-Fehler zu erhalten, rufen Sie CLGetInfoString auf. Der Fehlercode sollte über den Parameter *handle* übergeben werden.

Fehlerbeschreibung: https://registry.khronos.org/OpenCL/specs/3.0-unified/html/OpenCL_API.html#CL_SUCCESS. Wenn CL_LAST_ERROR anstelle des Fehlercodes übergeben wird, dann gibt die Funktion die Beschreibung des letzten Fehlers zurück. Zum Beispiel:

```
//--- liefert den Code des letzten OpenCL-Fehlers
int    code = (int)CLGetInfoInteger(0,CL_LAST_ERROR);
string desc; // um eine Fehlertextbeschreibung zu erhalten

//--- Verwenden des Fehlercodes, um die Fehlertextbeschreibung zu erhalten
if (!CLGetInfoString(code,CL_ERROR_DESCRIPTION,desc))
 desc = "Die Fehlerbeschreibung von OpenCL konnte nicht angerufen werden," + (string)
Print(desc);

//--- um die Beschreibung des letzten OpenCL-Fehlers zu erhalten, ohne zuerst den Code
if (!CLGetInfoString(CL_LAST_ERROR,CL_ERROR_DESCRIPTION,desc))
 desc = "Die Fehlerbeschreibung von OpenCL konnte nicht angerufen werden," + (string)
```

© 2000-2025, MetaQuotes Ltd.

```
Print(desc);
```

Bislang wird der Name der internen Enumeration als Fehlerbeschreibung angegeben. Ihre Dekodierung finden Sie hier: https://registry.khronos.org/OpenCL/specs/3.0-unified/html/OpenCL_API.html#CL_SUCCESS. Der Wert CL_INVALID_KERNEL_ARGS bedeutet zum Beispiel "Wird beim Enqueuing eines Kernels zurückgegeben, wenn einige Kernel-Argumente nicht gesetzt oder ungültig sind."

Funktionen zum Ausführen von Programmen auf OpenCL:

Funktion	Aktion
CLHandleType	Gibt den Typ des OpenCL-Handles als ein Wert aus Enumeration ENUM_OPENCL_HANDLE_TYPE zurück
CLGetInfoInteger	Gibt den Wer des ganzzahligen Eigenschafts für ein OpenCL-Objekt oder Gerät zurück
CLContextCreate	Erstellt einen Kontext für OpenCL
CLContextFree	Löscht den Kontext von OpenCL
CLGetDeviceInfo	Erhielt die Geräteeigenschaft aus dem OpenCL-Treiber
CLProgramCreate	Erstellt ein OpenCL-Programm aus dem Quellcode
CLProgramFree	Löscht das OpenCL-Programm
CLKernelCreate	Erstellt eine Funktion des Starts von OpenCL
CLKernelFree	Löscht die Funktion des Starts von OpenCL
CLSetKernelArg	Setzt den Parameter für die OpenCL-Funktion
CLSetKernelArgMem	Setzt einen OpenCL-Puffer als ein Parameter der OpenCL-Funktion
CLSetKernelArgMemLocal	Setzt den lokalen Puffer als Argument der kernel-Funktion
CLBufferCreate	Erstellt einen OpenCL-Puffer
CLBufferFree	Löscht einen OpenCL-Puffer
CLBufferWrite	Schreibt ein Array in einen OpenCL-Puffer
CLBufferRead	Liest einen OpenCL-Puffer in ein Array
CLExecute	Führt das OpenCL-Programm aus
CLExecutionStatus	Gibt den Ausführungsstatus eines OpenCL Programms zurück

Sehen Sie auch

OpenCL, Ressourcen

© 2000-2025, MetaQuotes Ltd.

CLHandleType

Gibt den Typ des OpenCL-Handles als ein Wert aus Enumeration ENUM_OPENCL_HANDLE_TYPE zurück.

```
ENUM_OPENCL_HANDLE_TYPE  CLHandleType(
   int  handle     // Handle des OpenCL-Objekts
   );
```

Parameter

handle

[in] Handle auf ein OpenCL-Objekt: Kontext, Kernel, Puffer oder OpenCL-Programm.

Rückgabewert

Typ des OpenCL-Handles als ein Wert aus Enumeration ENUM_OPENCL_HANDLE_TYPE.

ENUM_OPENCL_HANDLE_TYPE

Identifier	Beschreibung
OPENCL_INVALID	Ungültiges Handle
OPENCL_CONTEXT	Handle von OpenCL-Kontext
OPENCL_PROGRAM	Handle von OpenCL-Programm
OPENCL_KERNEL	Handle von OpenCL-Kernel
OPENCL_BUFFER	Handle von OpenCL-Puffer

© 2000-2025, MetaQuotes Ltd.

CLGetInfoInteger

Gibt den Wer des ganzzahligen Eigenschafts für ein OpenCL-Objekt oder Gerät zurück.

```
long  CLGetInfoInteger(
   int  handle,                          // Handle des OpenCL-Objekts oder Nummer des
   ENUM_OPENCL_PROPERTY_INTEGER  prop    // Die angeforderte Eigenschaft
   );
```

Parameter

handle

[in] Handle auf ein OpenCL-Objekt oder Nummer des OpenCL-Geräts. Nummerierung der OpenCL-Geräte startet mit Null.

prop

[in] Typ der angeforderten Eigenschaft von der Listing ENUM_OPENCL_PROPERTY_INTEGER, deren Wert Sie möchten erhalten.

Rückgabewert

Der Wert der angegebenen Eigenschaft beim Erfolg oder -1 bei einem Fehler. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

ENUM_OPENCL_PROPERTY_INTEGER

Identifikator	Beschreibung	Typ
CL_DEVICE_COUNT	Die Anzahl der Geräte mit Unterstützung für OpenCL. Diese Eigenschaft erfordert nicht Eingabe des ersten Parameter, ist es möglich, einen Null-Wert für den *handle* Parameter anzugeben.	int
CL_DEVICE_TYPE	Gerätetyp	ENUM_CL_DEVICE_TYPE
CL_DEVICE_VENDOR_ID	Eindeutiger Identifikator des Verkäufers	uint
CL_DEVICE_MAX_COMPUTE_UNITS	Anzahl der parallel berechneten Aufgaben im OpenCL-Gerät. Eine Arbeitsgruppe erfüllt eine Rechenaufgabe. Der Mindestwert ist 1.	uint
CL_DEVICE_MAX_CLOCK_FREQUENCY	Maximale Frequenz des Gerätes in MHz.	uint
CL_DEVICE_GLOBAL_MEM_SIZE	Größe des globalen Speichers des Gerätes in Bytes	ulong

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung	Typ
CL_DEVICE_LOCAL_MEM_SIZE	Größe des lokalen Speichers der verarbeiteten Daten (Szene) in Bytes	uint
CL_BUFFER_SIZE	CL_BUFFER_SIZE Tatsächliche Größe des OpenCL Puffers in Bytes ulong	ulong
CL_DEVICE_MAX_WORK_GROUP_SIZE	Gesamtzahl lokaler Arbeitsgruppen, die für ein OpenCL-Gerät verfügbar sind.	ulong
CL_KERNEL_WORK_GROUP_SIZE	Gesamtzahl lokaler Arbeitsgruppen, die für ein OpenCL-Programm verfügbar sind.	ulong
CL_KERNEL_LOCAL_MEM_SIZE	Größe des lokalen Speichers in Bytes, den das OpenCL-Programm für alle parallelen Aufgaben in der Gruppe nutzt. Verwenden Sie CL_DEVICE_LOCAL_MEM_SIZE für das Erhalten des verfügbaren Maximums	ulong
CL_KERNEL_PRIVATE_MEM_SIZE	Mindestgröße des privaten Speichers in Bytes, den jede Aufgabe im Kernel des OpenCL-Programms nutzt	ulong
CL_LAST_ERROR	Der Wert des letzten OpenCL-Fehlers	int

Enumeration ENUM_CL_DEVICE_TYPE enthält Typen der Geräte mit Unterstützung für OpenCL. Gerätetyp kann durch seine Nummer oder Handle des Objekts OpenCL durch den Aufruf von CLGetInfoInteger(handle_or_deviceN, CL_DEVICE_TYPE) erhalten werden.

ENUM_CL_DEVICE_TYPE

Identifikator	Beschreibung
CL_DEVICE_ACCELERATOR	Spezialisierter OpenCL-Beschleuniger (zum Beispiel IBM CELL Blade).
CL_DEVICE_CPU	Das OpenCL-Gerät ist ein Host-Prozessor. Der Host-Prozessor läuft die OpenCL-Implementierungen und ist ein Single- oder Multi-Core-CPU.
CL_DEVICE_GPU	Das OpenCL-Gerät ist eine GPU.

© 2000-2025, MetaQuotes Ltd.

Identifikator	Beschreibung
CL_DEVICE_DEFAULT	Das Standardgerät OpenCL. Das Standardgerät kann nicht ein CL_DEVICE_TYPE_CUSTOM-Gerät sein.
CL_DEVICE_CUSTOM	Spezielle Beschleunigern, die Programme in OpenCL C nicht unterstützen.

Beispiel:

```
void OnStart()
  {
   int cl_ctx;
//--- Initialisierung von OpenCL Kontext
   if((cl_ctx=CLContextCreate(CL_USE_GPU_ONLY))==INVALID_HANDLE)
     {
      Print("OpenCL not found");
      return;
     }
//--- Allgemeine Informationen über das OpenCL Gerät anzeigen
   Print("OpenCL type: ",EnumToString((ENUM_CL_DEVICE_TYPE)CLGetInfoInteger(cl_ctx,CL_
   Print("OpenCL vendor ID: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_VENDOR_ID));
   Print("OpenCL units: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_MAX_COMPUTE_UNITS));
   Print("OpenCL freq: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_MAX_CLOCK_FREQUENCY)," MHz"
   Print("OpenCL global mem: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_GLOBAL_MEM_SIZE)," by
   Print("OpenCL local mem: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_LOCAL_MEM_SIZE)," byte
//--- free OpenCL context
   CLContextFree(cl_ctx);
  }
```

© 2000-2025, MetaQuotes Ltd.

CLGetInfoString

Gibt den string Wert des Eigenschafts für ein OpenCL-Objekt oder Gerät zurück.

```
bool  CLGetInfoString(
   int  handle,                          // Handle des OpenCL-Objekts oder Nummer des
   ENUM_OPENCL_PROPERTY_STRING  prop,    // Die angeforderte Eigenschaft
   string&  value                        // String
   );
```

Optionen

handle

[in] Handle auf ein OpenCL-Objekt oder Nummer des OpenCL-Geräts. Nummerierung der OpenCL-Geräte startet mit Null.

prop

[in] Typ der angeforderten Eigenschaft von der Enumeration ENUM_OPENCL_PROPERTY_STRING, deren Wert Sie erhalten möchten.

value

[out] String den Wert der Eigenschaft zu erhalten.

Rückgabewert

true beim Erfolg oder false beim Fehler. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

ENUM_OPENCL_PROPERTY_STRING

Identifier	Beschreibung
CL_PLATFORM_PROFILE	CL_PLATFORM_PROFILE - Profil von OpenCL. Der Profilname kann ein der folgenden Werte sein: • FULL_PROFILE - Implementierung unterstützt OpenCL (Funktionalität ist als Teil der Kernspezifikation definiert und erfordert keine zusätzlichen Erweiterungen für OpenCL Unterstützung); • EMBEDDED_PROFILE - Implementierung unterstützt OpenCL in Form einer Ergänzung. Ergänztes Profil wird als Teilmenge für jede Version von OpenCL definiert.
CL_PLATFORM_VERSION	OpenCL-Version
CL_PLATFORM_VENDOR	Name des Geräteherstellers
CL_PLATFORM_EXTENSIONS	Liste der von der Plattform unterstützten Erweiterungen. Erweiterungennamen müssen von allen Geräten, die mit dieser Plattform verbunden sind, unterstützt werden

© 2000-2025, MetaQuotes Ltd.

Identifier	Beschreibung
CL_DEVICE_NAME	Gerätename
CL_DEVICE_VENDOR	Name des Herstellers
CL_DRIVER_VERSION	Version von OpenCL-Treiber in Format major_number.minor_number
CL_DEVICE_PROFILE	Profil des OpenCL-Gerätes. Der Profilname kann ein der folgenden Werte sein: • FULL_PROFILE - Implementierung unterstützt OpenCL (Funktionalität ist als Teil der Kernspezifikation definiert und erfordert keine zusätzlichen Erweiterungen für OpenCL Unterstützung); • EMBEDDED_PROFILE - Implementierung unterstützt OpenCL in Form einer Ergänzung. Ergänztes Profil wird als Teilmenge für jede Version von OpenCL definiert.
CL_DEVICE_VERSION	Version von OpenCL in Format "OpenCL<space><major_version.minor_version><space><vendor-specific information>"
CL_DEVICE_EXTENSIONS	Liste der vom Gerät unterstützten Erweiterungen. Die Liste kann vom Hersteller unterstützten Erweiterungen erhalten und kann einen oder mehrere zugelassene Namen haben: cl_khr_int64_base_atomics cl_khr_int64_extended_atomics cl_khr_fp16 cl_khr_gl_sharing cl_khr_gl_event cl_khr_d3d10_sharing cl_khr_dx9_media_sharing cl_khr_d3d11_sharing
CL_DEVICE_BUILT_IN_KERNELS	Liste der vom Gerät unterstützten Kernel, getrennten mit ";".
CL_DEVICE_OPENCL_C_VERSION	Maximale Version, die vom Compiler für dieses Gerät unterstützt ist. Format der Version: "OpenCL<space>C<space><major_version.minor_version><space><vendor-specific information> "
CL_ERROR_DESCRIPTION	Beschreibung eines OpenCL-Fehlers als Text

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   int cl_ctx;
   string str;
//--- Initialisierung von OpenCL Kontext
   if((cl_ctx=CLContextCreate(CL_USE_GPU_ONLY))==INVALID_HANDLE)
     {
      Print("OpenCL not found");
      return;
     }
//--- Information über die Plattform anzeigen
   if(CLGetInfoString(cl_ctx,CL_PLATFORM_NAME,str))
      Print("OpenCL platform name: ",str);
   if(CLGetInfoString(cl_ctx,CL_PLATFORM_VENDOR,str))
      Print("OpenCL platform vendor: ",str);
   if(CLGetInfoString(cl_ctx,CL_PLATFORM_VERSION,str))
      Print("OpenCL platform ver: ",str);
   if(CLGetInfoString(cl_ctx,CL_PLATFORM_PROFILE,str))
      Print("OpenCL platform profile: ",str);
   if(CLGetInfoString(cl_ctx,CL_PLATFORM_EXTENSIONS,str))
      Print("OpenCL platform ext: ",str);
//--- Information über das Gerät anzeigen
   if(CLGetInfoString(cl_ctx,CL_DEVICE_NAME,str))
      Print("OpenCL device name: ",str);
   if(CLGetInfoString(cl_ctx,CL_DEVICE_PROFILE,str))
      Print("OpenCL device profile: ",str);
   if(CLGetInfoString(cl_ctx,CL_DEVICE_BUILT_IN_KERNELS,str))
      Print("OpenCL device kernels: ",str);
   if(CLGetInfoString(cl_ctx,CL_DEVICE_EXTENSIONS,str))
      Print("OpenCL device ext: ",str);
   if(CLGetInfoString(cl_ctx,CL_DEVICE_VENDOR,str))
      Print("OpenCL device vendor: ",str);
   if(CLGetInfoString(cl_ctx,CL_DEVICE_VERSION,str))
      Print("OpenCL device ver: ",str);
   if(CLGetInfoString(cl_ctx,CL_DEVICE_OPENCL_C_VERSION,str))
      Print("OpenCL open c ver: ",str);
//--- Gemeinsame Information über das OpenCL-Gerät anzeigen
   Print("OpenCL type: ",EnumToString((ENUM_CL_DEVICE_TYPE)CLGetInfoInteger(cl_ctx,CL_
   Print("OpenCL vendor ID: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_VENDOR_ID));
   Print("OpenCL units: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_MAX_COMPUTE_UNITS));
   Print("OpenCL freq: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_MAX_CLOCK_FREQUENCY));
   Print("OpenCL global mem: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_GLOBAL_MEM_SIZE));
   Print("OpenCL local mem: ",CLGetInfoInteger(cl_ctx,CL_DEVICE_LOCAL_MEM_SIZE));
//--- free OpenCL context
   CLContextFree(cl_ctx);
  }
```

© 2000-2025, MetaQuotes Ltd.

CLContextCreate

Erstellt einen OpenCL-Kontext und gibt sein Handle zurück.

```
int  CLContextCreate(
   int  device=CL_USE_ANY     // Seriennummer von OpenCL-Gerät oder Makro
   );
```

Parameter

device

[in] Seriennummer von OpenCL-Gerät. Anstatt einer bestimmten Nummer können Sie einen der Werte angeben:

- CL_USE_ANY - ein beliebiges Gerät mit Unterstützung für OpenCL genutzt werden kann;
- CL_USE_CPU_ONLY - nur eine Emulation von OpenCL auf CPU ist erlaubt;
- CL_USE_GPU_ONLY - emulation von OpenCL ist verboten und es darf nur spezialisierte Geräte mit Unterstützung für OpenCL (Grafikkarte) zu verwenden;
- CL_USE_GPU_DOUBLE_ONLY - nur GPUs, die den Typ double unterstützen, sind erlaubt.

Rückgabewert

Handle auf den OpenCL-Kontext für den Erfolg, oder -1 bei einem Fehler. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLContextFree

Löscht den Kontext von OpenCL.

```
void  CLContextFree(
   int  context     // Handle auf den OpenCL-Kontext
   );
```

Parameter

context

[in] Handle auf den OpenCL-Kontext.

Rückgabewert

Kein. Im Falle eines internen Fehlers, ändert sich der Wert von _LastError. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLGetDeviceInfo

Erhielt die Geräteeigenschaft aus dem OpenCL-Treiber.

```
bool  CLGetDeviceInfo(
   int     handle,          // Handle von OpenCL-Gerät
   int     property_id,     // Identifikator der gefragten Eigenschaft
   uchar&  data[],          // Array für Daten
   uint&   size             // Offset im Array in Elemente, standardmäßig 0
   );
```

Optionen

handle

[in] Nummer von OpenCL-Gerät oder OpenCL-Handle, der durch die Funktion CLContextCreate() erstellt wurde.

property_id

[in] Identifikator der Eigenschaft, die Sie über das OpenCL-Gerät erhalten möchten. Kann einer der vordefinierten Werten aus der unteren Tabelle sein.

data[]

[out] Array, um die Daten über das gefragte Gerät zu erhalten.

size

[out] Die Größe der erhaltenen Daten im Array *data[]*.

Rückgabewert

true beim Erfolg oder false beim Fehler. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Für eindimensionale Arrays, wird die Nummer des Elements, mit dem Lesung von Daten für Schreibung in den OpenCL-Puffer beginnt, unter Berücksichtigung der Flagge AS_SERIES ausgerechnet.

Liste der gültigen Eigenschaftenbezeichner von OpenCL-Gerät

Die genaue Beschreibung der Eigenschaft und ihre Werte finden Sie auf die offizielle OpenCL Website.

Identifier	Wert
CL_DEVICE_TYPE	0x1000
CL_DEVICE_VENDOR_ID	0x1001
CL_DEVICE_MAX_COMPUTE_UNITS	0x1002
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS	0x1003
CL_DEVICE_MAX_WORK_GROUP_SIZE	0x1004

© 2000-2025, MetaQuotes Ltd.

Identifier	Wert
CL_DEVICE_MAX_WORK_ITEM_SIZES	0x1005
CL_DEVICE_PREFERRED_VECTOR_WIDTH_CHAR	0x1006
CL_DEVICE_PREFERRED_VECTOR_WIDTH_SHORT	0x1007
CL_DEVICE_PREFERRED_VECTOR_WIDTH_INT	0x1008
CL_DEVICE_PREFERRED_VECTOR_WIDTH_LONG	0x1009
CL_DEVICE_PREFERRED_VECTOR_WIDTH_FLOAT	0x100A
CL_DEVICE_PREFERRED_VECTOR_WIDTH_DOUBLE	0x100B
CL_DEVICE_MAX_CLOCK_FREQUENCY	0x100C
CL_DEVICE_ADDRESS_BITS	0x100D
CL_DEVICE_MAX_READ_IMAGE_ARGS	0x100E
CL_DEVICE_MAX_WRITE_IMAGE_ARGS	0x100F
CL_DEVICE_MAX_MEM_ALLOC_SIZE	0x1010
CL_DEVICE_IMAGE2D_MAX_WIDTH	0x1011
CL_DEVICE_IMAGE2D_MAX_HEIGHT	0x1012
CL_DEVICE_IMAGE3D_MAX_WIDTH	0x1013
CL_DEVICE_IMAGE3D_MAX_HEIGHT	0x1014
CL_DEVICE_IMAGE3D_MAX_DEPTH	0x1015
CL_DEVICE_IMAGE_SUPPORT	0x1016
CL_DEVICE_MAX_PARAMETER_SIZE	0x1017
CL_DEVICE_MAX_SAMPLERS	0x1018
CL_DEVICE_MEM_BASE_ADDR_ALIGN	0x1019
CL_DEVICE_MIN_DATA_TYPE_ALIGN_SIZE	0x101A
CL_DEVICE_SINGLE_FP_CONFIG	0x101B
CL_DEVICE_GLOBAL_MEM_CACHE_TYPE	0x101C
CL_DEVICE_GLOBAL_MEM_CACHELINE_SIZE	0x101D
CL_DEVICE_GLOBAL_MEM_CACHE_SIZE	0x101E
CL_DEVICE_GLOBAL_MEM_SIZE	0x101F

© 2000-2025, MetaQuotes Ltd.

Identifier	Wert
CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE	0x1020
CL_DEVICE_MAX_CONSTANT_ARGS	0x1021
CL_DEVICE_LOCAL_MEM_TYPE	0x1022
CL_DEVICE_LOCAL_MEM_SIZE	0x1023
CL_DEVICE_ERROR_CORRECTION_SUPPORT	0x1024
CL_DEVICE_PROFILING_TIMER_RESOLUTION	0x1025
CL_DEVICE_ENDIAN_LITTLE	0x1026
CL_DEVICE_AVAILABLE	0x1027
CL_DEVICE_COMPILER_AVAILABLE	0x1028
CL_DEVICE_EXECUTION_CAPABILITIES	0x1029
CL_DEVICE_QUEUE_PROPERTIES	0x102A
CL_DEVICE_NAME	0x102B
CL_DEVICE_VENDOR	0x102C
CL_DRIVER_VERSION	0x102D
CL_DEVICE_PROFILE	0x102E
CL_DEVICE_VERSION	0x102F
CL_DEVICE_EXTENSIONS	0x1030
CL_DEVICE_PLATFORM	0x1031
CL_DEVICE_DOUBLE_FP_CONFIG	0x1032
CL_DEVICE_PREFERRED_VECTOR_WIDTH_HALF	0x1034
CL_DEVICE_HOST_UNIFIED_MEMORY	0x1035
CL_DEVICE_NATIVE_VECTOR_WIDTH_CHAR	0x1036
CL_DEVICE_NATIVE_VECTOR_WIDTH_SHORT	0x1037
CL_DEVICE_NATIVE_VECTOR_WIDTH_INT	0x1038
CL_DEVICE_NATIVE_VECTOR_WIDTH_LONG	0x1039
CL_DEVICE_NATIVE_VECTOR_WIDTH_FLOAT	0x103A
CL_DEVICE_NATIVE_VECTOR_WIDTH_DOUBLE	0x103B
CL_DEVICE_NATIVE_VECTOR_WIDTH_HALF	0x103C
CL_DEVICE_OPENCL_C_VERSION	0x103D
CL_DEVICE_LINKER_AVAILABLE	0x103E

© 2000-2025, MetaQuotes Ltd.

Identifier	Wert
CL_DEVICE_BUILT_IN_KERNELS	0x103F
CL_DEVICE_IMAGE_MAX_BUFFER_SIZE	0x1040
CL_DEVICE_IMAGE_MAX_ARRAY_SIZE	0x1041
CL_DEVICE_PARENT_DEVICE	0x1042
CL_DEVICE_PARTITION_MAX_SUB_DEVICES	0x1043
CL_DEVICE_PARTITION_PROPERTIES	0x1044
CL_DEVICE_PARTITION_AFFINITY_DOMAIN	0x1045
CL_DEVICE_PARTITION_TYPE	0x1046
CL_DEVICE_REFERENCE_COUNT	0x1047
CL_DEVICE_PREFERRED_INTEROP_USER_SYNC	0x1048
CL_DEVICE_PRINTF_BUFFER_SIZE	0x1049
CL_DEVICE_IMAGE_PITCH_ALIGNMENT	0x104A
CL_DEVICE_IMAGE_BASE_ADDRESS_ALIGNMENT	0x104B

Beispiel:

```
void OnStart()
  {
//---
   int dCount= CLGetInfoInteger(0,CL_DEVICE_COUNT);
   for(int i = 0; i<dCount; i++)
     {
      int clCtx=CLContextCreate(i);
      if(clCtx == -1)
         Print("ERROR in CLContextCreate");
      string device;
      CLGetInfoString(clCtx,CL_DEVICE_NAME,device);
      Print(i,": ",device);
      uchar data[1024];
      uint size;
      CLGetDeviceInfo(clCtx,CL_DEVICE_VENDOR,data,size);
      Print("size = ",size);
      string str=CharArrayToString(data);
      Print(str);
    ;}
 ;}
//--- Beispiel der Ausgabe in Protokoll Experten
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      2: Advanced Micro Devices, Inc.
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      size = 32
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      Tahiti
```

© 2000-2025, MetaQuotes Ltd.

```
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      Intel(R) Corporation
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      size = 21
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      1:        Intel(R) Core(TM) i7-377
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      NVIDIA Corporation
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      size = 19
//  2013.07.24 10:50:48     opencl (EURUSD,H1)      0: GeForce GTX 580
```

© 2000-2025, MetaQuotes Ltd.

CLProgramCreate

Erstellt ein OpenCL-Programm aus dem Quellcode.

```
int  CLProgramCreate(
   int           context,     // Handle auf den OpenCL-Kontext
   const string  source       // Quellcode
   );
```

Die überladene Funktionsversion erstellt ein OpenCL-Programm und schreibt Compiler-Meldungen in die übergebene Zeichenfolge.

```
int  CLProgramCreate(
   int           context,     // Handle auf den OpenCL-Kontext
   const string  source,      // Quellcode
   string       &build_log    // eine Zeichenfolge für Empfang von Compilation-Log
   );
```

Parameter

context

[in] Handle auf den OpenCL-Kontext.

source

[in] String mit dem Quellcode des OpenCL-Programms.

&build_log

[in] String für Empfang den Meldungen von OpenCL-Compiler.

Rückgabewert

Handle auf das OpenCL-Objekt für beim Erfolg. Im Falle eines Fehlers wird -1 zurückgegeben. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Im Moment werden die folgenden Fehler-Codes verwendet:

- ERR_OPENCL_INVALID_HANDLE - ungültiges Handle auf context OpenCL.
- ERR_INVALID_PARAMETER - ungültiger String-Parameter.
- ERR_NOT_ENOUGH_MEMORY - nicht genügend Speicher um Operation abzuschließen.
- ERR_OPENCL_PROGRAM_CREATE - interner Fehler von OpenCL oder Kompilierung-Fehler.

Bei einigen Grafikkarten ist der Modus der Arbeit mit double-Zahlen deaktiviert, das zum Kompilierung-Fehler 5105 führen kann. Um Unterstützung von double-Zahlen zu aktivieren, fügen Sie eine Direktive #pragma OPENCL EXTENSION cl_khr_fp64 : enable in den Text des OpenCL-Programms hinzu. Aber wenn die Grafikkarte den double-Typ nicht unterstützt, die Einbeziehung dieser Direktive wird nicht helfen.

Beispiel:

```
//+------------------------------------------------------------------+
//| OpenCL kernel                                                    |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
const string
cl_src=
       //--- by default some GPU doesn't support doubles
       //--- cl_khr_fp64 directive is used to enable work with doubles
       "#pragma OPENCL EXTENSION cl_khr_fp64 : enable      \r\n"
       //--- OpenCL kernel function
       "__kernel void Test_GPU(__global double *data,      \r\n"
       "                       const    int N,             \r\n"
       "                       const    int total_arrays)  \r\n"
       "  {                                                \r\n"
       "   uint kernel_index=get_global_id(0);             \r\n"
       "   if (kernel_index>total_arrays) return;          \r\n"
       "   uint local_start_offset=kernel_index*N;         \r\n"
       "   for(int i=0; i<N; i++)                          \r\n"
       "     {                                             \r\n"
       "       data[i+local_start_offset] *= 2.0;          \r\n"
       "     }                                             \r\n"
       "  }                                                \r\n";
//+------------------------------------------------------------------+
//| Test_CPU                                                         |
//+------------------------------------------------------------------+
bool Test_CPU(double &data[],const int N,const int id,const int total_arrays)
  {
//--- check array size
   if(ArraySize(data)==0) return(false);
//--- check array index
   if(id>total_arrays) return(false);
//--- calculate local offset for array with index id
   int local_start_offset=id*N;
//--- multiply elements by 2
   for(int i=0; i<N; i++)
     {
      data[i+local_start_offset]*=2.0;
     }
   return true;
  }
//---
#define ARRAY_SIZE   100  // size of the array
#define TOTAL_ARRAYS 5    // total arrays
//--- OpenCL handles
int cl_ctx;  // OpenCL context handle
int cl_prg;  // OpenCL program handle
int cl_krn;  // OpenCL kernel handle
int cl_mem;  // OpenCL buffer handle
//---
double DataArray1[]; // data array for CPU calculation
double DataArray2[]; // data array for GPU calculation
//+------------------------------------------------------------------+
//| Script program start function                                    |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
int OnStart()
  {
//--- initialize OpenCL objects
//--- create OpenCL context
   if((cl_ctx=CLContextCreate())==INVALID_HANDLE)
     {
      Print("OpenCL not found. Error=",GetLastError());
      return(1);
     }
//--- create OpenCL program
   if((cl_prg=CLProgramCreate(cl_ctx,cl_src))==INVALID_HANDLE)
     {
      CLContextFree(cl_ctx);
      Print("OpenCL program create failed. Error=",GetLastError());
      return(1);
     }
//--- create OpenCL kernel
   if((cl_krn=CLKernelCreate(cl_prg,"Test_GPU"))==INVALID_HANDLE)
     {
      CLProgramFree(cl_prg);
      CLContextFree(cl_ctx);
      Print("OpenCL kernel create failed. Error=",GetLastError());
      return(1);
     }
//--- create OpenCL buffer
   if((cl_mem=CLBufferCreate(cl_ctx,ARRAY_SIZE*TOTAL_ARRAYS*sizeof(double),CL_MEM_REAI
     {
      CLKernelFree(cl_krn);
      CLProgramFree(cl_prg);
      CLContextFree(cl_ctx);
      Print("OpenCL buffer create failed. Error=",GetLastError());
      return(1);
     }
//--- set OpenCL kernel constant parameters
   CLSetKernelArgMem(cl_krn,0,cl_mem);
   CLSetKernelArg(cl_krn,1,ARRAY_SIZE);
   CLSetKernelArg(cl_krn,2,TOTAL_ARRAYS);
//--- prepare data arrays
   ArrayResize(DataArray1,ARRAY_SIZE*TOTAL_ARRAYS);
   ArrayResize(DataArray2,ARRAY_SIZE*TOTAL_ARRAYS);
//--- fill arrays with data
   for(int j=0; j<TOTAL_ARRAYS; j++)
     {
      //--- calculate local start offset for jth array
      uint local_offset=j*ARRAY_SIZE;
      //--- prepare array with index j
      for(int i=0; i<ARRAY_SIZE; i++)
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- fill arrays with function MathCos(i+j);
         DataArray1[i+local_offset]=MathCos(i+j);
         DataArray2[i+local_offset]=MathCos(i+j);
        }
     };
//--- test CPU calculation
   for(int j=0; j<TOTAL_ARRAYS; j++)
     {
      //--- calculation of the array with index j
      Test_CPU(DataArray1,ARRAY_SIZE,j,TOTAL_ARRAYS);
     }
//--- prepare CLExecute params
   uint  offset[]={0};
//--- global work size
   uint  work[]={TOTAL_ARRAYS};
//--- write data to OpenCL buffer
   CLBufferWrite(cl_mem,DataArray2);
//--- execute OpenCL kernel
   CLExecute(cl_krn,1,offset,work);
//--- read data from OpenCL buffer
   CLBufferRead(cl_mem,DataArray2);
//--- total error
   double total_error=0;
//--- compare results and calculate error
   for(int j=0; j<TOTAL_ARRAYS; j++)
     {
      //--- calculate local offset for jth array
      uint local_offset=j*ARRAY_SIZE;
      //--- compare the results
      for(int i=0; i<ARRAY_SIZE; i++)
        {
         double v1=DataArray1[i+local_offset];
         double v2=DataArray2[i+local_offset];
         double delta=MathAbs(v2-v1);
         total_error+=delta;
         //--- show first and last arrays
         if((j==0) || (j==TOTAL_ARRAYS-1))
            PrintFormat("array %d of %d, element [%d]:  %f, %f, [error]=%f",j+1,TOTAL_
        }
     }
   PrintFormat("Total error: %f",total_error);
//--- delete OpenCL objects
//--- free OpenCL buffer
   CLBufferFree(cl_mem);
//--- free OpenCL kernel
   CLKernelFree(cl_krn);
//--- free OpenCL program
   CLProgramFree(cl_prg);
//--- free OpenCL context
```

© 2000-2025, MetaQuotes Ltd.

```
   CLContextFree(cl_ctx);
//---
   return(0);
  }
```

© 2000-2025, MetaQuotes Ltd.

CLProgramFree

Löscht das OpenCL-Programm.

```
void  CLProgramFree(
   int  program     // Handle auf das OpenCL-Objekt
   );
```

Parameter

program

[in] Handle des OpenCL-Objekts.

Rückgabewert

Kein. Im Falle eines internen Fehlers, ändert sich der Wert von _LastError. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLKernelCreate

Erstellt eine Einsprungstelle in das OpenCL-Programm und gibt sein Handle zurück.

```
int  CLKernelCreate(
   int           program,        // Handle auf das OpenCL-Objekt
   const string  kernel_name     // Name des Kernels
   );
```

Parameter

program

[in] Handle auf das Objekt von OpenCL-Programm.

kernel_name

[in] Der Name der Kernel-Funktion, d.h. der Name der Einsprungstelle in das entsprechende OpenCL-Programm,, in dem Ausführung beginnt.

Rückgabewert

Handle auf das OpenCL-Objekt für beim Erfolg. Im Falle eines Fehlers wird -1 zurückgegeben. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Im Moment werden die folgenden Fehler-Codes verwendet:

- ERR_OPENCL_INVALID_HANDLE - ungültiges Handle auf *program* OpenCL.
- ERR_INVALID_PARAMETER - ungültiger String-Parameter.
- ERR_OPENCL_TOO_LONG_KERNEL_NAME - der Name des Kernels, enthält mehr als 127 Zeichen.
- ERR_OPENCL_KERNEL_CREATE - Interner Fehler beim Erstellen von OpenCL-Objekt.

© 2000-2025, MetaQuotes Ltd.

CLKernelFree

Löscht die Funktion des Starts von OpenCL.

```
void  CLKernelFree(
   int  kernel     // Handle auf den Kernel von OpenCL-Programm
   );
```

Parameter

kernel_name

[in] Handle des Objekts von Kernel.

Rückgabewert

Kein. Im Falle eines internen Fehlers, ändert sich der Wert von _LastError. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLSetKernelArg

Setzt den Parameter für die OpenCL-Funktion.

```
bool  CLSetKernelArg(
   int   kernel,       // Handle auf den Kernel von OpenCL-Programm
   uint  arg_index,    // Nummer des OpenCL-Funktionsarguments
   void  arg_value     // Quellcode
   );
```

Parameter

kernel

[in] Handle auf den Kernel von OpenCL-Programm.

arg_index

[in] Nummer des Funktionsarguments. Nummerierung beginnt mit Null.

arg_value

[in] Wert des Funktionsarguments.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Im Moment werden die folgenden Fehler-Codes verwendet:

- ERR_INVALID_PARAMETER,
- ERR_OPENCL_INVALID_HANDLE - ungültiges Handle auf den OpenCL-Kernel
- ERR_OPENCL_SET_KERNEL_PARAMETER - interner Fehler von OpenCL.

© 2000-2025, MetaQuotes Ltd.

CLSetKernelArgMem

Setzt einen OpenCL-Puffer als ein Parameter der OpenCL-Funktion.

```
bool  CLSetKernelArgMem(
   int   kernel,           // Handle auf den Kernel von OpenCL-Programm
   uint  arg_index,        // Nummer des OpenCL-Funktionsarguments
   int   cl_mem_handle     // Handle des OpenCL-Puffers
   );
```

Parameter

kernel

[in] Handle auf den Kernel von OpenCL-Programm.

arg_index

[in] Nummer des Funktionsarguments. Nummerierung beginnt mit Null.

cl_mem_handle

[in] Handle auf den OpenCL-Puffer.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLSetKernelArgMemLocal

Setzt den lokalen Puffer als Argument der kernel-Funktion.

```
bool  CLSetKernelArgMemLocal(
   int    kernel,           // Handle des Kernels des OpenCL Programms
   uint   arg_index,        // Nummer des Arguments der OpenCL-Funktion
   ulong  local_mem_size    // Puffergröße
   );
```

Parameter

kernel

[in] Handle des Kernels eines OpenCL Programms.

arg_index

[in] Nummer des Arguments der Funktion, die Nummerierung beginnt mit Null.

local_mem_size

[in] Puffergröße in Bytes.

Rückgabewert

Gibt true bei einer erfolgreichen Ausführung zurück, andernfalls false. Um Fehlerdetails abzurufen, verwenden Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLBufferCreate

Erstellt einen OpenCL-Puffer und gibt sein Handle zurück.

```
int  CLBufferCreate(
   int   context,     // Handle auf den OpenCL-Kontext
   uint  size,        // Größe des Puffers
   uint  flags        // Kombination von Flags, die Eigenschaften des Puffers angeben
   );
```

Parameter

context

[in] Handle auf den OpenCL-Kontext.

size

[in] Größe des Puffers in Bytes.

flags

[in] Eigenschaften des Puffers, sie durch eine Kombination von Flags definiert werden: CL_MEM_READ_WRITE, CL_MEM_WRITE_ONLY, CL_MEM_READ_ONLY, CL_MEM_ALLOC_HOST_PTR.

Rückgabewert

Handle auf den OpenCL-Puffer für den Erfolg. Im Falle eines Fehlers wird -1 zurückgegeben. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Im Moment werden die folgenden Fehler-Codes verwendet:

- ERR_OPENCL_INVALID_HANDLE - ungültiges Handle auf den OpenCL-Kontext.
- ERR_NOT_ENOUGH_MEMORY - nicht genügend Speicher.
- ERR_OPENCL_BUFFER_CREATE - interner Fehler beim Erstellen von Puffer.

© 2000-2025, MetaQuotes Ltd.

CLBufferFree

Löscht einen OpenCL-Puffer.

```
void  CLBufferFree(
   int   buffer     // Handle auf den OpenCL-Puffer
   );
```

Parameter

buffer

[in] Handle auf den OpenCL-Puffer.

Rückgabewert

Kein. Im Falle eines internen Fehlers, ändert sich der Wert von _LastError. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

CLBufferWrite

Schreibt ein Array in einen OpenCL-Puffer und gibt die Anzahl der geschriebenen Elemente zurück.

```
uint  CLBufferWrite(
   int           buffer,                    // Handle auf den OpenCL-Puffer
   const void&   data[],                    // Array von Werten
   uint          buffer_offset=0,           // Offset im OpenCL-Puffer in Bytes, standa
   uint          data_offset=0,             // Offset im Array in Elemente, standardmäß
   uint          data_count=WHOLE_ARRAY     // Anzahl der Werte im Array zu schreiben,
   );
```

Es gibt auch Versionen für die Verarbeitung von Matrizen und Vektoren.

Die Werte aus der Matrix werden in den Puffer geschrieben und bei Erfolg wird true zurückgegeben.

```
uint  CLBufferWrite(
   int           buffer,                    // ein Handle zum Puffer von OpenCL
   uint          buffer_offset,             // ein Offset im OpenCL-Puffer in Bytes
   matrix<T>     &mat                       // die Matrix mit den Werten, die dem Puf
   );
```

Die Werte des Vektors werden in den Puffer geschrieben und bei Erfolg wird true zurückgegeben.

```
uint  CLBufferWrite(
   int           buffer,                    // ein Handle zum Puffer von OpenCL
   uint          buffer_offset,             // ein Offset im OpenCL-Puffer in Bytes
   vector<T>     &vec                       // der Vektor mit den Werten, die dem Puf
   );
```

Parameter

buffer

[in] Handle auf den OpenCL-Puffer.

data[]

[in] Das Array von Werten, die Sie in den OpenCL-Puffer schreiben möchten. Wird als Referenz übergeben.

buffer_offset

[in] Offset im OpenCL-Puffer, mit dem Schreibung beginnt, in Bytes. Standardmäßig wird vom Anfang des Puffers geschrieben.

data_offset

[in] Der Index des ersten Elements des Arrays aus dem die Werte für Schreiben in den OpenCL-Puffer beginnen. Standardmäßig werden die Werte vom Anfang des Arrays genommen.

data_count

[in] Die Anzahl der Werte zu schreiben. Standardmäßig werden alle Werte des Arrays genommen.

mat

© 2000-2025, MetaQuotes Ltd.

[out] Die Matrix zum Lesen von Daten aus dem Puffer kann einer der drei Typen sein: matrix, matrixf oder matrixc.

vec

[out] Der Vektor zum Lesen von Daten aus dem Puffer kann von einem der drei Typen sein: vector, vectorf oder vectorc.

Rückgabewert

Anzahl der geschriebenen Elemente. Im Falle eines Fehlers wird 0 zurückgegeben. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

true, wenn eine Matrix oder ein Vektor erfolgreich verarbeitet wurde, ansonsten false.

Hinweis

Für eindimensionale Arrays, wird die Nummer des Elements, mit dem Lesung von Daten für Schreibung in den OpenCL-Puffer beginnt, unter Berücksichtigung der Flagge AS_SERIES ausgerechnet.

Ein Array von zwei oder mehr Dimensionen wird als eindimensional dargestellt. In diesem Fall ist *data_Offset* die Anzahl der Elemente, die in der Darstellung angegeben werden soll, nicht die Anzahl der Elemente in der ersten Dimension.

Beispiel für eine Matrixmultiplikation mit der Methode MatMul und paralleles Rechnen in OpenCL

```
#define M       3000      // die Anzahl der Zeilen der ersten Matrix
#define K       2000      // die Anzahl der Spalten in der ersten Matrix ist gleich de
#define N       3000      // die Anzahl der Spalten in der zweiten Matrix

//+------------------------------------------------------------------+
const string clSrc=
  "#define N     "+IntegerToString(N)+"                              \r\n"
  "#define K     "+IntegerToString(K)+"                              \r\n"
  "                                                                 \r\n"
  "__kernel void matricesMul( __global float *in1,                  \r\n"
  "                           __global float *in2,                  \r\n"
  "                           __global float *out  )                \r\n"
  "{                                                                \r\n"
  "  int m = get_global_id( 0 );                                    \r\n"
  "  int n = get_global_id( 1 );                                    \r\n"
  "  float sum = 0.0;                                               \r\n"
  "  for( int k = 0; k < K; k ++ )                                  \r\n"
  "     sum += in1[ m * K + k ] * in2[ k * N + n ];                 \r\n"
  "  out[ m * N + n ] = sum;                                        \r\n"
  "}                                                                \r\n";
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Initialisierung des Zufallszahlengenerators
  MathSrand((int)TimeCurrent());
//--- Füllen der Matrizen einer bestimmten Größe mit Zufallswerten
  matrixf mat1(M, K, MatrixRandom) ;     // erste Matrix
  matrixf mat2(K, N, MatrixRandom);      // zweite Matrix

//--- Berechnung des Produkts von Matrizen mit der naiven Methode
  uint start=GetTickCount();
  matrixf matrix_naive=matrixf::Zeros(M, N);// das Ergebnis der Multiplikation zweier
  for(int m=0; m<M; m++)
    for(int k=0; k<K; k++)
      for(int n=0; n<N; n++)
        matrix_naive[m][n]+=mat1[m][k]*mat2[k][n];
  uint time_naive=GetTickCount()-start;

//--- Berechnung des Produkts von Matrizen über MatMull
  start=GetTickCount();
  matrixf matrix_matmul=mat1.MatMul(mat2);
  uint time_matmul=GetTickCount()-start;

//--- Berechnung des Produkts von Matrizen in OpenCL
  matrixf matrix_opencl=matrixf::Zeros(M, N);
  int cl_ctx;              // Kontext-Handle
  if((cl_ctx=CLContextCreate(CL_USE_GPU_ONLY))==INVALID_HANDLE)
   {
    Print("OpenCL nicht gefunden, Funktion wird verlassen");
    return;
   }
  int cl_prg;              // Programm-Handle
  int cl_krn;              // Kernel-Handle
  int cl_mem_in1;          // Handle des ersten (Eingabe) Puffers
  int cl_mem_in2;          // Handle des zweiten (Eingabe) Puffers
  int cl_mem_out;          // Handle des dritten (Eingabe) Puffers
//--- das Programm und den Kernel erstellen
  cl_prg = CLProgramCreate(cl_ctx, clSrc);
  cl_krn = CLKernelCreate(cl_prg, "matricesMul");
//--- alle drei Puffer für drei Matrizen erstellen
  cl_mem_in1=CLBufferCreate(cl_ctx, M*K*sizeof(float), CL_MEM_READ_WRITE);
  cl_mem_in2=CLBufferCreate(cl_ctx, K*N*sizeof(float), CL_MEM_READ_WRITE);
//--- dritte Matrix - Ausgabe
  cl_mem_out=CLBufferCreate(cl_ctx, M*N*sizeof(float), CL_MEM_READ_WRITE);
//--- Setzen der Kernel-Argumente
  CLSetKernelArgMem(cl_krn, 0, cl_mem_in1);
  CLSetKernelArgMem(cl_krn, 1, cl_mem_in2);
  CLSetKernelArgMem(cl_krn, 2, cl_mem_out);
//--- Matrizen in die Gerätepuffer schreiben
  CLBufferWrite(cl_mem_in1, 0, mat1);
  CLBufferWrite(cl_mem_in2, 0, mat2);
  CLBufferWrite(cl_mem_out, 0, matrix_opencl);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- Startzeit der Ausführung des OpenCL-Codes
  start=GetTickCount();
//--- die Parameter des Arbeitsbereichs der Aufgabe festlegen und das OpenCL-Programm
  uint  offs[2] = {0, 0};
  uint works[2] = {M, N};
  start=GetTickCount();
  bool ex=CLExecute(cl_krn, 2, offs, works);
//--- Berechnen des Ergebnisses in der Matrix
  if(CLBufferRead(cl_mem_out, 0, matrix_opencl))
    PrintFormat("[%d x %d] matrix read: ", matrix_opencl.Rows(), matrix_opencl.Cols())
   else
      Print("CLBufferRead(cl_mem_out, 0, matrix_opencl failed. Error ",GetLastError())
  uint time_opencl=GetTickCount()-start;
  Print("Compare calculation time using each method");
  PrintFormat("Naive product time = %d ms",time_naive);
  PrintFormat("MatMul product time = %d ms",time_matmul);
  PrintFormat("OpenCl product time = %d ms",time_opencl);
//--- alle OpenCL-Kontexte freigeben
  CLFreeAll(cl_ctx, cl_prg, cl_krn, cl_mem_in1, cl_mem_in2, cl_mem_out);

//--- alle erhaltenen Ergebnismatrizen miteinander vergleichen
  Print("Wie viele Diskrepanzfehler gibt es zwischen den Ergebnismatrizen?");
  ulong errors=matrix_naive.Compare(matrix_matmul,(float)1e-12);
  Print("matrix_direct.Compare(matrix_matmul,1e-12)=",errors);
  errors=matrix_matmul.Compare(matrix_opencl,float(1e-12));
  Print("matrix_matmul.Compare(matrix_opencl,1e-12)=",errors);
/*
  Ergebnis:

   [3000 x 3000] matrix read:
  Vergleich der Berechnungszeit von jeder Methode
   Naive product time = 54750 ms
   MatMul product time = 4578 ms
   OpenCl product time = 922 ms
  Wie viele Diskrepanzfehler gibt es zwischen den Ergebnismatrizen?
   matrix_direct.Compare(matrix_matmul,1e-12)=0
   matrix_matmul.Compare(matrix_opencl,1e-12)=0
*/
 }
//+------------------------------------------------------------------+
//| Füllen der Matrix mit Zufallszahlen                              |
//+------------------------------------------------------------------+
void MatrixRandom(matrixf& m)
 {
  for(ulong r=0; r<m.Rows(); r++)
   {
    for(ulong c=0; c<m.Cols(); c++)
     {
      m[r][c]=(float)((MathRand()-16383.5)/32767.);
```

© 2000-2025, MetaQuotes Ltd.

```
      }
    }
  }
//+------------------------------------------------------------------+
//| Freigeben aller OpenCL Kontexte                                  |
//+------------------------------------------------------------------+
void CLFreeAll(int cl_ctx, int cl_prg, int cl_krn,
               int cl_mem_in1, int cl_mem_in2, int cl_mem_out)
  {
//--- alle von OpenCL erstellten Kontexte in umgekehrter Reihenfolge löschen
   CLBufferFree(cl_mem_in1);
   CLBufferFree(cl_mem_in2);
   CLBufferFree(cl_mem_out);
   CLKernelFree(cl_krn);
   CLProgramFree(cl_prg);
   CLContextFree(cl_ctx);
  }
```

© 2000-2025, MetaQuotes Ltd.

CLBufferRead

Liest den OpenCL-Puffer in ein Array und gibt die Anzahl der gelesenen Elemente zurück.

```
uint  CLBufferRead(
   int           buffer,                    // Handle auf den OpenCL-Puffer
   const void&   data[],                    // Array von Werten
   uint          buffer_offset=0,           // Offset im OpenCL-Puffer in Bytes, standa
   uint          data_offset=0,             // Offset im Array in Elemente, standardmäß
   uint          data_count=WHOLE_ARRAY     // Anzahl der Werte aus dem Puffer zu leser
   );
```

Es gibt auch Versionen für die Verarbeitung von Matrizen und Vektoren.

Lesen der OpenCL-Puffer in die Matrix und gibt bei Erfolg true zurück.

```
uint  CLBufferRead(
   int           buffer,                    // ein Handle zum Puffer von OpenCL
   uint          buffer_offset,             // ein Offset im OpenCL-Puffer in Bytes
   const matrxi& mat,                       // die Matrix erhält die Werte aus dem Puf
   ulong         rows=-1,                   // Die Anzahl der Zeilen in der Matrix
   ulong         cols=-1                    // Die Anzahl der Spalten in der Matrix
   );
```

Einlesen des OpenCL-Puffers in den Vektor und Rückgabe von true bei Erfolg.

```
uint  CLBufferRead(
   int           buffer,                    // ein Handle zum Puffer von OpenCL
   uint          buffer_offset,             // ein Offset im OpenCL-Puffer in Bytes
   const vector& vec,                       // der Vektor erhält die Werte aus dem Puf
   ulong         size-1,                    // Länge des Vektors
   );
```

Parameter

buffer

[in] Handle auf den OpenCL-Puffer.

data[]

[in] Array, um die Werte aus dem OpenCL-Puffer zu empfangen. Wird als Referenz übergeben.

buffer_offset

[in] Offset im OpenCL-Puffer, mit dem Lesung beginnt, in Bytes. Standardmäßig startet die Lesung am Anfang des Puffers.

data_offset

[in] Der Index des ersten Elements des Arrays, um Werte des OpenCL-Puffers zu schreiben. Standardmäßig startet die Schreibung von gelesenen Werten in ein Array mit dem Null-Index.

data_count

[in] Die Anzahl der Werte zu lesen. Standardmäßig wird der gesamte OpenCL-Puffer gelesen.

mat

© 2000-2025, MetaQuotes Ltd.

[out] Die Matrix zum Lesen von Daten aus dem Puffer kann einer der drei Typen sein: matrix, matrixf oder matrixc.

vec

[out] Der Vektor zum Lesen von Daten aus dem Puffer kann von einem der drei Typen sein: vector, vectorf oder vectorc.

rows=-1

[in] Wenn der Parameter angegeben wird, sollte auch der Parameter *cols* (Spalten) angegeben werden. Wenn die neuen Matrixdimensionen nicht angegeben werden, werden die aktuellen verwendet. Ist der Wert -1, so ändert sich die Anzahl der Zeilen nicht.

cols=-1

[in] Wenn der Parameter nicht angegeben wird, sollte auch der Parameter *rows* (Zeilen) übersprungen werden. Die Matrix hält sich an die Regel: entweder sind beide Parameter angegeben, oder keiner, sonst tritt ein Fehler auf. Werden beide Parameter (*rows* und *cols*) angegeben, wird die Größe der Matrix geändert. Im Falle von -1 ändert sich die Anzahl der Spalten nicht.

size=-1

[in] Wird der Parameter nicht angegeben oder ist sein Wert -1, ändert sich die Vektorlänge nicht.

Rückgabewert

Anzahl der gelesenen Elemente. Im Falle eines Fehlers wird 0 zurückgegeben. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

true, wenn eine Matrix oder ein Vektor erfolgreich verarbeitet wurde, ansonsten false.

Hinweis

Für eindimensionale Arrays, wird die Nummer des Elements, in das Schreiben von Daten aus dem OpenCL-Puffer beginnt, unter Berücksichtigung der Flagge AS_SERIES ausgerechnet.

Ein Array von zwei oder mehr Dimensionen wird als eindimensional dargestellt. In diesem Fall ist *data_Offset* die Anzahl der Elemente, die in der Darstellung angegeben werden soll, nicht die Anzahl der Elemente in der ersten Dimension.

Beispiel zur Berechnung von Pi anhand der Gleichung:

$$\pi = \lim_{N\to\infty}\frac{1}{N}\sum_{k=0}^{N-1}\frac{4}{1+\left(\frac{2k+1}{2N}\right)^2} = 16\lim_{N\to\infty} N\sum_{k=0}^{N-1}\frac{1}{4N^2+(2k+1)^2}$$

```
#define  _num_steps        1000000000
#define  _divisor          40000
#define  _step             1.0 / _num_steps
#define  _intrnCnt         _num_steps / _divisor
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                                  |
//+------------------------------------------------------------------+
string D2S(double arg, int digits) { return DoubleToString(arg, digits); }
string I2S(int arg)                { return IntegerToString(arg); }

//--- OpenCL-Programmcode
const string clSource=
  "#define _step "+D2S(_step, 12)+"                    \r\n"
  "#define _intrnCnt "+I2S(_intrnCnt)+"                \r\n"
  "                                                    \r\n"
  "__kernel void Pi( __global double *out )            \r\n"
  "{                                                   \r\n"
  "  int i = get_global_id( 0 );                       \r\n"
  "  double partsum = 0.0;                             \r\n"
  "  double x = 0.0;                                   \r\n"
  "  long from = i * _intrnCnt;                        \r\n"
  "  long to = from + _intrnCnt;                       \r\n"
  "  for( long j = from; j < to; j ++ )                \r\n"
  "  {                                                 \r\n"
  "     x = ( j + 0.5 ) * _step;                       \r\n"
  "     partsum += 4.0 / ( 1. + x * x );               \r\n"
  "  }                                                 \r\n"
  "  out[ i ] = partsum;                               \r\n"
  "}                                                   \r\n";
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
int OnStart()
 {
  Print("Pi-Berechnung: step = "+D2S(_step, 12)+"; _intrnCnt = "+I2S(_intrnCnt));
//--- Vorbereitung der OpenCL-Kontexte
  int clCtx;
  if((clCtx=CLContextCreate(CL_USE_GPU_ONLY))==INVALID_HANDLE)
   {
    Print("OpenCL not found");
    return(-1);
   }
  int clPrg = CLProgramCreate(clCtx, clSource);
  int clKrn = CLKernelCreate(clPrg, "Pi");
  int clMem=CLBufferCreate(clCtx, _divisor*sizeof(double), CL_MEM_READ_WRITE);
  CLSetKernelArgMem(clKrn, 0, clMem);

  const uint offs[1]  = {0};
  const uint works[1] = {_divisor};
//--- Starten des OpenCL-Programms
  ulong start=GetMicrosecondCount();
  if(!CLExecute(clKrn, 1, offs, works))
```

© 2000-2025, MetaQuotes Ltd.

```
    {
     Print("CLExecute(clKrn, 1, offs, works) failed! Fehlernummer ", GetLastError());
     CLFreeAll(clMem, clKrn, clPrg, clCtx);
     return(-1);
    }
//--- Abrufen der Ergebnisse vom OpenCL-Gerät
  vector buffer(_divisor);
  if(!CLBufferRead(clMem, 0, buffer))
    {
     Print("CLBufferRead(clMem, 0, buffer) failed! Fehlernummer ", GetLastError());
     CLFreeAll(clMem, clKrn, clPrg, clCtx);
     return(-1);
    }
//--- Summe aller Werte zur Berechnung von Pi
  double Pi=buffer.Sum()*_step;

  double time=(GetMicrosecondCount()-start)/1000.;
  Print("OpenCL: Pi calculated for "+D2S(time, 2)+" ms");
  Print("Pi = "+DoubleToString(Pi, 12));
//--- Freigeben des Speichers
  CLFreeAll(clMem, clKrn, clPrg, clCtx);
//--- Gelungen
  return(0);
 }
  /*
  Pi Calculation: step = 0.000000001000; _intrnCnt = 25000
  OpenCL: GPU device 'Ellesmere' selected
  OpenCL: Pi calculated for 99.98 ms
  Pi = 3.141592653590
  */
//+------------------------------------------------------------------+
//| Hilfsroutine zur Freigabe des Speichers                          |
//+------------------------------------------------------------------+
void CLFreeAll(const int clMem, const int clKrn, const int clPrg, const int clCtx)
 {
  CLBufferFree(clMem);
  CLKernelFree(clKrn);
  CLProgramFree(clPrg);
  CLContextFree(clCtx);
 }
```

© 2000-2025, MetaQuotes Ltd.

CLExecute

Führt das OpenCL-Programm aus. Es gibt drei Varianten der Funktion:

1. Lauf der Funktion kernel auf einem einzigen Kern

```
bool  CLExecute(
   int          kernel                  // Handle auf den Kernel von OpenCL-Programm
   );
```

2. Lauf mehrerer Instanzen von kernel (OpenCL-Funktion) mit einer Beschreibung des Problemraums

```
bool  CLExecute(
   int          kernel,                 // Handle auf den Kernel von OpenCL-Programm
   uint         work_dim,               // Dimension des Raumes von Aufgaben
   const uint&  global_work_offset[],   // Anfangsoffset im Raum von Aufgaben
   const uint&  global_work_size[]      // Gesamtzahl der Aufgaben
   );
```

3. Lauf mehrerer Instanzen von kernel (OpenCL-Funktion) mit einer Beschreibung des Problemraums und der Angabe der Größe des lokalen Teilmenge der Aufgaben in der Gruppe

```
bool  CLExecute(
   int          kernel,                 // Handle auf den Kernel von OpenCL-Programm
   uint         work_dim,               // Dimension des Raumes von Aufgaben
   const uint&  global_work_offset[],   // Anfangsoffset im Raum von Aufgaben
   const uint&  global_work_size[],     // Gesamtzahl der Aufgaben
   const uint&  local_work_size[]       // Anzahl der Aufgaben in der lokalen Gruppe
   );
```

Parameter

kernel

[in] Handle auf den OpenCL-Kernel.

work_dim

[in] Dimension des Raumes von Aufgaben.

global_work_offset[]

[in] Anfangsoffset im Raum von Aufgaben.

global_work_size[]

[in] Die Größe einer Teilmenge von Aufgaben.

local_work_size[]

[in] Die Größe des lokalen Teilmenge der Aufgaben in der Gruppe.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

© 2000-2025, MetaQuotes Ltd.

Betrachten wir das Beispiel der Bedeutung der Parameter:

- *work_dim* setzt die Dimension des Arrays *work_items[]*, das die Aufgaben beschreibt. Wenn *work_dim*=3, dann wird ein 3-dimensionales Array *work_items[N1, N2, N3]* verwendet.
- *global_work_size[]* enthält Werte, die die Größe des Arrays *work_items[]* angeben. Wenn work_dim = 3, und dementsprechend kann das Array *global_work_size[3]* 40 {100, 320} sein. Dann haben wir *work_items[40, 100, 320]*. Damit die Gesamtanzahl von Aufgaben ist gleich 40 x 100 x 320 = 1 280 000.
- *local_work_size[]* definiert eine Menge der Augaben, die der angegeben Kernel des OpenCL-Programms erfüllen wird. Seine Dimension ist der Dimension von *work_items[]* gleich und erlaubt eine gemeinsame Teilmenge von Aufgaben in kleinere Teilmengen ohne Reste aus der Teilung zu schneiden. In der Tat müssen die Dimensionen des Arrays *local_work_size[]* so gewählt werden, dass die globale Gruppe von Aufgaben *work_items[]* in kleinere Teilmengen geschnitten werden kann. In diesem Beispiel verwenden wir *local_work_size[3]={10, 10, 10}*, weil *work_items[40, 100, 320]* ohne Rest aus dem Array *local_items[10, 10, 10]* gesamt werden kann.

© 2000-2025, MetaQuotes Ltd.

CLExecutionStatus

Gibt den Ausführungsstatus eines OpenCL Programms zurück.

```
int  CLExecutionStatus(
   int   kernel            // Handle des Kernels des OpenCL Programms
   );
```

Parameter

kernel

[in] Handle des Kernels eines OpenCL Programms.

Rückgabewert

Gibt den Ausführungsstatus des OpenCL Programms zurück. Es kann einen der folgenden Werte einnehmen:

- CL_COMPLETE=0 - Programm beendet,
- CL_RUNNING=1 - wird ausgeführt,
- CL_SUBMITTED=2 - zur Ausführung übermittelt,
- CL_QUEUED=3 - steht auf der Warteliste zur Ausführung,
- -1 (minus eins) - bei derAusführung von CLExecutionStatus() ist ein Fehler aufgetreten.

© 2000-2025, MetaQuotes Ltd.

Arbeiten mit der Datenbank

Arbeiten mit der Datenbank

Die Funktionen für die Arbeit mit Datenbanken basieren auf der beliebten und einfach zu bedienende SQLite Engine. Die bequeme Eigenschaft dieser Engine besteht darin, dass sich die gesamte Datenbank in einer einzigen Datei auf der Festplatte des Nutzer-PCs befindet.

Die Funktionen ermöglichen das bequeme Erstellen von Tabellen, das Hinzufügen von Daten, die Durchführung von Änderungen und die Arbeit mit Stichproben über den einfachen SQL-Anfragen:

- die Abfrage von Handelshistorie und Kursdaten aus beliebigen Formaten,
- das Speichern von Optimierungs- und Testergebnissen,
- Vorbereitung und Austausch von Daten mit anderen Analysepaketen,
- Speicherung von MQL5-Anwendungseinstellungen und -Status.

Abfragen erlauben die Verwendung von statistischen und mathematischen Funktionen.

Die Funktionen für die Arbeit mit Datenbanken ermöglichen es, die sich am häufigsten wiederholenden Operationen zur Behandlung großer Datenfelder durch SQL-Anfragen zu ersetzen, so dass es oft möglich ist, die Aufrufe DatenbankAusführen/DatenbankVorbereiten zu verwenden, anstatt komplexe Schleifen und Vergleiche zu programmieren. Verwenden Sie die Funktion DatabaseReadBind, um ganz einfach die Abfrageergebnisse in einer vorgefertigten Struktur zu erhalten. Die Funktion ermöglicht das Lesen aller Datensatzfelder auf einmal innerhalb eines einzigen Aufrufs.

Um das Lesen, Schreiben und Modifizieren zu beschleunigen, kann eine Datenbank mit dem Flag DATABASE_OPEN_MEMORY im RAM geöffnet/angelegt werden, obwohl eine solche Datenbank nur für eine bestimmte Anwendung verfügbar ist und nicht gemeinsam genutzt wird. Bei der Arbeit mit Datenbanken, die sich auf der Festplatte befinden, sollten Einfügungen/Änderungen von Massendaten in Transaktionen mit DatabaseTransactionBegin/DatabaseTransactionCommit/DatabaseTransactionRollback umhüllt werden. Dies beschleunigt den Prozess um das Hundertfache.

Um die Arbeit mit den Funktionen zu beginnen, lesen Sie den Artikel SQLite: Natives Arbeiten mit SQL-Datenbanken in MQL5.

Funktion	Aktion
DatabaseOpen	Öffnet oder erstellt die angegebene Datenbankdatei
DatabaseClose	Schließt die Datenbank
DatabaseImport	Importieren von Daten aus einer Datei in eine Tabelle
DatabaseExport	Exportiert eine Tabelle oder das Ergebnis einer SQL-Anfrage in eine CSV-Datei
DatabasePrint	Druckt eine Tabelle oder ein Ergebnis der SQL-Anfrage im Experten-Journal aus
DatabaseTableExists	Prüft das Vorhandensein der Tabelle in einer Datenbank
DatabaseExecute	Ausführung einer Anfrage an die angegebene Datenbank

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
DatabasePrepare	Erstellt das Handle für die Anfragen, die durch DatabaseRead() ausgeführt werden kann
DatabaseReset	Rücksetzen einer Anfrage, wie nach dem Aufruf von DatabasePrepare()
DatabaseBind	Setzt einen Parameterwert in einer Anfrage
DatabaseBindArray	Setzt einen Parameterarray als Parameterwert
DatabaseRead	Wechselt zum nächsten Eintrag als Ergebnis einer Anfrage
DatabaseReadBind	Wechselt zum nächsten Datensatz und liest aus ihm Daten in die Struktur
DatabaseFinalize	Entfernt eine Anfrage, die durch DatabasePrepare() erstellt wurde
DatabaseTransactionBegin	Startet die Ausführung der Transaktion
DatabaseTransactionCommit	Schließt die Ausführung der Transaktion ab
RDatabaseTransactionRollback	Zurücksetzen der Transaktionen
DatabaseColumnsCount	Abrufen der Felderanzahl einer Anfrage
DatabaseColumnName	Abrufen des Feldnamens nach dem Index
DatabaseColumnType	Abrufen des Feldtyps nach dem Index
DatabaseColumnSize	Abrufen der Feldgröße in Bytes
DatabaseColumnText	Abrufen des Feldwerts als Zeichenkette aus dem aktuellen Datensatz
DatabaseColumnInteger	Abrufen des Integer-Werts aus dem aktuellen Datensatz
DatabaseColumnLong	Abrufen des Long-Werts aus dem aktuellen Datensatz
DatabaseColumnDouble	Abrufen des Double-Wertes aus dem aktuellen Datensatz
DatabaseColumnBlob	Abrufen des Feldwerts als Array aus dem aktuellen Datensatz

Statistische Funktionen:

- mode - Modus
- median - Median (50. Perzentil)
- percentile_25 - 25. Perzentil
- percentile_75
- percentile_90
- percentile_95
- percentile_99

© 2000-2025, MetaQuotes Ltd.

- stddev oder stddev_samp — Standardabweichung der Stichprobe
- stddev_pop — Standardabweichung der Grundgesamtheit
- variance or var_samp — Varianz der Stichprobe
- var_pop — Varianz der Grundgesamtheit

Mathematische Funktionen

- acos(X) - Arkuscosinus im Bogenmaß
- acosh(X) - hyperbolischer Arkuscosinus
- asin(X) - Arkussinus im Bogenmaß
- asinh(X) - hyperbolischer Arkussinus
- atan(X) - Arkustangens im Bogenmaß
- atan2(X,Y) - Arkustangens im Bogenmaß des Verhältnisses X/Y
- atanh(X) - hyperbolischer Arkustangens
- ceil(X) - Aufrunden auf eine ganze Zahl
- ceiling(X) - Aufrunden auf eine ganze Zahl
- cos(X) - Cosinus des Winkels im Bogenmaß
- cosh(X) - hyperbolischer Cosinus
- degrees(X) - Umrechnung eines Bogenmaßes in einen Winkel
- exp(X) - Exponent
- floor(X) - Abrunden auf eine ganze Zahl
- ln(X) - natürlicher Logarithmus
- log(B,X) - Logarithmus zur angegebenen Basis
- log(X) - dezimaler Logarithmus
- log10(X) - dezimaler Logarithmus
- log2(X) - Logarithmus zur Basis 2
- mod(X,Y) - Rest der Division
- pi() - Näherung von Pi
- pow(X,Y) - potenzieren der angegebenen Basis mit dem Exponenten
- power(X,Y) - potenzieren der angegebenen Basis mit dem Exponenten
- radians(X) - wandelt den Winkel in Bogenmaß um
- sin(X) - Sinus eines Winkels in Bogenmaß
- sinh(X) - hyperbolischer Sinus
- sqrt(X) - Quadratwurzel
- tan(X) - Tangens eines Winkels in Bogenmaß
- tanh(X) - hyperbolischer Tangens
- trunc(X) - Abschneiden auf die ganze Zahl, die näher 0 ist

Beispiel:

```
select
  count(*) as book_count,
```

© 2000-2025, MetaQuotes Ltd.

```
  cast(avg(parent) as integer) as mean,
  cast(median(parent) as integer) as median,
  mode(parent) as mode,
  percentile_90(parent) as p90,
  percentile_95(parent) as p95,
  percentile_99(parent) as p99
from moz_bookmarks;
```

© 2000-2025, MetaQuotes Ltd.

DatabaseOpen

Öffnet oder erstellt die angegebene Datenbankdatei.

```
int  DatabaseOpen(
   string  filename,       // Dateiname
   uint    flags           // Kombination der Flags
   );
```

Parameter

filename

[in] Dateiname in dem Verzeichnis "MQL5\Files".

flags

[in] Kombination der Flags gemaäß der Enumeration ENUM_DATABASE_OPEN_FLAGS.

Rückgabewert

Wenn die Funktion erfolgreich ausgeführt wurde, gibt sie das Datenbank-Handle zurück, mit dem dann auf die Datenbank zugegriffen wird. Andernfalls wird INVALID_HANDLE zurückgegeben. Um die Fehlernummer abzurufen, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_WRONG_INTERNAL_PARAMETER (4002) - interner Fehler beim Zugriff auf das Verzeichnis "MQL5\Files";
- ERR_INVALID_PARAMETER (4003) - Der Pfad zur Datenbankdatei ist leer oder es wurde eine inkompatible Kombination der Flags übergeben;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_WRONG_FILENAME (5002) - der Name der Datenbankdatei ist falsch;
- ERR_TOO_LONG_FILENAME (5003) - absoluter Pfad zur Datenbankdatei übersteigt die maximale Länge;
- ERR_DATABASE_TOO_MANY_OBJECTS (5122) - es wurde die Maximalzahl an Datenbankobjekten überschritten;
- ERR_DATABASE_CONNECT (5123) - Verbindungsfehler zur Datenbank;
- ERR_DATABASE_MISUSE (5621) - falsche Verwendung der SQLite-Bibliothek.

Hinweis

Wenn der Parameter *filename* NULL oder die leere Zeichenfolge "" enthält, wird eine temporäre Datei auf der Festplatte erstellt. Sie wird nach dem Schließen der Datenbankverbindung automatisch gelöscht.

Wenn der Parameter *filename* den Wert ":memory:" enthält, wird die Datenbank im Speicher angelegt und nach dem Schließen der Verbindung zu ihr automatisch gelöscht.

Wenn der Parameter *flags* keines der Flags DATABASE_OPEN_READONLY oder DATABASE_OPEN_READWRITE aufweist, wird das Flag DATABASE_OPEN_READWRITE verwendet.

Wenn die Dateierweiterung nicht angegeben wurde, wird ".sqlite" verwendet.

ENUM_DATABASE_OPEN_FLAGS

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
DATABASE_OPEN_READONLY	Schreibgeschützt
DATABASE_OPEN_READWRITE	Öffnen zum Lesen und Schreiben
DATABASE_OPEN_CREATE	Datei auf der Festplatte erstellen, falls nötig
DATABASE_OPEN_MEMORY	Datenbank im RAM erstellen
DATABASE_OPEN_COMMON	Die Datei befindet sich im Verzeichnis "Common" des Terminals

Siehe auch

DatabaseClose

© 2000-2025, MetaQuotes Ltd.

DatabaseClose

Schließt die Datenbank

```
void  DatabaseClose(
   int  database      // Handle der Datenbank erhalten von DatabaseOpen
   );
```

Parameter

database

[in] Handle der Datenbank erhalten von DatabaseOpen().

Rückgabewert

Keiner.

Hinweis

Nach dem Aufruf von DatabaseClose, werden alle Handles der Anfragen an die Datenbank automatisch entfernt und werden ungültig.

Sind die Handles ungültig, wirft die Funktion den Fehler ERR_DATABASE_INVALID_HANDLE aus. Das kann mit der Funktion GetLastError() geprüft werden.

Siehe auch

DatabaseOpen, DatabasePrepare

© 2000-2025, MetaQuotes Ltd.

DatabaseImport

Importieren von Daten aus einer Datei in eine Tabelle.

```
long  DatabaseImport(
   int           database,          // Handle der Datenbank, erhalten von DatabaseOper
   const string  table,             // Tabellenname der dort einzutragenden Daten
   const string  filename,          // Dateiname mit den zu importierenden Daten
   uint          flags,             // Kombination der Flags
   const string  separator,         // Trennzeichen der Daten
   ulong         skip_rows,         // Wieviele Zeichenketten sollen am Anfang übersp
   const string  skip_comments      // Satz von Zeichen zur Kennzeichnung eines Kommen
   );
```

Parameter

database

[in] Handle der Datenbank, erhalten von DatabaseOpen().

table

[in] Name der Tabelle, der die Daten aus einer Datei hinzugefügt werden sollen.

filename

[in] CSV-Datei oder ZIP-Archive mit den zu lesenden Daten. Der Name darf Unterverzeichnisse enthalten und ist relativ zu dem Verzeichnis MQL5\Files.

flags

[in] Kombination der Flags aus der Enumeration ENUM_DATABASE_IMPORT_FLAGS.

separator

[in] Trennzeichen der Daten in der CSV-Datei.

skip_rows

[in] Anzahl der anfangs zu überspringenden Zeilen, wenn Daten aus einer Datei gelesen werden sollen.

skip_comments

[in] Satz von Zeichen, die eine Zeichenkette als Kommentar kennzeichnen. Wenn ein Zeichen aus *skip_comments* am Anfang einer Zeichenkette erkannt wird, wird eine solche Zeichenkette als Kommentar betrachtet und nicht importiert.

Rückgabewert

Anzahl der importierten Einträge oder -1 im Fehlerfall. Um den Fehlercode zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - kein Tabellenname angegeben (leere Zeichenkette oder NULL);
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Handle der Datenbank.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Wenn es keine Tabelle mit dem Namen *table* gibt, wird sie automatisch generiert. Namen und Feldtypen in der erzeugten Tabelle werden automatisch auf der Grundlage der Dateidaten definiert.

ENUM_DATABASE_IMPORT_FLAGS

ID	Beschreibung
DATABASE_IMPORT_HEADER	Die erste Zeile enthält die Namen der Tabellenfelder
DATABASE_IMPORT_CRLF	CRLF (Standard ist LF) wird als Zeilenumbruch betrachtet
DATABASE_IMPORT_APPEND	Hinzufügen von Daten an das Ende einer bestehenden Tabelle
DATABASE_IMPORT_QUOTED_STRINGS	Zeichenkettenwerte werden in Anführungszeichen eingeschlossen.
DATABASE_IMPORT_COMMON_FOLDER	Die Datei wird im gemeinsamen Ordner aller Client-Terminals \Terminal\Common\File gespeichert.

Beispiel für das Lesen der Tabelle aus der Datei, die mit dem Code aus dem Beispiel DatabaseExport erstellt wurde:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
  string csv_filename;
//--- Abrufen der Namen der Textdateien zum Laden aus dem Common-Verzeichnis des Klier
  string filenames[];
  if(FileSelectDialog("Wählen einer CSV-Date, die in einer Tabelle gespeichert werden
                      "Textdateien (*.csv)|*.csv",
                      FSD_WRITE_FILE|FSD_COMMON_FOLDER, filenames, "data.csv")>0)
   {
    //--- Anzeige der Namen von jeder gewählten Datei
    if(ArraySize(filenames)==1)
      csv_filename=filenames[0];
    else
     {
      Print("Unbekannter Fehler während der Dateiauswahl. Fehlernummer: " GetLastError
      return;
     }
   }
  else
   {
    Print("Keine CSV-Datei ausgewählt");
    return;
```

© 2000-2025, MetaQuotes Ltd.

```
   }
//--- Erstellen oder Öffnen einer Datenbank
  string db_filename="test.sqlite";
  int db=DatabaseOpen(db_filename, DATABASE_OPEN_READWRITE|DATABASE_OPEN_CREATE);
//--- Prüfung, ob Testtabelle existiert
  if(DatabaseTableExists(db, "TEST"))
   {
    //--- entfernen der Tabelle TEST
    if(!DatabaseExecute(db, "TABELLE TEST LÖSCHEN, WENN VORHANDEN"))
     {
      Print("Löschen der Tabelle TEST ist fehlgeschlagen mit der Fehlernummer ", GetLa
      DatabaseClose(db);
      return;
     }
   }
  //--- Einträge aus der Datei in die Tabelle TEST importieren
  long imported=DatabaseImport(db, "TEST", csv_filename, DATABASE_IMPORT_HEADER|DATABA
  if(imported>0)
   {
    Print(imported," Zeilen in die Tabelle TEST importiert");
    DatabasePrint(db,"SELECT * FROM TEST",DATABASE_PRINT_NO_INDEX);
   }
  else
    {
     Print("DatabaseImport() ist fehlgeschlagen. Error ",GetLastError());
    }
//--- Schließen der Datenbankdatei und Informieren darüber
  DatabaseClose(db);
  PrintFormat("Database: %s wurde geschlossen", db_filename);
 }
```

Siehe auch

DatabaseOpen, DatabasePrint

© 2000-2025, MetaQuotes Ltd.

DatabaseExport

Exportiert eine Tabelle oder das Ergebnis einer SQL-Anfrage in eine CSV-Datei. Die erstellte Datei ist in UTF-8 encodiert.

```
long  DatabaseExport(
   int           database,          // Handle der Datenbank, erhalten von DatabaseOpe
   const string  table_or_sql,      // ein Tabellenname oder eine SQL-Anfrage
   const string  filename,          // der Name der CSV-Datei für den Datenexport
   uint          flags,             // Kombination der Flags
   const string  separator          // Trennzeichen in der CSV-Datei
   );
```

Parameter

database

[in] Handle der Datenbank, erhalten von DatabaseOpen().

table_or_sql

[in] Ein Name einer Tabelle oder ein Text einer SQL-Anfrage, deren Ergebnisse in eine bestimmte Datei exportiert werden sollen.

filename

[in] Ein Dateiname für den Datenexport. Der Pfad wird relativ zum Ordner MQL5\Files festgelegt.

flags

[in] Kombination von Flags aus der Enumeration ENUM_DATABASE_EXPORT_FLAGS.

separator

[in] Trennzeichen der Daten. Wenn NULL angegeben wurde, wird der Tabulator '\t' als Trennzeichen verwendet. Eine leere Zeichenkette "" wird als gültiges Trennzeichen betrachtet, aber die erhaltene CSV-Datei kann nicht als Tabelle gelesen werden - sie wird als eine Reihe von Zeichenfolgen betrachtet.

Rückgabewert

Anzahl der exportierten Einträge oder im Fehlerfall einen negativen Wert. Um den Fehlercode zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_INVALID_PARAMETER (4003) - Der Pfad zur Datenbankdatei ist leer oder es wurde eine inkompatible Kombination der Flags übergeben;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_FUNCTION_NOT_ALLOWED(4014) - angegebene Pipe ist nicht erlaubt;
- ERR_PROGRAM_STOPPED(4022) - Operation abgebrochen (MQL-Programm gestoppt);
- ERR_WRONG_FILENAME (5002) - Ungültiger Dateiname;
- ERR_TOO_LONG_FILENAME (5003) - absoluter Pfad überschreitet die Maximallänge;
- ERR_CANNOT_OPEN_FILE(5004) - Datei konnte nicht zum Schreiben geöffnet werden;
- ERR_FILE_WRITEERROR(5026) - In die Datei konnte nicht geschrieben werden;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Handle der Datenbank;

© 2000-2025, MetaQuotes Ltd.

- ERR_DATABASE_QUERY_PREPARE(5125) - Fehler bei der Anfrageerstellung;
- ERR_DATABASE_QUERY_NOT_READONLY - es sind Nur-Lese-Anfragen erlaubt.

Hinweis

Wenn die Abfrageergebnisse exportiert werden, sollte die SQL-Abfrage mit "SELECT" oder "select" beginnen. Mit anderen Worten, die SQL-Anfrage kann den Datenbankstatus nicht ändern, andernfalls schlägt DatabaseExport() mit einer Fehlermeldung fehl.

Datenbank-String-Werte können das Konvertierungszeichen ('\r' oder '\r\n') sowie den Satz der Trennzeichen für die Werte im Parameter *Trenner* enthalten. Stellen Sie in diesem Fall sicher, dass Sie das Flag DATABASE_EXPORT_QUOTED_STRINGS im Parameter 'flags' verwenden. Wenn dieses Flag vorhanden ist, werden alle angezeigten Zeichenfolgen in doppelte Anführungszeichen gesetzt. Wenn eine Zeichenkette ein doppeltes Anführungszeichen enthält, wird es durch zwei doppelte Anführungszeichen ersetzt.

ENUM_DATABASE_EXPORT_FLAGS

ID	Beschreibung
DATABASE_EXPORT_HEADER	Feldnamen in der ersten Zeil anzeigen
DATABASE_EXPORT_INDEX	Anzeige des Zeilenindex
DATABASE_EXPORT_NO_BOM	Keine "BOM"-Markierung am Anfang der Datei einfügen (BOM wird standardmäßig eingefügt)
DATABASE_EXPORT_CRLF	CRLF für den Zeilenumbruch verwenden (die Voreinstellung ist LF)
DATABASE_EXPORT_APPEND	Hinzufügen von Daten an das Ende einer bestehenden Datei (standardmäßig wird die Datei überschrieben). Wenn die Datei nicht existiert wird sie erstellt.
DATABASE_EXPORT_QUOTED_STRINGS	Zeichenketten in Anführungszeichen anzeigen.
DATABASE_EXPORT_COMMON_FOLDER	Eine CSV-Datei wird im gemeinsamen Ordner "Common" aller Client-Terminals \Terminal\Common\File erstellt.

Beispiel:

```
input int InpRates=100;
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
 {
  MqlRates rates[];
//--- Merken der Startzeit vor dem Empfang der Balken
  ulong start=GetMicrosecondCount();
//--- Abfrage der letzten 100 H1-Balken
```

© 2000-2025, MetaQuotes Ltd.

```
  if(CopyRates(Symbol(), PERIOD_H1, 1, InpRates, rates)<InpRates)
    {
     Print("CopyRates() ist fehlgeschlagen, Fehler: ", GetLastError());
     return;
    }
  else
    {
 //--- wie viele Balken empfangen wurden und wie lange es dauerte, sie zu empfangen
     PrintFormat("%s: CopyRates erhielt %d Balken in %d ms ",
                 _Symbol, ArraySize(rates), (GetMicrosecondCount()-start)/1000);
    }
//--- Setzen des Dateinamen zum Speichern der Datenbank
  string filename=_Symbol+"_"+EnumToString(PERIOD_H1)+"_"+TimeToString(TimeCurrent())-
  StringReplace(filename, ":", "-"); // ":" nichterlaubtes Zeichen in Dateinamen
//--- Erstellen/Öffnen einer Datenbank im Verzeichnis Common des Terminals
  int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE|DATABASE_OPEN_CREATE|DATABASE_
  if(db==INVALID_HANDLE)
    {
     Print("Database: ", filename, " Öffnen fehlgeschlagen, Fehlernummer: ", GetLastErr
     return;
    }
  else
     Print("Database: ", filename, " erfolgreich geöffnet");

//--- Prüfung ob die Tabelle RATES existiert
  if(DatabaseTableExists(db, "RATES"))
    {
     //--- Entfernen der Tabelle RATES
     if(!DatabaseExecute(db, "TABELLE RATES LÖSCHEN, WENN VORHANDEN"))
      {
       Print("Löschen der Tabelle TEST ist fehlgeschlagen mit der Fehlernummer ", GetLa
       DatabaseClose(db);
       return;
      }
    }
//--- Erstellen der Tabelle RATES
  if(!DatabaseExecute(db, "Erstellen der Tabelle RATES("
                      "SYMBOL             CHAR(10),"
                      "TIME               INT NOT NULL,"
                      "OPEN               REAL,"
                      "HIGH               REAL,"
                      "LOW                REAL,"
                      "CLOSE              REAL,"Nahe am realen
                      "TICK_VOLUME        INT,"
                      "SPREAD             INT,"
                      "REAL_VOLUME        INT);"))
    {
     Print("DB: ", filename, " Erstellen der Tabelle RATES, Fehlernummer: ", GetLastErr
     DatabaseClose(db);
```

© 2000-2025, MetaQuotes Ltd.

```
    return;
   }
//--- Anzeige der Liste aller Felder in der Tabelle RATES
  if(DatabasePrint(db, "PRAGMA TABLE_INFO(RATES)", 0)<0)
   {
    PrintFormat("DatabasePrint(\"PRAGMA TABLE_INFO(RATES)\") fehlgeschlagen, Fehlernur
    DatabaseClose(db);
    return;
   }
//--- Erstellen einer parametrisierten Anfrage zum Hinzufügen von Balken zur Tabelle R
  string sql="INSERT INTO RATES (SYMBOL,TIME,OPEN,HIGH,LOW,CLOSE,TICK_VOLUME,SPREAD,RE
             " VALUES (?1,?2,?3,?4,?5,?6,?7,?8,?9)"; // benötigte Parameter
  int request=DatabasePrepare(db, sql);
  if(request==INVALID_HANDLE)
   {
    PrintFormat("DatabasePrepare() fehlgeschlagen, Fehlernummer=%d", GetLastError());
    Print("SQL Anfrage: ", sql);
    DatabaseClose(db);
    return;
   }
//--- Setzen des Wertes des ersten Parameters der Anfrage
  DatabaseBind(request, 0, _Symbol);
//--- Merken der Startzeit, bevor die Balken der Tabelle RATES hinzugefügt werden
  start=GetMicrosecondCount();
  DatabaseTransactionBegin(db);
  int total=ArraySize(rates);
  bool request_error=false;
  for(int i=0; i<total; i++)
   {
    //--- Festlegen der übrigen Parameter vor dem Hinzufügen des Eintrags die Werte
    ResetLastError();
    if(!DatabaseBind(request, 1, rates[i].time))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    //--- wenn der vorherige DatabaseBind()-Aufruf erfolgreich war, wird der nächste P
    if(!request_error && !DatabaseBind(request, 2, rates[i].open))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    if(!request_error && !DatabaseBind(request, 3, rates[i].high))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    if(!request_error && !DatabaseBind(request, 4, rates[i].low))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    if(!request_error && !DatabaseBind(request, 5, rates[i].close))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    if(!request_error && !DatabaseBind(request, 6, rates[i].tick_volume))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    if(!request_error && !DatabaseBind(request, 7, rates[i].spread))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }
    if(!request_error && !DatabaseBind(request, 8, rates[i].real_volume))
     {
      PrintFormat("DatabaseBind() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      PrintFormat("Balken #%d Zeile=%d", i+1, __LINE__);
      request_error=true;
      break;
     }

    //--- eine Anfrage zum Einfügen des Eintrags ausführen und auf Fehler prüfen
    if(!request_error && !DatabaseRead(request) && (GetLastError()!=ERR_DATABASE_NO_MO
     {
      PrintFormat("DatabaseRead() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      DatabaseFinalize(request);
      request_error=true;
      break;
     }
    //--- die Anfrage vor der nächsten Aktualisierung der Parameter zurücksetzen
```

© 2000-2025, MetaQuotes Ltd.

```
    if(!request_error && !DatabaseReset(request))
     {
      PrintFormat("DatabaseReset() fehlgeschlagen, Fehlernummer=%d", GetLastError());
      DatabaseFinalize(request);
      request_error=true;
      break;
     }
   } //--- fertig, alle Balken durchlaufen

//--- Transaktionsstatus
  if(request_error)
   {
    PrintFormat("Tabelle RATES: hinzufügen von %d Balken ist fehlgeschlagen", ArraySiz
    DatabaseTransactionRollback(db);
    DatabaseClose(db);
    return;
   }
  else
   {
    DatabaseTransactionCommit(db);
    PrintFormat("Tabelle RATES: %d Balken in %d ms hinzugefügt",
                ArraySize(rates), (GetMicrosecondCount()-start)/1000);
   }
//--- Speichern der Tabelle RATES in einer CSV-Datei
  string csv_filename=Symbol()+".csv";
  long saved=DatabaseExport(db, "SELECT * FROM RATES", csv_filename, DATABASE_EXPORT_
  if(saved>0)
    Print("Tabelle RATES gespeichert in ", Symbol(), ".csv");
  else
    Print("DatabaseExport() fehlgeschlagen mit Fehlernummer ", GetLastError());
//--- Schließen der Datenbankdatei und Informieren darüber
  DatabaseClose(db);
  PrintFormat("Database: %s erstellt und geschlossen", filename);
 }
```

Siehe auch

DatabasePrint, DatabaseImport

© 2000-2025, MetaQuotes Ltd.

DatabasePrint

Druckt eine Tabelle oder ein Ergebnis der SQL-Anfrage im Experten-Journal aus.

```
long  DatabasePrint(
   int           database,          // Handle der Datenbank, erhalten von DatabaseOper
   const string  table_or_sql,      // eine Tabelle oder eine SQL-Anfrage
   uint          flags              // Kombination der Flags
   );
```

Parameter

database

[in] Handle der Datenbank, erhalten von DatabaseOpen().

table_or_sql

[in] Der Name einer Tabelle oder der Text einer SQL-Anfrage, deren Ergebnisse im Journal des Experten angezeigt wird.

flags

[in] Kombination von Flags, die die Ausgabeformatierung definieren. Die Flags sind wie folgt definiert:

DATABASE_PRINT_NO_HEADER - die Spaltenüberschriften werden nicht angezeigt (Feldnamen)
DATABASE_PRINT_NO_INDEX - die Indices der Zeichenketten werden nicht gezeigt
DATABASE_PRINT_NO_FRAME - es wird kein Rahmen gezeigt, der den Kopf von den Daten trennt
DATABASE_PRINT_STRINGS_RIGHT - rechtsbündige Zeichenketten.

Wenn flags=0, werden die Spalten und die Zeichenketten angezeigt, die Kopfzeile und die Daten werden durch einen Rahmen getrennt, während die Zeichenketten linksbündig ausgerichtet werden.

Rückgabewert

Anzahl der exportierten Zeichenfolgen oder -1 im Fehlerfall. Um den Fehlercode zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Handle der Datenbank;

Hinweis

Wenn das Journal Abfrageergebnisse anzeigt, sollte die SQL-Abfrage mit "SELECT" oder "select" beginnen. Mit anderen Worten, die SQL-Anforderung kann den Datenbankstatus nicht ändern, andernfalls schlägt DatabasePrint() mit einer Fehlermeldung fehl.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   string filename="departments.sqlite";
//--- Erstellen oder Öffnen einer Datenbank im Verzeichnis Common des Terminals
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE |DATAB
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " konnte nicht geöffnet werden, Fehlermeldung "GetLastEr
      return;
     }

//--- Erstellen der Datenbank COMPANY
   if(!CreataTableCompany(db))
     {
      DatabaseClose(db);
      return;
     }
//--- Erstellen der Datenbank DEPARTMENT
   if(!CreataTableDepartment(db))
     {
      DatabaseClose(db);
      return;
     }

//--- Darstellung der Liste aller Felder der Tabellen von COMPANY und DEPARTMENT
   PrintFormat("Ausdrucksversuch von \"PRAGMA TABLE_INFO(COMPANY);PRAGMA TABLE_INFO(DE
   if(DatabasePrint(db, "PRAGMA TABLE_INFO(COMPANY);PRAGMA TABLE_INFO(DEPARTMENT)", 0)
     {
      PrintFormat("DatabasePrint(\"PRAGMA TABLE_INFO()\") fehlgeschlagen, Fehlercode=%
      DatabaseClose(db);
      return;
     }
//--- Darstellung der Tabelle COMPANY im Log
   PrintFormat("Ausdrucksversuch von \"SELECT * from COMPANY\"");
   if(DatabasePrint(db, "SELECT * from COMPANY", 0)<0)
     {
      Print("DatabasePrint fehlgeschlagen, Fehlercode "GetLastError());
      DatabaseClose(db);
      return;
     }
//--- kombinierter Anfragetext der Tabellen von COMPANY und DEPARTMENT
   string request="SELECT EMP_ID, NAME, DEPT FROM COMPANY LEFT OUTER JOIN DEPARTMENT "
                  "ON COMPANY.ID = DEPARTMENT.EMP_ID";
//--- Darstellung des Ergebnisses der kombinierten Tabellen
   PrintFormat("Ausdrucksversuch von \"SELECT EMP_ID, NAME, DEPT FROM COMPANY LEFT OUT
   if(DatabasePrint(db, request, 0)<0)
     {
      Print("DatabasePrint fehlgeschlagen, Fehlercode "GetLastError());
      DatabaseClose(db);
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
     }
//--- Schließen der Datenbank
   DatabaseClose(db);
  }
/*
Ergebnis:
Try to print request "PRAGMA TABLE_INFO(COMPANY);PRAGMA TABLE_INFO(DEPARTMENT)"
#| cid name    type     notnull dflt_value pk
-+----------------------------------------
1|   0 ID      INT            1             1
2|   1 NAME    TEXT           1             0
3|   2 AGE     INT            1             0
4|   3 ADDRESS CHAR(50)       0             0
5|   4 SALARY  REAL           0             0
#| cid name   type     notnull dflt_value pk
-+----------------------------------------
1|   0 ID     INT            1             1
2|   1 DEPT   CHAR(50)       1             0
3|   2 EMP_ID INT            1             0
Try to print request "SELECT * from COMPANY"
#| ID NAME  AGE ADDRESS     SALARY
-+--------------------------------
1|  1 Paul   32 California 25000.0
2|  2 Allen  25 Texas      15000.0
3|  3 Teddy  23 Norway     20000.0
4|  4 Mark   25 Rich-Mond  65000.0
5|  5 David  27 Texas      85000.0
6|  6 Kim    22 South-Hall 45000.0
7|  7 James  24 Houston    10000.0
Try to print request "SELECT EMP_ID, NAME, DEPT FROM COMPANY LEFT OUTER JOIN DEPARTMEN
#| EMP_ID NAME  DEPT
-+-------------------------
1|      1 Paul  IT Billing
2|      2 Allen Engineering
3|        Teddy
4|        Mark
5|        David
6|        Kim
7|      7 James Finance
*/
//+------------------------------------------------------------------+
//| Erstellen der Tabelle COMPANY                                    |
//+------------------------------------------------------------------+
bool CreateTableCompany(int database)
  {
//--- Wenn die Tabelle COMPANY existiert, wird sie gelöscht
   if(DatabaseTableExists(database, "COMPANY"))
     {
      //--- löschen der Tabelle
```

© 2000-2025, MetaQuotes Ltd.

```
      if(!DatabaseExecute(database, "DROP TABLE COMPANY"))
        {
         Print("Löschen der Tabelle COMPANY ist fehlgeschlagen, Fehlermeldung "GetLast
         return(false);
        }
     }
//--- Erstellen der Datenbank COMPANY
   if(!DatabaseExecute(database, "CREATE TABLE COMPANY("
                       "ID INT PRIMARY KEY     NOT NULL,"
                       "NAME           TEXT    NOT NULL,"
                       "AGE            INT     NOT NULL,"
                       "ADDRESS        CHAR(50),"
                       "SALARY         REAL );"))
     {
      Print("DB: Erstellen der Tabelle COMPANY ist fehlgeschlagen, Fehlermeldung "GetI
      return(false);
     }

//--- Eintragen der Daten in die Tabelle COMPANY
   if(!DatabaseExecute(database, "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VAI
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'A
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'T
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'M
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (5, 'D
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (6, 'P
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (7, 'J
     {
      Print("Eintragen in COMPANY ist fehlgeschlagen, Fehlermeldung " GetLastError());
      return(false);
     }
//--- Gelungen
   return(true);
  }
//+------------------------------------------------------------------+
//| Erstellen der Tabelle DEPARTMENT                                 |
//+------------------------------------------------------------------+
bool CreateTableDepartment(int database)
  {
//--- Wenn die Tabelle DEPARTMENT existiert, wird sie gelöscht
   if(DatabaseTableExists(database, "DEPARTMENT"))
     {
      //--- löschen der Tabelle
      if(!DatabaseExecute(database, "DROP TABLE DEPARTMENT"))
        {
         Print("Failed to drop table DEPARTMENT  with code ", GetLastError());
         return(false);
        }
     }
//--- Erstellen der Datenbank DEPARTMENT
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!DatabaseExecute(database, "CREATE TABLE DEPARTMENT ("
                       "ID      INT PRIMARY KEY   NOT NULL,"
                       "DEPT    CHAR(50)          NOT NULL,"
                       "EMP_ID  INT               NOT NULL);"))
     {
      Print("DB: Erstellen der Tabelle DEPARTMENT ist fehlgeschlagen, Fehlermeldung "C
      return(false);
     }

//--- Eintragen der Daten in die Tabelle DEPARTMENT
   if(!DatabaseExecute(database, "INSERT INTO DEPARTMENT (ID,DEPT,EMP_ID) VALUES (1, '
                       "INSERT INTO DEPARTMENT (ID,DEPT,EMP_ID) VALUES (2, 'Engineerir
                       "INSERT INTO DEPARTMENT (ID,DEPT,EMP_ID) VALUES (3, 'Finance',
     {
      Print("Eintragen in DEPARTMENT ist fehlgeschlagen, Fehlermeldung " GetLastError
      return(false);
     }
//--- Gelungen
   return(true);
  }
//+------------------------------------------------------------------+
```

Siehe auch

DatabaseExport, DatabaseImport

© 2000-2025, MetaQuotes Ltd.

DatabaseTableExists

Prüft das Vorhandensein der Tabelle in einer Datenbank.

```
bool  DatabaseTableExists(
   int     database,      // Handle der Datenbank, erhalten von DatabaseOpen
   string  table          // Tabellennamen
   );
```

Parameter

database

[in] Handle der Datenbank, erhalten von DatabaseOpen().

table

[in] Tabellenname.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - Tabellenname wurde nicht angegeben (leer oder NULL);
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) - es existiert keine Tabelle (kein Fehler, normales Ende).

Siehe auch

DatabasePrepare, DatabaseFinalize

© 2000-2025, MetaQuotes Ltd.

DatabaseExecute

Ausführung einer Anfrage an die angegebene Datenbank.

```
bool  DatabaseExecute(
   int     database,      // Handle der Datenbank, erhalten von DatabaseOpen
   string  sql            // SQL-Anfrage
   );
```

Parameter

database

[in] Handle der Datenbank, erhalten vo nDatabaseOpen().

sql

[in] SQL-Anfrage.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_INVALID_PARAMETER (4003) - SQL-Parameter enthält eine leere Zeichenkette;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage.

Example:

```
//--- symbol statistics
struct Symbol_Stats
  {
   string            name;             // symbol name
   int               trades;           // number of trades for the symbol
   double            gross_profit;     // total profit for the symbol
   double            gross_loss;       // total loss for the symbol
   double            total_commission; // total commission for the symbol
   double            total_swap;       // total swaps for the symbol
   double            total_profit;     // total profit excluding swaps and commissions
   double            net_profit;       // net profit taking into account swaps and cor
   int               win_trades;       // number of profitable trades
   int               loss_trades;      // number of losing trades
   double            expected_payoff;  // expected payoff for the trade excluding swap
   double            win_percent;      // percentage of winning trades
   double            loss_percent;     // percentage of losing trades
   double            average_profit;   // average profit
   double            average_loss;     // average loss
```

© 2000-2025, MetaQuotes Ltd.

```
   double            profit_factor;    // profit factor
  };

//--- Magic Number statistics
struct Magic_Stats
  {
   long              magic;            // EA's Magic Number
   int               trades;           // number of trades for the symbol
   double            gross_profit;     // total profit for the symbol
   double            gross_loss;       // total loss for the symbol
   double            total_commission; // total commission for the symbol
   double            total_swap;       // total swaps for the symbol
   double            total_profit;     // total profit excluding swaps and commissions
   double            net_profit;       // net profit taking into account swaps and com
   int               win_trades;       // number of profitable trades
   int               loss_trades;      // number of losing trades
   double            expected_payoff;  // expected payoff for the trade excluding swap
   double            win_percent;      // percentage of winning trades
   double            loss_percent;     // percentage of losing trades
   double            average_profit;   // average profit
   double            average_loss;     // average loss
   double            profit_factor;    // profit factor
  };

//--- entry hour statistics
struct Hour_Stats
  {
   char              hour_in;          // market entry hour
   int               trades;           // number of trades in this entry hour
   double            volume;           // volume of trades in this entry hour
   double            gross_profit;     // total profit in this entry hour
   double            gross_loss;       // total loss in this entry hour
   double            net_profit;       // net profit taking into account swaps and com
   int               win_trades;       // number of profitable trades
   int               loss_trades;      // number of losing trades
   double            expected_payoff;  // expected payoff for the trade excluding swap
   double            win_percent;      // percentage of winning trades
   double            loss_percent;     // percentage of losing trades
   double            average_profit;   // average profit
   double            average_loss;     // average loss
   double            profit_factor;    // profit factor
  };

int ExtDealsTotal=0;;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- create the file name
   string filename=IntegerToString(AccountInfoInteger(ACCOUNT_LOGIN))+"_stats.sqlite";
//--- open/create the database in the common terminal folder
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE | DATA
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " open failed with code ", GetLastError());
      return;
     }
//--- create the DEALS table
   if(!CreateTableDeals(db))
     {
      DatabaseClose(db);
      return;
     }
   PrintFormat("Deals in the trading history: %d ", ExtDealsTotal);

//--- get trading statistics per symbols
   int request=DatabasePrepare(db, "SELECT r.*,"
                               "   (case when r.trades != 0 then (r.gross_profit+r.gro
                               "   (case when r.trades != 0 then r.win_trades*100.0/r.
                               "   (case when r.trades != 0 then r.loss_trades*100.0/r
                               "   r.gross_profit/r.win_trades as average_profit,"
                               "   r.gross_loss/r.loss_trades as average_loss,"
                               "   (case when r.gross_loss!=0.0 then r.gross_profit/(-
                               "FROM "
                               "   ("
                               "   SELECT SYMBOL,"
                               "   sum(case when entry =1 then 1 else 0 end) as trades
                               "   sum(case when profit > 0 then profit else 0 end) as
                               "   sum(case when profit < 0 then profit else 0 end) as
                               "   sum(swap) as total_swap,"
                               "   sum(commission) as total_commission,"
                               "   sum(profit) as total_profit,"
                               "   sum(profit+swap+commission) as net_profit,"
                               "   sum(case when profit > 0 then 1 else 0 end) as win_
                               "   sum(case when profit < 0 then 1 else 0 end) as loss
                               "   FROM DEALS "
                               "   WHERE SYMBOL <> '' and SYMBOL is not NULL "
                               "   GROUP BY SYMBOL"
                               "   ) as r");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
   Symbol_Stats stats[], symbol_stats;
   ArrayResize(stats, ExtDealsTotal);
```

© 2000-2025, MetaQuotes Ltd.

```
   int i=0;
//--- get records from request results
   for(; DatabaseReadBind(request, symbol_stats) ; i++)
     {
      stats[i].name=symbol_stats.name;
      stats[i].trades=symbol_stats.trades;
      stats[i].gross_profit=symbol_stats.gross_profit;
      stats[i].gross_loss=symbol_stats.gross_loss;
      stats[i].total_commission=symbol_stats.total_commission;
      stats[i].total_swap=symbol_stats.total_swap;
      stats[i].total_profit=symbol_stats.total_profit;
      stats[i].net_profit=symbol_stats.net_profit;
      stats[i].win_trades=symbol_stats.win_trades;
      stats[i].loss_trades=symbol_stats.loss_trades;
      stats[i].expected_payoff=symbol_stats.expected_payoff;
      stats[i].win_percent=symbol_stats.win_percent;
      stats[i].loss_percent=symbol_stats.loss_percent;
      stats[i].average_profit=symbol_stats.average_profit;
      stats[i].average_loss=symbol_stats.average_loss;
      stats[i].profit_factor=symbol_stats.profit_factor;
     }
   ArrayResize(stats, i);
   Print("Trade statistics by Symbol");
   ArrayPrint(stats);
   Print("");
//--- delete the request
   DatabaseFinalize(request);

//--- get trading statistics for Expert Advisors by Magic Numbers
   request=DatabasePrepare(db, "SELECT r.*,"
                           "   (case when r.trades != 0 then (r.gross_profit+r.gross_l
                           "   (case when r.trades != 0 then r.win_trades*100.0/r.trad
                           "   (case when r.trades != 0 then r.loss_trades*100.0/r.tra
                           "   r.gross_profit/r.win_trades as average_profit,"
                           "   r.gross_loss/r.loss_trades as average_loss,"
                           "   (case when r.gross_loss!=0.0 then r.gross_profit/(-r.gr
                           "FROM "
                           "   ("
                           "   SELECT MAGIC,"
                           "   sum(case when entry =1 then 1 else 0 end) as trades,"
                           "   sum(case when profit > 0 then profit else 0 end) as gro
                           "   sum(case when profit < 0 then profit else 0 end) as gro
                           "   sum(swap) as total_swap,"
                           "   sum(commission) as total_commission,"
                           "   sum(profit) as total_profit,"
                           "   sum(profit+swap+commission) as net_profit,"
                           "   sum(case when profit > 0 then 1 else 0 end) as win_trad
                           "   sum(case when profit < 0 then 1 else 0 end) as loss_tra
                           "   FROM DEALS "
```

© 2000-2025, MetaQuotes Ltd.

```
                              "   WHERE SYMBOL <> '' and SYMBOL is not NULL "
                              "   GROUP BY MAGIC"
                              "   ) as r");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
   Magic_Stats EA_stats[], magic_stats;
   ArrayResize(EA_stats, ExtDealsTotal);
   i=0;
//--- print
   for(; DatabaseReadBind(request, magic_stats) ; i++)
     {
      EA_stats[i].magic=magic_stats.magic;
      EA_stats[i].trades=magic_stats.trades;
      EA_stats[i].gross_profit=magic_stats.gross_profit;
      EA_stats[i].gross_loss=magic_stats.gross_loss;
      EA_stats[i].total_commission=magic_stats.total_commission;
      EA_stats[i].total_swap=magic_stats.total_swap;
      EA_stats[i].total_profit=magic_stats.total_profit;
      EA_stats[i].net_profit=magic_stats.net_profit;
      EA_stats[i].win_trades=magic_stats.win_trades;
      EA_stats[i].loss_trades=magic_stats.loss_trades;
      EA_stats[i].expected_payoff=magic_stats.expected_payoff;
      EA_stats[i].win_percent=magic_stats.win_percent;
      EA_stats[i].loss_percent=magic_stats.loss_percent;
      EA_stats[i].average_profit=magic_stats.average_profit;
      EA_stats[i].average_loss=magic_stats.average_loss;
      EA_stats[i].profit_factor=magic_stats.profit_factor;
     }
   ArrayResize(EA_stats, i);
   Print("Trade statistics by Magic Number");
   ArrayPrint(EA_stats);
   Print("");
//--- delete the request
   DatabaseFinalize(request);

//--- make sure that hedging system for open position management is used on the accou
   if((ENUM_ACCOUNT_MARGIN_MODE)AccountInfoInteger(ACCOUNT_MARGIN_MODE)!=ACCOUNT_MARGI
     {
      //--- deals cannot be transformed to trades using a simple method through transa
      DatabaseClose(db);
      return;
     }

//--- now create the TRADES table based on the DEALS table
   if(!CreateTableTrades(db))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       DatabaseClose(db);
       return;
      }
//--- fill in the TRADES table using an SQL query based on DEALS table data
   if(DatabaseTableExists(db, "DEALS"))
      //--- populate the TRADES table
      if(!DatabaseExecute(db, "INSERT INTO TRADES(TIME_IN,HOUR_IN,TICKET,TYPE,VOLUME,S
                              "SELECT "
                              "   d1.time as time_in,"
                              "   d1.hour as hour_in,"
                              "   d1.position_id as ticket,"
                              "   d1.type as type,"
                              "   d1.volume as volume,"
                              "   d1.symbol as symbol,"
                              "   d1.price as price_in,"
                              "   d2.time as time_out,"
                              "   d2.price as price_out,"
                              "   d1.commission+d2.commission as commission,"
                              "   d2.swap as swap,"
                              "   d2.profit as profit "
                              "FROM DEALS d1 "
                              "INNER JOIN DEALS d2 ON d1.position_id=d2.position_id "
                              "WHERE d1.entry=0 AND d2.entry=1     "))
        {
         Print("DB: fillng the table TRADES failed with code ", GetLastError());
         return;
        }

//--- get trading statistics by market entry hours
   request=DatabasePrepare(db, "SELECT r.*,"
                               "   (case when r.trades != 0 then (r.gross_profit+r.gross_l
                               "   (case when r.trades != 0 then r.win_trades*100.0/r.trad
                               "   (case when r.trades != 0 then r.loss_trades*100.0/r.tra
                               "   r.gross_profit/r.win_trades as average_profit,"
                               "   r.gross_loss/r.loss_trades as average_loss,"
                               "   (case when r.gross_loss!=0.0 then r.gross_profit/(-r.gr
                              "FROM "
                              "   ("
                              "   SELECT HOUR_IN,"
                              "   count() as trades,"
                              "   sum(volume) as volume,"
                              "   sum(case when profit > 0 then profit else 0 end) as gro
                              "   sum(case when profit < 0 then profit else 0 end) as gro
                              "   sum(profit) as net_profit,"
                              "   sum(case when profit > 0 then 1 else 0 end) as win_trad
                              "   sum(case when profit < 0 then 1 else 0 end) as loss_tra
                              "   FROM TRADES "
                              "   WHERE SYMBOL <> '' and SYMBOL is not NULL "
```

© 2000-2025, MetaQuotes Ltd.

```
                              "   GROUP BY HOUR_IN"
                              "   ) as r");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
   Hour_Stats hours_stats[], h_stats;
   ArrayResize(hours_stats, ExtDealsTotal);
   i=0;
//--- print
   for(; DatabaseReadBind(request, h_stats) ; i++)
     {
      hours_stats[i].hour_in=h_stats.hour_in;
      hours_stats[i].trades=h_stats.trades;
      hours_stats[i].volume=h_stats.volume;
      hours_stats[i].gross_profit=h_stats.gross_profit;
      hours_stats[i].gross_loss=h_stats.gross_loss;
      hours_stats[i].net_profit=h_stats.net_profit;
      hours_stats[i].win_trades=h_stats.win_trades;
      hours_stats[i].loss_trades=h_stats.loss_trades;
      hours_stats[i].expected_payoff=h_stats.expected_payoff;
      hours_stats[i].win_percent=h_stats.win_percent;
      hours_stats[i].loss_percent=h_stats.loss_percent;
      hours_stats[i].average_profit=h_stats.average_profit;
      hours_stats[i].average_loss=h_stats.average_loss;
      hours_stats[i].profit_factor=h_stats.profit_factor;
     }
   ArrayResize(hours_stats, i);
   Print("Trade statistics by entry hour");
   ArrayPrint(hours_stats);
   Print("");
//--- delete the request
   DatabaseFinalize(request);

//--- close database
   DatabaseClose(db);
   return;
  }
/*
Deals in the trading history: 2771
Trade statistics by Symbol
      [name] [trades] [gross_profit] [gross_loss] [total_commission] [total_swap] [tot
[0] "AUDUSD"      112      503.20000   -568.00000           -8.83000    -24.64000
[1] "EURCHF"      125      607.71000   -956.85000          -11.77000    -45.02000
[2] "EURJPY"      127     1078.49000  -1057.83000          -10.61000    -45.76000
[3] "EURUSD"      233     1685.60000  -1386.80000          -41.00000    -83.76000
[4] "GBPCHF"      125     1881.37000  -1424.72000          -22.60000    -51.56000
```

© 2000-2025, MetaQuotes Ltd.

```
[5] "GBPJPY"      127     1943.43000  -1776.67000          -18.84000    -52.46000
[6] "GBPUSD"      121     1668.50000  -1438.20000           -7.96000    -49.93000
[7] "USDCAD"       99      405.28000   -475.47000           -8.68000    -31.68000
[8] "USDCHF"      206     1588.32000  -1241.83000          -17.98000    -65.92000
[9] "USDJPY"      107      464.73000   -730.64000          -35.12000    -34.24000

Trade statistics by Magic Number
    [magic] [trades] [gross_profit] [gross_loss] [total_commission] [total_swap] [tota
[0]     100      242     2584.80000  -2110.00000          -33.36000    -93.53000
[1]     200      254     3021.92000  -2834.50000          -29.45000    -98.22000
[2]     300      250     2489.08000  -2381.57000          -34.37000    -96.58000
[3]     400      224     1272.50000  -1283.00000          -24.43000    -64.80000
[4]     500      198     1141.23000  -1051.91000          -27.66000    -63.36000
[5]     600      214     1317.10000  -1396.03000          -34.12000    -68.48000

Trade statistics by entry hour
     [hour_in] [trades] [volume] [gross_profit] [gross_loss] [net_profit] [win_trades]
[ 0]         0       50  5.00000      336.51000   -747.47000   -410.96000           21
[ 1]         1       20  2.00000      102.56000    -57.20000     45.36000           12
[ 2]         2        6  0.60000       38.55000    -14.60000     23.95000            5
[ 3]         3       38  3.80000      173.84000   -200.15000    -26.31000           22
[ 4]         4       60  6.00000      361.44000   -389.40000    -27.96000           27
[ 5]         5       32  3.20000      157.43000   -179.89000    -22.46000           20
[ 6]         6       18  1.80000       95.59000   -162.33000    -66.74000           11
[ 7]         7       14  1.40000       38.48000   -134.30000    -95.82000            9
[ 8]         8       42  4.20000      368.48000   -322.30000     46.18000           24
[ 9]         9      118 11.80000     1121.62000   -875.21000    246.41000           72
[10]        10      206 20.60000     2280.59000  -2021.80000    258.79000          115
[11]        11      138 13.80000     1377.02000   -994.18000    382.84000           84
[12]        12      152 15.20000     1247.56000  -1463.80000   -216.24000           84
[13]        13       64  6.40000      778.27000   -516.22000    262.05000           36
[14]        14       62  6.20000      536.93000   -427.47000    109.46000           38
[15]        15       50  5.00000      699.92000   -413.00000    286.92000           28
[16]        16       88  8.80000      778.55000   -514.00000    264.55000           51
[17]        17       76  7.60000      533.92000  -1019.46000   -485.54000           44
[18]        18       52  5.20000      237.17000   -246.78000     -9.61000           24
[19]        19       52  5.20000      407.67000   -150.36000    257.31000           30
[20]        20       18  1.80000       65.92000    -89.09000    -23.17000            9
[21]        21       10  1.00000       41.86000    -32.38000      9.48000            7
[22]        22       14  1.40000       45.55000    -83.72000    -38.17000            6
[23]        23        2  0.20000        1.20000     -1.90000     -0.70000            1
*/

//+------------------------------------------------------------------+
//| Creates the DEALS table                                          |
//+------------------------------------------------------------------+
bool CreateTableDeals(int database)
  {
//--- if the DEALS table already exists, delete it
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!DeleteTable(database, "DEALS"))
     {
      return(false);
     }
//--- check if the table exists
   if(!DatabaseTableExists(database, "DEALS"))
      //--- create the table
      if(!DatabaseExecute(database, "CREATE TABLE DEALS("
                          "ID          INT KEY NOT NULL,"
                          "ORDER_ID    INT     NOT NULL,"
                          "POSITION_ID INT     NOT NULL,"
                          "TIME        INT     NOT NULL,"
                          "TYPE        INT     NOT NULL,"
                          "ENTRY       INT     NOT NULL,"
                          "SYMBOL      CHAR(10),"
                          "VOLUME      REAL,"
                          "PRICE       REAL,"
                          "PROFIT      REAL,"
                          "SWAP        REAL,"
                          "COMMISSION  REAL,"
                          "MAGIC       INT,"
                          "HOUR        INT,"
                          "REASON      INT);"))
        {
         Print("DB: create the DEALS table  failed with code ", GetLastError());
         return(false);
        }
//---  request the entire trading history
   datetime from_date=0;
   datetime to_date=TimeCurrent();
//--- request the history of deals in the specified interval
   HistorySelect(from_date, to_date);
   ExtDealsTotal=HistoryDealsTotal();
//--- add deals to the table
   if(!InsertDeals(database))
      return(false);
//--- the table has been successfully created
   return(true);
  }
//+------------------------------------------------------------------+
//| Deletes a table with the specified name from the database        |
//+------------------------------------------------------------------+
bool DeleteTable(int database, string table_name)
  {
   if(!DatabaseExecute(database, "DROP TABLE IF EXISTS "+table_name))
     {
      Print("Failed to drop the DEALS table with code ", GetLastError());
      return(false);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- the table has been successfully deleted
   return(true);
  }
//+------------------------------------------------------------------+
//| Adds deals to the database table                                 |
//+------------------------------------------------------------------+
bool InsertDeals(int database)
  {
//--- Auxiliary variables
   ulong    deal_ticket;          // deal ticket
   long     order_ticket;         // the ticket of the order by which the deal was exec
   long     position_ticket;      // ID of the position to which the deal belongs
   datetime time;                 // deal execution time
   long     type ;                // deal type
   long     entry ;               // deal direction
   string   symbol;               // the symbol fro which the deal was executed
   double   volume;               // operation volume
   double   price;                // price
   double   profit;               // financial result
   double   swap;                 // swap
   double   commission;           // commission
   long     magic;                // Magic number (Expert Advisor ID)
   long     reason;               // deal execution reason or source
   char     hour;                 // deal execution hour
   MqlDateTime time_strusture;
//--- go through all deals and add them to the database
   bool failed=false;
   int deals=HistoryDealsTotal();
// --- lock the database before executing transactions
   DatabaseTransactionBegin(database);
   for(int i=0; i<deals; i++)
     {
      deal_ticket=    HistoryDealGetTicket(i);
      order_ticket=   HistoryDealGetInteger(deal_ticket, DEAL_ORDER);
      position_ticket=HistoryDealGetInteger(deal_ticket, DEAL_POSITION_ID);
      time= (datetime)HistoryDealGetInteger(deal_ticket, DEAL_TIME);
      type=           HistoryDealGetInteger(deal_ticket, DEAL_TYPE);
      entry=          HistoryDealGetInteger(deal_ticket, DEAL_ENTRY);
      symbol=         HistoryDealGetString(deal_ticket, DEAL_SYMBOL);
      volume=         HistoryDealGetDouble(deal_ticket, DEAL_VOLUME);
      price=          HistoryDealGetDouble(deal_ticket, DEAL_PRICE);
      profit=         HistoryDealGetDouble(deal_ticket, DEAL_PROFIT);
      swap=           HistoryDealGetDouble(deal_ticket, DEAL_SWAP);
      commission=     HistoryDealGetDouble(deal_ticket, DEAL_COMMISSION);
      magic=          HistoryDealGetInteger(deal_ticket, DEAL_MAGIC);
      reason=         HistoryDealGetInteger(deal_ticket, DEAL_REASON);
      TimeToStruct(time, time_strusture);
      hour= (char)time_strusture.hour;
      //--- add each deal to the table using the following request
```

© 2000-2025, MetaQuotes Ltd.

```
      string request_text=StringFormat("INSERT INTO DEALS (ID,ORDER_ID,POSITION_ID,TIM
                                       "VALUES (%d, %d, %d, %d, %d, %d, '%s', %G, %G,
                                       deal_ticket, order_ticket, position_ticket, tim
      if(!DatabaseExecute(database, request_text))
        {
         PrintFormat("%s: failed to insert deal #%d with code %d", __FUNCTION__, deal_
         PrintFormat("i=%d: deal #%d  %s", i, deal_ticket, symbol);
         failed=true;
         break;
        }
      }
//--- check for transaction execution errors
   if(failed)
     {
      //--- roll back all transactions and unlock the database
      DatabaseTransactionRollback(database);
      PrintFormat("%s: DatabaseExecute() failed with code ", __FUNCTION__, GetLastErro
      return(false);
     }
//--- all transactions have been performed successfully - record changes and unlock th
   DatabaseTransactionCommit(database);
   return(true);
  }
//+------------------------------------------------------------------+
//| Creates the TRADES table                                         |
//+------------------------------------------------------------------+
bool CreateTableTrades(int database)
  {
//--- if the TRADES table already exists, delete it
   if(!DeleteTable(database, "TRADES"))
      return(false);
//--- check if the table exists
   if(!DatabaseTableExists(database, "TRADES"))
      //--- create the table
      if(!DatabaseExecute(database, "CREATE TABLE TRADES("
                          "TIME_IN     INT     NOT NULL,"
                          "HOUR_IN     INT     NOT NULL,"
                          "TICKET      INT     NOT NULL,"
                          "TYPE        INT     NOT NULL,"
                          "VOLUME      REAL,"
                          "SYMBOL      CHAR(10),"
                          "PRICE_IN    REAL,"
                          "TIME_OUT    INT     NOT NULL,"
                          "PRICE_OUT   REAL,"
                          "COMMISSION  REAL,"
                          "SWAP        REAL,"
                          "PROFIT      REAL);"))
        {
         Print("DB: create the TRADES table failed with code ", GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
         return(false);
        }
//--- the table has been successfully created
   return(true);
  }
//+------------------------------------------------------------------+
```

Siehe auch

DatabasePrepare, DatabaseFinalize

© 2000-2025, MetaQuotes Ltd.

DatabasePrepare

Erstellt das Handle für die Anfragen, die durch DatabaseRead() ausgeführt werden kann.

```
int  DatabasePrepare(
   int     database,      // Handle der Datenbank, erhalten von DatabaseOpen
   string  sql,           // SQL-Anfrage
           ...            // Parameter der Anfrage
   );
```

Parameter

database

[in] Handle der Datenbank, erhalten von DatabaseOpen().

sql

[in] SQL-Anfrage, die u.U. automatisch ersetzte Parameternamen ?1,?2,... hat

...

[in] Automatisch ersetzte Parameter der Anfrage.

Rückgabewert

Bei Erfolg liefert die Funktion das Handle für die SQL-Anfrage. Andernfalls wird INVALID_HANDLE zurückgegeben. Um die Fehlernummer abzurufen, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - Der Pfad zur Datenbankdatei ist leer oder es wurde eine inkompatible Kombination der Flags übergeben;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler bei der Konvertierung der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_TOO_MANY_OBJECTS (5122) - es wurde die Maximalzahl an Datenbankobjekten überschritten;
- ERR_DATABASE_PREPARE (5125) - Fehler bei der Anfrageerstellung.

Hinweis

Die Funktion DatabasePrepare() stellt keine Anfrage an eine Datenbank. Ihr Zweck ist es, die Anfrageparameter zu überprüfen und das Handle für die Ausführung der SQL-Anfrage basierend auf den Verifikationsergebnissen zurückzugeben. Die Anfrage selbst wird während des ersten Aufrufs DatabaseRead() gestellt.

Example:

```
//--- Structure to store the deal
struct Deal
  {
   ulong             ticket;           // DEAL_TICKET
   long              order_ticket;     // DEAL_ORDER
   long              position_ticket;  // DEAL_POSITION_ID
   datetime          time;             // DEAL_TIME
```

© 2000-2025, MetaQuotes Ltd.

```
   char              type;             // DEAL_TYPE
   char              entry;            // DEAL_ENTRY
   string            symbol;           // DEAL_SYMBOL
   double            volume;           // DEAL_VOLUME
   double            price;            // DEAL_PRICE
   double            profit;           // DEAL_PROFIT
   double            swap;             // DEAL_SWAP
   double            commission;       // DEAL_COMMISSION
   long              magic;            // DEAL_MAGIC
   char              reason;           // DEAL_REASON
  };
//--- Structure to store the trade: the order of members corresponds to the position i
struct Trade
  {
   datetime          time_in;          // entry time
   ulong             ticket;           // position ID
   char              type;             // buy or sell
   double            volume;           // volume
   string            symbol;           // symbol
   double            price_in;         // entry price
   datetime          time_out;         // exit time
   double            price_out;        // exit price
   double            commission;       // entry and exit commission
   double            swap;             // swap
   double            profit;           // profit or loss
  };
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create the file name
   string filename=IntegerToString(AccountInfoInteger(ACCOUNT_LOGIN))+"_trades.sqlite"
//--- open/create the database in the common terminal folder
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE | DATA
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " open failed with code ", GetLastError());
      return;
     }
//--- create the DEALS table
   if(!CreateTableDeals(db))
     {
      DatabaseClose(db);
      return;
     }
//---  request the entire trading history
   datetime from_date=0;
   datetime to_date=TimeCurrent();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- request the history of deals in the specified interval
   HistorySelect(from_date, to_date);
   int deals_total=HistoryDealsTotal();
   PrintFormat("Deals in the trading history: %d ", deals_total);
//--- add deals to the table
   if(!InsertDeals(db))
      return;
//--- show the first 10 deals
   Deal deals[], deal;
   ArrayResize(deals, 10);
   int request=DatabasePrepare(db, "SELECT * FROM DEALS");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
   int i;
   for(i=0; DatabaseReadBind(request, deal); i++)
     {
      if(i>=10)
         break;
      deals[i].ticket=deal.ticket;
      deals[i].order_ticket=deal.order_ticket;
      deals[i].position_ticket=deal.position_ticket;
      deals[i].time=deal.time;
      deals[i].type=deal.type;
      deals[i].entry=deal.entry;
      deals[i].symbol=deal.symbol;
      deals[i].volume=deal.volume;
      deals[i].price=deal.price;
      deals[i].profit=deal.profit;
      deals[i].swap=deal.swap;
      deals[i].commission=deal.commission;
      deals[i].magic=deal.magic;
      deals[i].reason=deal.reason;
     }
//--- print the deals
   if(i>0)
     {
      ArrayResize(deals, i);
      PrintFormat("The first %d deals:", i);
      ArrayPrint(deals);
     }

//--- delete request after use
   DatabaseFinalize(request);

//--- make sure that hedging system for open position management is used on the accou
```

© 2000-2025, MetaQuotes Ltd.

```
   if((ENUM_ACCOUNT_MARGIN_MODE)AccountInfoInteger(ACCOUNT_MARGIN_MODE)!=ACCOUNT_MARGI
     {
      //--- deals cannot be transformed to trades using a simple method through transa
      DatabaseClose(db);
      return;
     }

//--- now create the TRADES table based on the DEALS table
   if(!CreateTableTrades(db))
     {
      DatabaseClose(db);
      return;
     }
//--- fill in the TRADES table using an SQL query based on DEALS table data
   ulong start=GetMicrosecondCount();
   if(DatabaseTableExists(db, "DEALS"))
      //--- populate the TRADES table
      if(!DatabaseExecute(db, "INSERT INTO TRADES(TIME_IN,TICKET,TYPE,VOLUME,SYMBOL,PR
                          "SELECT "
                          "   d1.time as time_in,"
                          "   d1.position_id as ticket,"
                          "   d1.type as type,"
                          "   d1.volume as volume,"
                          "   d1.symbol as symbol,"
                          "   d1.price as price_in,"
                          "   d2.time as time_out,"
                          "   d2.price as price_out,"
                          "   d1.commission+d2.commission as commission,"
                          "   d2.swap as swap,"
                          "   d2.profit as profit "
                          "FROM DEALS d1 "
                          "INNER JOIN DEALS d2 ON d1.position_id=d2.position_id "
                          "WHERE d1.entry=0 AND d2.entry=1     "))
        {
         Print("DB: fillng the TRADES table failed with code ", GetLastError());
         return;
        }
   ulong transaction_time=GetMicrosecondCount()-start;

//--- show the first 10 deals
   Trade trades[], trade;
   ArrayResize(trades, 10);
   request=DatabasePrepare(db, "SELECT * FROM TRADES");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   for(i=0; DatabaseReadBind(request, trade); i++)
     {
      if(i>=10)
         break;
      trades[i].time_in=trade.time_in;
      trades[i].ticket=trade.ticket;
      trades[i].type=trade.type;
      trades[i].volume=trade.volume;
      trades[i].symbol=trade.symbol;
      trades[i].price_in=trade.price_in;
      trades[i].time_out=trade.time_out;
      trades[i].price_out=trade.price_out;
      trades[i].commission=trade.commission;
      trades[i].swap=trade.swap;
      trades[i].profit=trade.profit;
     }
//--- print trades
   if(i>0)
     {
      ArrayResize(trades, i);
      PrintFormat("\r\nThe first %d trades:", i);
      ArrayPrint(trades);
      PrintFormat("Filling the TRADES table took %.2f milliseconds",double(transactior
     }
//--- delete request after use
   DatabaseFinalize(request);

//--- close the database
   DatabaseClose(db);
  }
/*
Results:
   Deals in the trading history: 2741
   The first 10 deals:
       [ticket] [order_ticket] [position_ticket]              [time] [type] [entry] [s
   [0] 34429573              0                 0 2019.09.05 22:39:59      2        0 ""
   [1] 34432127       51447238          51447238 2019.09.06 06:00:03      0        0 "U
   [2] 34432128       51447239          51447239 2019.09.06 06:00:03      1        0 "U
   [3] 34432450       51447565          51447565 2019.09.06 07:00:00      0        0 "E
   [4] 34432456       51447571          51447571 2019.09.06 07:00:00      1        0 "A
   [5] 34432879       51448053          51448053 2019.09.06 08:00:00      1        0 "U
   [6] 34432888       51448064          51448064 2019.09.06 08:00:00      0        0 "U
   [7] 34435147       51450470          51450470 2019.09.06 10:30:00      1        0 "E
   [8] 34435152       51450476          51450476 2019.09.06 10:30:00      0        0 "C
   [9] 34435154       51450479          51450479 2019.09.06 10:30:00      1        0 "E

   The first 10 trades:
                  [time_in] [ticket] [type] [volume] [symbol] [price_in]           [time
   [0] 2019.09.06 06:00:03 51447238      0  0.10000 "USDCAD"    1.32320 2019.09.06 18:
```

© 2000-2025, MetaQuotes Ltd.

```
   [1] 2019.09.06 06:00:03 51447239      1  0.10000 "USDCHF"    0.98697 2019.09.06 18:
   [2] 2019.09.06 07:00:00 51447565      0  0.10000 "EURUSD"    1.10348 2019.09.09 03:
   [3] 2019.09.06 07:00:00 51447571      1  0.10000 "AUDUSD"    0.68203 2019.09.09 03:
   [4] 2019.09.06 08:00:00 51448053      1  0.10000 "USDCHF"    0.98701 2019.09.06 18:
   [5] 2019.09.06 08:00:00 51448064      0  0.10000 "USDJPY"  106.96200 2019.09.06 18:
   [6] 2019.09.06 10:30:00 51450470      1  0.10000 "EURUSD"    1.10399 2019.09.06 14:
   [7] 2019.09.06 10:30:00 51450476      0  0.10000 "GBPUSD"    1.23038 2019.09.06 14:
   [8] 2019.09.06 10:30:00 51450479      1  0.10000 "EURJPY"  118.12000 2019.09.06 14:
   [9] 2019.09.06 10:30:00 51450480      0  0.10000 "GBPJPY"  131.65300 2019.09.06 14:
   Filling the TRADES table took 12.51 milliseconds
*/
//+------------------------------------------------------------------+
//| Creates the DEALS table                                          |
//+------------------------------------------------------------------+
bool CreateTableDeals(int database)
  {
//--- if the DEALS table already exists, delete it
   if(!DeleteTable(database, "DEALS"))
     {
      return(false);
     }
//--- check if the table exists
   if(!DatabaseTableExists(database, "DEALS"))
      //--- create the table
      if(!DatabaseExecute(database, "CREATE TABLE DEALS("
                          "ID          INT KEY NOT NULL,"
                          "ORDER_ID    INT     NOT NULL,"
                          "POSITION_ID INT     NOT NULL,"
                          "TIME        INT     NOT NULL,"
                          "TYPE        INT     NOT NULL,"
                          "ENTRY       INT     NOT NULL,"
                          "SYMBOL      CHAR(10),"
                          "VOLUME      REAL,"
                          "PRICE       REAL,"
                          "PROFIT      REAL,"
                          "SWAP        REAL,"
                          "COMMISSION  REAL,"
                          "MAGIC       INT,"
                          "REASON      INT );"))
        {
         Print("DB: create the DEALS table failed with code ", GetLastError());
         return(false);
        }
//--- the table has been successfully created
   return(true);
  }
//+------------------------------------------------------------------+
//| Deletes a table with the specified name from the database        |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
bool DeleteTable(int database, string table_name)
  {
   if(!DatabaseExecute(database, "DROP TABLE IF EXISTS "+table_name))
     {
      Print("Failed to drop the DEALS table  with code ", GetLastError());
      return(false);
     }
//--- the table has been successfully deleted
   return(true);
  }
//+------------------------------------------------------------------+
//| Adds deals to the database table                                 |
//+------------------------------------------------------------------+
bool InsertDeals(int database)
  {
//--- Auxiliary variables
   ulong    deal_ticket;          // deal ticket
   long     order_ticket;         // the ticket of the order by which the deal was exec
   long     position_ticket;      // ID of the position to which the deal belongs
   datetime time;                 // deal execution time
   long     type ;                // deal type
   long     entry ;               // deal direction
   string   symbol;               // the symbol fro which the deal was executed
   double   volume;               // operation volume
   double   price;                // price
   double   profit;               // financial result
   double   swap;                 // swap
   double   commission;           // commission
   long     magic;                // Magic number (Expert Advisor ID)
   long     reason;               // deal execution reason or source
//--- go through all deals and add them to the database
   bool failed=false;
   int deals=HistoryDealsTotal();
// --- lock the database before executing transactions
   DatabaseTransactionBegin(database);
   for(int i=0; i<deals; i++)
     {
      deal_ticket=    HistoryDealGetTicket(i);
      order_ticket=   HistoryDealGetInteger(deal_ticket, DEAL_ORDER);
      position_ticket=HistoryDealGetInteger(deal_ticket, DEAL_POSITION_ID);
      time= (datetime)HistoryDealGetInteger(deal_ticket, DEAL_TIME);
      type=           HistoryDealGetInteger(deal_ticket, DEAL_TYPE);
      entry=          HistoryDealGetInteger(deal_ticket, DEAL_ENTRY);
      symbol=         HistoryDealGetString(deal_ticket, DEAL_SYMBOL);
      volume=         HistoryDealGetDouble(deal_ticket, DEAL_VOLUME);
      price=          HistoryDealGetDouble(deal_ticket, DEAL_PRICE);
      profit=         HistoryDealGetDouble(deal_ticket, DEAL_PROFIT);
      swap=           HistoryDealGetDouble(deal_ticket, DEAL_SWAP);
      commission=     HistoryDealGetDouble(deal_ticket, DEAL_COMMISSION);
```

© 2000-2025, MetaQuotes Ltd.

```
      magic=          HistoryDealGetInteger(deal_ticket, DEAL_MAGIC);
      reason=         HistoryDealGetInteger(deal_ticket, DEAL_REASON);
      //--- add each deal to the table using the following request
      string request_text=StringFormat("INSERT INTO DEALS (ID,ORDER_ID,POSITION_ID,TIN
                                       "VALUES (%d, %d, %d, %d, %d, %d, '%s', %G, %G,
                                       deal_ticket, order_ticket, position_ticket, tir
      if(!DatabaseExecute(database, request_text))
        {
         PrintFormat("%s: failed to insert deal #%d with code %d", __FUNCTION__, deal_
         PrintFormat("i=%d: deal #%d  %s", i, deal_ticket, symbol);
         failed=true;
         break;
        }
     }
//--- check for transaction execution errors
   if(failed)
     {
      //--- roll back all transactions and unlock the database
      DatabaseTransactionRollback(database);
      PrintFormat("%s: DatabaseExecute() failed with code %d", __FUNCTION__, GetLastEr
      return(false);
     }
//--- all transactions have been performed successfully - record changes and unlock th
   DatabaseTransactionCommit(database);
   return(true);
  }
//+------------------------------------------------------------------+
//| Creates the TRADES table                                         |
//+------------------------------------------------------------------+
bool CreateTableTrades(int database)
  {
//--- if the TRADES table already exists, delete it
   if(!DeleteTable(database, "TRADES"))
      return(false);
//--- check if the table exists
   if(!DatabaseTableExists(database, "TRADES"))
      //--- create the table
      if(!DatabaseExecute(database, "CREATE TABLE TRADES("
                          "TIME_IN     INT     NOT NULL,"
                          "TICKET      INT     NOT NULL,"
                          "TYPE        INT     NOT NULL,"
                          "VOLUME      REAL,"
                          "SYMBOL      CHAR(10),"
                          "PRICE_IN    REAL,"
                          "TIME_OUT    INT     NOT NULL,"
                          "PRICE_OUT   REAL,"
                          "COMMISSION  REAL,"
                          "SWAP        REAL,"
                          "PROFIT      REAL);"))
```

© 2000-2025, MetaQuotes Ltd.

```
         {
          Print("DB: create the TRADES table failed with code ", GetLastError());
          return(false);
         }
//--- the table has been successfully created
   return(true);
  }
//+------------------------------------------------------------------+
```

Siehe auch

DatabaseExecute, DatabaseFinalize

© 2000-2025, MetaQuotes Ltd.

DatabaseReset

Rücksetzen einer Anfrage, wie nach dem Aufruf von DatabasePrepare().

```
int  DatabaseReset(
   int  request      // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   );
```

Parameter

request

[in] Das Handle der Anfrage, erhalten von DatabasePrepare().

Rückgabewert

Rückgabe von true bei Erfolg, sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Handle der Datenbank;
- Die Fehlernummern von SQLite beginnen mit ERR_DATABASE_ERROR(5601).

Hinweis

Die Funktion DatabaseReset() ist für die mehrfache Ausführung einer Anfrage mit unterschiedlichen Parameterwerten vorgesehen. Wenn beispielsweise mit dem Befehl INSERT der Tabelle Daten in großen Mengen hinzugefügt werden, sollte für jeden Eintrag ein nutzerdefinierter Satz von Feldwerten gebildet werden.

Im Gegensatz zu DatabasePrepare() wird von DatabaseReset() der String mit SQL-Befehlen nicht zu einer neuen Anfrage kompiliert, daher wird DatabaseReset() viel schneller ausgeführt als DatabasePrepare().

DatabaseReset() wird zusammen mit den Funktionen DatabaseBind() und/oder DatabaseBindArray() verwendet, wenn die Werte der Anfrageparameter nach der Ausführung von DatabaseRead() geändert werden sollen. Das bedeutet, dass vor dem Setzen neuer Werte der Anfrageparameter (vor dem Block von DatabaseBind/DatabaseBindArray-Aufrufen) DatabaseReset() aufgerufen werden sollte, um sie zurückzusetzen. Die parametrisierte Anfragen selbst sollte mit DatabasePrepare() erstellt werden.

Wie DatabasePrepare() macht DatabaseReset() keine Datenbankanfrage. Eine direkte Anfrage wird ausgeführt, wenn DatabaseRead() oder DatabaseReadBind() aufgerufen wird.

Der Aufruf DatabaseReset() führt nicht zum Zurücksetzen von Parameterwerten in der Anfrage, wenn diese durch den Aufruf DatabaseBind()/DatabaseBindArray() gesetzt wurden, d.h. die Parameter behalten ihre Werte bei. Somit kann nur der Wert eines einzelnen Parameters geändert werden. Es ist nicht notwendig, nach dem Aufruf von DatabaseReset() alle Parameter der Anfrage neu zu setzen.

Das Handle einer mit DatabaseFinalize() entfernten Anfrage kann nicht an DatabaseReset() übergeben werden. Dies führt zu einem Fehler.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
//--- Erstellen oder Öffnen einer Datenbank
   string filename="symbols.sqlite";
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE);
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " open failed with code ", GetLastError());
      return;
     }
   else
      Print("Database: ", filename, " opened successfully");
//--- Wenn die Tabelle SYMBOLS existiert, wird sie gelöscht
   if(DatabaseTableExists(db, "SYMBOLS"))
     {
      //--- löschen der Tabelle
      if(!DatabaseExecute(db, "DROP TABLE SYMBOLS"))
        {
         Print("Failed to drop table SYMBOLS with code ", GetLastError());
         DatabaseClose(db);
         return;
        }
     }
//--- Erstellen der Tabelle SYMBOLS
   if(!DatabaseExecute(db, "CREATE TABLE SYMBOLS("
                       "NAME           TEXT    NOT NULL,"
                       "DESCRIPTION    TEXT            ,"
                       "PATH           TEXT            ,"
                       "SPREAD         INT             ,"
                       "POINT          REAL    NOT NULL,"
                       "DIGITS         INT     NOT NULL,"
                       "JSON           BLOB );"))
     {
      Print("DB: ", filename, " create table failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
//--- Darstellung der Liste aller Felder der Tabellen der SYMBOLS
   if(DatabasePrint(db, "PRAGMA TABLE_INFO(SYMBOLS)", 0)<0)
     {
      PrintFormat("DatabasePrint(\"PRAGMA TABLE_INFO(SYMBOLS)\") failed, error code=%d
      DatabaseClose(db);
      return;
     }

//--- Erstellen einer parametrisierten Anfrage, um Ticks der Tabelle SYMBOLS hinzuzufü
   string sql="INSERT INTO SYMBOLS (NAME,DESCRIPTION,PATH,SPREAD,POINT,DIGITS,JSON)"
              " VALUES (?1,?2,?3,?4,?5,?6,?7);"; // request parameters
   int request=DatabasePrepare(db, sql);
```

© 2000-2025, MetaQuotes Ltd.

```
   if(request==INVALID_HANDLE)
     {
      PrintFormat("DatabasePrepare() failed with code=%d", GetLastError());
      Print("SQL request: ", sql);
      DatabaseClose(db);
      return;
     }

//--- Schleife über alle Symbole, um sie der Tabelle SYMBOLS hinzuzufügen
   int symbols=SymbolsTotal(false);
   bool request_error=false;
   DatabaseTransactionBegin(db);
   for(int i=0; i<symbols; i++)
     {
      //--- Setzen der Werte der verbliebenen Parameter vor dem Hinzufügen eines Symbo
      ResetLastError();
      string symbol=SymbolName(i, false);
      if(!DatabaseBind(request, 0, symbol))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }
      //--- Falls der vorherige Aufruf von DatabaseBind() erfolgreich war, wird der nä
      if(!DatabaseBind(request, 1, SymbolInfoString(symbol, SYMBOL_DESCRIPTION)))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 2, SymbolInfoString(symbol, SYMBOL_PATH)))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 3, SymbolInfoInteger(symbol, SYMBOL_SPREAD)))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 4, SymbolInfoDouble(symbol, SYMBOL_POINT)))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 5, SymbolInfoInteger(symbol, SYMBOL_DIGITS)))
```

© 2000-2025, MetaQuotes Ltd.

```
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 6, GetSymBolAsJson(symbol)))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, GetLas
         request_error=true;
         break;
        }

      //--- Ausführen der Anfrage, um die Eingabe einzutragen, einschließlich einer Fe
      if(!DatabaseRead(request)&&(GetLastError()!=ERR_DATABASE_NO_MORE_DATA))
        {
         PrintFormat("DatabaseRead() failed with code=%d", GetLastError());
         DatabaseFinalize(request);
         request_error=true;
         break;
        }
      else
         PrintFormat("%d: added %s", i+1, symbol);
      //--- Rücksetzen der Anfrage vor der nächsten Parameteraktualisierung
      if(!DatabaseReset(request))
        {
         PrintFormat("DatabaseReset() failed with code=%d", GetLastError());
         DatabaseFinalize(request);
         request_error=true;
         break;
        }
     } //--- Ende der Schleife über alle Symbole

//--- Transaktionsstatus
   if(request_error)
     {
      PrintFormat("Table SYMBOLS: failed to add %d symbols", symbols);
      DatabaseTransactionRollback(db);
      DatabaseClose(db);
      return;
     }
   else
     {
      DatabaseTransactionCommit(db);
      PrintFormat("Table SYMBOLS: added %d symbols",symbols);
     }

//--- Sichern der Tabelle SYMBOLS in einer csv-Datei
   string csv_filename="symbols.csv";
   if(DatabaseExport(db, "SELECT * FROM SYMBOLS", csv_filename,
```

© 2000-2025, MetaQuotes Ltd.

```
                     DATABASE_EXPORT_HEADER|DATABASE_EXPORT_INDEX|DATABASE_EXPORT_QUOT
      Print("Database: table SYMBOLS saved in ", csv_filename);
   else
      Print("Database: DatabaseExport(\"SELECT * FROM SYMBOLS\") failed with code", Ge

//--- Schließen der Datenbankdatei und Informieren darüber
   DatabaseClose(db);
   PrintFormat("Database: %s created and closed", filename);
  }
//+------------------------------------------------------------------+
//| Rückgabe der Symbolspezifikation als JSON                        |
//+------------------------------------------------------------------+
string GetSymBolAsJson(string symbol)
  {
//--- Einzüge
   string indent1=Indent(1);
   string indent2=Indent(2);
   string indent3=Indent(3);
//---
   int digits=(int)SymbolInfoInteger(symbol, SYMBOL_DIGITS);
   string json="{"+
               "\n"+indent1+"\"ConfigSymbols\":["+
               "\n"+indent2+"{"+
               "\n"+indent3+"\"Symbol\":\""+symbol+"\","+
               "\n"+indent3+"\"Path\":\""+SymbolInfoString(symbol, SYMBOL_PATH)+"\","+
               "\n"+indent3+"\"CurrencyBase\":\""+SymbolInfoString(symbol, SYMBOL_CURR
               "\n"+indent3+"\"CurrencyProfit\":\""+SymbolInfoString(symbol, SYMBOL_CU
               "\n"+indent3+"\"CurrencyMargin\":\""+SymbolInfoString(symbol, SYMBOL_CU
               "\n"+indent3+"\"ColorBackground\":\""+ColorToString((color)SymbolInfoIn
               "\n"+indent3+"\"Digits\":\""+IntegerToString(SymbolInfoInteger(symbol,
               "\n"+indent3+"\"Point\":\""+DoubleToString(SymbolInfoDouble(symbol, SYM
               "\n"+indent3+"\"TickBookDepth\":\""+IntegerToString(SymbolInfoInteger(s
               "\n"+indent3+"\"ChartMode\":\""+IntegerToString(SymbolInfoInteger(symbo
               "\n"+indent3+"\"TradeMode\":\""+IntegerToString(SymbolInfoInteger(symbo
               "\n"+indent3+"\"TradeCalcMode\":\""+IntegerToString(SymbolInfoInteger(s
               "\n"+indent3+"\"OrderMode\":\""+IntegerToString(SymbolInfoInteger(symbo
               "\n"+indent3+"\"CalculationMode\":\""+IntegerToString(SymbolInfoIntege
               "\n"+indent3+"\"ExecutionMode\":\""+IntegerToString(SymbolInfoInteger(s
               "\n"+indent3+"\"ExpirationMode\":\""+IntegerToString(SymbolInfoInteger
               "\n"+indent3+"\"FillFlags\":\""+IntegerToString(SymbolInfoInteger(symbo
               "\n"+indent3+"\"ExpirFlags\":\""+IntegerToString(SymbolInfoInteger(symb
               "\n"+indent3+"\"Spread\":\""+IntegerToString(SymbolInfoInteger(symbol,
               "\n"+indent3+"\"TickValue\":\""+StringFormat("%G", (SymbolInfoDouble(sy
               "\n"+indent3+"\"TickSize\":\""+StringFormat("%G", (SymbolInfoDouble(syr
               "\n"+indent3+"\"ContractSize\":\""+StringFormat("%G",(SymbolInfoDouble
               "\n"+indent3+"\"StopsLevel\":\""+IntegerToString(SymbolInfoInteger(symb
               "\n"+indent3+"\"VolumeMin\":\""+StringFormat("%G",(SymbolInfoDouble(syr
               "\n"+indent3+"\"VolumeMax\":\""+StringFormat("%G",(SymbolInfoDouble(syr
               "\n"+indent3+"\"VolumeStep\":\""+StringFormat("%G",(SymbolInfoDouble(sy
```

© 2000-2025, MetaQuotes Ltd.

```
                "\n"+indent3+"\"VolumeLimit\":\""+StringFormat("%G",(SymbolInfoDouble(s
                "\n"+indent3+"\"SwapMode\":\""+IntegerToString(SymbolInfoInteger(symbol
                "\n"+indent3+"\"SwapLong\":\""+StringFormat("%G",(SymbolInfoDouble(symb
                "\n"+indent3+"\"SwapShort\":\""+StringFormat("%G",(SymbolInfoDouble(syr
                "\n"+indent3+"\"Swap3Day\":\""+IntegerToString(SymbolInfoInteger(symbol
                "\n"+indent2+"}"+
                "\n"+indent1+"]"+
                "\n}";
   return(json);
  }
//+------------------------------------------------------------------+
//| Einzüge durch Leerzeichen                                        |
//+------------------------------------------------------------------+
string Indent(const int number, const int characters=3)
  {
   int length=number*characters;
   string indent=NULL;
   StringInit(indent, length, ' ');
   return indent;
  }
/*
 Ergebnis:
  Database: symbols.sqlite opened successfully
  #| cid name        type notnull dflt_value pk
  -+-----------------------------------------
  1|   0 NAME        TEXT       1             0
  2|   1 DESCRIPTION TEXT       0             0
  3|   2 PATH        TEXT       0             0
  4|   3 SPREAD      INT        0             0
  5|   4 POINT       REAL       1             0
  6|   5 DIGITS      INT        1             0
  7|   6 JSON        BLOB       0             0
  1: added EURUSD
  2: added GBPUSD
  3: added USDCHF
  ...
  82: added USDCOP
  83: added USDARS
  84: added USDCLP
  Table SYMBOLS: added 84 symbols
  Database: table SYMBOLS saved in symbols.csv
  Database: symbols.sqlite created and closed
*/
```

Siehe auch

DatabasePrepare, DatabaseBind, DatabaseBindArray, DatabaseFinalize

© 2000-2025, MetaQuotes Ltd.

DatabaseBind

Setzt einen Parameterwert in einer Anfrage.

```
bool  DatabaseBind(
   int   request,      // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int   index,        // der Parameterindex in der Anfrage
   T     value         // der Wert eines Parameters eines einfachen Typs
   );
```

Parameter

request

[in] Das Handle einer Anfrage, das in DatabasePrepare() erstellt wurde.

index

[in] Der Parameterindex der Anfrage, dessen Werte gesetzt werden soll. Die Nummerierung beginnt mit Null.

value

[in] Der Wert, der zu setzen ist. Mögliche Typen: bool, char, uchar, short, ushart, int, uint, color, datetime, long, ulong, float, double, string.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - Typ wird nicht unterstützt;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Handle der Datenbank;
- ERR_DATABASE_NOT_READY (5128) - Funktion kann zur Zeit nicht für eine Anfrage verwendet werden. Die Anfrage wird ausgeführt oder ist bereits beendet. DatabaseReset() sollte aufgerufen werden.

Hinweis

Die Funktion wird verwendet, wenn eine SQL-Anfrage parametrisierbare Werte der Form "?" oder "?N" enthält, wobei N den Parameterindex (beginnend mit Eins) bedeutet. Gleichzeitig beginnt die Indizierung der Parameter in DatabaseBind() bei Null.

Zum Beispiel:

```
INSERT INTO table VALUES (?,?,?)
SELECT * FROM table WHERE id=?
```

Die Funktion kann unmittelbar nach der Erzeugung einer parametrisierten Anfrage in DatabasePrepare() oder nach dem Zurücksetzen der Anforderung mit DatabaseReset() aufgerufen werden.

Verwenden Sie diese Funktion zusammen mit DatabaseReset(), um die Anfrage so oft wie nötig mit verschiedenen Parameterwerten auszuführen.

Die Funktion ist so konzipiert, dass sie mit den einfachen Typen der Parameter arbeitet. Wenn ein Parameter gegen ein Array geprüft werden soll, verwenden Sie die Funktion DatabaseBindArray().

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
void OnStart()
  {
   MqlTick ticks[];
//--- sichern der Startzeit vor dem Erhalt der Ticks
   uint start=GetTickCount();
//--- Anfordern der Tick-Historie pro Tag
   ulong to=TimeCurrent()*1000;
   ulong from=to-PeriodSeconds(PERIOD_D1)*1000;
   if(CopyTicksRange(_Symbol, ticks, COPY_TICKS_ALL, from, to)==-1)
     {
      PrintFormat("%s: CopyTicksRange(%s - %s) failed, error=%d",
                  _Symbol, TimeToString(datetime(from/1000)), TimeToString(datetime(to
      return;
     }
   else
     {
      //--- wie viele Ticks wurden erhalten und wie lange dauerte es, sie zu bekommen
      PrintFormat("%s: CopyTicksRange received %d ticks in %d ms (from %s to %s)",
                  _Symbol, ArraySize(ticks), GetTickCount()-start,
                  TimeToString(datetime(from/1000)), TimeToString(datetime(to/1000)));
     }

//--- Setzen des Dateinamen zum Speichern der Datenbank
   string filename=_Symbol+" "+TimeToString(datetime(from/1000))+" - "+TimeToString(da
   StringReplace(filename, ":", "."); // ":" character is not allowed in file names
//--- Erstellen/Öffnen einer Datenbank im Verzeichnis Common des Terminals
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE | DATA
   if(db==INVALID_HANDLE)
     {
      Print("Database: ", filename, " open failed with code ", GetLastError());
      return;
     }
   else
      Print("Database: ", filename, " opened successfully");

//--- Erstellen der Tabelle der Ticks
   if(!DatabaseExecute(db, "CREATE TABLE TICKS("
                       "SYMBOL             CHAR(10),"
                       "TIME               INT NOT NULL,"
                       "BID                REAL,"
                       "ASK                REAL,"
                       "LAST               REAL,"
                       "VOLUME             INT,"
                       "TIME_MSC           INT,"
```

© 2000-2025, MetaQuotes Ltd.

```
                        "VOLUME_REAL        REAL);"))
     {
      Print("DB: ", filename, " create table TICKS failed with code ", GetLastError())
      DatabaseClose(db);
      return;
     }
//--- Darstellung der Liste aller Felder der Tabellen der Ticks
   if(DatabasePrint(db, "PRAGMA TABLE_INFO(TICKS)", 0)<0)
     {
      PrintFormat("DatabasePrint(\"PRAGMA TABLE_INFO(TICKS)\") failed, error code=%d a
      DatabaseClose(db);
      return;
     }
//--- Erstellen einer parametrisierten Anfrage, um Ticks der Tabelle der Ticks hinzuzu
   string sql="INSERT INTO TICKS (SYMBOL,TIME,BID,ASK,LAST,VOLUME,TIME_MSC,VOLUME_REAL
              " VALUES (?1,?2,?3,?4,?5,?6,?7,?8)"; // Parameter der Anfrage
   int request=DatabasePrepare(db, sql);
   if(request==INVALID_HANDLE)
     {
      PrintFormat("DatabasePrepare() failed with code=%d", GetLastError());
      Print("SQL request: ", sql);
      DatabaseClose(db);
      return;
     }
//--- Setzen des Wertes des ersten Parameters der Anfrage
   DatabaseBind(request, 0, _Symbol);
//--- sichern der Startzeit vor dem Hinzufügen der Ticks zu Tabelle der Ticks
   start=GetTickCount();
   DatabaseTransactionBegin(db);
   int total=ArraySize(ticks);
   bool request_error=false;
   for(int i=0; i<total; i++)
     {
      //--- Setzen der Werte der verbliebenen Parameter vor dem Hinzufügen der Eingabe
      ResetLastError();
      if(!DatabaseBind(request, 1, ticks[i].time))
        {
         PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
         PrintFormat("Tick #%d line=%d", i+1, __LINE__);
         request_error=true;
         break;
        }
      //--- Falls der vorherige Aufruf von DatabaseBind() erfolgreich war, wird der nä
      if(!request_error && !DatabaseBind(request, 2, ticks[i].bid))
        {
         PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
         PrintFormat("Tick #%d line=%d", i+1, __LINE__);
         request_error=true;
         break;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   if(!request_error && !DatabaseBind(request, 3, ticks[i].ask))
      {
       PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
       PrintFormat("Tick #%d line=%d", i+1, __LINE__);
       request_error=true;
       break;
      }
   if(!request_error && !DatabaseBind(request, 4, ticks[i].last))
      {
       PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
       PrintFormat("Tick #%d line=%d", i+1, __LINE__);
       request_error=true;
       break;
      }
   if(!request_error && !DatabaseBind(request, 5, ticks[i].volume))
      {
       PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
       PrintFormat("Tick #%d line=%d", i+1, __LINE__);
       request_error=true;
       break;
      }
   if(!request_error && !DatabaseBind(request, 6, ticks[i].time_msc))
      {
       PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
       PrintFormat("Tick #%d line=%d", i+1, __LINE__);
       request_error=true;
       break;
      }
   if(!request_error && !DatabaseBind(request, 7, ticks[i].volume_real))
      {
       PrintFormat("DatabaseBind() failed with code=%d", GetLastError());
       PrintFormat("Tick #%d line=%d", i+1, __LINE__);
       request_error=true;
       break;
      }

   //--- Ausführen der Anfrage, um die Eingabe einzutragen, einschließlich einer Fe
   if(!request_error && !DatabaseRead(request) && (GetLastError()!=ERR_DATABASE_NO_
      {
       PrintFormat("DatabaseRead() failed with code=%d", GetLastError());
       DatabaseFinalize(request);
       request_error=true;
       break;
      }
   //--- Rücksetzen der Anfrage vor der nächsten Parameteraktualisierung
   if(!request_error && !DatabaseReset(request))
      {
       PrintFormat("DatabaseReset() failed with code=%d", GetLastError());
```

© 2000-2025, MetaQuotes Ltd.

```
         DatabaseFinalize(request);
         request_error=true;
         break;
        }
     } //--- Fertig, alle Ticks wurden verarbeitet

//--- Transaktionsstatus
   if(request_error)
     {
      PrintFormat("Table TICKS: failed to add %d ticks ", ArraySize(ticks));
      DatabaseTransactionRollback(db);
      DatabaseClose(db);
      return;
     }
   else
     {
      DatabaseTransactionCommit(db);
      PrintFormat("Table TICKS: added %d ticks in %d ms",
                  ArraySize(ticks), GetTickCount()-start);
     }

//--- Schließen der Datenbankdatei und Informieren darüber
   DatabaseClose(db);
   PrintFormat("Database: %s created and closed", filename);
  }
/*
 Ergebnis:
  EURUSD: CopyTicksRange received 268061 ticks in 47 ms (from 2020.03.18 12:40 to 2020
  Database: EURUSD 2020.03.18 12.40 - 2020.03.19 12.40.sqlite opened successfully
  #| cid name        type     notnull dflt_value pk
  -+-----------------------------------------------
  1|   0 SYMBOL      CHAR(10)       0              0
  2|   1 TIME        INT            1              0
  3|   2 BID         REAL           0              0
  4|   3 ASK         REAL           0              0
  5|   4 LAST        REAL           0              0
  6|   5 VOLUME      INT            0              0
  7|   6 TIME_MSC    INT            0              0
  8|   7 VOLUME_REAL REAL           0              0
  Table TICKS: added 268061 ticks in 797 ms
  Database: EURUSD 2020.03.18 12.40 - 2020.03.19 12.40.sqlite created and closed
  OnCalculateCorrelation=0.87 2020.03.19 13:00:  EURUSD vs GBPUSD  PERIOD_M30
*/
```

Siehe auch

DatabasePrepare, DatabaseReset, DatabaseRead, DatabaseBindArray

© 2000-2025, MetaQuotes Ltd.

DatabaseBindArray

Setzt einen Parameterarray als Parameterwert.

```
bool  DatabaseBind(
   int   request,      // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int   index,        // der Parameterindex in der Anfrage
   T&    array[]       // Array mit den Parameterwerten
   );
```

Parameter

request

[in] Das Handle einer Anfrage, das in DatabasePrepare() erstellt wurde.

index

[in] Der Parameterindex der Anfrage, dessen Werte gesetzt werden soll. Die Nummerierung beginnt mit Null.

array[]

[in] Der Array als Parameter der Anfrage.

Rückgabewert

Liefert bei Erfolg true, ansonsten false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - Typ wird nicht unterstützt;
- ERR_ARRAY_BAD_SIZE (4011) - die Arraygröße in Bytes überschreitet INT_MAX;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Handle der Datenbank;
- ERR_DATABASE_NOT_READY (5128) - die Funktion kann im Moment nicht für eine Anfrage verwenden (die Anfrage wird gerade ausgeführt oder ist bereits beendet, DatabaseReset sollte aufgerufen werden).

Hinweis

Die Funktion wird verwendet, wenn eine SQL-Anfrage parametrisierbare Werte der Form "?" oder "?N" enthält, wobei N den Parameterindex (beginnend mit Eins) bedeutet. Gleichzeitig beginnt die Indizierung der Parameter in DatabaseBindArray() bei Null.

Zum Beispiel:

```
INSERT INTO table VALUES (?,?,?)
```

Die Funktion kann unmittelbar nach der Erzeugung einer parametrisierten Anfrage in DatabasePrepare() oder nach dem Zurücksetzen der Anforderung mit DatabaseReset() aufgerufen werden.

Verwenden Sie diese Funktion zusammen mit DatabaseReset(), um die Anfrage so oft wie nötig mit verschiedenen Parameterwerten auszuführen.

Beispiel:

```
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Öffnen des Dialogs zur Dateiauswahl mit der Erweiterung DAT
   string selected_files[];
   if(!FileSelectDialog("Select files to download", NULL,
                        "Data files (*.dat)|*.dat|All files (*.*)|*.*",
                        FSD_ALLOW_MULTISELECT, selected_files, "tester.dat")>0)
     {
      Print("Files not selected. Exit");
      return;
     }
//--- Abrufen der Dateigröße
   ulong filesize[];
   int filehandle[];
   int files=ArraySize(selected_files);
   ArrayResize(filesize, files);
   ZeroMemory(filesize);
   ArrayResize(filehandle, files);
   double total_size=0;
   for(int i=0; i<files; i++)
     {
      filehandle[i]=FileOpen(selected_files[i], FILE_READ|FILE_BIN);
      if(filehandle[i]!=INVALID_HANDLE)
        {
         filesize[i]=FileSize(filehandle[i]);
         //PrintFormat("%d, %s handle=%d %d bytes", i, selected_files[i], filehandle[i
         total_size+=(double)filesize[i];
        }
     }
//--- Prüfen der Gesamtgröße aller Dateien
   if(total_size==0)
     {
      PrintFormat("Total files size is 0. Exit");
      return;
     }

//--- Erstellen oder Öffnen einer Datenbank im Verzeichnis Common des Terminals
   string filename="dat_files.sqlite";
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE);
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " open failed with code ", GetLastError());
      return;
     }
   else
      Print("Database: ", filename, " opened successfully");
//--- Falls die Tabelle FILES existiert, wird sie gelöscht
   if(DatabaseTableExists(db, "FILES"))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- löschen der Tabelle
       if(!DatabaseExecute(db, "DROP TABLE FILES"))
         {
          Print("Failed to drop table FILES with code ", GetLastError());
          DatabaseClose(db);
          return;
         }
      }
//--- Erstellen der Tabelle FILES
   if(!DatabaseExecute(db, "CREATE TABLE FILES("
                           "NAME           TEXT NOT NULL,"
                           "SIZE           INT  NOT NULL,"
                           "PERCENT_SIZE   REAL NOT NULL,"
                           "DATA           BLOB NOT NULL);"))
      {
       Print("DB: failed to create table FILES with code ", GetLastError());
       DatabaseClose(db);
       return;
      }
//--- Anzeige der Liste aller Felder der Tabelle FILES
   if(DatabasePrint(db, "PRAGMA TABLE_INFO(FILES)", 0)<0)
      {
       PrintFormat("DatabasePrint(\"PRAGMA TABLE_INFO(FILES)\") failed, error code=%d a
       DatabaseClose(db);
       return;
      }

//--- Erstellen einer parametrisierten Anfrage, um Dateien der Tabelle FILES hinzuzufü
   string sql="INSERT INTO FILES (NAME,SIZE,PERCENT_SIZE,DATA)"
              " VALUES (?1,?2,?3,?4);"; // Parameter der Anfrage
   int request=DatabasePrepare(db, sql);
   if(request==INVALID_HANDLE)
      {
       PrintFormat("DatabasePrepare() failed with code=%d", GetLastError());
       Print("SQL request: ", sql);
       DatabaseClose(db);
       return;
      }

//--- Schleife über alle Dateien, um sie der Tabelle FILES hinzuzufügen
   bool request_error=false;
   DatabaseTransactionBegin(db);
   int count=0;
   uint size;
   for(int i=0; i<files; i++)
      {
       if(filehandle[i]!=INVALID_HANDLE)
         {
```

© 2000-2025, MetaQuotes Ltd.

```
      char data[];
      size=FileReadArray(filehandle[i], data);
      if(size==0)
        {
         PrintFormat("FileReadArray(%s) failed with code %d", selected_files[i], Ge
         continue;
        }

      count++;
      //--- Setzen der Parameterwerte vor dem Hinzufügen der Datei zur Tabelle
      if(!DatabaseBind(request, 0, selected_files[i]))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, Get
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 1, size))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, Get
         request_error=true;
         break;
        }
      if(!DatabaseBind(request, 2, double(size)*100./total_size))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, Get
         request_error=true;
         break;
        }
      if(!DatabaseBindArray(request, 3, data))
        {
         PrintFormat("DatabaseBind() failed at line %d with code=%d", __LINE__, Get
         request_error=true;
         break;
        }
      //--- Ausführen der Anfrage, um die Eingabe einzutragen, einschließlich einer
      if(!DatabaseRead(request)&&(GetLastError()!=ERR_DATABASE_NO_MORE_DATA))
        {
         PrintFormat("DatabaseRead() failed with code=%d", GetLastError());
         DatabaseFinalize(request);
         request_error=true;
         break;
        }
      else
         PrintFormat("%d. %s: %d bytes", count, selected_files[i],size);
      //--- Rücksetzen der Anfrage vor der nächsten Parameteraktualisierung
      if(!DatabaseReset(request))
        {
         PrintFormat("DatabaseReset() failed with code=%d", GetLastError());
         DatabaseFinalize(request);
```

© 2000-2025, MetaQuotes Ltd.

```
            request_error=true;
            break;
           }
        }
     }
//--- Transaktionsstatus
   if(request_error)
     {
      PrintFormat("Table FILES: failed to add %d files", count);
      DatabaseTransactionRollback(db);
      DatabaseClose(db);
      return;
     }
   else
     {
      DatabaseTransactionCommit(db);
      PrintFormat("Table FILES: added %d files", count);
     }

//--- Schließen der Datenbankdatei und Informieren darüber
   DatabaseClose(db);
   PrintFormat("Database: %s created and closed", filename);
  }
```

Siehe auch

DatabasePrepare, DatabaseReset, DatabaseRead, DatabaseBind

© 2000-2025, MetaQuotes Ltd.

DatabaseRead

Wechselt zum nächsten Eintrag als Ergebnis einer Anfrage.

```
bool  DatabaseRead(
   int  request      // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   );
```

Parameter

request

[in] Handle der Anfrage, das durch DatabasePrepare() erhalten wurde.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - Tabellenname wurde nicht angegeben (leer oder NULL);
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) - es existiert keine Tabelle (kein Fehler, normales Ende).

Siehe auch

DatabasePrepare, DatabaseReadBind

© 2000-2025, MetaQuotes Ltd.

DatabaseReadBind

Wechselt zum nächsten Datensatz und liest aus ihm Daten in die Struktur.

```
bool  DatabaseReadBind(
   int    request,           // Handle der Anfrage, das durch DatabasePrepare erhalten
   void&  struct_object      // die Referenz der Struktur, der das Ergebnis zugewiesen
   );
```

Parameter

request

[in] Das Handle einer Anfrage, das in DatabasePrepare() erstellt wurde.

struct_object

[out] Die Referenz auf die Struktur, der die Daten aus dem aktuellen Datensatz zugewiesen werden sollen. Die Struktur sollte nur numerische Typen und/oder Zeichenketten (Arrays sind nicht erlaubt) als Mitglieder haben und darf nicht abgeleitet worden sein.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INVALID_PARAMETER (4003) - Tabellenname wurde nicht angegeben (leer oder NULL);
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) - es existiert keine Tabelle (kein Fehler, normales Ende).

Hinweis

Anzahl der Felder in der Struktur *struct_object* sollte DatenbankColumnsCount() nicht überschreiten. Wenn die Anzahl der Felder in der Struktur *struct_object* kleiner ist als die Anzahl der Felder im Datensatz, werden nur Teile des Datensatzes zugewiesen. Die restlichen Daten können explizit über DatabaseColumnText(), DatabaseColumnInteger() und den anderen, geeigneten Funktionen abgerufen werden.

Beispiel:

```
struct Person
  {
   int               id;
   string            name;
   int               age;
   string            address;
   double            salary;
  };
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int db;
   string filename="company.sqlite";
//--- open
   db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE |DATABASE_
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " open failed with code ", GetLastError());
      return;
     }
//--- if the table COMPANY exists then drop the table
   if(DatabaseTableExists(db, "COMPANY"))
     {
      //--- delete the table
      if(!DatabaseExecute(db, "DROP TABLE COMPANY"))
        {
         Print("Failed to drop table COMPANY with code ", GetLastError());
         DatabaseClose(db);
         return;
        }
     }
//--- create table
   if(!DatabaseExecute(db, "CREATE TABLE COMPANY("
                       "ID INT PRIMARY KEY     NOT NULL,"
                       "NAME           TEXT    NOT NULL,"
                       "AGE            INT     NOT NULL,"
                       "ADDRESS        CHAR(50),"
                       "SALARY         REAL );"))
     {
      Print("DB: ", filename, " create table failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }

//--- insert data
   if(!DatabaseExecute(db, "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (1
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'A
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'T
                       "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'M
     {
      Print("DB: ", filename, " insert failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }

//--- prepare the request
```

© 2000-2025, MetaQuotes Ltd.

```
   int request=DatabasePrepare(db, "SELECT * FROM COMPANY WHERE SALARY>15000");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
//--- print records
   Person person;
   Print("Persons with salary > 15000:");
   for(int i=0; DatabaseReadBind(request, person); i++)
      Print(i, ":  ", person.id, " ", person.name, " ", person.age, " ", person.addres
//--- delete request after use
   DatabaseFinalize(request);

   Print("Some statistics:");
//--- prepare new request about total salary
   request=DatabasePrepare(db, "SELECT SUM(SALARY) FROM COMPANY");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      DatabaseClose(db);
      return;
     }
   while(DatabaseRead(request))
     {
      double total_salary;
      DatabaseColumnDouble(request, 0, total_salary);
      Print("Total salary=", total_salary);
     }
//--- delete request after use
   DatabaseFinalize(request);

//--- prepare new request about average salary
   request=DatabasePrepare(db, "SELECT AVG(SALARY) FROM COMPANY");
   if(request==INVALID_HANDLE)
     {
      Print("DB: ", filename, " request failed with code ", GetLastError());
      ResetLastError();
      DatabaseClose(db);
      return;
     }
   while(DatabaseRead(request))
     {
      double aver_salary;
      DatabaseColumnDouble(request, 0, aver_salary);
      Print("Average salary=", aver_salary);
     }
//--- delete request after use
```

© 2000-2025, MetaQuotes Ltd.

```
   DatabaseFinalize(request);

//--- close database
   DatabaseClose(db);
  }
//+------------------------------------------------------------------+
/*
Output:
Persons with salary > 15000:
0:  1 Paul 32 California 25000.0
1:  3 Teddy 23 Norway 20000.0
2:  4 Mark 25 Rich-Mond  65000.0
Some statistics:
Total salary=125000.0
Average salary=31250.0
*/
```

Siehe auch

DatabasePrepare, DatabaseRead

© 2000-2025, MetaQuotes Ltd.

DatabaseFinalize

Entfernt eine Anfrage, die durch DatabasePrepare() erstellt wurde.

```
void  DatabaseFinalize(
   int  request      // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   );
```

Parameter

request

[in] Handle der Anfrage, erhalten von DatabasePrepare().

Rückgabewert

Keiner.

Hinweis

Sind die Handles ungültig, wirft die Funktion den Fehler ERR_DATABASE_INVALID_HANDLE aus. Das kann mit der Funktion GetLastError() geprüft werden.

Siehe auch

DatabasePrepare, DatabaseExecute

© 2000-2025, MetaQuotes Ltd.

DatabaseTransactionBegin

Startet die Ausführung der Transaktion.

```
bool  DatabaseTransactionBegin(
   int  database      // Handle der Datenbank erhalten von DatabaseOpen
   );
```

Parameter

database

[in] Handle der Datenbank erhalten von DatabaseOpen().

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_INVALID_PARAMETER (4003) - SQL-Parameter enthält eine leere Zeichenkette;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage.

Hinweis

DatabaseTransactionBegin() sollte vor einer Transaktionsausführung aufgerufen werden. Jede Transaktion sollte mit dem Aufruf von DatabaseTransactionBegin() beginnen und mit dem Aufruf von DatabaseTransactionCommit() enden.

Example:

```
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create the file name
   string filename=AccountInfoString(ACCOUNT_SERVER) +"_"+IntegerToString(AccountInfo
//--- open/create the database in the common terminal folder
   int db=DatabaseOpen(filename, DATABASE_OPEN_READWRITE | DATABASE_OPEN_CREATE | DATA
   if(db==INVALID_HANDLE)
     {
      Print("DB: ", filename, " open failed with code ", GetLastError());
      return;
     }
//--- if the DEALS table already exists, delete it
   if(!DeleteTable(db, "DEALS"))
     {
      DatabaseClose(db);
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
//--- create the DEALS table
   if(!CreateTableDeals(db))
     {
      DatabaseClose(db);
      return;
     }
//---  request the entire trading history
   datetime from_date=0;
   datetime to_date=TimeCurrent();
//--- request the history of deals in the specified interval
   HistorySelect(from_date, to_date);
   int deals_total=HistoryDealsTotal();
   PrintFormat("Deals in the trading history: %d ", deals_total);

//--- measure the transaction execution speed using DatabaseTransactionBegin/DatabaseT
   ulong start=GetMicrosecondCount();
   bool fast_transactions=true;
   InsertDeals(db, fast_transactions);
   double fast_transactions_time=double(GetMicrosecondCount()-start)/1000;
   PrintFormat("Transations WITH    DatabaseTransactionBegin/DatabaseTransactionCommit

//--- delete the DEALS table, and then create it again
   if(!DeleteTable(db, "DEALS"))
     {
      DatabaseClose(db);
      return;
     }
//--- create a new DEALS table
   if(!CreateTableDeals(db))
     {
      DatabaseClose(db);
      return;
     }

//--- test again, this time without using DatabaseTransactionBegin/DatabaseTransactior
   fast_transactions=false;
   start=GetMicrosecondCount();
   InsertDeals(db, fast_transactions);
   double slow_transactions_time=double(GetMicrosecondCount()-start)/1000;
   PrintFormat("Transations WITHOUT DatabaseTransactionBegin/DatabaseTransactionCommit
//--- report gain in time
   PrintFormat("Use of DatabaseTransactionBegin/DatabaseTransactionCommit provided acc
//--- close the database
   DatabaseClose(db);
  }
/*
Results:
   Deals in the trading history: 2737
```

© 2000-2025, MetaQuotes Ltd.

```
   Transations WITH    DatabaseTransactionBegin/DatabaseTransactionCommit: time=48.5 r
   Transations WITHOUT DatabaseTransactionBegin/DatabaseTransactionCommit: time=25818.
   Use of DatabaseTransactionBegin/DatabaseTransactionCommit provided acceleration by
*/
//+------------------------------------------------------------------+
//| Deletes a table with the specified name from the database        |
//+------------------------------------------------------------------+
bool DeleteTable(int database, string table_name)
  {
   if(!DatabaseExecute(database, "DROP TABLE IF EXISTS "+table_name))
     {
      Print("Failed to drop table DEALS with code ", GetLastError());
      return(false);
     }
//--- the table has been successfully deleted
   return(true);
  }
//+------------------------------------------------------------------+
//| Creates the DEALS table                                          |
//+------------------------------------------------------------------+
bool CreateTableDeals(int database)
  {
//--- check if the table exists
   if(!DatabaseTableExists(database, "DEALS"))
      //--- create the table
      if(!DatabaseExecute(database, "CREATE TABLE DEALS("
                          "ID          INT KEY NOT NULL,"
                          "ORDER_ID    INT     NOT NULL,"
                          "POSITION_ID INT     NOT NULL,"
                          "TIME        INT     NOT NULL,"
                          "TYPE        INT     NOT NULL,"
                          "ENTRY       INT     NOT NULL,"
                          "SYMBOL      CHAR(10),"
                          "VOLUME      REAL,"
                          "PRICE       REAL,"
                          "PROFIT      REAL,"
                          "SWAP        REAL,"
                          "COMMISSION  REAL,"
                          "MAGIC       INT,"
                          "REASON      INT );"))
        {
         Print("DB: create table failed with code ", GetLastError());
         return(false);
        }
//--- the table has been successfully created
   return(true);
  }
//+------------------------------------------------------------------+
//| Adds deals to the database table                                 |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool InsertDeals(int database, bool begintransaction=true)
  {
//--- Auxiliary variables
   ulong    deal_ticket;         // deal ticket
   long     order_ticket;        // the ticket of the order by which the deal was exec
   long     position_ticket;     // ID of the position to which the deal belongs
   datetime time;                // deal execution time
   long     type ;               // deal type
   long     entry ;              // deal direction
   string   symbol;              // the symbol fro which the deal was executed
   double   volume;              // operation volume
   double   price;               // price
   double   profit;              // financial result
   double   swap;                // swap
   double   commission;          // commission
   long     magic;               // Magic number
   long     reason;              // deal execution reason or source
//--- go through all deals and add to the database
   bool failed=false;
   int deals=HistoryDealsTotal();
//--- if fast transaction performance method is used
   if(begintransaction)
     {
      // --- lock the database before executing transactions
      DatabaseTransactionBegin(database);
     }
   for(int i=0; i<deals; i++)
     {
      deal_ticket=    HistoryDealGetTicket(i);
      order_ticket=   HistoryDealGetInteger(deal_ticket, DEAL_ORDER);
      position_ticket=HistoryDealGetInteger(deal_ticket, DEAL_POSITION_ID);
      time= (datetime)HistoryDealGetInteger(deal_ticket, DEAL_TIME);
      type=           HistoryDealGetInteger(deal_ticket, DEAL_TYPE);
      entry=          HistoryDealGetInteger(deal_ticket, DEAL_ENTRY);
      symbol=         HistoryDealGetString(deal_ticket, DEAL_SYMBOL);
      volume=         HistoryDealGetDouble(deal_ticket, DEAL_VOLUME);
      price=          HistoryDealGetDouble(deal_ticket, DEAL_PRICE);
      profit=         HistoryDealGetDouble(deal_ticket, DEAL_PROFIT);
      swap=           HistoryDealGetDouble(deal_ticket, DEAL_SWAP);
      commission=     HistoryDealGetDouble(deal_ticket, DEAL_COMMISSION);
      magic=          HistoryDealGetInteger(deal_ticket, DEAL_MAGIC);
      reason=         HistoryDealGetInteger(deal_ticket, DEAL_REASON);
      //--- add each deal using the following request
      string request_text=StringFormat("INSERT INTO DEALS (ID,ORDER_ID,POSITION_ID,TIM
                                       "VALUES (%d, %d, %d, %d, %d, %d, '%s', %G, %G,
                                       deal_ticket, order_ticket, position_ticket, tim
      if(!DatabaseExecute(database, request_text))
        {
```

© 2000-2025, MetaQuotes Ltd.

```
         PrintFormat("%s: failed to insert deal #%dwith code %d", __FUNCTION__, deal_t
         PrintFormat("i=%d: deal #%d  %s", i, deal_ticket, symbol);
         failed=true;
         break;
        }
     }
//--- check for transaction execution errors
   if(failed)
     {
      //--- if fast transaction performance method is used
      if(begintransaction)
        {
         //--- roll back all transactions and unlock the database
         DatabaseTransactionRollback(database);
        }
      Print("%s: DatabaseExecute() failed with code ", __FUNCTION__, GetLastError());
      return(false);
     }
//--- if fast transaction performance method is used
   if(begintransaction)
     {
      //--- all transactions have been performed successfully - record changes and unl
      DatabaseTransactionCommit(database);
     }
//--- successful completion
   return(true);
  }
//+------------------------------------------------------------------+
```

Siehe auch

DatabaseExecute, DatabasePrepare, DatabaseTransactionCommit, DatabaseTransactionRollback

© 2000-2025, MetaQuotes Ltd.

DatabaseTransactionCommit

Schließt die Ausführung der Transaktion ab.

```
bool  DatabaseTransactionCommit(
   int  database      // Handle der Datenbank erhalten von DatabaseOpen
   );
```

Parameter

database

[in] Handle der Datenbank erhalten von DatabaseOpen().

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_INVALID_PARAMETER (4003) - SQL-Parameter enthält eine leere Zeichenkette;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage.

Hinweis

Die Funktion DatabaseTransactionCommit() schließt alle Transaktionen ab, die nach dem Aufruf der Funktion DatabaseBeginTransaction() ausgeführt werden. Jede Transaktion sollte mit dem Aufruf von DatabaseTransactionBegin() beginnen und mit dem Aufruf von DatabaseTransactionCommit() für einen erfolgreichen Abschluss enden.

Siehe auch

DatabaseExecute, DatabasePrepare, DatabaseTransactionBegin, DatabaseTransactionRollback

© 2000-2025, MetaQuotes Ltd.

RDatabaseTransactionRollback

Zurücksetzen der Transaktionen.

```
bool  DatabaseTransactionRollback(
   int  database      // Handle der Datenbank erhalten von DatabaseOpen
   );
```

Parameter

database

[in] Handle der Datenbank erhalten von DatabaseOpen().

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_INTERNAL_ERROR (4001) - kritischer Laufzeitfehler;
- ERR_INVALID_PARAMETER (4003) - SQL-Parameter enthält eine leere Zeichenkette;
- ERR_NOT_ENOUGH_MEMORY (4004) - ungenügend Speicher;
- ERR_WRONG_STRING_PARAMETER (5040) - Fehler beim Konvertieren der Anfrage in eine UTF-8 Zeichenkette;
- ERR_DATABASE_INTERNAL (5120) - interner Datenbankfehler;
- ERR_DATABASE_INVALID_HANDLE (5121) - ungültiges Datenbankhandle;
- ERR_DATABASE_EXECUTE (5124) - Fehler bei der Ausführung der Anfrage.

Hinweis

Der Aufruf von DatabaseTransactionRollback() bricht alle Transaktionen ab, die nach dem Aufruf von DatabaseTransactionBegin() ausgeführt werden. Die Funktion DatabaseTransactionRollback() ist notwendig, um Änderungen in einer Datenbank zurückzusetzen, falls beim Ausführen einer Transaktion Fehler auftreten.

Siehe auch

DatabaseExecute, DatabasePrepare, DatabaseTransactionBegin, DatabaseTransactionCommit

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnsCount

Abrufen der Felderanzahl einer Anfrage.

```
int  DatabaseColumnsCount(
   int  request      // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

Rückgabewert

Anzahl der Felder oder -1 im Fehlerfall. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage.

Hinweis

Es ist nicht notwendig, die Funktion DatabaseRead() aufzurufen, um die Anzahl der Felder einer in DatabasePrepare() erstellten Anfrage zu erhalten. Für die restlichen Funktionen DatabaseColumnXXX() sollte vorher DatabaseRead() aufgerufen werden.

Siehe auch

DatabasePrepare, DatabaseFinalize, DatabaseClose

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnName

Abrufen des Feldnamens nach dem Index.

```
bool  DatabaseColumnName(
   int      request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wu
   int      column,      // Feldindex in der Anfrage
   string&  name         // die Referenz der Variablen, der der Feldname zugewiesen w
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

name

[out] Variable, der der Feldname zugewiesen wird.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Siehe auch

DatabasePrepare, DatabaseColumnsCount, DatabaseColumnType

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnType

Abrufen des Feldtyps nach dem Index.

```
ENUM_DATABASE_FIELD_TYPE  DatabaseColumnType(
   int  request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int  column       // Feldindex der Anfrage
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

Rückgabewert

Es wird der Feldtyp aus der Enumeration ENUM_DATABASE_FIELD_TYPE zurückgegeben. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

ENUM_DATABASE_FIELD_TYPE

ID	Beschreibung
DATABASE_FIELD_TYPE_INVALID	Error Getting Type, die Fehlernummer kann mit GetLastError() abgerufen werden.
DATABASE_FIELD_TYPE_INTEGER	Typ ganze zahl
DATABASE_FIELD_TYPE_FLOAT	Typ reelle Zahl
DATABASE_FIELD_TYPE_TEXT	Typ Zeichenkette
DATABASE_FIELD_TYPE_BLOB	Typ binär
DATABASE_FIELD_TYPE_NULL	Sondertyp NULL

Siehe auch

DatabasePrepare, DatabaseColumnsCount, DatabaseColumnName

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnSize

Abrufen der Feldgröße in Bytes.

```
int  DatabaseColumnSize(
   int  request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int  column       // Feldindex der Anfrage
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

Rückgabewert

Bei Erfolg wird wird die Feldgröße in Bytes zurückgegeben, andernfalls -1. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Siehe auch

DatabasePrepare, DatabaseColumnBlob, DatabaseColumnsCount, DatabaseColumnName, DatabaseColumnType

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnText

Abrufen des Feldwerts als Zeichenkette aus dem aktuellen Datensatz.

```
bool  DatabaseColumnText(
   int      request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wu
   int      column,      // Feldindex in der Anfrage
   string&  value        // die Referenz der Variablen, der der Wert zugewiesen wird
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

value

[out] Referenz der Variablen, der der Feldwert zugewiesen wird.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Um den Wert aus dem nächsten Datensatz zu lesen, rufen Sie vorher DatabaseRead() auf.

Siehe auch

DatabasePrepare, DatabaseColumnsCount, DatabaseColumnType, DatabaseColumnName

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnInteger

Abrufen des Integer-Werts aus dem aktuellen Datensatz.

```
bool  DatabaseColumnInteger(
   int   request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int   column,      // Feldindex in der Anfrage
   int&  value        // die Referenz der Variablen, der der Wert zugewiesen wird
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

value

[out] Referenz der Variablen, der der Feldwert zugewiesen wird.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Um den Wert aus dem nächsten Datensatz zu lesen, rufen Sie vorher DatabaseRead() auf.

Siehe auch

DatabasePrepare, DatabaseColumnsCount, DatabaseColumnType, DatabaseColumnName

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnLong

Abrufen des Long-Werts aus dem aktuellen Datensatz.

```
bool  DatabaseColumnLong(
   int    request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int    column,      // Feldindex in der Anfrage
   long&  value        // die Referenz der Variablen, der der Wert zugewiesen wird
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

value

[out] Referenz der Variablen, der der Feldwert zugewiesen wird.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Um den Wert aus dem nächsten Datensatz zu lesen, rufen Sie vorher DatabaseRead() auf.

Siehe auch

DatabasePrepare, DatabaseColumnsCount, DatabaseColumnType, DatabaseColumnName

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnDouble

Abrufen des Double-Wertes aus dem aktuellen Datensatz.

```
bool  DatabaseColumnDouble(
   int      request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wu
   int      column,      // Feldindex in der Anfrage
   double&  value        // die Referenz der Variablen, der der Wert zugewiesen wird
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

value

[out] Referenz der Variablen, der der Feldwert zugewiesen wird.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Um den Wert aus dem nächsten Datensatz zu lesen, rufen Sie vorher DatabaseRead() auf.

Siehe auch

DatabasePrepare, DatabaseColumnsCount, DatabaseColumnType, DatabaseColumnName

© 2000-2025, MetaQuotes Ltd.

DatabaseColumnBlob

Ruft den Feldwert als Array aus dem aktuellen Datensatz ab.

```
bool  DatabaseColumnBlob(
   int    request,     // Handle der Anfrage, das durch DatabasePrepare erhalten wurde
   int    column,      // Feldindex in der Anfrage
   void&  data[]       // die Referenz der Variablen, der der Wert zugewiesen wird
   );
```

Parameter

request

[in] Handle der Anfrage, das von DatabasePrepare() erhalten wurde.

column

[in] Feldindex der Anfrage. Die Feldnummerierung beginnt bei Null und darf DatenbankColumnsCount() - 1. nicht überschreiten.

data[]

[out] Referenz des Arrays, dem der Feldwert zugewiesen wird.

Rückgabewert

Rückgabe, bei Erfolg, von true sonst false. Um die Fehlernummer zu erhalten, verwenden Sie GetLastError(), die möglichen Antworten sind:

- ERR_DATABASE_INVALID_HANDLE (5121) — ungültiges Handle der Anfrage;
- ERR_DATABASE_NO_MORE_DATA (5126) — Der 'Spaltenindex' überschreitet DatabaseColumnsCount()-1.

Hinweis

Der Wert kann nur erhalten werden, wenn vorher mindestens ein Aufruf von DatabaseRead() für die 'Anfrage' erfolgte.

Um den Wert aus dem nächsten Datensatz zu lesen, rufen Sie vorher DatabaseRead() auf.

Siehe auch

DatabasePrepare, DatabaseColumnSize, DatabaseColumnsCount, DatabaseColumnType, DatabaseColumnName

© 2000-2025, MetaQuotes Ltd.

Arbeiten mit DirectX

Die Funktionen von DirectX 11 und des Shaders (erzeugt die Schatten) sind für die 3D-Visualisierung direkt auf einem Preischart konzipiert.

Die Erstellung von 3D-Grafiken erfordert einen grafischen Kontext (DXContextCreate) mit der erforderlichen Bildgröße. Außerdem ist es notwendig, Vertex- und Indexpuffer (DXBufferCreate) vorzubereiten, sowie Vertex- und Pixel-Shader (DXShaderCreate) zu erstellen. Dies reicht aus, um Grafiken in Farbe darzustellen.

Die nächste Grafikstufe benötigt die Eingaben (DXInputSet) zur Übergabe zusätzlicher Renderingparameter an den Shader. Dies ermöglicht die Einstellung der Position der Kamera und des 3D-Objekts, die Beschreibung der Lichtquellen und die Steuerung von Maus und Tastatur.

Mit den integrierten MQL5-Funktionen können Sie somit animierte 3D-Diagramme direkt in MetaTrader 5 erstellen, ohne dass Sie Werkzeuge von Drittanbietern benötigen. Eine Grafikkarte sollte DX 11 und Shader Model 5.0 unterstützen, damit die Funktionen funktionieren.

Um die Arbeit mit der Bibliothek zu beginnen, lesen Sie einfach den Artikel Wie man 3D-Grafiken mit DirectX in MetaTrader 5 erstellt.

Funktion	Aktion
DXContextCreate	Erzeugt einen grafischen Kontext zum Rendern von Frames einer bestimmten Größe.
DXContextSetSize	Ändert die Frame-Größe eines mit DXContextCreate() erstellten Grafikkontextes.
DXContextSetSize	Liefert die Rahmengröße eines mit DXContextCreate() erstellten Grafikkontextes.
DXContextClearColors	Weist eine bestimmte Farbe allen Pixeln für den Rendering-Puffer.
DXContextClearDepth	Löscht den Tiefenpuffer
DXContextGetColors	Ruft ein Bild einer bestimmten Größe und eines bestimmten Offsets aus einem Grafikkontext ab.
DXContextGetDepth	Ruft den Tiefenpuffer eines gerenderten Frames ab
DXBufferCreate	Erstellt einen Puffer eines bestimmten Typs basierend auf einem Daten-Array
DXTextureCreate	Erstellt eine 2D-Textur aus einem Rechteck mit einer bestimmten Größe, das aus einem übergebenen Bild ausgeschnitten wurde.
DXInputCreate	Erstellt die Eingaben des Shaders
DXInputSet	Setzt die Eingaben des Shaders
DXShaderCreate	Erstellt einen Shader eines bestimmten Typs
DXShaderSetLayout	Legt das Vertex-Layout für den Vertex-Shader fest

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
DXShaderInputsSet	Setzt die Eingaben des Shaders
DXShaderTexturesSet	Legt die Texturen des Shaders fest
DXDraw	Rendert die Eckpunkte des in DXBufferSet() gesetzten Vertex-Puffers
DXDrawIndexed	Rendert grafische Primitive, die durch den Indexpuffer von DXBufferSet() beschrieben werden
DXPrimiveTopologySet	Setzt den Typ der Primitive für das Rendern mit DXDrawIndexed()
DXBufferSet	Setzt einen Puffer für das aktuelle Rendering
DXShaderSet	Setzt einen Shader für das Rendern
DXHandleType	Liefert einen Handle-Typ zurück
DXRelease	Gibt ein Handle frei

© 2000-2025, MetaQuotes Ltd.

DXContextCreate

Erzeugt einen grafischen Kontext zum Rendern von Frames einer bestimmten Größe.

```
int  DXContextCreate(
   uint  width,      // Breite in Pixel
   uint  height      // Höhe in Pixel
   );
```

Parameter

width

[in] Frame-Breite in Pixel.

height

[in] Frame-Höhe in Pixel.

Rückgabewert

Das Handle des erstellten Kontextes oder INVALID_HANDLE im Fehlerfall. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Alle grafischen Objekte, die mit den Funktionen DXBufferCreate, DXInputCreate, DXShaderCreate und DXTextureCreate erstellt wurden, können nur in dem grafischen Kontext verwendet werden, in dem sie erstellt wurden.

Eine Frame-Größe kann nachträglich auf DXContextSetSize() geändert werden.

Ein erstelltes Handle, das nicht mehr verwendet wird, sollte explizit durch die Funktion DXRelease() freigegeben werden.

© 2000-2025, MetaQuotes Ltd.

DXContextSetSize

Ändert die Frame-Größe eines mit DXContextCreate() erstellten Grafikkontextes.

```
bool  DXContextSetSize(
   int    context,      // Handle des grafischen Kontextes
   uint&  width,        // Breite in Pixel
   uint&  height        // Höhe in Pixel
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

width

[in] Frame-Breite in Pixel.

height

[in] Frame-Höhe in Pixel.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Frame-Größe eines grafischen Kontextes sollte nur zwischen dem Rendern der Frames geändert werden.

© 2000-2025, MetaQuotes Ltd.

DXContextGetSize

Liefert die Rahmengröße eines mit DXContextCreate() erstellten Grafikkontextes.

```
bool  DXContextGetSize(
   int    context,      // Handle des grafischen Kontextes
   uint&  width,        // Breite in Pixel
   uint&  height        // Höhe in Pixel
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

width

[out] Frame-Breite in Pixel.

height

[out] Frame-Höhe in Pixel.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

© 2000-2025, MetaQuotes Ltd.

DXContextClearColors

Setzt die angegebene Farbe für alle Pixel im Puffer für das Rendern.

```
bool  DXContextClearColors(
   int              context,      // Handle des grafischen Kontextes
   const DXVector&  color         // Farbe
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

color

[in] Farbe für das Rendern.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Funktion DXContextClearColors() kann verwendet werden, um die Farbe des Puffers vor dem Rendern des nächsten Frames zu löschen.

© 2000-2025, MetaQuotes Ltd.

DXContextClearDepth

Löscht den Tiefenpuffer.

```
bool  DXContextClearDepth(
   int  context      // Handle des grafischen Kontextes
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Funktion DXContextClearDepth() kann verwendet werden, um die Farbe des Puffers vor dem Rendern des nächsten Frames zu löschen.

© 2000-2025, MetaQuotes Ltd.

DXContextGetColors

Ruft ein Bild einer bestimmten Größe und eines bestimmten Offsets aus einem Grafikkontext ab.

```
bool  DXContextGetColors(
   int    context,                      // Handle des grafischen Kontextes
   uint&  image[],                      // Array der Pixel des Bildes
   int    image_width=WHOLE_ARRAY,      // Bildbreite in Pixel
   int    image_height=WHOLE_ARRAY,     // Bildhöhe in Pixel
   int    image_offset_x=0,             // X-Abstand
   int    image_offset_y=0              // Y-Abstand offset
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

image

[out] Ds Array von *image_width*image_height* Pixel im Format ARGB.

image_width=WHOLE_ARRAY

[in] Bildbreite in Pixel.

image_height=WHOLE_ARRAY

[in] Bildhöhe in Pixel.

image_offset_x=0

[in] X-Abstand.

image_offset_y=0

[in] Y-Abstand.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

© 2000-2025, MetaQuotes Ltd.

DXContextGetDepth

Ruft den Tiefenpuffer eines gerenderten Frames ab.

```
bool  DXContextGetDepth(
   int     context,      // Handle des grafischen Kontextes
   float&  image[]       // Array der Tiefe
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

image

[out] Array mit den Werten für die Tiefe des gerenderten Frames.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Der zurückgegebene Puffer enthält die Tiefe jedes Pixels eines gerenderten Frames, die in DXContextGetColors() in relativen Einheiten (von 0,0 bis 1,0) erhalten werden kann.

© 2000-2025, MetaQuotes Ltd.

DXBufferCreate

Erstellt einen Puffer eines bestimmten Typs basierend auf einem Daten-Array.

```
int  DXBufferCreate(
   int                  context,              // Handle des grafischen Kontextes
   ENUM_DX_BUFFER_TYPE  buffer_type,          // Typ des zu erstellenden Puffers
   const void&          data[],               // Daten des Puffers
   uint                 start=0,              // Startindex
   uint                 count=WHOLE_ARRAY     // Anzahl der Elemente
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

buffer_type

[in] Puffertyp für die Enumeration ENUM_DX_BUFFER_TYPE.

data[]

[in] Daten für den zu erstellenden Puffer.

start=0

[in] Index des ersten Elements des Arrays. Ab dieser Stelle werden die Daten des Arrays für den Puffer verwendet. Standardmäßig werden die Daten vom Beginn des Arrays verwendet.

count=WHOLE_ARRAY

[in] Anzahl der Werte. Standardmäßig wird das ganze Array verwendet (count=WHOLE_ARRAY).

Rückgabewert

Das Handle eines erstellten Puffers oder INVALID_HANDLE im Fehlerfall. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Für den Indexpuffer sollte das Array *data[]* vom Typ 'uint' sein, während der Vertexpuffer das Array der Strukturen empfängt, die Vertices beschreiben.

Ein erstelltes Handle, das nicht mehr verwendet wird, sollte explizit durch die Funktion DXRelease() freigegeben werden.

ENUM_DX_BUFFER_TYPE

ID	Wert	Beschreibung
DX_BUFFER_VERTEX	1	Vertex-Puffer
DX_BUFFER_INDEX	2	Index-Puffer

© 2000-2025, MetaQuotes Ltd.

DXTextureCreate

Erstellt eine 2D-Textur aus einem Rechteck mit einer bestimmten Größe, das aus einem übergebenen Bild ausgeschnitten wurde.

```
int  DXTextureCreate(
   int             context,          // Handle des grafischen Kontextes
   ENUM_DX_FORMAT  format,           // Format der Farbpixel
   uint            width,            // Breite des Ursprungsbildes
   uint            height,           // Höhe des Ursprungsbildes
   const void&     data[],           // Array der Pixel des Ursprungsbildes
   uint            data_x,           // X-Koordinate des Rechtecks für das Erstellen de
   uint            data_y,           // Y-Koordinate des Rechtecks für das Erstellen de
   uint            data_width,       // Breite des Rechtecks für das Erstellen der Text
   uint            data_height       // Höhe des Rechtecks für das Erstellen der Textur
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

format

[in] Format der Pixelfarbe nach der Enumeration ENUM_DX_FORMAT.

width

[in] Die Breite eines Bildest, auf dem die Textur basiert.

height

[in] Die Höhe eines Bildest, auf dem die Textur basiert.

data

[in] Pixel-Array eines Bildes, auf dem die Textur basiert.

data_x

[in] X-Koordinate eines Rechtecks (X-Achsen-Abstand) für das Erstellen einer Textur.

data_y

[in] Y-Koordinate eines Rechtecks (Y-Achsen-Abstand) für das Erstellen einer Textur.

data_width

[in] Breite eines Rechtecks für das Erstellen einer Textur.

data_height

[in] Höhe eines Rechtecks für das Erstellen einer Textur.

Rückgabewert

Handle der Textur oder INVALID_HANDLE im Fehlerfall. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

© 2000-2025, MetaQuotes Ltd.

Ein erstelltes Handle, das nicht mehr verwendet wird, sollte explizit durch die Funktion DXRelease() freigegeben werden.

ENUM_DX_FORMAT

ID	Wert	Passt in DXGI_FORMAT
DX_FORMAT_UNKNOWN	0	DXGI_FORMAT_UNKNOWN
DX_FORMAT_R32G32B32A32_TYPELESS	1	DXGI_FORMAT_R32G32B32A32_TYPELESS
DX_FORMAT_R32G32B32A32_FLOAT	2	DXGI_FORMAT_R32G32B32A32_FLOAT
DX_FORMAT_R32G32B32A32_UINT	3	DXGI_FORMAT_R32G32B32A32_UINT
DX_FORMAT_R32G32B32A32_SINT	4	DXGI_FORMAT_R32G32B32A32_SINT
DX_FORMAT_R32G32B32_TYPELESS	5	DXGI_FORMAT_R32G32B32_TYPELESS
DX_FORMAT_R32G32B32_FLOAT	6	DXGI_FORMAT_R32G32B32_FLOAT
DX_FORMAT_R32G32B32_UINT	7	DXGI_FORMAT_R32G32B32_UINT
DX_FORMAT_R32G32B32_SINT	8	DXGI_FORMAT_R32G32B32_SINT
DX_FORMAT_R16G16B16A16_TYPELESS	9	DXGI_FORMAT_R16G16B16A16_TYPELESS
DX_FORMAT_R16G16B16A16_FLOAT	10	DXGI_FORMAT_R16G16B16A16_FLOAT
DX_FORMAT_R16G16B16A16_UNORM	11	DXGI_FORMAT_R16G16B16A16_UNORM
DX_FORMAT_R16G16B16A16_UINT	12	DXGI_FORMAT_R16G16B16A16_UINT
DX_FORMAT_R16G16B16A16_SNORM	13	DXGI_FORMAT_R16G16B16A16_SNORM
DX_FORMAT_R16G16B16A16_SINT	14	DXGI_FORMAT_R16G16B16A16_SINT
DX_FORMAT_R32G32_TYPELESS	15	DXGI_FORMAT_R32G32_TYPELESS
DX_FORMAT_R32G32_FLOAT	16	DXGI_FORMAT_R32G32_FLOAT
DX_FORMAT_R32G32_UINT	17	DXGI_FORMAT_R32G32_UINT
DX_FORMAT_R32G32_SINT	18	DXGI_FORMAT_R32G32_SINT
DX_FORMAT_R32G8X24_TYPELESS	19	DXGI_FORMAT_R32G8X24_TYPELESS
DX_FORMAT_D32_FLOAT_S8X24_UINT	20	DXGI_FORMAT_D32_FLOAT_S8X24_UINT
DX_FORMAT_R32_FLOAT_X8X24_TYPELESS	21	DXGI_FORMAT_R32_FLOAT_X8X24_TYPELESS
DX_FORMAT_X32_TYPELESS_G8X24_UINT	22	DXGI_FORMAT_X32_TYPELESS_G8X24_UINT

© 2000-2025, MetaQuotes Ltd.

ID	Wert	Passt in DXGI_FORMAT
DX_FORMAT_R10G10B10A2_TYPELESS	23	DXGI_FORMAT_R10G10B10A2_TYPELESS
DX_FORMAT_R10G10B10A2_UNORM	24	DXGI_FORMAT_R10G10B10A2_UNORM
DX_FORMAT_R10G10B10A2_UINT	25	DXGI_FORMAT_R10G10B10A2_UINT
DX_FORMAT_R11G11B10_FLOAT	26	DXGI_FORMAT_R11G11B10_FLOAT
DX_FORMAT_R8G8B8A8_TYPELESS	27	DXGI_FORMAT_R8G8B8A8_TYPELESS
DX_FORMAT_R8G8B8A8_UNORM	28	DXGI_FORMAT_R8G8B8A8_UNORM
DX_FORMAT_R8G8B8A8_UNORM_SRGB	29	DXGI_FORMAT_R8G8B8A8_UNORM_SRGB
DX_FORMAT_R8G8B8A8_UINT	30	DXGI_FORMAT_R8G8B8A8_UINT
DX_FORMAT_R8G8B8A8_SNORM	31	DXGI_FORMAT_R8G8B8A8_SNORM
DX_FORMAT_R8G8B8A8_SINT	32	DXGI_FORMAT_R8G8B8A8_SINT
DX_FORMAT_R16G16_TYPELESS	33	DXGI_FORMAT_R16G16_TYPELESS
DX_FORMAT_R16G16_FLOAT	34	DXGI_FORMAT_R16G16_FLOAT
DX_FORMAT_R16G16_UNORM	35	DXGI_FORMAT_R16G16_UNORM
DX_FORMAT_R16G16_UINT	36	DXGI_FORMAT_R16G16_UINT
DX_FORMAT_R16G16_SNORM	37	DXGI_FORMAT_R16G16_SNORM
DX_FORMAT_R16G16_SINT	38	DXGI_FORMAT_R16G16_SINT
DX_FORMAT_R32_TYPELESS	39	DXGI_FORMAT_R32_TYPELESS
DX_FORMAT_D32_FLOAT	40	DXGI_FORMAT_D32_FLOAT
DX_FORMAT_R32_FLOAT	41	DXGI_FORMAT_R32_FLOAT
DX_FORMAT_R32_UINT	42	DXGI_FORMAT_R32_UINT
DX_FORMAT_R32_SINT	43	DXGI_FORMAT_R32_SINT
DX_FORMAT_R24G8_TYPELESS	44	DXGI_FORMAT_R24G8_TYPELESS
DX_FORMAT_D24_UNORM_S8_UINT	45	DXGI_FORMAT_D24_UNORM_S8_UINT
DX_FORMAT_R24_UNORM_X8_TYPELESS	46	DXGI_FORMAT_R24_UNORM_X8_TYPELESS
DX_FORMAT_X24_TYPELESS_G8_UINT	47	DXGI_FORMAT_X24_TYPELESS_G8_UINT
DX_FORMAT_R8G8_TYPELESS	48	DXGI_FORMAT_R8G8_TYPELESS
DX_FORMAT_R8G8_UNORM	49	DXGI_FORMAT_R8G8_UNORM
DX_FORMAT_R8G8_UINT	50	DXGI_FORMAT_R8G8_UINT

© 2000-2025, MetaQuotes Ltd.

ID	Wert	Passt in DXGI_FORMAT
DX_FORMAT_R8G8_SNORM	51	DXGI_FORMAT_R8G8_SNORM
DX_FORMAT_R8G8_SINT	52	DXGI_FORMAT_R8G8_SINT
DX_FORMAT_R16_TYPELESS	53	DXGI_FORMAT_R16_TYPELESS
DX_FORMAT_R16_FLOAT	54	DXGI_FORMAT_R16_FLOAT
DX_FORMAT_D16_UNORM	55	DXGI_FORMAT_D16_UNORM
DX_FORMAT_R16_UNORM	56	DXGI_FORMAT_R16_UNORM
DX_FORMAT_R16_UINT	57	DXGI_FORMAT_R16_UINT
DX_FORMAT_R16_SNORM	58	DXGI_FORMAT_R16_SNORM
DX_FORMAT_R16_SINT	59	DXGI_FORMAT_R16_SINT
DX_FORMAT_R8_TYPELESS	60	DXGI_FORMAT_R8_TYPELESS
DX_FORMAT_R8_UNORM	61	DXGI_FORMAT_R8_UNORM
DX_FORMAT_R8_UINT	62	DXGI_FORMAT_R8_UINT
DX_FORMAT_R8_SNORM	63	DXGI_FORMAT_R8_SNORM
DX_FORMAT_R8_SINT	64	DXGI_FORMAT_R8_SINT
DX_FORMAT_A8_UNORM	65	DXGI_FORMAT_A8_UNORM
DX_FORMAT_R1_UNORM	66	DXGI_FORMAT_R1_UNORM
DX_FORMAT_R9G9B9E5_SHAREDEXP	67	DXGI_FORMAT_R9G9B9E5_SHAREDEXP
DX_FORMAT_R8G8_B8G8_UNORM	68	DXGI_FORMAT_R8G8_B8G8_UNORM
DX_FORMAT_G8R8_G8B8_UNORM	69	DXGI_FORMAT_G8R8_G8B8_UNORM
DX_FORMAT_BC1_TYPELESS	70	DXGI_FORMAT_BC1_TYPELESS
DX_FORMAT_BC1_UNORM	71	DXGI_FORMAT_BC1_UNORM
DX_FORMAT_BC1_UNORM_SRGB	72	DXGI_FORMAT_BC1_UNORM_SRGB
DX_FORMAT_BC2_TYPELESS	73	DXGI_FORMAT_BC2_TYPELESS
DX_FORMAT_BC2_UNORM	74	DXGI_FORMAT_BC2_UNORM
DX_FORMAT_BC2_UNORM_SRGB	75	DXGI_FORMAT_BC2_UNORM_SRGB
DX_FORMAT_BC3_TYPELESS	76	DXGI_FORMAT_BC3_TYPELESS
DX_FORMAT_BC3_UNORM	77	DXGI_FORMAT_BC3_UNORM
DX_FORMAT_BC3_UNORM_SRGB	78	DXGI_FORMAT_BC3_UNORM_SRGB
DX_FORMAT_BC4_TYPELESS	79	DXGI_FORMAT_BC4_TYPELESS
DX_FORMAT_BC4_UNORM	80	DXGI_FORMAT_BC4_UNORM
DX_FORMAT_BC4_SNORM	81	DXGI_FORMAT_BC4_SNORM

© 2000-2025, MetaQuotes Ltd.

ID	Wert	Passt in DXGI_FORMAT
DX_FORMAT_BC5_TYPELESS	82	DXGI_FORMAT_BC5_TYPELESS
DX_FORMAT_BC5_UNORM	83	DXGI_FORMAT_BC5_UNORM
DX_FORMAT_BC5_SNORM	84	DXGI_FORMAT_BC5_SNORM
DX_FORMAT_B5G6R5_UNORM	85	DXGI_FORMAT_B5G6R5_UNORM
DX_FORMAT_B5G5R5A1_UNORM	86	DXGI_FORMAT_B5G5R5A1_UNORM
DX_FORMAT_B8G8R8A8_UNORM	87	DXGI_FORMAT_B8G8R8A8_UNORM
DX_FORMAT_B8G8R8X8_UNORM	88	DXGI_FORMAT_B8G8R8X8_UNORM
DX_FORMAT_R10G10B10_XR_BIAS_A2_UNORM	89	DXGI_FORMAT_R10G10B10_XR_BIAS_A2_UNORM
DX_FORMAT_B8G8R8A8_TYPELESS	90	DXGI_FORMAT_B8G8R8A8_TYPELESS
DX_FORMAT_B8G8R8A8_UNORM_SRGB	91	DXGI_FORMAT_B8G8R8A8_UNORM_SRGB
DX_FORMAT_B8G8R8X8_TYPELESS	92	DXGI_FORMAT_B8G8R8X8_TYPELESS
DX_FORMAT_B8G8R8X8_UNORM_SRGB	93	DXGI_FORMAT_B8G8R8X8_UNORM_SRGB
DX_FORMAT_BC6H_TYPELESS	94	DXGI_FORMAT_BC6H_TYPELESS
DX_FORMAT_BC6H_UF16	95	DXGI_FORMAT_BC6H_UF16
DX_FORMAT_BC6H_SF16	96	DXGI_FORMAT_BC6H_SF16
DX_FORMAT_BC7_TYPELESS	97	DXGI_FORMAT_BC7_TYPELESS
DX_FORMAT_BC7_UNORM	98	DXGI_FORMAT_BC7_UNORM
DX_FORMAT_BC7_UNORM_SRGB	99	DXGI_FORMAT_BC7_UNORM_SRGB
DX_FORMAT_AYUV	100	DXGI_FORMAT_AYUV
DX_FORMAT_Y410	101	DXGI_FORMAT_Y410
DX_FORMAT_Y416	102	DXGI_FORMAT_Y416
DX_FORMAT_NV12	103	DXGI_FORMAT_NV12
DX_FORMAT_P010	104	DXGI_FORMAT_P010
DX_FORMAT_P016	105	DXGI_FORMAT_P016
DX_FORMAT_420_OPAQUE	106	DXGI_FORMAT_420_OPAQUE
DX_FORMAT_YUY2	107	DXGI_FORMAT_YUY2
DX_FORMAT_Y210	108	DXGI_FORMAT_Y210
DX_FORMAT_Y216	109	DXGI_FORMAT_Y216
DX_FORMAT_NV11	110	DXGI_FORMAT_NV11

© 2000-2025, MetaQuotes Ltd.

ID	Wert	Passt in DXGI_FORMAT
DX_FORMAT_AI44	111	DXGI_FORMAT_AI44
DX_FORMAT_IA44	112	DXGI_FORMAT_IA44
DX_FORMAT_P8	113	DXGI_FORMAT_P8
DX_FORMAT_A8P8	114	DXGI_FORMAT_A8P8
DX_FORMAT_B4G4R4A4_UNORM	115	DXGI_FORMAT_B4G4R4A4_UNORM
DX_FORMAT_P208	130	DXGI_FORMAT_P208
DX_FORMAT_V208	131	DXGI_FORMAT_V208
DX_FORMAT_V408	132	DXGI_FORMAT_V408
DX_FORMAT_FORCE_UINT	0xffffffff	DXGI_FORMAT_FORCE_UINT

© 2000-2025, MetaQuotes Ltd.

DXInputCreate

Erstellen der Eingaben des Shaders.

```
int  DXInputCreate(
   int   context,        // Handle des grafischen Kontextes
   uint  input_size      // Größe der Eingaben in Bytes
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

input_size

[in] Größe in Bytes der Parameterstruktur.

Rückgabewert

Das Handle der Eingaben des Shaders oder INVALID_HANDLE im Fehlerfall. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Ein erstelltes Handle, das nicht mehr verwendet wird, sollte explizit durch die Funktion DXRelease() freigegeben werden.

© 2000-2025, MetaQuotes Ltd.

DXInputSet

Setzt die Eingaben des Shaders.

```
bool  DXInputSet(
   int           input,      // Handle des grafischen Kontextes
   const void&   data        // Daten der Eingaben
   );
```

Parameter

input

[in] Handle der Eingaben für den Shader, das von DXInputCreate() erstellt wurde.

data

[in] Daten der Eingaben zur Einstellung des Shaders.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

© 2000-2025, MetaQuotes Ltd.

DXShaderCreate

Erstellt einen Shader eines bestimmten Typs.

```
int  DXShaderCreate(
   int                  context,          // Handle des grafischen Kontextes
   ENUM_DX_SHADER_TYPE  shader_type,      // Typ des Shaders
   const string         source,           // Quellcode des Shader
   const string         entry_point,      // Einstiegspunkt
   string&              compile_error     // Zeichenkette für Kompilernachricht
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

shader_type

[out] Der Wert der Enumeration ENUM_DX_SHADER_TYPE.

source

[in] Quellcode des Shaders in HLSL 5.

entry_point

[in] Einstiegspunkt - Funktionsname im Quellcode.

compile_error

[in] Zeichenketten, denen die Kompilerfehler zugewiesen werden.

Rückgabewert

Handle des Shaders oder INVALID_HANDLE im Fehlerfall. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Ein erstelltes Handle, das nicht mehr verwendet wird, sollte explizit durch die Funktion DXRelease() freigegeben werden.

ENUM_DX_SHADER_TYPE

ID	Wert	Beschreibung
DX_SHADER_VERTEX	0	Vertex shader
DX_SHADER_GEOMETRY	1	Geometry shader
DX_SHADER_PIXEL	2	Pixel shader

© 2000-2025, MetaQuotes Ltd.

DXShaderSetLayout

Legt das Vertex-Layout für den Vertex-Shader fest.

```
bool  DXShaderSetLayout(
   int                    shader,       // Handle des Shaders
   const DXVertexLayout&  layout[]      //
   );
```

Parameter

shader

[in] Handle des Vertex-Shaders erstellt in DXShaderCreate().

layout[]

[in] Array der Vertex-Feldbeschreibungen. Die Beschreibungen werden durch die Struktur DXVertexLayout festgelegt:

```
struct DXVertexLayout
  {
   string         semantic_name;       // Die Semantik von HLSL assoziiert mit die
   uint           semantic_index;      // Der Index für das Semantik-Element. Ein
   ENUM_DX_FORMAT format; /// Der Datentyp der Elementdaten.
  };
```

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Das Layout sollte mit dem Typ der Knoten in einem angegebenen Knotenpuffer übereinstimmen. Es sollte auch mit der Art der Eingabe von Knoten übereinstimmen, die an der Einstiegsstelle im Vertex-Shader-Code verwendet werden.

Der Vertex-Puffer für einen Shader wird in DXBufferSet() eingestellt.

Die DXVertexLayout-Struktur ist eine Version der MSDN-Struktur D3D11_INPUT_ELEMENT_DESC.

© 2000-2025, MetaQuotes Ltd.

DXShaderInputsSet

Setzt die Eingaben des Shaders.

```
bool  DXShaderInputsSet(
   int          shader,        // Handle des Shaders
   const int&   inputs[]       // Handle des Arrays der Eingaben
   );
```

Parameter

shader

[in] Handle des Shader erstellt in DXShaderCreate().

inputs[]

[in] Handle des Arrays der Eingabenof, erstellt mit DXInputCreate().

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Größe der Eingaben sollte gleich cbuffer, der Anzahl der Objekte, sein, die im Shader-Code angegeben sind.

© 2000-2025, MetaQuotes Ltd.

DXShaderTexturesSet

Legt die Texturen des Shaders fest

```
bool  DXShaderTexturesSet(
   int          shader,         // Handle des Shaders
   const  int&  textures[]      // Array der Handles der Strukturen
   );
```

Parameter

shader

[in] Handle des Shader erstellt in DXShaderCreate().

textures[]

[in] Array der Handles der Texturen, die mitDXTextureCreate() erstellt wurden.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Größe des Textur-Arrays sollte gleich der Anzahl der im Shader-Code deklarierten Objekte Texture2D sein.

© 2000-2025, MetaQuotes Ltd.

DXDraw

Rendert die Eckpunkte des in DXBufferSet() gesetzten Vertex-Puffers.

```
bool  DXDraw(
   int   context,               // Handle des grafischen Kontextes
   uint  start=0,               // Erster Vertexindex
   uint  count=WHOLE_ARRAY      // Vertex-Anzahl
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

start=0

[in] Index des ersten Vertex für das Rendern.

count=WHOLE_ARRAY

[in] Vertex-Anzahl, die zu rendern sind.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Shaders sollten vorher mit DXShaderSet() festgelegt werden, für das Rendern der Vertexe.

© 2000-2025, MetaQuotes Ltd.

DXDrawIndexed

Rendert grafische Primitive, die durch den Indexpuffer von DXBufferSet() beschrieben werden.

```
bool  DXDrawIndexed(
   int   context,               // Handle des grafischen Kontextes
   uint  start=0,               // Erstindex der Primitiven
   uint  count=WHOLE_ARRAY      // Primitiven-Anzahl
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

Rstart

[in] Index des ersten Primitiven für das Rendern.

count

[in] Primitiven-Anzahl, die zu rendern sind.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Art der Primitive, die durch den Indexpuffer beschrieben werden, wird über DXPrimiveTopologySet() eingestellt.

Der Vertex-Puffer in DXBufferSet() sollte vorläufig so eingestellt werden, dass er Primitive rendert.

Auchder Shader sollten vorher mit DXShaderSet() eingestellt werden.

© 2000-2025, MetaQuotes Ltd.

DXPrimiveTopologySet

Setzt den Typ der Primitive für das Rendern mit DXDrawIndexed().

```
bool  DXPrimiveTopologySet(
   int                          context,               // Handle des grafischen Kontex
   ENUM_DX_PRIMITIVE_TOPOLOGY   primitive_topology     // Typ der Primitiven
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

primitive_topology

[in] Der Wert der Enumeration ENUM_DX_PRIMITIVE_TOPOLOGY.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

ENUM_DX_PRIMITIVE_TOPOLOGY

ID	Wert	Passt in D3D11_PRIMITIVE_TOPOLOGY
DX_PRIMITIVE_TOPOLOGY_POINTLIST	1	D3D11_PRIMITIVE_TOPOLOGY_POINTLIST
DX_PRIMITIVE_TOPOLOGY_LINELIST	2	D3D11_PRIMITIVE_TOPOLOGY_LINELIST
DX_PRIMITIVE_TOPOLOGY_LINESTRIP	3	D3D11_PRIMITIVE_TOPOLOGY_LINESTRIP
DX_PRIMITIVE_TOPOLOGY_TRIANGLELIST	4	D3D11_PRIMITIVE_TOPOLOGY_TRIANGLELIST
DX_PRIMITIVE_TOPOLOGY_TRIANGLESTRIP	5	D3D11_PRIMITIVE_TOPOLOGY_TRIANGLESTRIP
DX_PRIMITIVE_TOPOLOGY_LINELIST_ADJ	6	D3D11_PRIMITIVE_TOPOLOGY_LINELIST_ADJ
DX_PRIMITIVE_TOPOLOGY_LINESTRIP_ADJ	7	D3D11_PRIMITIVE_TOPOLOGY_LINESTRIP_ADJ
DX_PRIMITIVE_TOPOLOGY_TRIANGLELIST_ADJ	8	D3D11_PRIMITIVE_TOPOLOGY_TRIANGLELIST_ADJ
DX_PRIMITIVE_TOPOLOGY_TRIANGLESTRIP_ADJ	9	D3D11_PRIMITIVE_TOPOLOGY_TRIANGLESTRIP_ADJ

© 2000-2025, MetaQuotes Ltd.

DXBufferSet

Setzt einen Puffer für das aktuelle Rendering.

```
bool  DXBufferSet(
   int   context,               // Handle des grafischen Kontextes
   int   buffer,                // Handle des Vertex- oder Index-Puffers
   uint  start=0,               // Startindex
   uint  count=WHOLE_ARRAY      // Anzahl der Elemente
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

buffer

[in] Handle des Vertex- oder Index-Puffers erstellt von DXBufferCreate().

start=0

[in] Index des ersten Elements des Puffers. Standardmäßig werden die Daten vom Anfang des Arrays gelesen.

count=WHOLE_ARRAY

[in] Anzahl der zu verwendenden Elemente. Standardmäßig werden alle Werte des Puffers verwendet.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Die Funktion DXBufferSet() sollte aufgerufen werden, um die Vertex- oder Index-Puffer mit der Funktion DXDraw() zu rendern.

© 2000-2025, MetaQuotes Ltd.

DXShaderSet

Setzt einen Shader für das Rendern.

```
bool  DXShaderSet(
   int  context,      // Handle des grafischen Kontextes
   int  shader        // Handle des Shaders
   );
```

Parameter

context

[in] Handle des grafischen Kontextes, das mit DXContextCreate() erstellt wurde.

shader

[in] Handle des Shader erstellt in DXShaderCreate().

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Für das Rendern können mehrere Arten von Shadern gleichzeitig verwendet werden (Vertex, Geometrie und Pixel).

© 2000-2025, MetaQuotes Ltd.

DXHandleType

Liefert einen Handle-Typ zurück.

```
ENUM_DX_HANDLE_TYPE  DXHandleType(
   int  handle      // Handle
   );
```

Parameter

handle

[in] Handle.

Rückgabewert

Der Wert der Enumeration ENUM_DX_HANDLE_TYPE

ENUM_DX_HANDLE_TYPE

ID	Wert	Beschreibung
DX_HANDLE_INVALID	0	Ungültiges Handle
DX_HANDLE_CONTEXT	1	Handle eines grafischen Kontextes
DX_HANDLE_SHADER	2	Handle eines Shaders
DX_HANDLE_BUFFER	3	Handle des Vertex- oder Index-Puffers
DX_HANDLE_INPUT	4	Handle der Eingaben des Shaders
DX_HANDLE_TEXTURE	5	Handle für Texture

© 2000-2025, MetaQuotes Ltd.

DXRelease

Gibt ein Handle frei.

```
bool  DXRelease(
   int  handle      // Handle
   );
```

Parameter

context

[in] das freizugebende Handle.

Rückgabewert

Im Falle einer erfolgreichen Ausführung wird 'true' zurückgegeben, andernfalls - 'false'. Um eine Fehlernummer zu erhalten, muss man die Funktion GetLastError() aufrufen.

Hinweis

Alle erstellten Handle, die nicht mehr verwendet werden, sollten explizit durch die Funktion DXRelease() freigegeben werden.

© 2000-2025, MetaQuotes Ltd.

Das MetaTrader-Modul für die Integration mit Python

MQL5 wurde für die Entwicklung von leistungsstarken Handelsanwendungen an den Finanzmärkten entwickelt und ist beispiellos unter anderen spezialisierten Sprachen, die im algorithmischen Handel eingesetzt werden. Die Syntax und Geschwindigkeit von MQL5-Programmen sind so nah wie möglich an C++, es gibt Unterstützung für OpenCL und Integration mit MS Visual Studio. Die Bibliotheken Statistics, fuzzy logic und ALGLIB sind ebenfalls verfügbar. Die MetaEditor-Entwicklungsumgebung bietet native Unterstützung für .NET-Bibliotheken mit "intelligentem" Funktionsimport, wodurch die Entwicklung spezieller Wrapper entfällt. C++-DLLs von Drittanbietern können ebenfalls verwendet werden. C++ Quellcodedateien (CPP und H) können direkt aus dem Editor heraus bearbeitet und in eine DLL kompiliert werden. Microsoft Visual Studio, das auf dem PC des Benutzers installiert ist, kann dafür verwendet werden.

Python ist eine moderne High-Level-Programmiersprache zur Entwicklung von Skripten und Anwendungen. Es enthält mehrere Bibliotheken für maschinelles Lernen, Prozessautomatisierung sowie Datenanalyse und Visualisierung.

Das MetaTrader-Paket für Python wurde für das komfortable und schnelle Abrufen von Austauschdaten über das Interprozessorkommunikation direkt vom MetaTrader 5 Terminal aus entwickelt. Die so erhaltenen Daten können für statistische Berechnungen und maschinelles Lernen weiterverwendet werden.

Installieren des Pakets von der Kommandozeile:

```
pip install MetaTrader5
```

Updating the package from the command line:

```
pip install --upgrade MetaTrader5
```

Funktionen zur Integration von MetaTrader 5 und Python

Funktion	Aktion
initialize	Stellt eine Verbindung mit dem MetaTrader 5 Terminal her
login	Verbinden mit einem Handelskonto mit den angegebenen Parametern
shutdown	Schließt die vorher hergestellte Verbindung zu MetaTrader 5 Terminal wieder
version	Rückgabe der Version des MetaTrader 5 Terminals
last_error	Datenrückgabe des letzten Fehlers
account_info	Informationsabruf des aktuellen Handelskontos
terminal_Info	Abfrage des Status' und der Parameter des verbundenen MetaTrader 5 Terminals
symbols_total	Abrufen der Anzahl aller Finanzinstrumente des MetaTrader 5 Terminals
symbols_get	Abrufen aller Finanzinstrumente des MetaTrader 5 Terminals
symbol_info	Abrufen der Daten des angegebenen Finanzinstruments

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
symbol_info_tick	Abrufen des letzten Ticks des angegebenen Finanzinstruments
symbol_select	Auswahl eines Symbols im Fenster MarketWatch oder es aus demselben entfernen
market_book_add	Abonniert das MetaTrader 5-Terminal auf die Markttiefe-Änderungsereignisse für ein angegebenes Symbol
market_book_get	Gibt ein Tupel aus BookInfo mit den Einträgen der Markttiefe für das angegebene Symbol zurück
market_book_release	Beenden des Abonnements der Markttiefe des MetaTrader 5-Terminals für ein bestimmtes Symbol ab
copy_rates_from	Abrufen der Bars vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Datum
copy_rates_from_pos	Abrufen der Bars vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Index
copyrates_range	Abrufen der Bars der angegebenen Zeitspanne vom MetaTrader 5 Terminal
copy_ticks_from	Abrufen der Ticks vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Datum
copy_ticks_range	Abrufen der Ticks der angegebenen Zeitspanne vom MetaTrader 5 Terminal
orders_total	Abrufen der Anzahl der aktiven Orders.
orders_get	Abrufen der aktiven Orders mit der Fähigkeit nach Symbol oder Ticket zu filtern
order_calc_margin	Rückgabe der Marge in der Kontowährung zur Durchführung der angegebenen Handelsoperation
order_calc_profit	Rückgabe des Gewinns in Kontowährung für die angegebene Handelsoperation
order_check	Prüfen, ob die Geldmittel ausreichen für die angestrebte Handelsoperation
order_send	Senden einer Anfrage zur Durchführung einer Handelsoperation.
positions_total	Abrufen der Anzahl der offenen Positionen
positions_get	Abrufen der offenen Positionen, mit der Fähigkeit nach Symbol oder Ticket zu filtern
history_orders_total	Abrufen der Anzahl der Orders in der Handelshistorie innerhalb des angegebenen Intervalls
history_orders_get	Abrufen der Orders in der Handelshistorie, mit der Fähigkeit nach Symbol oder Ticket zu filtern

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion
history_deals_total	Abrufen der Anzahl der Deals in der Handelshistorie innerhalb des angegebenen Intervalls
history_deals_get	Abrufen der Deals in der Handelshistorie, mit der Fähigkeit nach Symbol oder Ticket zu filtern

Beispiel einer Verbindung von Python zum MetaTrader 5

1. Holen Sie sich die letzte Version von Python 3.8 von https://www.python.org/downloads/windows
2. Beim Installieren von Python prüfen Sie "Add Python 3.8 to PATH%", damit Sie die Python-Skripts aus der Kommandozeile heraus starten können.
3. Installieren des MetaTrader 5 Moduls aus der Kommandozeile

```
pip install MetaTrader5
```

4. Hinzufügen der Pakete von matplotlib und pandas

```
pip install matplotlib
pip install pandas
```

5. Starten des Testskripts

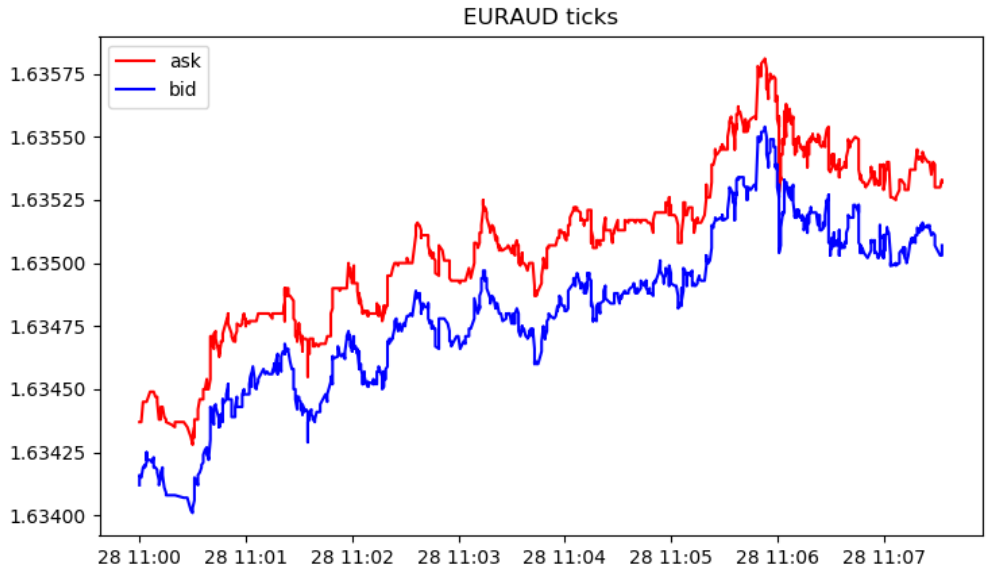
```
from datetime import datetime
import matplotlib.pyplot as plt
import pandas as pd
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()
import MetaTrader5 as mt5

# mit MetaTrader 5 verbinden
if not mt5.initialize():
    print("initialize() failed")
    mt5.shutdown()

# Abfrage des Status und der Parameter der Verbindung
print(mt5.terminal_info())
# Abrufen der Version des MetaTrader 5
print(mt5.version())

# Abrufen von 1000 Ticks von EURAUD
euraud_ticks = mt5.copy_ticks_from("EURAUD", datetime(2020,1,28,13), 1000, mt5.COPY_TI
# Abrufen von Ticks von AUDUSD zwischen 2019.04.01 13:00 - 2019.04.02 13:00
audusd_ticks = mt5.copy_ticks_range("AUDUSD", datetime(2020,1,27,13), datetime(2020,1,

# Abrufen der Bars eines anderen Symbols auf verschiedenen Wegen
eurusd_rates = mt5.copy_rates_from("EURUSD", mt5.TIMEFRAME_M1, datetime(2020,1,28,13),
eurgbp_rates = mt5.copy_rates_from_pos("EURGBP", mt5.TIMEFRAME_M1, 0, 1000)
eurcad_rates = mt5.copy_rates_range("EURCAD", mt5.TIMEFRAME_M1, datetime(2020,1,27,13)

# Schließen der Verbindung zum MetaTrader 5
```

© 2000-2025, MetaQuotes Ltd.

```
mt5.shutdown()

# Daten
print('euraud_ticks(', len(euraud_ticks), ')')
for val in euraud_ticks[:10]: print(val)

print('audusd_ticks(', len(audusd_ticks), ')')
for val in audusd_ticks[:10]: print(val)

print('eurusd_rates(', len(eurusd_rates), ')')
for val in eurusd_rates[:10]: print(val)

print('eurgbp_rates(', len(eurgbp_rates), ')')
for val in eurgbp_rates[:10]: print(val)

print('eurcad_rates(', len(eurcad_rates), ')')
for val in eurcad_rates[:10]: print(val)

#ANZEIGEN
# Erstellen von DataFrame aus den erhaltenen Daten
ticks_frame = pd.DataFrame(euraud_ticks)
# Konvertieren der Zeit in Sekunden im Datumsformat
ticks_frame['time']=pd.to_datetime(ticks_frame['time'], unit='s')
# Ticks anzeigen auf dem Chart
plt.plot(ticks_frame['time'], ticks_frame['ask'], 'r-', label='ask')
plt.plot(ticks_frame['time'], ticks_frame['bid'], 'b-', label='bid')

# Anzeigen der Legende
plt.legend(loc='upper left')

# Hinzufügen des Headers
plt.title('EURAUD ticks')

# Anzeigen des Charts
plt.show()
```

© 2000-2025, MetaQuotes Ltd.

6. Abrufen der Daten und des Charts




```
[2, 'MetaQuotes-Demo', '16167573']
[500, 2325, '19 Feb 2020']

euraud_ticks( 1000 )
(1580209200, 1.63412, 1.63437, 0., 0, 1580209200067, 130, 0.)
(1580209200, 1.63416, 1.63437, 0., 0, 1580209200785, 130, 0.)
(1580209201, 1.63415, 1.63437, 0., 0, 1580209201980, 130, 0.)
(1580209202, 1.63419, 1.63445, 0., 0, 1580209202192, 134, 0.)
(1580209203, 1.6342, 1.63445, 0., 0, 1580209203004, 130, 0.)
(1580209203, 1.63419, 1.63445, 0., 0, 1580209203487, 130, 0.)
(1580209203, 1.6342, 1.63445, 0., 0, 1580209203694, 130, 0.)
(1580209203, 1.63419, 1.63445, 0., 0, 1580209203990, 130, 0.)
(1580209204, 1.63421, 1.63445, 0., 0, 1580209204194, 130, 0.)
(1580209204, 1.63425, 1.63445, 0., 0, 1580209204392, 130, 0.)
audusd_ticks( 40449 )
(1580122800, 0.67858, 0.67868, 0., 0, 1580122800244, 130, 0.)
(1580122800, 0.67858, 0.67867, 0., 0, 1580122800429, 4, 0.)
(1580122800, 0.67858, 0.67865, 0., 0, 1580122800817, 4, 0.)
(1580122801, 0.67858, 0.67866, 0., 0, 1580122801618, 4, 0.)
(1580122802, 0.67858, 0.67865, 0., 0, 1580122802928, 4, 0.)
(1580122809, 0.67855, 0.67865, 0., 0, 1580122809526, 130, 0.)
(1580122809, 0.67855, 0.67864, 0., 0, 1580122809699, 4, 0.)
(1580122813, 0.67855, 0.67863, 0., 0, 1580122813576, 4, 0.)
(1580122815, 0.67856, 0.67863, 0., 0, 1580122815190, 130, 0.)
(1580122815, 0.67855, 0.67863, 0., 0, 1580122815479, 130, 0.)
eurusd_rates( 1000 )
(1580149260, 1.10132, 1.10151, 1.10131, 1.10149, 44, 1, 0)
(1580149320, 1.10149, 1.10161, 1.10143, 1.10154, 42, 1, 0)
(1580149380, 1.10154, 1.10176, 1.10154, 1.10174, 40, 2, 0)
(1580149440, 1.10174, 1.10189, 1.10168, 1.10187, 47, 1, 0)
```

© 2000-2025, MetaQuotes Ltd.

```
(1580149500, 1.10185, 1.10191, 1.1018, 1.10182, 53, 1, 0)
(1580149560, 1.10182, 1.10184, 1.10176, 1.10183, 25, 3, 0)
(1580149620, 1.10183, 1.10187, 1.10177, 1.10187, 49, 2, 0)
(1580149680, 1.10187, 1.1019, 1.1018, 1.10187, 53, 1, 0)
(1580149740, 1.10187, 1.10202, 1.10187, 1.10198, 28, 2, 0)
(1580149800, 1.10198, 1.10198, 1.10183, 1.10188, 39, 2, 0)
eurgbp_rates( 1000 )
(1582236360, 0.83767, 0.83767, 0.83764, 0.83765, 23, 9, 0)
(1582236420, 0.83765, 0.83765, 0.83764, 0.83765, 15, 8, 0)
(1582236480, 0.83765, 0.83766, 0.83762, 0.83765, 19, 7, 0)
(1582236540, 0.83765, 0.83768, 0.83758, 0.83763, 39, 6, 0)
(1582236600, 0.83763, 0.83768, 0.83763, 0.83767, 21, 6, 0)
(1582236660, 0.83767, 0.83775, 0.83765, 0.83769, 63, 5, 0)
(1582236720, 0.83769, 0.8377, 0.83758, 0.83764, 40, 7, 0)
(1582236780, 0.83766, 0.83769, 0.8376, 0.83766, 37, 6, 0)
(1582236840, 0.83766, 0.83772, 0.83763, 0.83772, 22, 6, 0)
(1582236900, 0.83772, 0.83773, 0.83768, 0.8377, 36, 5, 0)
eurcad_rates( 1441 )
(1580122800, 1.45321, 1.45329, 1.4526, 1.4528, 146, 15, 0)
(1580122860, 1.4528, 1.45315, 1.45274, 1.45301, 93, 15, 0)
(1580122920, 1.453, 1.45304, 1.45264, 1.45264, 82, 15, 0)
(1580122980, 1.45263, 1.45279, 1.45231, 1.45277, 109, 15, 0)
(1580123040, 1.45275, 1.4528, 1.45259, 1.45271, 53, 14, 0)
(1580123100, 1.45273, 1.45285, 1.45269, 1.4528, 62, 16, 0)
(1580123160, 1.4528, 1.45284, 1.45267, 1.45282, 64, 14, 0)
(1580123220, 1.45282, 1.45299, 1.45261, 1.45272, 48, 14, 0)
(1580123280, 1.45272, 1.45275, 1.45255, 1.45275, 74, 14, 0)
(1580123340, 1.45275, 1.4528, 1.4526, 1.4528, 94, 13, 0)
```

© 2000-2025, MetaQuotes Ltd.

initialize

Stellt eine Verbindung mit dem MetaTrader 5 Terminal her. Es gibt drei Aufrufoptionen.

Aufruf ohne Parameter. Das Terminal für die Verbindung wird automatisch gefunden.

```
initialize()
```

Aufruf mit angegebenen Pfad zum MetaTrader 5 Terminal, mit dem sich verbunden werden soll.

```
initialize(
   path                       // Pfad zur EXE-Date des MetaTrader 5 Terminals
   )
```

Aufruf mit angegebenen Handelskonto Pfad und Parametern.

```
initialize(
   path,                      // Pfad zur EXE-Date des MetaTrader 5 Terminals
   login=LOGIN,               // Kontonummer
   password="PASSWORD",       // Passwort
   server="SERVER",           // Servername, wie er im Terminal angegeben wurde
   timeout=TIMEOUT,           // Timeout
   portable=False             // Portable-Mode
   )
```

Parameter

path

[in] Pfad zur Datei metatrader.exe oder metatrader64.exe. Optionale unbenannte Parameter. Er wird zunächst ohne Parameternamen angegeben. Wird der Pfad nicht angegeben, versucht das Modul die ausführbaren Dateien selber zu finden.

login=LOGIN

[in] Nummer des Handelskontos. Optionale benannte Parameter. Falls nicht angegeben wird das letzte Handelskonto verwendet.

password="PASSWORD"

[in] Passwort des Handelskontos. Optionale benannte Parameter. Wenn das Passwort nicht festgelegt ist, wird automatisch das in der Terminal-Datenbank gespeicherte Passwort für ein angegebenes Handelskonto angewendet.

server="SERVER"

[in] Name des Handelsservers. Optionale benannte Parameter. Wenn der Server nicht angegebenen ist, wird automatisch der in der Terminal-Datenbank gespeicherte Server für ein bestimmtes Handelskonto verwendet.

timeout=TIMEOUT

[in] Timeout der Verbindung im Millisekunden. Optionale benannte Parameter. Falls nicht angegeben wir ein Wert von 60 000 (60 Sekunden) verwendet.

portable=False

[in] Flag des Terminals, das im Portable-Mode gestartet wurde. Optionale benannte Parameter. Falls nicht angegeben wir der Wert False verwendet.

© 2000-2025, MetaQuotes Ltd.

Rückgabewert

'True' im Falle einer erfolgreichen Verbindung zum MetaTrader 5 Terminal - sonst 'False'.

Hinweis

Falls notwendig wird das MetaTrader 5 Terminal gestartet, um den Aufruf von initialize() auszuführen.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Herstellen der Verbindung zum angegebenen Handelskonto im MetaTrader 5
if not mt5.initialize(login=25115284, server="MetaQuotes-Demo",password="4zatlbqx"):
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Anzeige des Status', des Servernamens und des Handelskontos
print(mt5.terminal_info())
# Anzeige der Version des MetaTrader 5
print(mt5.version())

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

Siehe auch

shutdown, terminal_info, version

© 2000-2025, MetaQuotes Ltd.

login

Verbinden mit einem Handelskonto mittels der angegebenen Parameter.

```
login(
   login,                    // Kontonummer
   password="PASSWORD",      // Passwort
   server="SERVER",          // Servername wie im Terminal angegeben
   timeout=TIMEOUT           // Timeout
   )
```

Parameter

login

[in] Nummer des Handelskontos. Benötigter unbenannter Parameter.

password

[in] Passwort des Handelskontos. Optionale benannte Parameter. Wenn das Passwort nicht angegeben wurde, wird automatisch das in der Datenbank des Terminals verwendet.

server

[in] Name des Handelsservers. Optionale benannte Parameter. Wenn der Server nicht angegeben wurde, wird automatisch der in der Datenbank des Terminals verwendet.

timeout=TIMEOUT

[in] Timeout der Verbindung im Millisekunden. Optionale benannte Parameter. Falls nicht angegeben wir ein Wert von 60 000 (60 Sekunden) verwendet. Wenn die Verbindung nicht innerhalb der angegebenen Zeit hergestellte werden konnte, wird der Aufruf zwangsweise beendet und ein Fehler erzeugt.

Rückgabewert

True falls mit dem Handelskonto verbunden, andernfalls - False.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Anzeige der Version des MetaTrader 5
print(mt5.version())
# Verbindung mit dem Handelskonto unter Angabe von Passwort und Server herstellen
account=17221085
authorized=mt5.login(account)  # das Passwort der Datenbank des Terminals wird verwend
if authorized:
    print("connected to account #{}".format(account))
```

© 2000-2025, MetaQuotes Ltd.

```
else:
    print("failed to connect at account #{}, error code: {}".format(account, mt5.last_

# jetzt verbinden mit einem anderen Handelskonto mittels eines Passworts
account=25115284
authorized=mt5.login(account, password="gqrtz0lbdm")
if authorized:
    # Anzeige der Daten des Handelskontos 'as is'
    print(mt5.account_info())
    # Anzeige der Daten des Handelskontos als Liste
    print("Show account_info()._asdict():")
    account_info_dict = mt5.account_info()._asdict()
    for prop in account_info_dict:
        print("  {}={}".format(prop, account_info_dict[prop]))
else:
    print("failed to connect at account #{}, error code: {}".format(account, mt5.last_

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()

Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
[500, 2367, '23 Mar 2020']

connected to account #17221085

connected to account #25115284
AccountInfo(login=25115284, trade_mode=0, leverage=100, limit_orders=200, margin_so_mo
account properties:
   login=25115284
   trade_mode=0
   leverage=100
   limit_orders=200
   margin_so_mode=0
   trade_allowed=True
   trade_expert=True
   margin_mode=2
   currency_digits=2
   fifo_close=False
   balance=99588.33
   credit=0.0
   profit=-45.23
   equity=99543.1
   margin=54.37
   margin_free=99488.73
   margin_level=183084.6054809638
   margin_so_call=50.0
```

© 2000-2025, MetaQuotes Ltd.

```
    margin_so_so=30.0
    margin_initial=0.0
    margin_maintenance=0.0
    assets=0.0
    liabilities=0.0
    commission_blocked=0.0
    name=James Smith
    server=MetaQuotes-Demo
    currency=USD
    company=MetaQuotes Software Corp.
```

Siehe auch

initialize, shutdown

© 2000-2025, MetaQuotes Ltd.

shutdown

Schließt die vorher hergestellte Verbindung zu MetaTrader 5 Terminal wieder.

```
shutdown()
```

Rückgabewert

Keiner.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed")
    quit()

# Anzeige des Status', des Servernamens und des Handelskontos
print(mt5.terminal_info())
# Anzeige der Version des MetaTrader 5
print(mt5.version())

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

Siehe auch

initialize, login_py, terminal_info, version

© 2000-2025, MetaQuotes Ltd.

version

Rückgabe der Version des MetaTrader 5 Terminals.

```
version()
```

Rückgabewert

Rückgabe der Version des MetaTrader 5 Terminals, das Datum von build und release. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion version() gibt die Version des Terminals, das Datum von build und release als Tupel dieser drei Werte zurück.

Typ	Beschreibung	Beispielwert
Ganzzahl (integer)	Version des MetaTrader 5 Terminals	500
Ganzzahl (integer)	Build	2007
string	Veröffentlichungszeitpunkt des Build	'25 Feb 2019'

Beispiel:

```
import MetaTrader5 as mt5
import pandas as pd
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Anzeige der Version des MetaTrader 5
print(mt5.version())

# Darstellung des Status', des Servernamens und des Handelskontos 'as is'.
print(mt5.terminal_info())
print()

# Abrufen der Eigenschaften als Liste
terminal_info_dict=mt5.terminal_info()._asdict()
# Konvertieren und Ausdrucken der Liste als DataFrame
df=pd.DataFrame(list(terminal_info_dict.items()),columns=['property','value'])
print("terminal_info() as dataframe:")
```

© 2000-2025, MetaQuotes Ltd.

```
print(df[:-1])

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()

Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
[500, 2367, '23 Mar 2020']
TerminalInfo(community_account=True, community_connection=True, connected=True, dlls_a

terminal_info() as dataframe:
                 property                          value
0       community_account                           True
1    community_connection                           True
2               connected                           True
3            dlls_allowed                          False
4           trade_allowed                          False
5       tradeapi_disabled                          False
6           email_enabled                          False
7             ftp_enabled                          False
8   notifications_enabled                          False
9                    mqid                          False
10                  build                           2367
11                maxbars                           5000
12               codepage                           1251
13             ping_last                          77881
14      community_balance                        707.107
15         retransmission                              0
16                company     MetaQuotes Software Corp.
17                   name                   MetaTrader 5
18               language                        Russian
19                   path  E:\ProgramFiles\MetaTrader 5
20              data_path  E:\ProgramFiles\MetaTrader 5
```

Siehe auch

initialize, shutdown, terminal_info

© 2000-2025, MetaQuotes Ltd.

last_error

Datenrückgabe des letzten Fehlers.

```
last_error()
```

Rückgabewert

Es wird die letzte Fehlernummer mit Beschreibung als Tupel zurückgegeben.

Hinweis

last_error() erlaubt denn Erhalt der Fehlernummer im Falle einer fehlgeschlagenen Ausführung einer Bibliotheksfunktion des MetaTrader 5. Sie ist ähnlich GetLastError(). Jedoch werden eigene Fehlernummern verwendet. Mögliche Werte:

Konstante	Wert	Beschreibung
RES_S_OK	1	allgemeiner Erfolg
RES_E_FAIL	-1	allgemeiner Fehler
RES_E_INVALID_PARAMS	-2	ungültige Argumente/Parameter
RES_E_NO_MEMORY	-3	kein ausreichender Speicher
RES_E_NOT_FOUND	-4	Keine Historie
RES_E_INVALID_VERSION	-5	ungültige Version
RES_E_AUTH_FAILED	-6	fehlgeschlagene Authentifizierung
RES_E_UNSUPPORTED	-7	nicht unterstütze Methode
RES_E_AUTO_TRADING_DISABLED	-8	Auto-Trading deaktiviert
RES_E_INTERNAL_FAIL	-10000	interner IPC, allgemeiner Fehler
RES_E_INTERNAL_FAIL_SEND	-10001	interner IPC, Senden ist fehlgeschlagen
RES_E_INTERNAL_FAIL_RECEIVE	-10002	interner IPC, Empfang fehlgeschlagen
RES_E_INTERNAL_FAIL_INIT	-10003	interner IPC, Initialisierung fehlgeschlagen
RES_E_INTERNAL_FAIL_CONNECT	-10003	interner IPC, kein ipc
RES_E_INTERNAL_FAIL_TIMEOUT	-10005	interner Timeout

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
```

© 2000-2025, MetaQuotes Ltd.

```
    quit()

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

Siehe auch

version, GetLastError

© 2000-2025, MetaQuotes Ltd.

account_info

Informationsabruf des aktuellen Handelskontos.

```
account_info()
```

Rückgabewert

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion gibt alle Daten in einem Aufruf zurück, die einzeln mit AccountInfoInteger, AccountInfoDouble und AccountInfoString erhalten werden könnten.

Beispiel:

```
import MetaTrader5 as mt5
import pandas as pd
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Verbindung mit dem Handelskonto unter Angabe von Passwort und Server herstellen
authorized=mt5.login(25115284, password="gqz0343lbdm")
if authorized:
    account_info=mt5.account_info()
    if account_info!=None:
        # Anzeige der Daten des Handelskontos 'as is'
        print(account_info)
        # Anzeige der Daten des Handelskontos als Liste
        print("Show account_info()._asdict():")
        account_info_dict = mt5.account_info()._asdict()
        for prop in account_info_dict:
            print("  {}={}".format(prop, account_info_dict[prop]))
        print()

       # Konvertieren und Ausdrucken der Liste als DataFrame
        df=pd.DataFrame(list(account_info_dict.items()),columns=['property','value'])
        print("account_info() as dataframe:")
        print(df)
else:
    print("failed to connect to trade account 25115284 with password=gqz0343lbdm, erro
```

© 2000-2025, MetaQuotes Ltd.

```
# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
AccountInfo(login=25115284, trade_mode=0, leverage=100, limit_orders=200, margin_so_mo
Show account_info()._asdict():
  login=25115284
  trade_mode=0
  leverage=100
  limit_orders=200
  margin_so_mode=0
  trade_allowed=True
  trade_expert=True
  margin_mode=2
  currency_digits=2
  fifo_close=False
  balance=99511.4
  credit=0.0
  profit=41.82
  equity=99553.22
  margin=98.18
  margin_free=99455.04
  margin_level=101398.67590140559
  margin_so_call=50.0
  margin_so_so=30.0
  margin_initial=0.0
  margin_maintenance=0.0
  assets=0.0
  liabilities=0.0
  commission_blocked=0.0
  server=MetaQuotes-Demo
  currency=USD
  company=MetaQuotes Software Corp.

account_info() as dataframe
              property                     value
0                login                  25115284
1           trade_mode                         0
2             leverage                       100
3         limit_orders                       200
4       margin_so_mode                         0
5        trade_allowed                      True
6         trade_expert                      True
7          margin_mode                         2
8      currency_digits                         2
9           fifo_close                     False
```

© 2000-2025, MetaQuotes Ltd.

```
10             balance                      99588.3
11              credit                            0
12              profit                       -45.13
13              equity                       99543.2
14              margin                         54.37
15         margin_free                      99488.8
16        margin_level                       183085
17      margin_so_call                           50
18        margin_so_so                           30
19      margin_initial                            0
20  margin_maintenance                            0
21              assets                            0
22         liabilities                            0
23  commission_blocked                            0
24                name                   James Smith
25              server               MetaQuotes-Demo
26            currency                          USD
27             company  MetaQuotes Software Corp.
```

Siehe auch

initialize, shutdown, login

© 2000-2025, MetaQuotes Ltd.

terminal_info

Abfragen des Verbindungsstatus zum MetaTrader 5 Terminal und den Einstellungen.

```
terminal_info()
```

Rückgabewert

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion gibt in einem Aufruf alle Daten zurück, die mit TerminalInfoInteger, TerminalInfoDouble und TerminalInfoDouble erhalten werden könnten.

Beispiel:

```
import MetaTrader5 as mt5
import pandas as pd
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Anzeige der Version des MetaTrader 5
print(mt5.version())
# Anzeige der Information über die Einstellungen und den Status des Terminals
terminal_info=mt5.terminal_info()
if terminal_info!=None:
    # Anzeige der Terminaldaten 'as is'
    print(terminal_info)
    # Anzeige der Daten des Handelskontos als Liste
    print("Show terminal_info()._asdict():")
    terminal_info_dict = mt5.terminal_info()._asdict()
    for prop in terminal_info_dict:
        print("  {}={}".format(prop, terminal_info_dict[prop]))
    print()
   # Konvertieren und Ausdrucken der Liste als DataFrame
    df=pd.DataFrame(list(terminal_info_dict.items()),columns=['property','value'])
    print("terminal_info() as dataframe:")
    print(df)
```

© 2000-2025, MetaQuotes Ltd.

```
# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
[500, 2366, '20 Mar 2020']
TerminalInfo(community_account=True, community_connection=True, connected=True,....
Show terminal_info()._asdict():
  community_account=True
  community_connection=True
  connected=True
  dlls_allowed=False
  trade_allowed=False
  tradeapi_disabled=False
  email_enabled=False
  ftp_enabled=False
  notifications_enabled=False
  mqid=False
  build=2366
  maxbars=5000
  codepage=1251
  ping_last=77850
  community_balance=707.10668201585
  retransmission=0.0
  company=MetaQuotes Software Corp.
  name=MetaTrader 5
  language=Russian
  path=E:\ProgramFiles\MetaTrader 5
  data_path=E:\ProgramFiles\MetaTrader 5
  commondata_path=C:\Users\Rosh\AppData\Roaming\MetaQuotes\Terminal\Common

terminal_info() as dataframe:
                  property                      value
0       community_account                       True
1    community_connection                       True
2               connected                       True
3            dlls_allowed                      False
4           trade_allowed                      False
5       tradeapi_disabled                      False
6           email_enabled                      False
7             ftp_enabled                      False
8   notifications_enabled                      False
9                    mqid                      False
10                  build                       2367
11                maxbars                       5000
12               codepage                       1251
13             ping_last                      80953
```

© 2000-2025, MetaQuotes Ltd.

```
14      community_balance                    707.107
15         retransmission                   0.063593
16                company  MetaQuotes Software Corp.
17                   name                MetaTrader 5
18               language                     Russian
```

Siehe auch

initialize, shutdown, version

© 2000-2025, MetaQuotes Ltd.

symbols_total

Abrufen der Anzahl aller Finanzinstrumente des MetaTrader 5 Terminals.

```
symbols_total()
```

Rückgabewert

Integer Wert

Hinweis

Die Funktion ist ähnlich SymbolsTotal. Sie gibt jedoch die Anzahl aller Symbole zurück, einschließlich der nutzerdefinierten und der im Fenster MarketWatch deaktivierten Symbole.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Anzahl der Finanzinstrumente
symbols=mt5.symbols_total()
if symbols>0:
    print("Total symbols =",symbols)
else:
    print("symbols not found")

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

Siehe auch

symbols_get, symbol_select, symbol_info

© 2000-2025, MetaQuotes Ltd.

symbols_get

Abrufen aller Finanzinstrumente des MetaTrader 5 Terminals.

```
symbols_get(
   group="GROUP"      // Filter zur Symbolauswahl
)
```

group="GROUP"

[in] Der Filter für die Gruppe der angeforderten Symbole. Optionaler Parameter. Wenn die Gruppe angegeben wurde, liefert die Funktion nur die Symbole, die das angegebene Kriterium erfüllen.

Rückgabewert

Rückgabe der Symbole als Tupel. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Der Parameter *group* erlaubt das Filtern der Deals nach Symbolen. '*' kann am Anfang und am Ende einer Zeichenkette verwendet werden.

Der Parameter *group* kann als benannter oder unbenannter Parameter verwendet werden. Beide Optionen funktionieren auf die gleiche Weise. Die benannte Option (group="GROUP") macht den Code leichter lesbar.

Der Parameter *group* kann mehrere durch Komma getrennte Bedingungen enthalten. Eine Bedingung kann mit '*' als Maske gesetzt werden. Das logische Negationssymbol '!' kann für einen Ausschluss verwendet werden. Alle Bedingungen werden sequentiell angewendet, d.h. Bedingungen der Aufnahme in eine Gruppe sollten zuerst angegeben werden, gefolgt von einer Ausschlussbedingung. Zum Beispiel bedeutet *group*="*, !EUR", dass alle Symbole zuerst ausgewählt werden sollen und diejenigen, die "EUR" im Namen enthalten, danach ausgeschlossen werden sollen.

Im Gegensatz zu symbol_info(), gibt die Funktion symbols_get() Daten zu allen angeforderten Symbolen innerhalb eines einzigen Aufrufs zurück.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abruf aller Symbole
symbols=mt5.symbols_get()
print('Symbols: ', len(symbols))
count=0
# Anzeige der ersten Fünf
```

© 2000-2025, MetaQuotes Ltd.

```
for s in symbols:
    count+=1
    print("{}. {}".format(count,s.name))
    if count==5: break
print()

# Abruf aller Symbole mit RU im Namen
ru_symbols=mt5.symbols_get("*RU*")
print('len(*RU*): ', len(ru_symbols))
for s in ru_symbols:
    print(s.name)
print()

# Abrufen der Symbole, deren Name nicht USD, EUR, JPY und GBP beinhalten.
group_symbols=mt5.symbols_get(group="*,!*USD*,!*EUR*,!*JPY*,!*GBP*")
print('len(*,!*USD*,!*EUR*,!*JPY*,!*GBP*):', len(group_symbols))
for s in group_symbols:
    print(s.name,":",s)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()

Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
Symbols:  84
1. EURUSD
2. GBPUSD
3. USDCHF
4. USDJPY
5. USDCNH

len(*RU*):  8
EURUSD
USDRUB
USDRUR
EURRUR
EURRUB
FORTS.RUB.M5
EURUSD_T20
EURUSD4

len(*,!*USD*,!*EUR*,!*JPY*,!*GBP*):  13
AUDCAD : SymbolInfo(custom=False, chart_mode=0, select=True, visible=True, session_dea
AUDCHF : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_d
AUDNZD : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_d
CADCHF : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_d
NZDCAD : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_d
NZDCHF : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_d
```

© 2000-2025, MetaQuotes Ltd.

```
NZDSGD : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_c
CADMXN : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_c
CHFMXN : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_c
NZDMXN : SymbolInfo(custom=False, chart_mode=0, select=False, visible=False, session_c
FORTS.RTS.M5 : SymbolInfo(custom=True, chart_mode=0, select=False, visible=False, sess
FORTS.RUB.M5 : SymbolInfo(custom=True, chart_mode=0, select=False, visible=False, sess
FOREX.CHF.M5 : SymbolInfo(custom=True, chart_mode=0, select=False, visible=False, sess
```

Siehe auch

symbols_total, symbol_select, symbol_info

© 2000-2025, MetaQuotes Ltd.

symbol_info

Abrufen der Daten des angegebenen Finanzinstruments.

```
symbol_info(
   symbol      // Name des Finanzinstruments
)
```

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion gibt alle Daten in einem Aufruf zurück, die einzeln mit SymbolInfoInteger, SymbolInfoDouble und SymbolInfoString erhalten werden könnten.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Versuch das Symbol EURJPY im MarketWatch anzuzeigen
selected=mt5.symbol_select("EURJPY",True)
if not selected:
    print("Failed to select EURJPY")
    mt5.shutdown()
    quit()

# Darstellung der Eigenschaften des Symbols EURJPY
symbol_info=mt5.symbol_info("EURJPY")
if symbol_info!=None:
    # Anzeige der Terminaldaten 'as is'
    print(symbol_info)
    print("EURJPY: spread =",symbol_info.spread,"  digits =",symbol_info.digits)
    # Anzeige der Symboleigenschaften als Liste
    print("Show symbol_info(\"EURJPY\")._asdict():")
    symbol_info_dict = mt5.symbol_info("EURJPY")._asdict()
    for prop in symbol_info_dict:
```

© 2000-2025, MetaQuotes Ltd.

```
        print("  {}={}".format(prop, symbol_info_dict[prop]))

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
SymbolInfo(custom=False, chart_mode=0, select=True, visible=True, session_deals=0, ses
EURJPY: spread = 17   digits = 3
Show symbol_info()._asdict():
  custom=False
  chart_mode=0
  select=True
  visible=True
  session_deals=0
  session_buy_orders=0
  session_sell_orders=0
  volume=0
  volumehigh=0
  volumelow=0
  time=1585069682
  digits=3
  spread=17
  spread_float=True
  ticks_bookdepth=10
  trade_calc_mode=0
  trade_mode=4
  start_time=0
  expiration_time=0
  trade_stops_level=0
  trade_freeze_level=0
  trade_exemode=1
  swap_mode=1
  swap_rollover3days=3
  margin_hedged_use_leg=False
  expiration_mode=7
  filling_mode=1
  order_mode=127
  order_gtc_mode=0
  option_mode=0
  option_right=0
  bid=120.024
  bidhigh=120.506
  bidlow=118.798
  ask=120.041
  askhigh=120.526
  asklow=118.828
```

© 2000-2025, MetaQuotes Ltd.

```
last=0.0
lasthigh=0.0
lastlow=0.0
volume_real=0.0
volumehigh_real=0.0
volumelow_real=0.0
option_strike=0.0
point=0.001
trade_tick_value=0.8977708350166538
trade_tick_value_profit=0.8977708350166538
trade_tick_value_loss=0.8978272580355541
trade_tick_size=0.001
trade_contract_size=100000.0
trade_accrued_interest=0.0
trade_face_value=0.0
trade_liquidity_rate=0.0
volume_min=0.01
volume_max=500.0
volume_step=0.01
volume_limit=0.0
swap_long=-0.2
swap_short=-1.2
margin_initial=0.0
margin_maintenance=0.0
session_volume=0.0
session_turnover=0.0
session_interest=0.0
session_buy_orders_volume=0.0
session_sell_orders_volume=0.0
session_open=0.0
session_close=0.0
session_aw=0.0
session_price_settlement=0.0
session_price_limit_min=0.0
session_price_limit_max=0.0
margin_hedged=100000.0
price_change=0.0
price_volatility=0.0
price_theoretical=0.0
price_greeks_delta=0.0
price_greeks_theta=0.0
price_greeks_gamma=0.0
price_greeks_vega=0.0
price_greeks_rho=0.0
price_greeks_omega=0.0
price_sensitivity=0.0
basis=
category=
currency_base=EUR
```

© 2000-2025, MetaQuotes Ltd.

```
currency_profit=JPY
currency_margin=EUR
bank=
description=Euro vs Japanese Yen
exchange=
formula=
isin=
name=EURJPY
page=http://www.google.com/finance?q=EURJPY
path=Forex\EURJPY
```

Siehe auch

account_info, terminal_info

© 2000-2025, MetaQuotes Ltd.

symbol_info_tick

Abrufen des letzten Ticks des angegebenen Finanzinstruments.

```
symbol_info_tick(
   symbol      // Name des Finanzinstruments
)
```

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe der Information als Tupel. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion ist ähnlich SymbolInfoTick.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Versuch das Symbol GBPUSD im MarketWatch anzuzeigen
selected=mt5.symbol_select("GBPUSD",True)
if not selected:
    print("Failed to select GBPUSD")
    mt5.shutdown()
    quit()

# Anzeige des letzten Ticks von GBPUSD
lasttick=mt5.symbol_info_tick("GBPUSD")
print(lasttick)
# Darstellung der Tick-Feldwerte als Liste
print("Show symbol_info_tick(\"GBPUSD\")._asdict():")
symbol_info_tick_dict = mt5.symbol_info_tick("GBPUSD")._asdict()
for prop in symbol_info_tick_dict:
    print("  {}={}".format(prop, symbol_info_tick_dict[prop]))

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

© 2000-2025, MetaQuotes Ltd.

```
Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
Tick(time=1585070338, bid=1.17264, ask=1.17279, last=0.0, volume=0, time_msc=158507033
Show symbol_info_tick._asdict():
  time=1585070338
  bid=1.17264
  ask=1.17279
  last=0.0
  volume=0
  time_msc=1585070338728
  flags=2
  volume_real=0.0
```

Siehe auch

symbol_info, symbol_info

© 2000-2025, MetaQuotes Ltd.

symbol_select

Auswahl eines Symbols im Fenster MarketWatch oder es aus demselben entfernen.

```
symbol_select(
   symbol,      // Name des Finanzinstruments
   enable=None  // aktivieren/deaktivieren
)
```

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

enable

[in] Auswahl. Optionale unbenannte Parameter. Wenn 'false' wird das Symbol aus dem Fenster MarketWatch entfernt. Andernfalls wird es im Fenster MarketWatch gezeigt und aktiviert. Ein Symbol kann nicht entfernt werden, wenn aktuell offene Charts mit diesem Symbol vorhanden sind oder Positionen darauf eröffnet sind.

Rückgabewert

True im Erfolgsfall, ansonsten - False.

Hinweis

Die Funktion ist ähnlich wie SymbolSelect.

Beispiel:

```
import MetaTrader5 as mt5
import pandas as pd
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
print()
# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize(login=25115284, server="MetaQuotes-Demo",password="4zatlbqx"):
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Versuch das Symbol EURCAD im MarketWatch anzuzeigen und zu aktivieren.
selected=mt5.symbol_select("EURCAD",True)
if not selected:
    print("Failed to select EURCAD, error code =",mt5.last_error())
else:
    symbol_info=mt5.symbol_info("EURCAD")
    print(symbol_info)
    print("EURCAD: currency_base =",symbol_info.currency_base,"  currency_profit =",sy
    print()

    # get symbol properties in the form of a dictionary
    print("Show symbol_info()._asdict():")
    symbol_info_dict = symbol_info._asdict()
```

© 2000-2025, MetaQuotes Ltd.

```
    for prop in symbol_info_dict:
        print("  {}={}".format(prop, symbol_info_dict[prop]))
    print()

   # Konvertieren und Ausdrucken der Liste als DataFrame
    df=pd.DataFrame(list(symbol_info_dict.items()),columns=['property','value'])
    print("symbol_info_dict() as dataframe:")
    print(df)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
SymbolInfo(custom=False, chart_mode=0, select=True, visible=True, session_deals=0, ses
EURCAD: currency_base = EUR   currency_profit = CAD   currency_margin = EUR

Show symbol_info()._asdict():
  custom=False
  chart_mode=0
  select=True
  visible=True
  session_deals=0
  session_buy_orders=0
  session_sell_orders=0
  volume=0
  volumehigh=0
  volumelow=0
  time=1585217595
  digits=5
  spread=39
  spread_float=True
  ticks_bookdepth=10
  trade_calc_mode=0
  trade_mode=4
  start_time=0
  expiration_time=0
  trade_stops_level=0
  trade_freeze_level=0
  trade_exemode=1
  swap_mode=1
  swap_rollover3days=3
  margin_hedged_use_leg=False
  expiration_mode=7
  filling_mode=1
  order_mode=127
  order_gtc_mode=0
```

© 2000-2025, MetaQuotes Ltd.

```
option_mode=0
option_right=0
bid=1.55192
bidhigh=1.55842
bidlow=1.5419800000000001
ask=1.5523099999999999
askhigh=1.55915
asklow=1.5436299999999998
last=0.0
lasthigh=0.0
lastlow=0.0
volume_real=0.0
volumehigh_real=0.0
volumelow_real=0.0
option_strike=0.0
point=1e-05
trade_tick_value=0.7043642408362214
trade_tick_value_profit=0.7043642408362214
trade_tick_value_loss=0.7044535553770941
trade_tick_size=1e-05
trade_contract_size=100000.0
trade_accrued_interest=0.0
trade_face_value=0.0
trade_liquidity_rate=0.0
volume_min=0.01
volume_max=500.0
volume_step=0.01
volume_limit=0.0
swap_long=-1.1
swap_short=-0.9
margin_initial=0.0
margin_maintenance=0.0
session_volume=0.0
session_turnover=0.0
session_interest=0.0
session_buy_orders_volume=0.0
session_sell_orders_volume=0.0
session_open=0.0
session_close=0.0
session_aw=0.0
session_price_settlement=0.0
session_price_limit_min=0.0
session_price_limit_max=0.0
margin_hedged=100000.0
price_change=0.0
price_volatility=0.0
price_theoretical=0.0
price_greeks_delta=0.0
price_greeks_theta=0.0
```

© 2000-2025, MetaQuotes Ltd.

```
  price_greeks_gamma=0.0
  price_greeks_vega=0.0
  price_greeks_rho=0.0
  price_greeks_omega=0.0
  price_sensitivity=0.0
  basis=
  category=
  currency_base=EUR
  currency_profit=CAD
  currency_margin=EUR
  bank=
  description=Euro vs Canadian Dollar
  exchange=
  formula=
  isin=
  name=EURCAD
  page=http://www.google.com/finance?q=EURCAD
  path=Forex\EURCAD

symbol_info_dict() as dataframe:
         property                                      value
0          custom                                      False
1      chart_mode                                          0
2          select                                       True
3         visible                                       True
4   session_deals                                          0
..            ...                                        ...
91        formula
92           isin
93           name                                     EURCAD
94           page  http://www.google.com/finance?q=EURCAD
95           path                               Forex\EURCAD

[96 rows x 2 columns]
```

Siehe auch

symbol_info

© 2000-2025, MetaQuotes Ltd.

market_book_add

Abonniert das MetaTrader 5-Terminal auf die Markttiefe-Änderungsereignisse für ein angegebenes Symbol.

```
Market_book_add(
   symbol      // Name des Finanzinstruments
)
```

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

Rückgabewert

True im Erfolgsfall, ansonsten - False.

Hinweis

Die Funktion ist ähnlich wie MarketBookAdd.

Siehe auch

market_book_get, market_book_release, Market Depth structure

© 2000-2025, MetaQuotes Ltd.

market_book_get

Gibt ein Tupel aus BookInfo mit den Einträgen der Markttiefe für das angegebene Symbol zurück.

```
market_book_get(
   symbol      // Name des Finanzinstruments
)
```

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

Rückgabewert

Gibt den Inhalt der Markttiefe als Tupel der Einträge aus BookInfo zurück, die den Ordertyp, Preis und Volumen in Lots enthalten. BookInfo ist ähnlich der Struktur MqlBookInfo.

Rückgabe von None im Falle eines Fehlers. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Das Abonnieren der Änderungsereignisse der Markttiefe sollte vorab mit der Funktion market_book_add() erfolgen.

Die Funktion ist ähnlich wie MarketBookGet.

Beispiel:

```
import MetaTrader5 as mt5
import time
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
print("")

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
   # Schließen der Verbindung zum MetaTrader 5
    mt5.shutdown()
    quit()

# Abonnieren der Aktualisierungen der Markttiefe für EURUSD (Depth of Market)
if mt5.market_book_add('EURUSD'):
  # Abrufen der Daten der Markttiefe 10 mal in einer Schleife
   for i in range(10):
        # Abrufen des Inhalts der Markttiefe (Depth of Market)
        items = mt5.market_book_get('EURUSD')
        # Darstellung der gesamten Markttiefe 'wie besehen' in einer einzigen Zeichenk
        print(items)
        # hier die Einzeldarstellung von jeder Order für mehr Klarheit
```

© 2000-2025, MetaQuotes Ltd.

```
        if items:
            for it in items:
                # Order-Inhalt
                print(it._asdict())
        # 5 Sek. Pause vor der nächsten Abfrage der Daten der Markttiefe
        time.sleep(5)
  # Beenden des Abonnements für die Aktualisierungen der Markttiefe (Depth of Market)
   mt5.market_book_release('EURUSD')
else:
    print("mt5.market_book_add('EURUSD') failed, error code =",mt5.last_error())

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.34

(BookInfo(type=1, price=1.20038, volume=250, volume_dbl=250.0), BookInfo(type=1, price
{'type': 1, 'price': 1.20038, 'volume': 250, 'volume_dbl': 250.0}
{'type': 1, 'price': 1.20032, 'volume': 100, 'volume_dbl': 100.0}
{'type': 1, 'price': 1.2003, 'volume': 50, 'volume_dbl': 50.0}
{'type': 1, 'price': 1.20028, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20026, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20025, 'volume': 50, 'volume_dbl': 50.0}
{'type': 2, 'price': 1.20023, 'volume': 100, 'volume_dbl': 100.0}
{'type': 2, 'price': 1.20017, 'volume': 250, 'volume_dbl': 250.0}
(BookInfo(type=1, price=1.2004299999999999, volume=250, volume_dbl=250.0), BookInfo(ty
{'type': 1, 'price': 1.2004299999999999, 'volume': 250, 'volume_dbl': 250.0}
{'type': 1, 'price': 1.20037, 'volume': 100, 'volume_dbl': 100.0}
{'type': 1, 'price': 1.20036, 'volume': 50, 'volume_dbl': 50.0}
{'type': 1, 'price': 1.20034, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20031, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20029, 'volume': 50, 'volume_dbl': 50.0}
{'type': 2, 'price': 1.20028, 'volume': 100, 'volume_dbl': 100.0}
{'type': 2, 'price': 1.20022, 'volume': 250, 'volume_dbl': 250.0}
(BookInfo(type=1, price=1.2004299999999999, volume=250, volume_dbl=250.0), BookInfo(ty
{'type': 1, 'price': 1.2004299999999999, 'volume': 250, 'volume_dbl': 250.0}
{'type': 1, 'price': 1.20037, 'volume': 100, 'volume_dbl': 100.0}
{'type': 1, 'price': 1.20036, 'volume': 50, 'volume_dbl': 50.0}
{'type': 1, 'price': 1.20034, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20031, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20029, 'volume': 50, 'volume_dbl': 50.0}
{'type': 2, 'price': 1.20028, 'volume': 100, 'volume_dbl': 100.0}
{'type': 2, 'price': 1.20022, 'volume': 250, 'volume_dbl': 250.0}
(BookInfo(type=1, price=1.20036, volume=250, volume_dbl=250.0), BookInfo(type=1, price
{'type': 1, 'price': 1.20036, 'volume': 250, 'volume_dbl': 250.0}
{'type': 1, 'price': 1.20029, 'volume': 100, 'volume_dbl': 100.0}
```

© 2000-2025, MetaQuotes Ltd.

```
{'type': 1, 'price': 1.20028, 'volume': 50, 'volume_dbl': 50.0}
{'type': 1, 'price': 1.20026, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20023, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20022, 'volume': 50, 'volume_dbl': 50.0}
{'type': 2, 'price': 1.20021, 'volume': 100, 'volume_dbl': 100.0}
{'type': 2, 'price': 1.20014, 'volume': 250, 'volume_dbl': 250.0}
(BookInfo(type=1, price=1.20035, volume=250, volume_dbl=250.0), BookInfo(type=1, price
{'type': 1, 'price': 1.20035, 'volume': 250, 'volume_dbl': 250.0}
{'type': 1, 'price': 1.20029, 'volume': 100, 'volume_dbl': 100.0}
{'type': 1, 'price': 1.20027, 'volume': 50, 'volume_dbl': 50.0}
{'type': 1, 'price': 1.20025, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20023, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20022, 'volume': 50, 'volume_dbl': 50.0}
{'type': 2, 'price': 1.20021, 'volume': 100, 'volume_dbl': 100.0}
{'type': 2, 'price': 1.20014, 'volume': 250, 'volume_dbl': 250.0}
(BookInfo(type=1, price=1.20037, volume=250, volume_dbl=250.0), BookInfo(type=1, price
{'type': 1, 'price': 1.20037, 'volume': 250, 'volume_dbl': 250.0}
{'type': 1, 'price': 1.20031, 'volume': 100, 'volume_dbl': 100.0}
{'type': 1, 'price': 1.2003, 'volume': 50, 'volume_dbl': 50.0}
{'type': 1, 'price': 1.20028, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20025, 'volume': 36, 'volume_dbl': 36.0}
{'type': 2, 'price': 1.20023, 'volume': 50, 'volume_dbl': 50.0}
{'type': 2, 'price': 1.20022, 'volume': 100, 'volume_dbl': 100.0}
{'type': 2, 'price': 1.20016, 'volume': 250, 'volume_dbl': 250.0}
```

Siehe auch

market_book_add, market_book_release, Market Depth structure

© 2000-2025, MetaQuotes Ltd.

market_book_release

Beenden des Abonnements der Markttiefe des MetaTrader 5-Terminals für ein bestimmtes Symbol ab.

```
market_book_release(
   symbol      // Name des Finanzinstruments
)
```

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

Rückgabewert

True im Erfolgsfall, ansonsten - False.

Hinweis

Die Funktion ist ähnlich wie MarketBookRelease.

Siehe auch

market_book_add, market_book_get, Market Depth structure

© 2000-2025, MetaQuotes Ltd.

copy_rates_from

Abrufen der Bars vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Datum.

```
copy_rates_from(
   symbol,       // Symbolname
   timeframe,    // Zeitrahmen
   date_from,    // Datum der Anfangsbar
   count         // Anzahl der Bars
   )
```

Parameter

symbol

[in] Name des Finanzinstruments, zum Beispiel "EURUSD". Benötigter unbenannter Parameter.

timeframe

[in] Zeitrahmen der benötigten Bars Bestimmt durch einen Wert der Enumeration TIMEFRAME. Benötigter unbenannter Parameter.

date_from

[in] Eröffnungszeitpunkt der ersten Bar der gewünschten Zeitspanne. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

count

[in] Verlangte Anzahl der Bars. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe de Bars als numpy Array mit den angegebenen Spalten von time, open, high, low, close, tick_volume, spread und real_volume. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Siehe auch die Funktion CopyRates() für weitere Information.

Nur die Daten zurück, deren Datum kleiner (früher) oder gleich dem angegebenen Datum ist. Dabei wird das Interval mit Genauigkeit bis zur Sekunde vorgegeben und berücksichtigt. D.h. die Zeit der Öffnung jeder Bar ist immer gleich oder kleiner als das angegebene Datum.

MetaTrader 5 Terminal stellt die Balken nur innerhalb der Historie zur Verfügung, die einem Benutzer auf den Charts zur Verfügung stehen. Die Anzahl der dem Benutzer zur Verfügung stehenden Balken wird durch den Parameter "Max. Balken im Chart" bestimmt.

Beim Erstellen des Objekts 'datetime' verwendet Python die lokale Zeitzone, MetaTrader 5 hingegen die Zeitzone von UTC (ohne Zeitverschiebung). Daher sollte 'datetime' in UTC-Zeit für die Funktionen erstellt werden, die die Zeit benötigen. Die Daten vom MetaTrader 5 Terminal haben UTC-Zeit.

TIMEFRAME ist eine Enumeration mit den möglichen Zeitrahmen eines Charts

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
TIMEFRAME_M1	1 Minute
TIMEFRAME_M2	2 Minuten
TIMEFRAME_M3	3 Minuten
TIMEFRAME_M4	4 Minuten
TIMEFRAME_M5	5 Minuten
TIMEFRAME_M6	6 Minuten
TIMEFRAME_M10	10 Minuten
TIMEFRAME_M12	12 Minuten
TIMEFRAME_M12	15 Minuten
TIMEFRAME_M20	20 Minuten
TIMEFRAME_M30	30 Minuten
TIMEFRAME_H1	1 Stunde
TIMEFRAME_H2	2 Stunden
TIMEFRAME_H3	3 Stunden
TIMEFRAME_H4	4 Stunden
TIMEFRAME_H6	6 Stunden
TIMEFRAME_H8	8 Stunden
TIMEFRAME_H12	12 Stunden
TIMEFRAME_D1	1 Tag
TIMEFRAME_W1	1 Woche
TIMEFRAME_MN1	1 Monat

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Importieren des Moduls 'pandas' für die Darstellung der erhaltenen Daten in Tabeller
import pandas as pd
pd.set_option('display.max_columns', 500) # Anzahl der anzuzeigenden Spalten
pd.set_option('display.width', 1500)      # maximale Tabellenbreite der Darstellung
# Importieren der Moduls pytz für die Arbeit mit Zeitzonen
```

© 2000-2025, MetaQuotes Ltd.

```
import pytz

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Setzen der Zeitzone auf UTC
timezone = pytz.timezone("Etc/UTC")
# Erstellen des Objekts 'datetime' in UTC-Zeit, um die lokale Zeitzone zu vermeiden
utc_from = datetime(2020, 1, 10, tzinfo=timezone)
# Abrufen von 10 Bars von EURUSD H4 ab dem 01.10.2020 in UTC-Zeit
rates = mt5.copy_rates_from("EURUSD", mt5.TIMEFRAME_H4, utc_from, 10)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
# Anzeigen von jedem erhaltenen Datenelement in einer neuen Zeile
print("Anzeige der erhaltenen Daten 'as is'")
for rate in rates:
    print(rate)

# Erstellen von DataFrame aus den erhaltenen Daten
rates_frame = pd.DataFrame(rates)
# Konvertieren der Zeit in Sekunden im Datumsformat
rates_frame['time']=pd.to_datetime(rates_frame['time'], unit='s')

# Datenanzeige
print("\nAnzeige von dataframe mit Daten")
print(rates_frame)


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Datenanzeige 'as is'
(1578484800, 1.11382, 1.11385, 1.1111, 1.11199, 9354, 1, 0)
(1578499200, 1.11199, 1.11308, 1.11086, 1.11179, 10641, 1, 0)
(1578513600, 1.11178, 1.11178, 1.11016, 1.11053, 4806, 1, 0)
(1578528000, 1.11053, 1.11193, 1.11033, 1.11173, 3480, 1, 0)
(1578542400, 1.11173, 1.11189, 1.11126, 1.11182, 2236, 1, 0)
(1578556800, 1.11181, 1.11203, 1.10983, 1.10993, 7984, 1, 0)
(1578571200, 1.10994, 1.11173, 1.10965, 1.11148, 7406, 1, 0)
(1578585600, 1.11149, 1.11149, 1.10923, 1.11046, 7468, 1, 0)
(1578600000, 1.11046, 1.11097, 1.11033, 1.11051, 3450, 1, 0)
(1578614400, 1.11051, 1.11093, 1.11017, 1.11041, 2448, 1, 0)

Darstellung von dataframe mit Daten
                 time     open     high      low    close  tick_volume  spread  real_v
```

© 2000-2025, MetaQuotes Ltd.

```
0 2020-01-08 12:00:00  1.11382  1.11385  1.11110  1.11199         9354       1
1 2020-01-08 16:00:00  1.11199  1.11308  1.11086  1.11179        10641       1
2 2020-01-08 20:00:00  1.11178  1.11178  1.11016  1.11053         4806       1
3 2020-01-09 00:00:00  1.11053  1.11193  1.11033  1.11173         3480       1
4 2020-01-09 04:00:00  1.11173  1.11189  1.11126  1.11182         2236       1
5 2020-01-09 08:00:00  1.11181  1.11203  1.10983  1.10993         7984       1
6 2020-01-09 12:00:00  1.10994  1.11173  1.10965  1.11148         7406       1
7 2020-01-09 16:00:00  1.11149  1.11149  1.10923  1.11046         7468       1
8 2020-01-09 20:00:00  1.11046  1.11097  1.11033  1.11051         3450       1
9 2020-01-10 00:00:00  1.11051  1.11093  1.11017  1.11041         2448       1
```

Siehe auch

CopyRates, copy_rates_from_pos, copy_rates_range, copy_ticks_from, copy_ticks_range

© 2000-2025, MetaQuotes Ltd.

copy_rates_from_pos

Abrufen der Bars vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Index.

```
copy_rates_from_pos(
   symbol,       // Symbolname
   timeframe,    // Zeitrahmen
   start_pos,    // Index der ersten Bar
   count         // Anzahl der Bars
   )
```

Parameter

symbol

[in] Name des Finanzinstruments, zum Beispiel "EURUSD". Benötigter unbenannter Parameter.

timeframe

[in] Zeitrahmen der benötigten Bars Bestimmt durch einen Wert der Enumeration TIMEFRAME. Benötigter unbenannter Parameter.

start_pos

[in] Index der ersten Bar der benötigten Daten. Die Zählung der Bars läuft von der Gegenwart in die Vergangenheit. Daher ist die Bar Null die aktuelle Bar. Benötigter unbenannter Parameter.

count

[in] Verlangte Anzahl der Bars. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe de Bars als numpy Array mit den angegebenen Spalten von time, open, high, low, close, tick_volume, spread und real_volume. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Siehe auch die Funktion CopyRates() für weitere Information.

MetaTrader 5 Terminal stellt die Balken nur innerhalb der Historie zur Verfügung, die einem Benutzer auf den Charts zur Verfügung stehen. Die Anzahl der dem Benutzer zur Verfügung stehenden Balken wird durch den Parameter "Max. Balken im Chart" bestimmt.

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Importieren des Moduls 'pandas' für die Darstellung der erhaltenen Daten in Tabeller
import pandas as pd
pd.set_option('display.max_columns', 500) # Anzahl der anzuzeigenden Spalten
pd.set_option('display.width', 1500)      # maximale Tabellenbreite der Darstellung
```

© 2000-2025, MetaQuotes Ltd.

```
# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen von 10 Bars von GBPUSD D1 ab dem aktuellen Tag
rates = mt5.copy_rates_from_pos("GBPUSD", mt5.TIMEFRAME_D1, 0, 10)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
# Anzeigen von jedem erhaltenen Datenelement in einer neuen Zeile
print("Anzeige der erhaltenen Daten 'as is'")
for rate in rates:
    print(rate)

# Erstellen von DataFrame aus den erhaltenen Daten
rates_frame = pd.DataFrame(rates)
# Konvertieren der Zeit in Sekunden im Datumsformat
rates_frame['time']=pd.to_datetime(rates_frame['time'], unit='s')

# Datenanzeige
print("\nAnzeige von dataframe mit Daten")
print(rates_frame)
```

Ergebnis:

```
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Datenanzeige 'as is'
(1581552000, 1.29568, 1.30692, 1.29441, 1.30412, 68228, 0, 0)
(1581638400, 1.30385, 1.30631, 1.3001, 1.30471, 56498, 0, 0)
(1581897600, 1.30324, 1.30536, 1.29975, 1.30039, 49400, 0, 0)
(1581984000, 1.30039, 1.30486, 1.29705, 1.29952, 62288, 0, 0)
(1582070400, 1.29952, 1.3023, 1.29075, 1.29187, 57909, 0, 0)
(1582156800, 1.29186, 1.29281, 1.28489, 1.28792, 61033, 0, 0)
(1582243200, 1.28802, 1.29805, 1.28746, 1.29566, 66386, 0, 0)
(1582502400, 1.29426, 1.29547, 1.28865, 1.29283, 66933, 0, 0)
(1582588800, 1.2929, 1.30178, 1.29142, 1.30037, 80121, 0, 0)
(1582675200, 1.30036, 1.30078, 1.29136, 1.29374, 49286, 0, 0)

Darstellung von dataframe mit Daten
        time     open     high      low    close  tick_volume  spread  real_volume
0 2020-02-13  1.29568  1.30692  1.29441  1.30412        68228       0            0
1 2020-02-14  1.30385  1.30631  1.30010  1.30471        56498       0            0
2 2020-02-17  1.30324  1.30536  1.29975  1.30039        49400       0            0
3 2020-02-18  1.30039  1.30486  1.29705  1.29952        62288       0            0
4 2020-02-19  1.29952  1.30230  1.29075  1.29187        57909       0            0
5 2020-02-20  1.29186  1.29281  1.28489  1.28792        61033       0            0
```

© 2000-2025, MetaQuotes Ltd.

```
6 2020-02-21  1.28802  1.29805  1.28746  1.29566        66386        0            0
7 2020-02-24  1.29426  1.29547  1.28865  1.29283        66933        0            0
8 2020-02-25  1.29290  1.30178  1.29142  1.30037        80121        0            0
9 2020-02-26  1.30036  1.30078  1.29136  1.29374        49286        0            0
```

Siehe auch

CopyRates, copy_rates_from, copy_rates_range, copy_ticks_from, copy_ticks_range

© 2000-2025, MetaQuotes Ltd.

copy_rates_range

Abrufen der Bars der angegebenen Zeitspanne vom MetaTrader 5 Terminal.

```
copy_rates_range(
   symbol,       // Symbolname
   timeframe,    // Zeitrahmen
   date_from,    // Zeitpunkt der ersten benötigten Bar
   date_to       // Zeitpunkt, bis zu dem die Bars angefordert werden
   )
```

Parameter

symbol

[in] Name des Finanzinstruments, zum Beispiel "EURUSD". Benötigter unbenannter Parameter.

timeframe

[in] Zeitrahmen der benötigten Bars Bestimmt durch einen Wert der Enumeration TIMEFRAME. Benötigter unbenannter Parameter.

date_from

[in] Zeitpunkt der ersten benötigten Bar. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Bars mit einer Eröffnungszeit >= date_from werden zurückgegeben. Benötigter unbenannter Parameter.

date_to

[in] Zeitpunkt, bis zu dem die Bars angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Bars mit einer Eröffnungszeit <= date_to werden zurückgegeben. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe de Bars als numpy Array mit den angegebenen Spalten von time, open, high, low, close, tick_volume, spread und real_volume. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Siehe auch die Funktion CopyRates() für weitere Information.

MetaTrader 5 Terminal stellt die Balken nur innerhalb der Historie zur Verfügung, die einem Benutzer auf den Charts zur Verfügung stehen. Die Anzahl der dem Benutzer zur Verfügung stehenden Balken wird durch den Parameter "Max. Balken im Chart" bestimmt.

Beim Erstellen des Objekts 'datetime' verwendet Python die lokale Zeitzone, MetaTrader 5 hingegen die Zeitzone von UTC (ohne Zeitverschiebung). Daher sollte 'datetime' in UTC-Zeit für die Funktionen erstellt werden, die die Zeit benötigen. Die Daten vom MetaTrader 5 Terminal haben UTC-Zeit.

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
```

© 2000-2025, MetaQuotes Ltd.

```
print("MetaTrader5 package version: ",mt5.__version__)

# Importieren des Moduls 'panda' für eine Darstellung der erhaltenen Daten in Tabeller
import pandas as pd
pd.set_option('display.max_columns', 500) # Anzahl der anzuzeigenden Spalten
pd.set_option('display.width', 1500)      # Maximale Tabellenbreite für die Anzeige
# Importieren der Moduls pytz für die Arbeit mit Zeitzonen
import pytz

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Setzen der Zeitzone auf UTC
timezone = pytz.timezone("Etc/UTC")
# Erstellen des Objekts 'datetime' in der UTC-Zeitzone, um eine Verwendung der lokaler
utc_from = datetime(2020, 1, 10, tzinfo=timezone)
utc_to = datetime(2020, 1, 11, hour = 13, tzinfo=timezone)
# Abrufen der Bars von USDJPY M5 des Zeitintervalls 2020.01.10 00:00 - 2020.01.11 13:0
rates = mt5.copy_rates_range("USDJPY", mt5.TIMEFRAME_M5, utc_from, utc_to)

# Schließen der Verbindung zum MetaTrader 5 Terminal
mt5.shutdown()

# Zeilenweise Darstellung von jedem Element der erhaltenen Daten
print("Display obtained data 'as is'")
counter=0
for rate in rates:
    counter+=1
    if counter<=10:
        print(rate)

# Erstellen des DataFrame aus den erhaltenen Daten
rates_frame = pd.DataFrame(rates)
# Konvertieren der Zeit in Sekunden in das Format 'datetime'
rates_frame['time']=pd.to_datetime(rates_frame['time'], unit='s')

# Datenanzeige
print("\nDisplay dataframe with data")
print(rates_frame.head(10))
```

Ergebnis:

```
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Datenanzeige 'as is'
(1578614400, 109.513, 109.527, 109.505, 109.521, 43, 2, 0)
(1578614700, 109.521, 109.549, 109.518, 109.543, 215, 8, 0)
(1578615000, 109.543, 109.543, 109.466, 109.505, 98, 10, 0)
(1578615300, 109.504, 109.534, 109.502, 109.517, 155, 8, 0)
(1578615600, 109.517, 109.539, 109.513, 109.527, 71, 4, 0)
(1578615900, 109.526, 109.537, 109.484, 109.52, 106, 9, 0)
(1578616200, 109.52, 109.524, 109.508, 109.51, 205, 7, 0)
(1578616500, 109.51, 109.51, 109.491, 109.496, 44, 8, 0)
```

© 2000-2025, MetaQuotes Ltd.

```
(1578616800, 109.496, 109.509, 109.487, 109.5, 85, 5, 0)
(1578617100, 109.5, 109.504, 109.487, 109.489, 82, 7, 0)

Darstellung von dataframe mit Daten
                 time     open     high      low    close  tick_volume  spread  real_v
0 2020-01-10 00:00:00  109.513  109.527  109.505  109.521           43       2
1 2020-01-10 00:05:00  109.521  109.549  109.518  109.543          215       8
2 2020-01-10 00:10:00  109.543  109.543  109.466  109.505           98      10
3 2020-01-10 00:15:00  109.504  109.534  109.502  109.517          155       8
4 2020-01-10 00:20:00  109.517  109.539  109.513  109.527           71       4
5 2020-01-10 00:25:00  109.526  109.537  109.484  109.520          106       9
6 2020-01-10 00:30:00  109.520  109.524  109.508  109.510          205       7
7 2020-01-10 00:35:00  109.510  109.510  109.491  109.496           44       8
8 2020-01-10 00:40:00  109.496  109.509  109.487  109.500           85       5
9 2020-01-10 00:45:00  109.500  109.504  109.487  109.489           82       7
```

Siehe auch

CopyRates, copy_rates_from, copy_rates_range, copy_ticks_from, copy_ticks_range

© 2000-2025, MetaQuotes Ltd.

copy_ticks_from

Abrufen von Ticks aus dem MetaTrader 5 Terminal ab dem angegebenen Zeitpunkt.

```
copy_ticks_from(
   symbol,       // Symbolname
   date_from,    // Anfangsdatum, ab dem die Ticks angefordert werden
   count,        // Anzahl der benötigten Ticks
   flags         // Kombination der Flags, die den Typ der benötigten Ticks spezifizie
   )
```

Parameter

symbol

[in] Name des Finanzinstruments, zum Beispiel "EURUSD". Benötigter unbenannter Parameter.

from

[in] Zeitpunkt, ab dem die Ticks benötigt werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

count

[in] Anzahl der benötigten Ticks. Benötigter unbenannter Parameter.

flags

[in] Die Flag, die den Typ der benötigten Ticks spezifiziert. COPY_TICKS_INFO - Ticks mit den Änderungen von Bid und/oder Ask, COPY_TICKS_TRADE - Ticks mit den Änderungen von Last und Volume, COPY_TICKS_ALL - alle Ticks. Die Werte der Flags sind beschreiben in der Enumeration COPY_TICKS. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe der Ticks als numpy Array mit den benannten Spalten time, bid, ask, last und flags. Die Werte der 'flags' kann eine Kombination der Flags der Enumeration TICK_FLAG sein. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Siehe die Funktion CopyTicks für weitere Informationen.

Beim Erstellen des Objekts 'datetime' verwendet Python die lokale Zeitzone, MetaTrader 5 hingegen die Zeitzone von UTC (ohne Zeitverschiebung). Daher sollte 'datetime' in UTC-Zeit für die Funktionen erstellt werden, die die Zeit benötigen. Die Daten vom MetaTrader 5 Terminal haben UTC-Zeit.

COPY_TICKS definiert den Typ der Ticks, die mit den Funktionen copy_ticks_from() und copy_ticks_range() angefordert werden können.

ID	Beschreibung
COPY_TICKS_ALL	alle Ticks
COPY_TICKS_INFO	Ticks mit Bid- und/oder Ask-Preisänderungen

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
COPY_TICKS_TRADE	Ticks mit letzter Preisänderung und/oder Volumensänderung

TICK_FLAG definiert die möglichen Flags der Ticks. Diese Flags werden zur Beschreibung der mit den Funktionen copy_ticks_from() und copy_ticks_range() erhaltenen Flags verwendet.

ID	Beschreibung
TICK_FLAG_BID	Bid-Preisänderung
TICK_FLAG_ASK	Ask-Preisänderung
TICK_FLAG_LAST	Letzte Preisänderung
TICK_FLAG_VOLUME	Volumensänderung
TICK_FLAG_BUY	letzte Kaufpreisänderung
TICK_FLAG_SELL	letzte Verkaufpreisänderung

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Importieren des Moduls 'pandas' für die Darstellung der erhaltenen Daten in Tabeller
import pandas as pd
pd.set_option('display.max_columns', 500) # Anzahl der anzuzeigenden Spalten
pd.set_option('display.width', 1500)      # maximale Tabellenbreite der Darstellung
# Importieren der Moduls pytz für die Arbeit mit Zeitzonen
import pytz

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Setzen der Zeitzone auf UTC
timezone = pytz.timezone("Etc/UTC")
>#  Erstellen des Objekts 'datetime' in UTC-Zeit, um die lokale Zeitzone zu vermeiden
utc_from = datetime(2020, 1, 10, tzinfo=timezone)
# Anfordern von 100 000 EURUSD Ticks ab 10.01.2019 in UTC-Zeit
ticks = mt5.copy_ticks_from("EURUSD", utc_from, 100000, mt5.COPY_TICKS_ALL)
print("Ticks received:",len(ticks))

# Schließen der Verbindung zum MetaTrader 5 Terminal
mt5.shutdown()
```

© 2000-2025, MetaQuotes Ltd.

```
# Darstellung der Daten von jedem Tick in einer neuen Zeile
print("Display obtained ticks 'as is'")
count = 0
for tick in ticks:
    count+=1
    print(tick)
    if count >= 10:
        break

# Erstellen des DataFrame aus den erhaltenen Daten
ticks_frame = pd.DataFrame(ticks)
# Konvertieren der Zeit in Sekunden im Datumsformat
ticks_frame['time']=pd.to_datetime(ticks_frame['time'], unit='s')

# Datenanzeige
print("\nDisplay dataframe with ticks")
print(ticks_frame.head(10))
```

Ergebnis:

```
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Ticks received: 100000
Anzeige der erhaltenen Ticks 'as is'
(1578614400, 1.11051, 1.11069, 0., 0, 1578614400987, 134, 0.)
(1578614402, 1.11049, 1.11067, 0., 0, 1578614402025, 134, 0.)
(1578614404, 1.1105, 1.11066, 0., 0, 1578614404057, 134, 0.)
(1578614404, 1.11049, 1.11067, 0., 0, 1578614404344, 134, 0.)
(1578614412, 1.11052, 1.11064, 0., 0, 1578614412106, 134, 0.)
(1578614418, 1.11039, 1.11051, 0., 0, 1578614418265, 134, 0.)
(1578614418, 1.1104, 1.1105, 0., 0, 1578614418905, 134, 0.)
(1578614419, 1.11039, 1.11051, 0., 0, 1578614419519, 134, 0.)
(1578614456, 1.11037, 1.11065, 0., 0, 1578614456011, 134, 0.)
(1578614456, 1.11039, 1.11051, 0., 0, 1578614456015, 134, 0.)

Darstellung von dataframe mit Ticks
                 time      bid      ask  last  volume       time_msc  flags  volume_re
0 2020-01-10 00:00:00  1.11051  1.11069   0.0       0  1578614400987    134          0
1 2020-01-10 00:00:02  1.11049  1.11067   0.0       0  1578614402025    134          0
2 2020-01-10 00:00:04  1.11050  1.11066   0.0       0  1578614404057    134          0
3 2020-01-10 00:00:04  1.11049  1.11067   0.0       0  1578614404344    134          0
4 2020-01-10 00:00:12  1.11052  1.11064   0.0       0  1578614412106    134          0
5 2020-01-10 00:00:18  1.11039  1.11051   0.0       0  1578614418265    134          0
6 2020-01-10 00:00:18  1.11040  1.11050   0.0       0  1578614418905    134          0
7 2020-01-10 00:00:19  1.11039  1.11051   0.0       0  1578614419519    134          0
8 2020-01-10 00:00:56  1.11037  1.11065   0.0       0  1578614456011    134          0
9 2020-01-10 00:00:56  1.11039  1.11051   0.0       0  1578614456015    134          0
```

Siehe auch

CopyRates, copy_rates_from_pos, copy_rates_range, copy_ticks_from, copy_ticks_range

© 2000-2025, MetaQuotes Ltd.

copy_ticks_range

Abrufen der Ticks der angegebenen Zeitspanne vom MetaTrader 5 Terminal.

```
copy_ticks_range(
   symbol,       // Symbolname
   date_from,    // Zeitpunkt, ab dem die Ticks angefordert werden
   date_to,      // Zeitpunkt, bis zu dem die Ticks angefordert werden
   flags         // Kombination der Flags, die den Typ der benötigten Ticks spezifizie
   )
```

Parameter

symbol

[in] Name des Finanzinstruments, zum Beispiel "EURUSD". Benötigter unbenannter Parameter.

date_from

[in] Zeitpunkt, ab dem die Ticks benötigt werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

date_to

[in] Zeitpunkt, bis zu dem die Bars benötigt werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

flags

[in] Die Flag, die den Typ der benötigten Ticks spezifiziert. COPY_TICKS_INFO - Ticks mit den Änderungen von Bid und/oder Ask, COPY_TICKS_TRADE - Ticks mit den Änderungen von Last und Volume, COPY_TICKS_ALL - alle Ticks. Die Werte der Flags sind beschreiben in der Enumeration COPY_TICKS. Benötigter unbenannter Parameter.

Rückgabewert

Rückgabe der Ticks als numpy Array mit den benannten Spalten time, bid, ask, last und flags. Die Werte der 'flags' kann eine Kombination der Flags der Enumeration TICK_FLAG sein. Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Siehe die Funktion CopyTicks für weitere Informationen.

Beim Erstellen des Objekts 'datetime' verwendet Python die lokale Zeitzone, MetaTrader 5 hingegen die Zeitzone von UTC (ohne Zeitverschiebung). Daher sollte 'datetime' in UTC-Zeit für die Funktionen erstellt werden, die die Zeit benötigen. Die aus MetaTrader 5 erhaltenen Daten haben UTC-Zeit, aber Python wendet beim Versuch, sie zu drucken, wieder die lokale Zeit (inkl. Verschiebung) an. Daher sollten die gewonnenen Daten auch für die visuelle Darstellung korrigiert werden.

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
```

© 2000-2025, MetaQuotes Ltd.

```
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Importieren des Moduls 'panda' für eine Darstellung der erhaltenen Daten in Tabeller
import pandas as pd
pd.set_option('display.max_columns', 500) # Anzahl der anzuzeigenden Spalten
pd.set_option('display.width', 1500)      # Maximale Tabellenbreite für die Anzeige
# Importieren der Moduls pytz für die Arbeit mit Zeitzonen
import pytz

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Setzen der Zeitzone auf UTC
timezone = pytz.timezone("Etc/UTC")
# Erstellen des Objekts 'datetime' in der UTC-Zeitzone, um eine Verwendung der lokaler
utc_from = datetime(2020, 1, 10, tzinfo=timezone)
utc_to = datetime(2020, 1, 11, tzinfo=timezone)
# Anforderung der Ticks von AUDUSD im Zeitintervall 11.01.2020 - 11.01.2020
ticks = mt5.copy_ticks_range("AUDUSD", utc_from, utc_to, mt5.COPY_TICKS_ALL)
print("Ticks received:",len(ticks))

# Schließen der Verbindung zum MetaTrader 5 Terminal
mt5.shutdown()

# Darstellung der Daten von jedem Tick in einer neuen Zeile
print("Display obtained ticks 'as is'")
count = 0
for tick in ticks:
    count+=1
    print(tick)
    if count >= 10:
        break

# Erstellen des DataFrame aus den erhaltenen Daten
ticks_frame = pd.DataFrame(ticks)
# Konvertieren der Zeit in Sekunden im Datumsformat
ticks_frame['time']=pd.to_datetime(ticks_frame['time'], unit='s')

# Datenanzeige
print("\nDisplay dataframe with ticks")
print(ticks_frame.head(10))
```

Ergebnis:

```
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Ticks received: 37008
Anzeige der erhaltenen Ticks 'as is'
(1578614400, 0.68577, 0.68594, 0., 0, 1578614400820, 134, 0.)
(1578614401, 0.68578, 0.68594, 0., 0, 1578614401128, 130, 0.)
(1578614401, 0.68575, 0.68594, 0., 0, 1578614401128, 130, 0.)
(1578614411, 0.68576, 0.68594, 0., 0, 1578614411388, 130, 0.)
(1578614411, 0.68575, 0.68594, 0., 0, 1578614411560, 130, 0.)
```

© 2000-2025, MetaQuotes Ltd.

```
(1578614414, 0.68576, 0.68595, 0., 0, 1578614414973, 134, 0.)
(1578614430, 0.68576, 0.68594, 0., 0, 1578614430188, 4, 0.)
(1578614450, 0.68576, 0.68595, 0., 0, 1578614450408, 4, 0.)
(1578614450, 0.68576, 0.68594, 0., 0, 1578614450519, 4, 0.)
(1578614456, 0.68575, 0.68594, 0., 0, 1578614456363, 130, 0.)

Darstellung von dataframe mit Ticks
                 time      bid      ask  last  volume       time_msc  flags  volume_re
0 2020-01-10 00:00:00  0.68577  0.68594   0.0       0  1578614400820    134          0
1 2020-01-10 00:00:01  0.68578  0.68594   0.0       0  1578614401128    130          0
2 2020-01-10 00:00:01  0.68575  0.68594   0.0       0  1578614401128    130          0
3 2020-01-10 00:00:11  0.68576  0.68594   0.0       0  1578614411388    130          0
4 2020-01-10 00:00:11  0.68575  0.68594   0.0       0  1578614411560    130          0
5 2020-01-10 00:00:14  0.68576  0.68595   0.0       0  1578614414973    134          0
6 2020-01-10 00:00:30  0.68576  0.68594   0.0       0  1578614430188      4          0
7 2020-01-10 00:00:50  0.68576  0.68595   0.0       0  1578614450408      4          0
8 2020-01-10 00:00:50  0.68576  0.68594   0.0       0  1578614450519      4          0
9 2020-01-10 00:00:56  0.68575  0.68594   0.0       0  1578614456363    130          0
```

Siehe auch

CopyRates, copy_rates_from_pos, copy_rates_range, copy_ticks_from, copy_ticks_range

© 2000-2025, MetaQuotes Ltd.

orders_total

Abrufen der Anzahl der aktiven Orders.

```
orders_total()
```

Rückgabewert

Integer Wert

Hinweis

Die Funktion ist ähnlich OrdersTotal.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Prüfen des Vorhandenseins aktiver Orders
orders=mt5.orders_total()
if orders>0:
    print("Total orders=",orders)
else:
    print("Orders not found")

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

Siehe auch

orders_get, positions_total

© 2000-2025, MetaQuotes Ltd.

orders_get

Abrufen der aktiven Orders mit der Fähigkeit nach Symbol oder Ticket zu Filtern Es gibt drei Aufrufoptionen.

Aufruf ohne Parameter. Rückgabe der aktiven Orders aller Symbole.

```
orders_get()
```

Ein Abruf der aktiven Orders mit angegebenem Symbol.

```
orders_get(
   symbol="SYMBOL"       // Symbolname
)
```

Ein Abruf der aktiven Orders mit einer angegebenen Symbolgruppe.

```
orders_get(
   group="GROUP"         // Symbolfilter der Orderauswahl
)
```

Aufruf unter Angabe des Orderticket.

```
orders_get(
   ticket=TICKET         // Ticket
)
```

symbol="SYMBOL"

[in] Symbolname. Optionale benannte Parameter. Wein ein Symbol angegeben ist, wird das *Ticket* ignoriert.

group="GROUP"

[in] Der Filter für die Gruppe der angeforderten Symbole. Optionale benannte Parameter. Wenn die Gruppe angegeben ist, gibt die Funktion nur Orders zurück, die das angegebene Kriterium für Symbolnamen erfüllen.

ticket=TICKET

[in] Order ticket (ORDER_TICKET). Optionale benannte Parameter.

Rückgabewert

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion ermöglicht den Empfang aller aktiven Orders innerhalb eines Anrufs, ähnlich wie bei dem Funktionspaar OrdersTotal und OrderSelect.

Der Parameter *group* erlaubt das Filtern von Orders nach Symbolen. '*' kann am Anfang und am Ende einer Zeichenkette verwendet werden.

Der Parameter *group* kann mehrere durch Komma getrennte Bedingungen enthalten. Eine Bedingung kann mit '*' als Maske gesetzt werden. Das logische Negationssymbol '!' kann für einen Ausschluss verwendet werden. Alle Bedingungen werden sequentiell angewendet, d.h. Bedingungen der

© 2000-2025, MetaQuotes Ltd.

Aufnahme in eine Gruppe sollten zuerst angegeben werden, gefolgt von einer Ausschlussbedingung. Zum Beispiel bedeutet *group*="*, !EUR", dass die Orders für alle Symbole zuerst ausgewählt werden sollten und diejenigen, die "EUR" in den Symbolnamen enthalten, danach ausgeschlossen werden sollten.

Beispiel:

```
import MetaTrader5 as mt5
import pandas as pd
pd.set_option('display.max_columns', 500) # darzustellende Spaltenanzahl
pd.set_option('display.width', 1500)      # maximale darzustellende Tabellenbreite
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
print()
# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Datenanzeige der Orders mit GBPUSD
orders=mt5.orders_get(symbol="GBPUSD")
if orders is None:
    print("No orders on GBPUSD, error code={}".format(mt5.last_error()))
else:
    print("Total orders on GBPUSD:",len(orders))
    # Darstellung aller aktiven Orders
    for order in orders:
        print(order)
print()

# Abrufen der Orderliste aller Symbole mit "*GBP*" im Namen
gbp_orders=mt5.orders_get(group="*GBP*")
if gbp_orders is None:
    print("No orders with group=\"*GBP*\", error code={}".format(mt5.last_error()))
else:
    print("orders_get(group=\"*GBP*\")={}".format(len(gbp_orders)))
    # Darstellung der Orders als Tabelle mittels pandas.DataFrame
    df=pd.DataFrame(list(gbp_orders),columns=gbp_orders[0]._asdict().keys())
    df.drop(['time_done', 'time_done_msc', 'position_id', 'position_by_id', 'reason',
    df['time_setup'] = pd.to_datetime(df['time_setup'], unit='s')
    print(df)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()

Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
```

© 2000-2025, MetaQuotes Ltd.

```
Total orders on GBPUSD: 2
TradeOrder(ticket=554733548, time_setup=1585153667, time_setup_msc=1585153667718, time
TradeOrder(ticket=554733621, time_setup=1585153671, time_setup_msc=1585153671419, time

orders_get(group="*GBP*")=4
      ticket           time_setup  time_setup_msc  time_expiration  type  type_time  ty
0  554733548 2020-03-25 16:27:47   1585153667718                0     3          0
1  554733621 2020-03-25 16:27:51   1585153671419                0     2          0
2  554746664 2020-03-25 16:38:14   1585154294401                0     3          0
3  554746710 2020-03-25 16:38:17   1585154297022                0     2          0
```

Siehe auch

orders_total, positions_get

© 2000-2025, MetaQuotes Ltd.

order_calc_margin

Rückgabe der Marge in der Kontowährung zur Durchführung der angegebenen Handelsoperation.

```
order_calc_margin(
   action,      // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYPE_SELL)
   symbol,      // Symbolname
   volume,      // Volumen
   price        // Eröffnungspreis
   )
```

Parameter

action

[in] Ordertyp akzeptiert Werte der Enumeration ORDER_TYPE. Benötigter unbenannter Parameter.

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

volume

[in] Volumen der Handelsoperation. Benötigter unbenannter Parameter.

price

[in] Eröffnungspreis. Benötigter unbenannter Parameter.

Rückgabewert

Double-Wert im Erfolgsfall, andernfalls nichts. Die Fehlernummer kann über last_error() abgefragt werden.

Hinweis

Die Funktion ermöglicht das Schätzen der für eine bestimmte Auftragsart auf dem aktuellen Konto und im aktuellen Marktumfeld erforderlichen Marge, ohne die aktuell Pending-Orders und offenen Positionen zu berücksichtigen. Die Funktion ist ähnlich wie OrderCalcMargin.

ORDER_TYPE

ID	Beschreibung
ORDER_TYPE_BUY	Marktorder, Kauf
ORDER_TYPE_SELL	Marktorder, Verkauf
ORDER_TYPE_BUY_LIMIT	Pending-Order, Buy-Limit
ORDER_TYPE_SELL_LIMIT	Pending-Order, Sell-Limit
ORDER_TYPE_BUY_STOP	Pending-Order, Buy-Stop
ORDER_TYPE_SELL_STOP	Pending-Order Sell-Stop
ORDER_TYPE_BUY_STOP_LIMIT	Beim Erreichen des Order-Preis, wird eine Pending-Order Buy-Limit zum Preis von StopLimit platziert

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
ORDER_TYPE_SELL_STOP_LIMIT	Beim Erreichen des Order-Preis, wird eine Pending-Order Sell-Limit zum Preis von StopLimit platziert
ORDER_TYPE_CLOSE_BY	Auftrag eine Position durch eine entgegengesetzte zu schließen.

Beispiel:

```
import MetaTrader5 as mt5
# Anzeige der Daten des Pakets MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Kontowährung
account_currency=mt5.account_info().currency
print("Account currency:",account_currency)

# Auflistung der Symbole
symbols=("EURUSD","GBPUSD","USDJPY", "USDCHF","EURJPY","GBPJPY")
print("Symbols to check margin:", symbols)
action=mt5.ORDER_TYPE_BUY
lot=0.1
for symbol in symbols:
    symbol_info=mt5.symbol_info(symbol)
    if symbol_info is None:
        print(symbol,"not found, skipped")
        continue
    if not symbol_info.visible:
        print(symbol, "is not visible, trying to switch on")
        if not mt5.symbol_select(symbol,True):
            print("symbol_select({}}) failed, skipped",symbol)
            continue
    ask=mt5.symbol_info_tick(symbol).ask
    margin=mt5.order_calc_margin(action,symbol,lot,ask)
    if margin != None:
        print("   {} buy {} lot margin: {} {}".format(symbol,lot,margin,account_currer
    else:
        print("order_calc_margin failed: , error code =", mt5.last_error())

# Schließen der Verbindung zum MetaTrader 5
```

© 2000-2025, MetaQuotes Ltd.

```
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Account currency: USD

Symbols to check margin: ('EURUSD', 'GBPUSD', 'USDJPY', 'USDCHF', 'EURJPY', 'GBPJPY')
   EURUSD buy 0.1 lot margin: 109.91 USD
   GBPUSD buy 0.1 lot margin: 122.73 USD
   USDJPY buy 0.1 lot margin: 100.0 USD
   USDCHF buy 0.1 lot margin: 100.0 USD
   EURJPY buy 0.1 lot margin: 109.91 USD
   GBPJPY buy 0.1 lot margin: 122.73 USD
```

Siehe auch

order_calc_profit, order_check

© 2000-2025, MetaQuotes Ltd.

order_calc_profit

Rückgabe des Gewinns in Kontowährung für die angegebene Handelsoperation.

```
order_calc_profit(
   action,          // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYPE_SELL)
   symbol,          // Symbolname
   volume,          // Volume
   price_open,      // Eröffnungspreis
   price_close      // Schließpreis
   );
```

Parameter

action

[in] Ordertyp kann eine der beiden Werte der Enumeration ORDER_TYPE annehmen: ORDER_TYPE_BUY or ORDER_TYPE_SELL. Benötigter unbenannter Parameter.

symbol

[in] Name des Finanzinstruments. Benötigter unbenannter Parameter.

volume

[in] Volumen der Handelsoperation. Benötigter unbenannter Parameter.

price_open

[in] Eröffnungspreis. Benötigter unbenannter Parameter.

price_close

[in] SchließpreisClose price. Benötigter unbenannter Parameter.

Rückgabewert

Double-Wert im Erfolgsfall, andernfalls nichts. Die Fehlernummer kann über last_error() abgefragt werden.

Hinweis

Die Funktion ermöglicht die Schätzung des Ergebnisses einer Handelsoperation auf dem aktuellen Konto und der aktuellen Handelsumgebung. Die Funktion ist ähnlich wie OrderCalcProfit.

Beispiel:

```
import MetaTrader5 as mt5
# Anzeige der Daten des Pakets MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
```

© 2000-2025, MetaQuotes Ltd.

```
    quit()

# Abrufen der Kontowährung
account_currency=mt5.account_info().currency
print("Account currency:",account_currency)

# Auflistung der Symbole
symbols = ("EURUSD","GBPUSD","USDJPY")
print("Symbols to check margin:", symbols)
# Gewinnschätzung für Kauf und Verkauf
lot=1.0
distance=300
for symbol in symbols:
    symbol_info=mt5.symbol_info(symbol)
    if symbol_info is None:
        print(symbol,"not found, skipped")
        continue
    if not symbol_info.visible:
        print(symbol, "is not visible, trying to switch on")
        if not mt5.symbol_select(symbol,True):
            print("symbol_select({}}) failed, skipped",symbol)
            continue
    point=mt5.symbol_info(symbol).point
    symbol_tick=mt5.symbol_info_tick(symbol)
    ask=symbol_tick.ask
    bid=symbol_tick.bid
    buy_profit=mt5.order_calc_profit(mt5.ORDER_TYPE_BUY,symbol,lot,ask,ask+distance*po
    if buy_profit!=None:
        print("   buy {} {} lot: profit on {} points => {} {}".format(symbol,lot,dista
    else:
        print("order_calc_profit(ORDER_TYPE_BUY) failed, error code =",mt5.last_error
    sell_profit=mt5.order_calc_profit(mt5.ORDER_TYPE_SELL,symbol,lot,bid,bid-distance*
    if sell_profit!=None:
        print("   sell {} {} lots: profit on {} points => {} {}".format(symbol,lot,dis
    else:
        print("order_calc_profit(ORDER_TYPE_SELL) failed, error code =",mt5.last_erro
    print()

# Schließen der Verbindung zum MetaTrader 5 Terminal
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Account currency: USD
Symbols to check margin: ('EURUSD', 'GBPUSD', 'USDJPY')
   buy EURUSD 1.0 lot: profit on 300 points => 300.0 USD
   sell EURUSD 1.0 lot: profit on 300 points => 300.0 USD

   buy GBPUSD 1.0 lot: profit on 300 points => 300.0 USD
   sell GBPUSD 1.0 lot: profit on 300 points => 300.0 USD
```

© 2000-2025, MetaQuotes Ltd.

```
buy USDJPY 1.0 lot: profit on 300 points => 276.54 USD
sell USDJPY 1.0 lot: profit on 300 points => 278.09 USD
```

Siehe auch

order_calc_margin, order_check

© 2000-2025, MetaQuotes Ltd.

order_check

Prüfung auf ausreichende Geldmittel zur Durchführung der gewünschten Handelsoperation. Das Prüfungsergebnis wird als Struktur MqlTradeCheckResult zurückgegeben.

```
order_check(
   request      // request Struktur
   );
```

Parameter

request

[in] MqlTradeRequest Strukturtyp, die die Handelsaktion definiert. Benötigter unbenannter Parameter. Ein Beispiel für das Ausfüllen einer Anfrage und der Inhalt der Enumeration werden im Folgenden beschrieben.

Rückgabewert

Prüfungsergebnis als Struktur MqlTradeCheckResult. Das Feld *request* in der Antwort enthält die Struktur einer an order_check() übergebenen Handelsanfrage.

Hinweis

Erfolgreiches Senden einer Anfrage **hat nicht zur Folge, dass die angeforderte Handelsoperation erfolgreich ausgeführt wird**. Die Funktion order_check ist ähnlich wie OrderCheck.

TRADE_REQUEST_ACTIONS

ID	Beschreibung
TRADE_ACTION_DEAL	Platzieren Sie eine Order für einen sofortigen Deal mit den angegebenen Parametern (setzen Sie eine Marktorder)
TRADE_ACTION_PENDING	Einen Auftrag zur Durchführung eines Deals zu bestimmten Bedingungen erteilen (Pending-Order)
TRADE_ACTION_SLTP	Stop-Loss und Take-Profit offener Positionen ändern
TRADE_ACTION_MODIFY	Parameter des zuvor platzierten Handelsauftrags ändern
TRADE_ACTION_REMOVE	Entfernen Sie zuvor aufgegebene Pending-Orders
TRADE_ACTION_CLOSE_BY	Schließen einer Position durch eine entgegengesetzte

ORDER_TYPE_FILLING

ID	Beschreibung
ORDER_FILLING_FOK	Diese Ausführungspolitik bedeutet, dass ein Auftrag nur im angegebenen Volumen ausgeführt werden kann. Wenn die erforderliche Menge eines Finanzinstruments derzeit nicht auf dem Markt verfügbar ist, wird der Auftrag nicht ausgeführt. Das gewünschte Volumen kann sich aus mehreren verfügbaren Angeboten zusammensetzen.

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
ORDER_FILLING_IOC	Eine Vereinbarung zur Ausführung eines Deals zum maximal auf dem Markt verfügbaren Volumen innerhalb des in der Order angegebenen Volumens. Wenn die Anfrage nicht vollständig erfüllt werden kann, wird ein Auftrag mit dem verfügbaren Volumen ausgeführt und das verbleibende Volumen wird annulliert.
ORDER_FILLING_RETURN	Diese Politik wird nur für Markt- (ORDER_TYPE_BUY und ORDER_TYPE_SELL), Limit- und Stop-Limit-Orders (ORDER_TYPE_BUY_LIMIT, ORDER_TYPE_SELL_LIMIT, ORDER_TYPE_BUY_STOP_LIMIT und ORDER_TYPE_SELL_STOP_LIMIT) und nur für Symbole mit den Modi Markt- oder Börsenausführung verwendet. Bei teilweiser Ausführung wird eine Markt- oder Limitorder mit dem Restvolumen nicht storniert und weiter verarbeitet. Bei der Aktivierung einer Order vom Typ ORDER_TYPE_BUY_STOP_LIMIT und ORDER_TYPE_SELL_STOP_LIMIT wird ein entsprechender Limitauftrag ORDER_TYPE_BUY_LIMIT/ORDER_TYPE_SELL_LIMIT mit dem Typ ORDER_FILLING_RETURN angelegt.

ORDER_TYPE_TIME

ID	Beschreibung
ORDER_TIME_GTC	Der Auftrag bleibt in der Warteschlange, bis er manuell storniert wird.
ORDER_TIME_DAY	Die Order ist nur während des aktuellen Handelstages aktiv.
ORDER_TIME_SPECIFIED	Der Auftrag ist bis zum angegebenen Datum aktiv
ORDER_TIME_SPECIFIED_DAY	Die Order ist bis 23:59:59 Uhr des angegebenen Tages aktiv. Wenn diese Zeit außerhalb einer Handelssitzung zu liegen scheint, wird der Verfall in der nächstgelegenen Handelszeit verarbeitet.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Kontowährung
account_currency=mt5.account_info().currency
print("Account currency:",account_currency)
```

© 2000-2025, MetaQuotes Ltd.

```
# Vorbereitung der Struktur request
symbol="USDJPY"
symbol_info = mt5.symbol_info(symbol)
if symbol_info is None:
    print(symbol, "not found, can not call order_check()")
    mt5.shutdown()
    quit()

# Wenn das Symbol im MarketWatch nicht verfügbar ist, wird es hinzugefügt
if not symbol_info.visible:
    print(symbol, "is not visible, trying to switch on")
    if not mt5.symbol_select(symbol,True):
        print("symbol_select({}}) failed, exit",symbol)
        mt5.shutdown()
        quit()

# Vorbereitung von request
point=mt5.symbol_info(symbol).point
request = {
    "action": mt5.TRADE_ACTION_DEAL,
    "symbol": symbol,
    "volume": 1.0,
    "type": mt5.ORDER_TYPE_BUY,
    "price": mt5.symbol_info_tick(symbol).ask,
    "sl": mt5.symbol_info_tick(symbol).ask-100*point,
    "tp": mt5.symbol_info_tick(symbol).ask+100*point,
    "deviation": 10,
    "magic": 234000,
    "comment": "python script",
    "type_time": mt5.ORDER_TIME_GTC,
    "type_filling": mt5.ORDER_FILLING_RETURN,
}

# Prüfen und anzeigen des Ergebnisses 'as is'
result = mt5.order_check(request)
print(result);
# Abrufen des Ergebnisses als Liste mit einer Einzeldarstellung der Elemente
result_dict=result._asdict()
for field in result_dict.keys():
    print("   {}={}".format(field,result_dict[field]))
    # falls es die Struktur einer Handelsanfrage ist, zeige es auch Element für Elemer
    if field=="request":
        traderequest_dict=result_dict[field]._asdict()
        for tradereq_filed in traderequest_dict:
            print("       traderequest: {}={}".format(tradereq_filed,traderequest_dict
```

© 2000-2025, MetaQuotes Ltd.

```
# Schließen der Verbindung zum MetaTrader 5 Terminal
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

Account currency: USD
   retcode=0
   balance=101300.53
   equity=68319.53
   profit=-32981.0
   margin=51193.67
   margin_free=17125.86
   margin_level=133.45308121101692
   comment=Done
   request=TradeRequest(action=1, magic=234000, order=0, symbol='USDJPY', volume=1.0,
       traderequest: action=1
       traderequest: magic=234000
       traderequest: order=0
       traderequest: symbol=USDJPY
       traderequest: volume=1.0
       traderequest: price=108.081
       traderequest: stoplimit=0.0
       traderequest: sl=107.98100000000001
       traderequest: tp=108.181
       traderequest: deviation=10
       traderequest: type=0
       traderequest: type_filling=2
       traderequest: type_time=0
       traderequest: expiration=0
       traderequest: comment=python script
       traderequest: position=0
       traderequest: position_by=0
```

Siehe auch

order_send, OrderCheck, Trading operation types, Trading request structure, Structure of the trading request check results, Structure of the trading request result

© 2000-2025, MetaQuotes Ltd.

order_send

Senden einer Auftrags zur Durchführung einer Handelsoperation vom Terminal zum Handelsserver. Die Funktion ist ähnlich wie OrderSend.

```
order_send(
   request      // request Struktur
   );
```

Parameter

request

[in] MqlTradeRequest Strukturtyp, die die Handelsaktion definiert. Benötigter unbenannter Parameter. Ein Beispiel für das Ausfüllen einer Anfrage und der Inhalt der Enumeration werden im Folgenden beschrieben.

Rückgabewert

Ausführungsergebnis in der Struktur MqlTradeResult. Das Feld *request* in der Antwort enthält die Struktur einer an order_send() übergebenen Handelsanforderung. Die Information über den Fehler kann über last_error() abgerufen werden.

Die Struktur MqlTradeRequest **der Handelsanfrage**

Feld	Beschreibung
action	Typ er Handelsoperation. Der Wert kann einer der Werte der Enumeration TRADE_REQUEST_ACTIONS sein.
magic	EA-ID. Ermöglicht das Erkennen von Handelsaufträgen für eine anlytische Behandlung. Jeder EA kann beim Senden einer Handelsanfrage eine eindeutige ID festlegen.
order	Order-Ticket. Erforderlich für die Änderung von Pending-Orders
symbol	Der Name des Handelsinstruments, für das der Auftrag erteilt wird. Nicht erforderlich bei der Änderung von Aufträgen und Schließung von Positionen.
volume	Gefordertes Volumen eines Geschäfts in Losen. Das tatsächliche Volumen eines Deals hängt von einem Auftragstyp.
price	Preis, zu dem ein Auftrag ausgeführt werden soll. Der Preis wird nicht bei Marktorders für Instrumente des Typs "Market Execution" (SYMBOL_TRADE_EXECUTION_MARKET) mit dem Typ TRADE_ACTION_DEAL verwendet.
stoplimit	Ein Preis, zu dem ein Pending-Limit-Order aktiviert wird, wenn der Preis den Wert von "price" erreicht (diese Bedingung ist obligatorisch). Die Pending-Order wird erst dann an das Handelssystem weitergeleitet.
sl	Ein Preis, zu dem eine Stop-Loss-Order aktiviert wird, wenn sich der Preis in eine ungünstige Richtung bewegt.
tp	Ein Preis, zu dem ein Take-Profit-Auftrag aktiviert wird, wenn sich der Preis in eine günstige Richtung bewegt.

© 2000-2025, MetaQuotes Ltd.

Feld	Beschreibung
deviation	Maximal akzeptierte Abweichung vom gewünschten Preis, angegeben in Punkten
type	Auftragsart. Der Wert kann einer der Werte der Enumeration ORDER_TYPE sein.
type_filling	Typ der Auftragsfüllung. Der Wert kann einer der Werte der Enumeration ORDER_TYPE_FILLING sein.
type_time	Auftragsart nach Zeitablauf. Der Wert kann einer der folgenden Werte der Enumeration ORDER_TYPE_TIME sein.
expiration	Ablaufzeit der Pending-Order (für den Ordertyp TIME_SPECIFIED)
comment	Kommentar zu einem Auftrag
position	Positionsticket. Füllen Sie es beim Ändern und Schließen einer Position aus, um diese eindeutig zu identifizieren. Normalerweise ist es das gleiche wie das Ticket des Auftrags, der die Position eröffnet hat.
position_by	Ticket der entgegengesetzten Position. Es wird verwendet, wenn eine Position durch eine entgegengesetzte Position geschlossen wird (die mit demselben Symbol, aber in entgegengesetzter Richtung geöffnet wird).

Hinweis

Eine Handelsanfrage durchläuft mehrere Verifizierungsstufen auf dem Handelsserver. Zuerst wird die Gültigkeit aller notwendigen Felder der *Anfrage* überprüft. Wenn keine Fehler vorliegen, nimmt der Server den Auftrag zur weiteren Bearbeitung an. Siehe die Funktionsbeschreibung OrderSend für die Einzelheiten der Ausführung von Handelsoperationen.

Beispiel:

```
import time
import MetaTrader5 as mt5

# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ", mt5.__author__)
print("MetaTrader5 package version: ", mt5.__version__)

# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Vorbereitung der Struktur request für einen Kauf
symbol = "USDJPY"
symbol_info = mt5.symbol_info(symbol)
if symbol_info is None:
    print(symbol, "not found, can not call order_check()")
    mt5.shutdown()
    quit()
```

© 2000-2025, MetaQuotes Ltd.

```
# Wenn das Symbol im MarketWatch nicht verfügbar ist, wird es hinzugefügt
if not symbol_info.visible:
    print(symbol, "is not visible, trying to switch on")
    if not mt5.symbol_select(symbol,True):
        print("symbol_select({}}) failed, exit",symbol)
        mt5.shutdown()
        quit()

lot = 0.1
point = mt5.symbol_info(symbol).point
price = mt5.symbol_info_tick(symbol).ask
deviation = 20
request = {
    "action": mt5.TRADE_ACTION_DEAL,
    "symbol": symbol,
    "volume": lot,
    "type": mt5.ORDER_TYPE_BUY,
    "price": price,
    "sl": price - 100 * point,
    "tp": price + 100 * point,
    "deviation": deviation,
    "magic": 234000,
    "comment": "python script open",
    "type_time": mt5.ORDER_TIME_GTC,
    "type_filling": mt5.ORDER_FILLING_RETURN,
}

# Senden eines Handelsauftrags
result = mt5.order_send(request)
# Prüfen des Ausführungsergebnisses
print("1. order_send(): by {} {} lots at {} with deviation={} points".format(symbol,lo
if result.retcode != mt5.TRADE_RETCODE_DONE:
    print("2. order_send failed, retcode={}".format(result.retcode))
   # Abrufen des Ergebnisses als Liste und Darstellung Element für Element
    result_dict=result._asdict()
    for field in result_dict.keys():
        print("   {}={}".format(field,result_dict[field]))
        # Wenn es die Struktur eines Handelsauftrags ist, werden die Elemente auch ein
        if field=="request":
            traderequest_dict=result_dict[field]._asdict()
            for tradereq_filed in traderequest_dict:
                print("       traderequest: {}={}".format(tradereq_filed,traderequest_
    print("shutdown() and quit")
    mt5.shutdown()
    quit()

print("2. order_send done, ", result)
print("   opened position with POSITION_TICKET={}".format(result.order))
print("   sleep 2 seconds before closing position #{}".format(result.order))
```

© 2000-2025, MetaQuotes Ltd.

```
time.sleep(2)
# Erstellen eines Schließauftrages
position_id=result.order
price=mt5.symbol_info_tick(symbol).bid
deviation=20
request={
    "action": mt5.TRADE_ACTION_DEAL,
    "symbol": symbol,
    "volume": lot,
    "type": mt5.ORDER_TYPE_SELL,
    "position": position_id,
    "price": price,
    "deviation": deviation,
    "magic": 234000,
    "comment": "python script close",
    "type_time": mt5.ORDER_TIME_GTC,
    "type_filling": mt5.ORDER_FILLING_RETURN,
}
# Senden eines Handelsauftrags
result=mt5.order_send(request)
# Prüfen des Ausführungsergebnisses
print("3. close position #{}: sell {} {} lots at {} with deviation={} points".format(p
if result.retcode != mt5.TRADE_RETCODE_DONE:
    print("4. order_send failed, retcode={}".format(result.retcode))
    print("   result",result)
else:
    print("4. position #{} closed, {}".format(position_id,result))
   # Abrufen des Ergebnisses als Liste und Darstellung Element für Element
    result_dict=result._asdict()
    for field in result_dict.keys():
        print("   {}={}".format(field,result_dict[field]))
        # Wenn es die Struktur eines Handelsauftrags ist, werden die Elemente auch ein
        if field=="request":
            traderequest_dict=result_dict[field]._asdict()
            for tradereq_filed in traderequest_dict:
                print("       traderequest: {}={}".format(tradereq_filed,traderequest_

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

```
Ergebnis
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29
1. order_send(): by USDJPY 0.1 lots at 108.023 with deviation=20 points
2. order_send done,  OrderSendResult(retcode=10009, deal=535084512, order=557416535, v
   opened position with POSITION_TICKET=557416535
   sleep 2 seconds before closing position #557416535
3. close position #557416535: sell USDJPY 0.1 lots at 108.018 with deviation=20 points
4. position #557416535 closed, OrderSendResult(retcode=10009, deal=535084631, order=55
```

© 2000-2025, MetaQuotes Ltd.

```
    retcode=10009
    deal=535084631
    order=557416654
    volume=0.1
    price=108.015
    bid=108.015
    ask=108.02
    comment=Request executed
    request_id=55
    retcode_external=0
    request=TradeRequest(action=1, magic=234000, order=0, symbol='USDJPY', volume=0.1,
        traderequest: action=1
        traderequest: magic=234000
        traderequest: order=0
        traderequest: symbol=USDJPY
        traderequest: volume=0.1
        traderequest: price=108.018
        traderequest: stoplimit=0.0
        traderequest: sl=0.0
        traderequest: tp=0.0
        traderequest: deviation=20
        traderequest: type=1
        traderequest: type_filling=2
        traderequest: type_time=0
        traderequest: expiration=0
        traderequest: comment=python script close
        traderequest: position=557416535
        traderequest: position_by=0
```

Siehe auch

order_check, OrderSend,Trading operation types, Trading request structure, Structure of the trading request check results, Structure of the trading request result

© 2000-2025, MetaQuotes Ltd.

positions_total

Abrufen der Anzahl der offenen Positionen

```
positions_total()
```

Rückgabewert

Integer Wert

Hinweis

Die Funktion ist ähnlich PositionsTotal.

Beispiel:

```
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Prüfen auf die Existenz von offenen Positionen
positions_total=mt5.positions_total()
if positions_total>0:
    print("Total positions=",positions_total)
else:
    print("Positions not found")

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()
```

Siehe auch

positions_get, orders_total

© 2000-2025, MetaQuotes Ltd.

positions_get

Abrufen der offenen Positionen, mit der Fähigkeit nach Symbol oder Ticket zu filtern. Es gibt drei Aufrufoptionen.

Aufruf ohne Parameter. Rückgabe der offenen Positionen aller Symbole.

```
positions_get()
```

Ein Abruf der offenen Positionen mit angegebenem Symbol.

```
positions_get(
   symbol="SYMBOL"       // Symbolname
)
```

Ein Abruf der offenen Positionen der angegebenen Symbolgruppe.

```
positions_get(
   group="GROUP"         // Filter zur Positionsauswahl nach Symbolen
)
```

Aufruf unter Angabe des Positionsticket.

```
positions_get(
   ticket=TICKET         // Ticket
)
```

Parameter

symbol="SYMBOL"

[in] Symbolname. Optionale benannte Parameter. Wein ein Symbol angegeben ist, wird das *Ticket* ignoriert.

group="GROUP"

[in] Der Filter für die Gruppe der angeforderten Symbole. Optionale benannte Parameter. Wenn die Gruppe angegeben wurde, liefert die Funktion nur die Positionen, die die angegebenen Kriterien für das Symbol erfüllen.

ticket=TICKET

[in] Position ticket (POSITION_TICKET). Optionale benannte Parameter.

Rückgabewert

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion ermöglicht den Empfang aller offenen Positionen innerhalb eines Anrufs, ähnlich wie bei dem Funktionspaar PositionsTotal und PositionSelect.

Der Parameter *group* kann mehrere durch Komma getrennte Bedingungen enthalten. Eine Bedingung kann mit '*' als Maske gesetzt werden. Das logische Negationssymbol '!' kann für einen Ausschluss verwendet werden. Alle Bedingungen werden sequentiell angewendet, d.h. Bedingungen der Aufnahme in eine Gruppe sollten zuerst angegeben werden, gefolgt von einer Ausschlussbedingung.

© 2000-2025, MetaQuotes Ltd.

Zum Beispiel bedeutet *group*="*, !EUR", dass zuerst alle Positionen aller Symbole ausgewählt und diejenigen, die "EUR" in den Symbolnamen enthalten, danach ausgeschlossen werden.

Beispiel:

```
import MetaTrader5 as mt5
import pandas as pd
pd.set_option('display.max_columns', 500) # darzustellende Spaltenanzahl
pd.set_option('display.width', 1500)      # maximale darzustellende Tabellenbreite
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
print()
# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abruf der offenen USDCHF-Positionen
positions=mt5.positions_get(symbol="USDCHF")
if positions==None:
    print("No positions on USDCHF, error code={}".format(mt5.last_error()))
elif len(positions)>0:
    print("Total positions on USDCHF =",len(positions))
    # Anzeige aller offenen Positionen
    for position in positions:
        print(position)

# Abrufen der Positionsliste aller Symbole mit "*USD*" im Namen
usd_positions=mt5.positions_get(group="*USD*")
if usd_positions==None:
    print("No positions with group=\"*USD*\", error code={}".format(mt5.last_error()))
elif len(usd_positions)>0:
    print("positions_get(group=\"*USD*\")={}".format(len(usd_positions)))
    # Anzeige dieser Positionen als Tabelle mittels pandas.DataFrame
    df=pd.DataFrame(list(usd_positions),columns=usd_positions[0]._asdict().keys())
    df['time'] = pd.to_datetime(df['time'], unit='s')
    df.drop(['time_update', 'time_msc', 'time_update_msc', 'external_id'], axis=1, inp
    print(df)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()


Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

positions_get(group="*USD*")=5
```

© 2000-2025, MetaQuotes Ltd.

	ticket	time	type	magic	identifier	reason	volume	price_open
0	548297723	2020-03-18 15:00:55	1	0	548297723	3	0.01	1.09301
1	548655158	2020-03-18 20:31:26	0	0	548655158	3	0.01	1.08676
2	548663803	2020-03-18 20:40:04	0	0	548663803	3	0.01	1.08640
3	548847168	2020-03-19 01:10:05	0	0	548847168	3	0.01	1.09545
4	548847194	2020-03-19 01:10:07	0	0	548847194	3	0.02	1.09536

Siehe auch

positions_total, orders_get

© 2000-2025, MetaQuotes Ltd.

history_orders_total

Abrufen der Anzahl der Orders in der Handelshistorie innerhalb des angegebenen Intervalls

```
history_orders_total(
   date_from,     // Zeitpunkt, ab dem die Orders abgerufen werden
   date_to        // Zeitpunkt, bis zu dem die Orders angefordert werden
   )
```

Parameter

date_from

[in] Zeitpunkt, ab dem die Orders angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

date_to

[in] Zeitpunkt, bis zu dem die Orders angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

Rückgabewert

Integer Wert

Hinweis

Die Funktion ist ähnlich wie HistoryOrdersTotal.

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Anzahl der Orders in der Historie
from_date=datetime(2020,1,1)
to_date=datetime.now()
history_orders=mt5.history_orders_total(from_date, datetime.now())
if history_orders>0:
    print("Total history orders=",history_orders)
else:
    print("Orders not found in history")

# Schließen der Verbindung zum MetaTrader 5
```

© 2000-2025, MetaQuotes Ltd.

```
mt5.shutdown()
```

Siehe auch

history_orders_get, history_deals_total

© 2000-2025, MetaQuotes Ltd.

history_orders_get

Abrufen der Orders in der Handelshistorie, mit der Fähigkeit nach Ticket oder Position zu filtern Es gibt drei Aufrufoptionen.

Aufruf unter Angabe eines Zeitintervalls. Rückgabe aller Orders innerhalb des angegebenen Zeitraums.

```
history_orders_get(
   date_from,                 // Zeitpunkt, ab dem die Orders abgerufen werden
   date_to,                   // Zeitpunkt, bis zu dem die Orders angefordert werden
   group="GROUP"         // Filter zu Selektion der Orders nach Symbolen
   )
```

Aufruf unter Angabe des Orderticket. Rückgabe einer Order mit dem angegebenen Ticket.

```
history_orders_get(
   ticket=TICKET         // Orderticket
)
```

Aufruf unter Angabe des Positionsticket. Rückgabe aller Orders mit dem angebenden Positionsticket der Eigenschaft ORDER_POSITION_ID.

```
history_orders_get(
   position=POSITION     // Positionsticket
)
```

Parameter

date_from

[in] Zeitpunkt, ab dem die Orders angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter werden als erstes angegeben.

date_to

[in] Zeitpunkt, bis zu dem die Orders angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter werden als zweites angegeben.

group="GROUP"

[in] Der Filter für die Gruppe der angeforderten Symbole. Optionale benannte Parameter. Wenn die Gruppe angegeben ist, gibt die Funktion nur Orders zurück, die ein bestimmtes Kriterium für einen Symbolnamen erfüllen.

ticket=TICKET

[in] Order-Ticket, das erhalten werden sollte. Optionaler Parameter. Wenn nicht angegeben, wird der Filter nicht angewendet.

position=POSITION

[in] Ticket einer Position (gespeichert in ORDER_POSITION_ID), die alle Orders erhalten sollen. Optionaler Parameter. Wenn nicht angegeben, wird der Filter nicht angewendet.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion ermöglicht den Empfang aller Orders der Historie innerhalb eines bestimmten Zeitraums in einem einzigen Aufruf, ähnlich wie bei HistoryOrdersTotal und HistoryOrderSelect.

Der Parameter *group* kann mehrere durch Komma getrennte Bedingungen enthalten. Eine Bedingung kann mit '*' als Maske gesetzt werden. Das logische Negationssymbol '!' kann für einen Ausschluss verwendet werden. Alle Bedingungen werden sequentiell angewendet, d.h. Bedingungen der Aufnahme in eine Gruppe sollten zuerst angegeben werden, gefolgt von einer Ausschlussbedingung. Zum Beispiel bedeutet *group*="*, !EUR", dass zuerst alle Deals aller Symbole ausgewählt und diejenigen, die "EUR" in den Symbolnamen enthalten, danach ausgeschlossen werden.

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
import pandas as pd
pd.set_option('display.max_columns', 500) # darzustellende Spaltenanzahl
pd.set_option('display.width', 1500)      # maximale darzustellende Tabellenbreite
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
print()
# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Anzahl der Orders in der Historie
from_date=datetime(2020,1,1)
to_date=datetime.now()
history_orders=mt5.history_orders_get(from_date, to_date, group="*GBP*")
if history_orders==None:
    print("No history orders with group=\"*GBP*\", error code={}".format(mt5.last_erro
elif len(history_orders)>0:
    print("history_orders_get({}, {}, group=\"*GBP*\")={}".format(from_date,to_date,le
print()

# Anzeige aller historischen Orders nach dem Positionsticket
position_id=530218319
position_history_orders=mt5.history_orders_get(position=position_id)
if position_history_orders==None:
    print("No orders with position #{}".format(position_id))
    print("error code =",mt5.last_error())
elif len(position_history_orders)>0:
    print("Total history orders on position #{}: {}".format(position_id,len(position_h
   # Anzeige aller historischen Orders mit dem angegebenen Positionsticket
```

© 2000-2025, MetaQuotes Ltd.

```
    for position_order in position_history_orders:
        print(position_order)
    print()
    # Darstellung der Orders als Tabelle mittels pandas.DataFrame
    df=pd.DataFrame(list(position_history_orders),columns=position_history_orders[0]._
    df.drop(['time_expiration','type_time','state','position_by_id','reason','volume_
    df['time_setup'] = pd.to_datetime(df['time_setup'], unit='s')
    df['time_done'] = pd.to_datetime(df['time_done'], unit='s')
    print(df)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()

Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

history_orders_get(2020-01-01 00:00:00, 2020-03-25 17:17:32.058795, group="*GBP*")=14

Total history orders on position #530218319: 2
TradeOrder(ticket=530218319, time_setup=1582282114, time_setup_msc=1582282114681, time
TradeOrder(ticket=535548147, time_setup=1583176242, time_setup_msc=1583176242265, time

      ticket           time_setup  time_setup_msc           time_done  time_done_msc  t
0  530218319 2020-02-21 10:48:34   1582282114681 2020-02-21 16:49:37  1582303777582
1  535548147 2020-03-02 19:10:42   1583176242265 2020-03-02 19:10:42  1583176242265
```

Siehe auch

history_deals_total, history_deals_get

© 2000-2025, MetaQuotes Ltd.

history_deals_total

Abrufen der Anzahl der Deals aus der Handelshistorie innerhalb des angegebenen Zeitraums.

```
history_deals_total(
   date_from,     // Zeitpunkt, ab dem die Deals abgerufen werden
   date_to        // Zeitpunkt, bis zu dem die Deals abgerufen werden
   )
```

Parameter

date_from

[in] Zeitpunkt, ab dem die Deals abgerufen werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

date_to

[in] Zeitpunkt, bis zu dem die Deals abgerufen werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter.

Rückgabewert

Integer Wert

Hinweis

Die Funktion ist ähnlich HistoryDealsTotal.

Beispiel:

```
from datetime import datetime
import MetaTrader5 as mt5
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)

# Verbindung herstellen zum MetaTrader 5 Terminal
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Anzahl der Deals in der Historie
from_date=datetime(2020,1,1)
to_date=datetime.now()
deals=mt5.history_deals_total(from_date, to_date)
if deals>0:
    print("Total deals=",deals)
else:
    print("Deals not found in history")

# Schließen der Verbindung zum MetaTrader 5
```

© 2000-2025, MetaQuotes Ltd.

```
mt5.shutdown()
```

Siehe auch

history_deals_get, history_orders_total

© 2000-2025, MetaQuotes Ltd.

history_deals_get

Abrufen der Deals aus der Handelshistorie innerhalb des angegebenen Intervalls mit der Möglichkeit, nach Ticket oder Position zu filtern.

Aufruf unter Angabe eines Zeitintervalls. Rückgabe aller Deals, die in das angegebene Intervall fallen.

```
history_deals_get(
   date_from,                 // Zeitpunkt, ab dem die Deals abgerufen werden
   date_to,                   // Zeitpunkt, bis zu dem die Deals abgerufen werden
   group="GROUP"          // Filter zur Auswahl der Deals nach einem Symbol
   )
```

Aufruf unter Angabe des Auftragsticket. Rückgabe aller Deals mit dem angegebenen Auftragsticket der Eigenschaft DEAL_ORDER.

```
history_deals_get(
   ticket=TICKET          // Orderticket
)
```

Aufruf unter Angabe des Positionsticket. Rückgabe aller Deals mit dem angegebenen Positionsticket der Eigenschaft DEAL_POSITION_ID.

```
history_deals_get(
   position=POSITION      // Positionsticket
)
```

Parameter

date_from

[in] Zeitpunkt, ab dem die Orders angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter werden als erstes angegeben.

date_to

[in] Zeitpunkt, bis zu dem die Orders angefordert werden. Ausgewählt und gesetzt durch ein Objekt vom Typ 'datetime' oder der Anzahl der Sekunden seit dem 1970.01.01. Benötigter unbenannter Parameter werden als zweites angegeben.

group="GROUP"

[in] Der Filter für die Gruppe der angeforderten Symbole. Optionale benannte Parameter. Wenn die Gruppe angegeben wurde, liefert die Funktion nur die Deals, die die angegebenen Kriterien für das Symbol erfüllen.

ticket=TICKET

[in] Ticket eines Deals (gespeichert in DEAL_ORDER), die alle Deals erhalten sollen. Optionaler Parameter. Wenn nicht angegeben, wird der Filter nicht angewendet.

position=POSITION

[in] Ticket einer Position (gespeichert in DEAL_POSITION_ID), die alle Positionen erhalten sollen. Optionaler Parameter. Wenn nicht angegeben, wird der Filter nicht angewendet.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Rückgabe der Information als Struktur eines benannten Tupels (namedtuple). Im Falle eines Fehlers wird nichts zurückgegeben. Die Information über den Fehler kann über last_error() abgerufen werden.

Hinweis

Die Funktion ermöglicht den Erhalt aller Deals in der Historie innerhalb eines angegebenen Zeitraums in einem einzigen Aufruf, ähnlich wie bei HistoryDealsTotal und HistoryDealSelect.

Der Parameter *group* erlaubt das Aussortieren von Deals nach Symbolen. '*' kann am Anfang und am Ende einer Zeichenkette verwendet werden.

Der Parameter *group* kann mehrere durch Komma getrennte Bedingungen enthalten. Eine Bedingung kann mit '*' als Maske gesetzt werden. Das logische Negationssymbol '!' kann für einen Ausschluss verwendet werden. Alle Bedingungen werden sequentiell angewendet, d.h. Bedingungen der Aufnahme in eine Gruppe sollten zuerst angegeben werden, gefolgt von einer Ausschlussbedingung. Zum Beispiel bedeutet *group*="*, !EUR", dass zuerst alle Deals aller Symbole ausgewählt und diejenigen, die "EUR" in den Symbolnamen enthalten, danach ausgeschlossen werden.

Beispiel:

```
import MetaTrader5 as mt5
from datetime import datetime
import pandas as pd
pd.set_option('display.max_columns', 500) # darzustellende Spaltenanzahl
pd.set_option('display.width', 1500)      # maximale darzustellende Tabellenbreite
# Datenanzeige des Pakets von MetaTrader 5
print("MetaTrader5 package author: ",mt5.__author__)
print("MetaTrader5 package version: ",mt5.__version__)
print()
# Verbindung zum MetaTrader 5 Terminal herstellen
if not mt5.initialize():
    print("initialize() failed, error code =",mt5.last_error())
    quit()

# Abrufen der Anzahl der Deals in der Historie
from_date=datetime(2020,1,1)
to_date=datetime.now()
# Abrufen der Deals der Symbole, deren Name "GBP" beinhalten innerhalb des angegebener
deals=mt5.history_deals_get(from_date, to_date, group="*GBP*")
if deals==None:
    print("No deals with group=\"*USD*\", error code={}".format(mt5.last_error()))
elif len(deals)> 0:
    print("history_deals_get({}, {}, group=\"*GBP*\")={}".format(from_date,to_date,ler

# Abrufen der Deals der Symbole, deren Name weder "EUR" noch "GBP" beinhalten
deals = mt5.history_deals_get(from_date, to_date, group="*,!*EUR*,!*GBP*")
if deals == None:
    print("No deals, error code={}".format(mt5.last_error()))
elif len(deals) > 0:
    print("history_deals_get(from_date, to_date, group=\"*,!*EUR*,!*GBP*\") =", len(de
```

© 2000-2025, MetaQuotes Ltd.

```
    # Anzeige der erhaltenen Deals 'as is'
    for deal in deals:
        print("  ",deal)
    print()
    # Anzeige dieser Deals als Tabelle mittels pandas.DataFrame
    df=pd.DataFrame(list(deals),columns=deals[0]._asdict().keys())
    df['time'] = pd.to_datetime(df['time'], unit='s')
    print(df)
print("")

# Abrufen aller Deals mit Bezug zur Position #530218319
position_id=530218319
position_deals = mt5.history_deals_get(position=position_id)
if position_deals == None:
    print("No deals with position #{}".format(position_id))
    print("error code =", mt5.last_error())
elif len(position_deals) > 0:
    print("Deals with position id #{}: {}".format(position_id, len(position_deals)))
    # Anzeige dieser Deals als Tabelle mittels pandas.DataFrame
    df=pd.DataFrame(list(position_deals),columns=position_deals[0]._asdict().keys())
    df['time'] = pd.to_datetime(df['time'], unit='s')
    print(df)

# Schließen der Verbindung zum MetaTrader 5
mt5.shutdown()

Ergebnis:
MetaTrader5 package author:  MetaQuotes Software Corp.
MetaTrader5 package version:  5.0.29

history_deals_get(from_date, to_date, group="*GBP*") = 14

history_deals_get(from_date, to_date, group="*,!*EUR*,!*GBP*") = 7
   TradeDeal(ticket=506966741, order=0, time=1582202125, time_msc=1582202125419, type=
   TradeDeal(ticket=507962919, order=530218319, time=1582303777, time_msc=158230377758
   TradeDeal(ticket=513149059, order=535548147, time=1583176242, time_msc=158317624226
   TradeDeal(ticket=516943494, order=539349382, time=1583510003, time_msc=158351000389
   TradeDeal(ticket=516943915, order=539349802, time=1583510025, time_msc=158351002505
   TradeDeal(ticket=517139682, order=539557870, time=1583520201, time_msc=158352020122
   TradeDeal(ticket=517139716, order=539557909, time=1583520202, time_msc=158352020297

      ticket      order                 time       time_msc  type  entry  magic  positi
0  506966741          0 2020-02-20 12:35:25  1582202125419     2      0      0
1  507962919  530218319 2020-02-21 16:49:37  1582303777582     0      0      0    5302
2  513149059  535548147 2020-03-02 19:10:42  1583176242265     1      1      0    5302
3  516943494  539349382 2020-03-06 15:53:23  1583510003895     1      0      0    5393
4  516943915  539349802 2020-03-06 15:53:45  1583510025054     0      0      0    5393
5  517139682  539557870 2020-03-06 18:43:21  1583520201227     0      1      0    5393
6  517139716  539557909 2020-03-06 18:43:22  1583520202971     1      1      0    5393
```

© 2000-2025, MetaQuotes Ltd.

```
Deals with position id #530218319: 2
      ticket      order                time       time_msc  type  entry  magic  positi
0  507962919  530218319 2020-02-21 16:49:37  1582303777582     0      0      0    5302
1  513149059  535548147 2020-03-02 19:10:42  1583176242265     1      1      0    5302
```

Siehe auch

history_deals_total, history_orders_get

© 2000-2025, MetaQuotes Ltd.

ONNX-Modelle im maschinellen Lernen

ONNX (Open Neural Network Exchange) ist ein Open-Source-Format für Modelle des maschinellen Lernens. Dieses Projekt hat mehrere große Vorteile:

- ONNX wird von großen Unternehmen wie Microsoft, Facebook, Amazon und anderen Partnern unterstützt.
- Sein offenes Format ermöglicht Formatkonvertierungen zwischen verschiedenen Machine-Learning-Toolkits, während die ONNXMLTools von Microsoft die Konvertierung von Modellen in das ONNX-Format ermöglichen.
- MQL5 bietet automatische Datentypkonvertierung für Modelleingaben und -ausgaben, wenn der übergebene Parametertyp nicht mit dem Modell übereinstimmt.
- ONNX-Modelle können mit verschiedenen maschinellen Lerntools erstellt werden. Sie werden derzeit in Caffe2, Microsoft Cognitive Toolkit, MXNet, PyTorch und OpenCV unterstützt. Schnittstellen für andere gängige Frameworks und Bibliotheken sind ebenfalls verfügbar.
- Mit der MQL5-Sprache können Sie ein ONNX-Modell in einer Handelsstrategie implementieren und es zusammen mit allen Vorteilen der MetaTrader 5-Plattform für effiziente Abläufe in der verwenden Finanzmärkte.
- Bevor Sie ein Modell für den Live-Handel optimieren, können Sie das Modellverhalten anhand historischer Daten testen im Strategietester, ohne Tools von Drittanbietern zu verwenden.

MQL5 bietet die folgenden Funktionen für die Arbeit mit ONNX:

Funktion	Aktion
OnnxCreate	Erstellen einer ONNX-Sitzung, laden eines Modells aus einer *.onnx-Datei
OnnxCreateFromBuffer	Erstellen einer ONNX-Sitzung, laden eines Modells aus einem Datenarray
OnnxRelease	Schließen Sie eine ONNX-Sitzung
OnnxRun	Ausführen eines ONNX-Modells
OnnxGetInputCount	Ermitteln der Anzahl der Eingänge eines ONNX-Modells
OnnxGetOutputCount	Ermitteln der Anzahl der Ausgänge eines ONNX-Modells
OnnxGetInputName	Abrufen des Namens des Eingangs eines Modells anhand des Index
OnnxGetOutputName	Abrufen des Namens des Ausgangs eines Modells anhand des Index
OnnxGetInputTypeInfo	Abrufen der Beschreibung des Eingangstyps aus dem Modell
OnnxGetOutputTypeInfo	Abrufen der Beschreibung des Ausgangstyps aus dem Modell
OnnxSetInputShape	Festlegen der Form der Eingangsdaten eines Modells anhand des Index
OnnxSetOutputShape	Festlegen der Form der Ausgangsdaten eines Modells anhand des Index

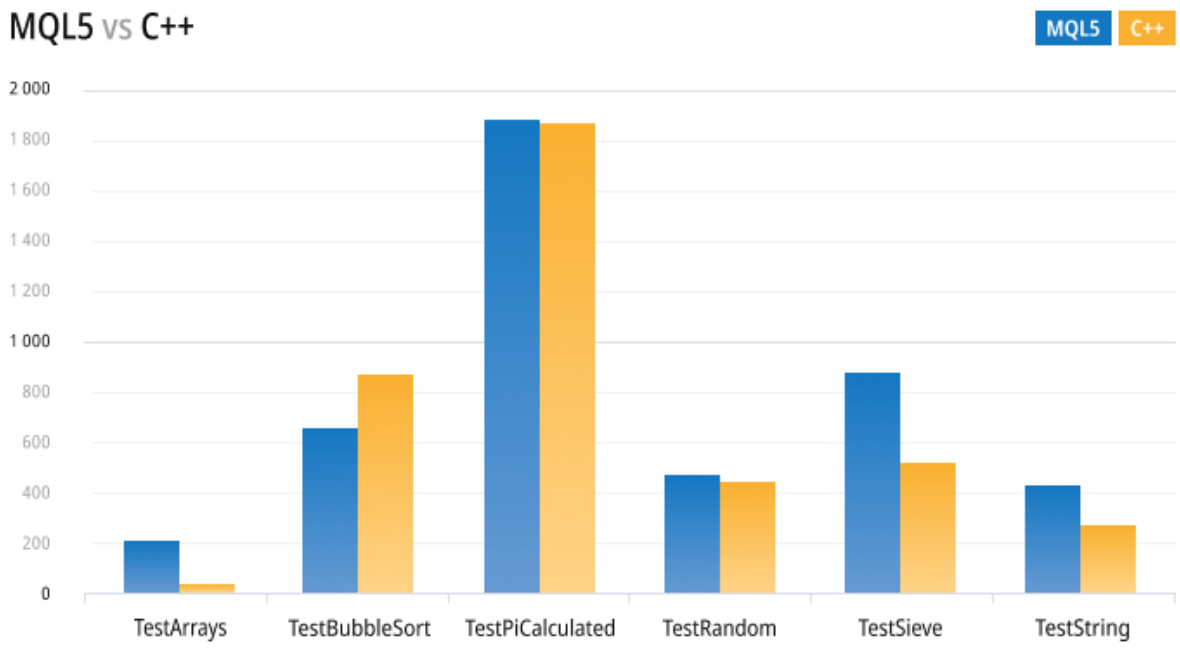
© 2000-2025, MetaQuotes Ltd.

ONNX-Unterstützung in MQL5

ONNX ist ein offenes Format, das zur Darstellung von Modellen für maschinelles Lernen entwickelt wurde. Dieser Standard definiert einen gemeinsamen Satz von Operatoren und ein gemeinsames Dateiformat, um es Entwicklern zu ermöglichen, Modelle mit verschiedenen Frameworks, Tools, Laufzeiten und Compilern zu verwenden.

Das offene ONNX-Format ermöglicht somit den Empfang und die Übertragung von Machine-Learning-Modellen zwischen verschiedenen Plattformen und Machine-Learning-Toolkits. ONNX ist in der Sprache MQL5 implementiert, damit KI-Entwickler die erstellten Modelle in der leistungsstarken Ausführungsumgebung der MetaTrader 5-Plattform ausführen können.

Die Ausführungsgeschwindigkeit von MQL5 ist mit der von C++-Anwendungen vergleichbar. Das zeigen die Ausführungsergebnisse von Standardtests mit MQL5 und C++. Je niedriger der Balken, desto weniger Zeit (in Millisekunden) wird für die Ausführung benötigt und desto besser ist das Ergebnis. Die Tests wurden auf Windows 10 (build 17763) x64, Xeon E5-2630 v4 @ 2.20GHz, Speicher: 65457 Mb durchgeführt.



Die neuen asynchronen Handelsoperationen und die native ONNX-Unterstützung bieten Ihnen neue Möglichkeiten, die bisher nur einer Handvoll professioneller KI-Entwickler und institutioneller Händler zur Verfügung standen. Die ONNX-Unterstützung in MQL5 ermöglicht es Händlern, Modelle für den Finanzmarkthandel in ihrer bevorzugten Entwicklungsumgebung zu trainieren und dann mit niedrigen Netzwerkkosten, hohen Orderbuch-Aktualisierungsgeschwindigkeiten und asynchroner Orderübermittlung zu handeln.

Derzeit wird ONNX von Partnerunternehmen wie Microsoft, Facebook, Amazon und anderen entwickelt und gepflegt, was die weitere Entwicklung dieses offenen Projekts garantiert.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Format-Konvertierung

ONNX ist ein offenes Format, das die Verwendung von Modellen aus verschiedenen Machine Learning Toolkits ermöglicht. Dieses Format wird von vielen Frameworks unterstützt, darunter Chainer, Caffee2 und PyTorch.

Eines der verbreitetsten Tools zur Konvertierung von Modellen in das ONNX-Format ist Microsofts ONNXMLTools.

Anweisungen zur Installation und Verwendung von ONNXMLTools sind im GitHub repo verfügbar. Die folgenden Toolkits werden derzeit unterstützt:

- Keras (ein Wrapper des keras2onnx converter)
- Tensorflow (ein Wrapper des tf2onnx converter)
- scikit-learn (ein Wrapper des skl2onnx converter)
- Apple Core ML
- Spark ML (experimentell)
- LightGBM
- libscm;
- XGBoost;
- H2O
- CatBoost

ONNXMLTools kann einfach installiert werden. Details zur Installation und Beispiele zur Modellkonvertierung finden Sie auf der Projektseite unter https://github.com/onnx/onnxmltools#install.

© 2000-2025, MetaQuotes Ltd.

Automatische Konvertierung von Eingangs- und Ausgangswerten bei der Ausführung von ONNX-Modellen

Die aktuelle ONNX-Version in MQL5 unterstützt nur Tensoren für Eingangs/Ausgangs-Werte. Tensoren sind Datenarrays mit den Elementen der folgenden Datentypen:

ONNX-Typ	Entspricht dem MQL5-Typ
ONNX_DATA_TYPE_BOOL	Bool
ONNX_DATA_TYPE_FLOAT	float
ONNX_DATA_TYPE_UINT8	uchar
ONNX_DATA_TYPE_INT8	char
ONNX_DATA_TYPE_UINT16	ushort
ONNX_DATA_TYPE_INT16	short
ONNX_DATA_TYPE_INT32	int
ONNX_DATA_TYPE_INT64	long
ONNX_DATA_TYPE_FLOAT16	—
ONNX_DATA_TYPE_DOUBLE	double
ONNX_DATA_TYPE_UINT32	uint
ONNX_DATA_TYPE_UINT64	ulong
ONNX_DATA_TYPE_COMPLEX64	—
ONNX_DATA_TYPE_COMPLEX128	complex
ONNX_DATA_TYPE_BFLOAT16	—
ONNX_DATA_TYPE_STRING	—

Nur Arrays, Vektoren und Matrizen (wir bezeichnen sie als **Daten**) können in ONNX-Modellen als Eingangs-/Ausgangswerte eingegeben werden.

Wenn die Parametertypen nicht mit dem Parametertyp des ONNX-Modells übereinstimmen und OnnxRun ohne das Flag ONNX_NO_CONVERSION aufgerufen wird, wird eine automatische Datenkonvertierung durchgeführt. Automatische Konvertierung bedeutet, dass vor der Ausführung eines ONNX-Modells die **Nutzerdaten** in ONNX-Tensoren mit der entsprechenden Konvertierung kopiert werden.

Wenn ein ONNX-Modell ohne die Autokonvertierung ausgeführt wird, wird das Modell mit den **Daten** ohne zusätzliches Kopieren berechnet.

> **WICHTIG!** Die Autokonvertierung kontrolliert keinen Überlauf (Kürzen), daher sollten Sie die Daten und die Datentypen, die in das ONNX-Modell eingegeben werden, sorgfältig überwachen.

Autokonvertierung unterstützt die folgenden ONNX-Typen:

© 2000-2025, MetaQuotes Ltd.

- ONNX_DATA_TYPE_BOOL
- ONNX_DATA_TYPE_FLOAT
- ONNX_DATA_TYPE_UINT8
- ONNX_DATA_TYPE_INT8
- ONNX_DATA_TYPE_UINT16
- ONNX_DATA_TYPE_INT16
- ONNX_DATA_TYPE_INT32
- ONNX_DATA_TYPE_INT64
- ONNX_DATA_TYPE_FLOAT16
- ONNX_DATA_TYPE_DOUBLE
- ONNX_DATA_TYPE_UINT32
- ONNX_DATA_TYPE_UINT64
- ONNX_DATA_TYPE_COMPLEX64
- ONNX_DATA_TYPE_COMPLEX128

Nichtunterstütze Typen

- ONNX_DATA_TYPE_BFLOAT16
- ONNX_DATA_TYPE_STRING

Autokonvertierungsregeln der Tensortypen

Wenn der MQL5-Typ nicht in der Liste der vom Modell unterstützten Typen enthalten ist, gibt die Ausführung des ONNX-Modells den Fehler ERR_ONNX_NOT_SUPPORTED zurück (Fehlercode 5802).

Hinweis: Bei der Autokonvertierung wird der Typ color als uint verarbeitet, während datetime als long verarbeitet wird.

Autokonvertierung von Eingangswerten

ONNX-Typ (Tensorelementtyp)	Von der Autokonvertierung unterstützter MQL5-Typ
ONNX_DATA_TYPE_BOOL	bool, char, uchar, short, ushort, int, color, uint, datetime, long, folat, double, complex Bei der Konvertierung werden **Daten**-Elemente durch einen einfachen Vergleich mit 0 überprüft
ONNX_DATA_TYPE_FLOAT16	float, double
ONNX_DATA_TYPE_FLOAT	char, uchar, short, ushort, int, color, uint, datetime, long, ulong, float, double
ONNX_DATA_TYPE_UINT8	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT8	Siehe ONNX_DATA_TYPE_FLOAT

© 2000-2025, MetaQuotes Ltd.

ONNX-Typ (Tensorelementtyp)	Von der Autokonvertierung unterstützter MQL5-Typ
ONNX_DATA_TYPE_UINT16	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT16	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT32	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT64	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_DOUBLE	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_UINT32	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_UINT64	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_COMPLEX64	complex
ONNX_DATA_TYPE_COMPLEX128	complex

Autokonvertierung von Ausgangswerten

ONNX-Typ (Tensorelementtyp)	Von der Autokonvertierung unterstützter MQL5-Typ
ONNX_DATA_TYPE_BOOL	bool, char, uchar, short, ushort, int, color, uint, datetime, long, folat, double, complex Wenn das Tensorelement Null ist, wird das Datenelement auf 0 gesetzt, andernfalls ist der Wert 1
ONNX_DATA_TYPE_FLOAT16	float, double
ONNX_DATA_TYPE_FLOAT	char, uchar, short, ushort, int, color, uint, datetime, long, ulong, float, double
ONNX_DATA_TYPE_UINT8	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT8	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_UINT16	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT16	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT32	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_INT64	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_DOUBLE	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_UINT32	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_UINT64	Siehe ONNX_DATA_TYPE_FLOAT
ONNX_DATA_TYPE_COMPLEX64	complex

© 2000-2025, MetaQuotes Ltd.

ONNX-Typ (Tensorelementtyp)	Von der Autokonvertierung unterstützter MQL5-Typ
ONNX_DATA_TYPE_COMPLEX128	complex

Siehe auch

Type Casting

© 2000-2025, MetaQuotes Ltd.

Erstellen eines Modells

Mehrere Methoden sind verfügbar, um ein fertiges Modell im ONNX-Format zu erhalten. Die verbreitete Bibliothek ONNX Modell Zoo enthält mehrere vortrainierte ONNX-Modelle für verschiedene Aufgabentypen. Der Vorteil dieser Zusammenstellung ist, dass das Notizbuch jedes Modells Links zum Trainingsdatensatz und Verweise auf die Originalarbeit enthält, die die Modellarchitektur beschreibt.

Die meisten Frameworks für maschinelles Lernen verwenden Python. Um die ONNX-Laufzeitumgebung für Python zu installieren, verwenden Sie einen der folgenden Befehle:

```
pip install onnxruntime        # CPU build
pip install onnxruntime-gpu    # GPU build
```

Um die ONNX-Laufzeit in Python aufzurufen, verwenden Sie den folgenden Befehl

```
import onnxruntime
Session = onnxruntime.InferenceSession("Pfad zum Modell")
```

Für die Eingänge und Ausgänge des Modells sehen Sie sich die Dokumentation des jeweiligen Modells an. Es können auch Visualisierungswerkzeuge verwendet werden, um das Modell zu betrachten, wie Netron oder WinML Dashboard. In der ONNX-Laufzeit können Sie auch die Metadaten eines Modells sowie seine Eingänge und Ausgänge abfragen:

```
results = session.run(["output1", "output2"], {
                      "input1": indata1, "input2": indata2})
results = session.run([], {"input1": indata1, "input2": indata2})
```

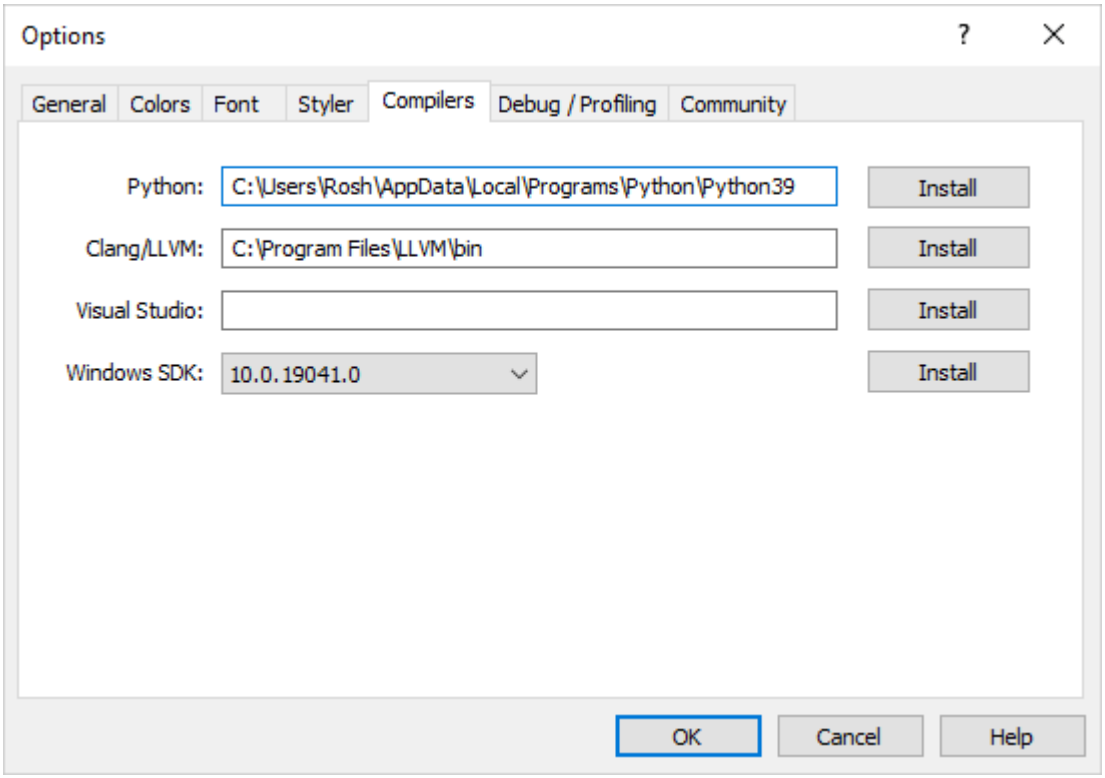
Sie können ONNX-Modelle direkt im MetaTrader 5-Terminal oder im MetaEditor mit Python erstellen.

Python im MetaTrader 5

MetaTrader 5 bietet sofort einsatzbereite Unterstützung für Python-Skripte. Um diese Vorgänge zu ermöglichen, stellen die Terminalentwickler das MetaTrader5-Modul für Python bereit: https://pypi.org/project/MetaTrader5.

In der integrierten Entwicklungsumgebung MetaEditor können Sie nicht nur Anwendungen in MQL5 erstellen, sondern auch Python-Skripte direkt aus dem Editor ausführen. Geben Sie dazu in den MetaEditor-Einstellungen den Pfad zur ausführbaren Datei an:

© 2000-2025, MetaQuotes Ltd.



Wenn Python nicht auf Ihrem Computer installiert ist, klicken Sie auf Installieren, um die Installationsdatei herunterzuladen.

Sie können ein Python-Skript in MetaEdtior erstellen oder es in den Datenordner des Terminals hochladen und es sofort mit der Taste F7 (Kompilieren) ausführen. Dadurch wird das MetaTrader 5-Terminal geöffnet, wobei das Skript auf dem aktuellen Diagramm ausgeführt wird. Meldungen von der Python-Konsole (stdout, stderr) werden unter dem Abschnitt Fehler angezeigt.

Operationen mit Modellen in MetaTrader 5

Mit der MQL5-Sprache können Sie ONNX-Modelle direkt im MetaTrader 5-Terminal ausführen. Dies erfolgt in drei Schritten:

1. Trainieren Sie das Modell auf der Plattform eines Drittanbieters, z. B. Python.
2. Konvertieren Sie das Modell zu ONNX.
3. Fügen Sie das ONNX-Modell mithilfe der ONNX-Funktion in einen Expert Advisor ein und führen Sie es im MetaTrader 5-Terminal aus.

Die Python-Integration in MQL5 ermöglicht die Ausführung eines Python-Skripts und das Speichern eines ONNX-Modells im MetaEditor oder die direkte Ausführung in einem Diagramm in MetaTrader 5. Sie können das Modell mithilfe eines vorgefertigten Python-Skripts beliebig oft direkt im Terminal trainieren. Die Bibliothek enthält vorgefertigte Funktionen zum Erhalt von Preisdaten, die in ein ONNX-Modell eingegeben werden können:

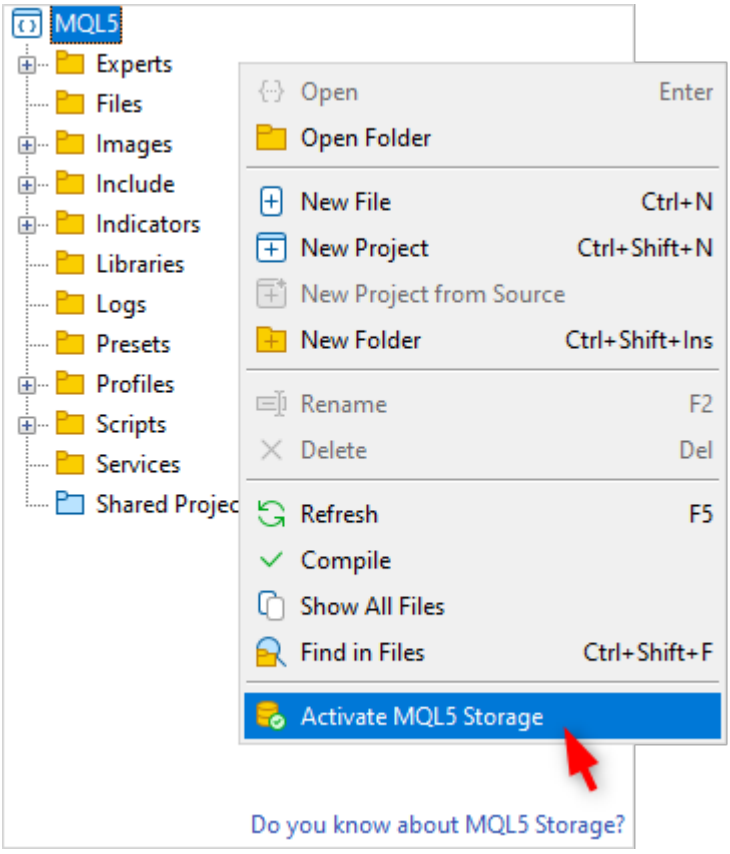
- copy_rates_from - Abruf der Balken ab dem angegebenen Datum
- copy_rates_from_pos - Abruf der Balken ab dem angegebenen Index
- copy_ticks_range - Abruf der Balken für die angegebene Zeitspanne
- copy_ticks_from - Abruf der Ticks ab dem angegebenen Datum

© 2000-2025, MetaQuotes Ltd.

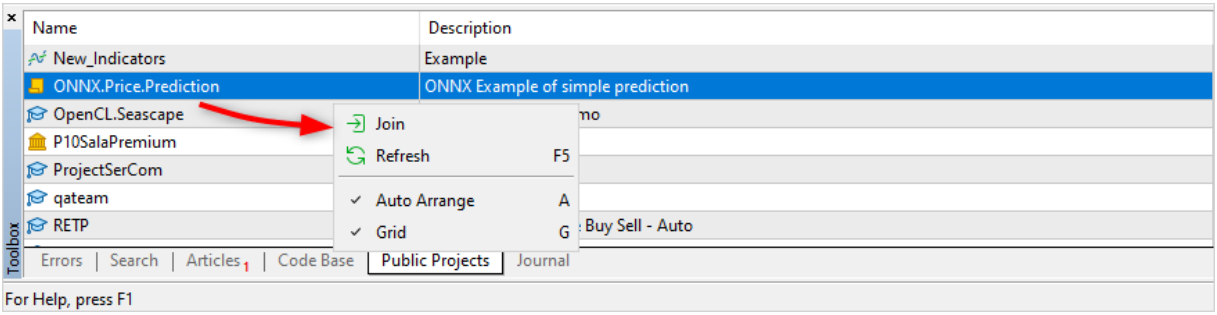
- copy_ticks_range - Abruf der Ticks für die angegebene Zeitspanne

Beispiel eines Modells

Ein Beispiel eines fertigen ONNX-Modells ist in öffentlichen Projekten verfügbar. Sie müssten zunächst im Navigator den MQL5 Storage aktivieren, indem Sie in den MetaEditor-Einstellungen Ihr MQL5-Login angeben (Groß-/Kleinschreibung beachten).

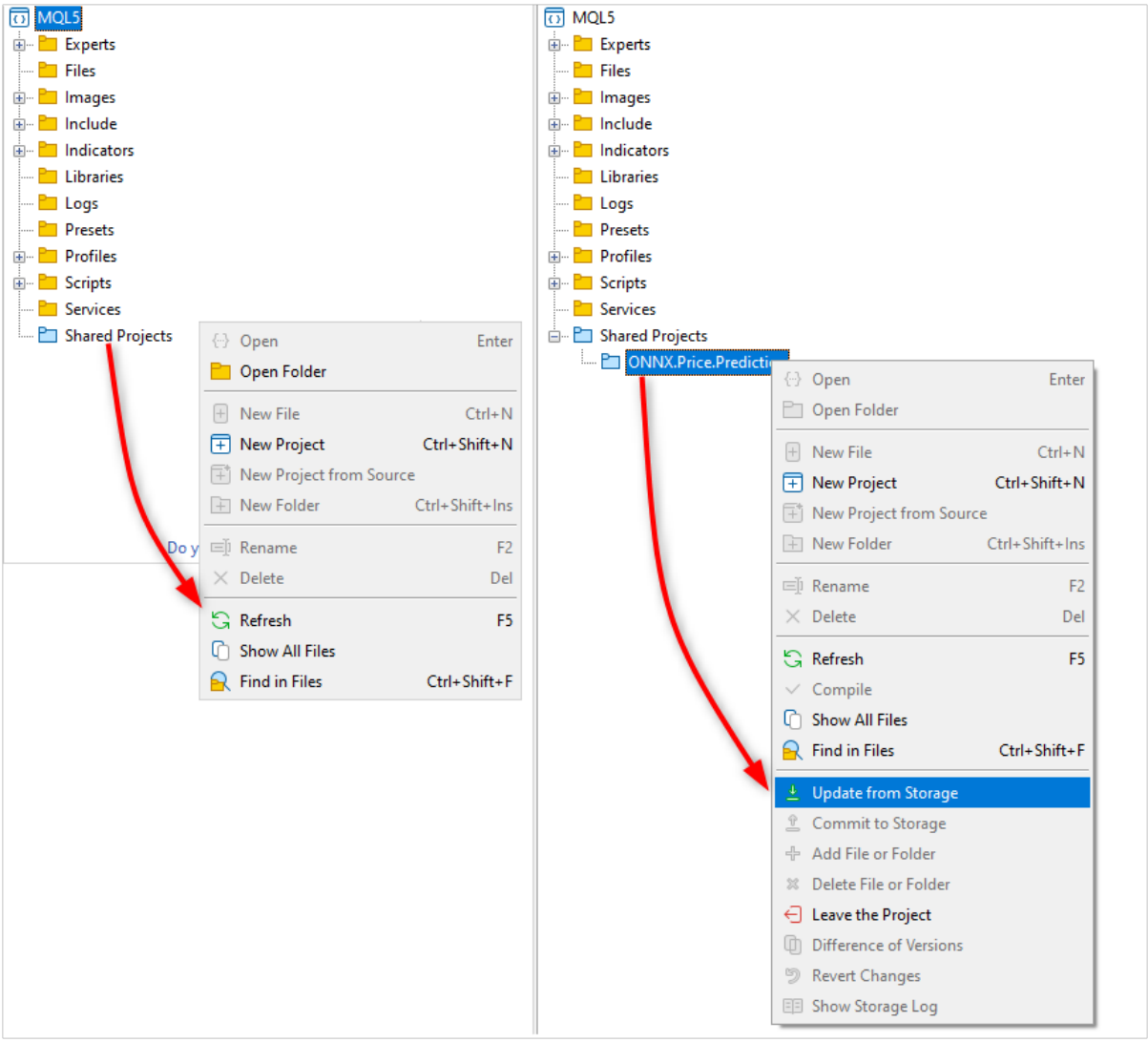


Suchen Sie nach der Aktivierung das Projekt ONNX.Price.Prediction und verbinden Sie es über den Kontextmenübefehl.

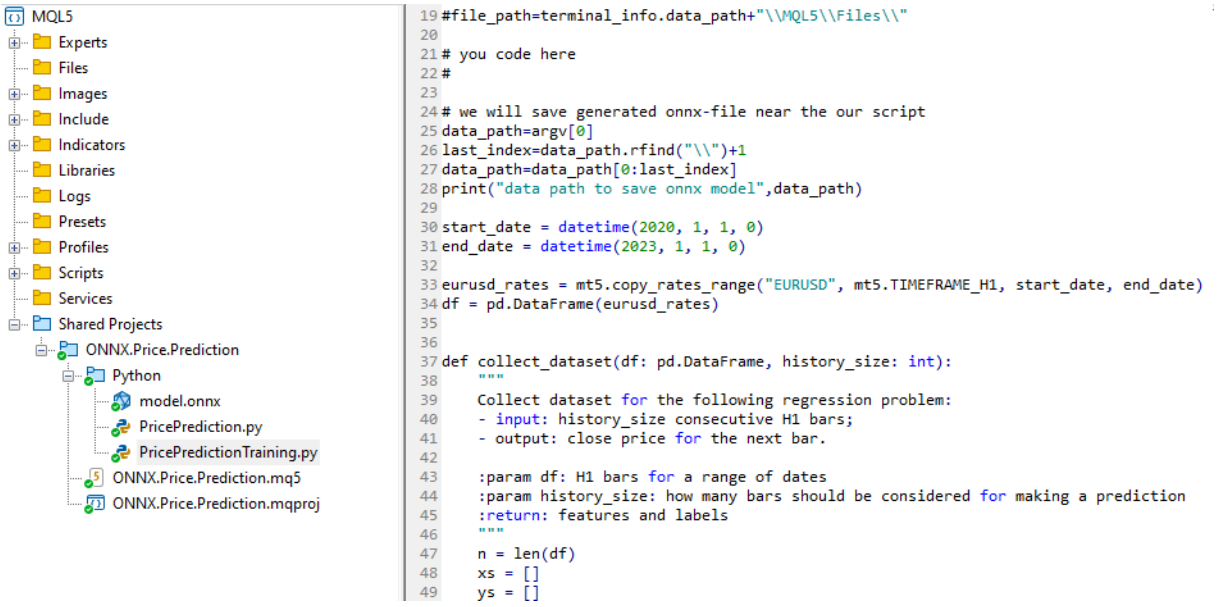


Als Nächstes aktualisieren Sie das Projekt aus dem MQL5-Speicher.

© 2000-2025, MetaQuotes Ltd.



Das Projekt enthält ein ONNX-Modell, zwei Python-Skripte, ein MQL5-Skript für den Projektbetrieb und eine MQL5-Projektdatei (ONNX.Price.Prediction.mqproj).



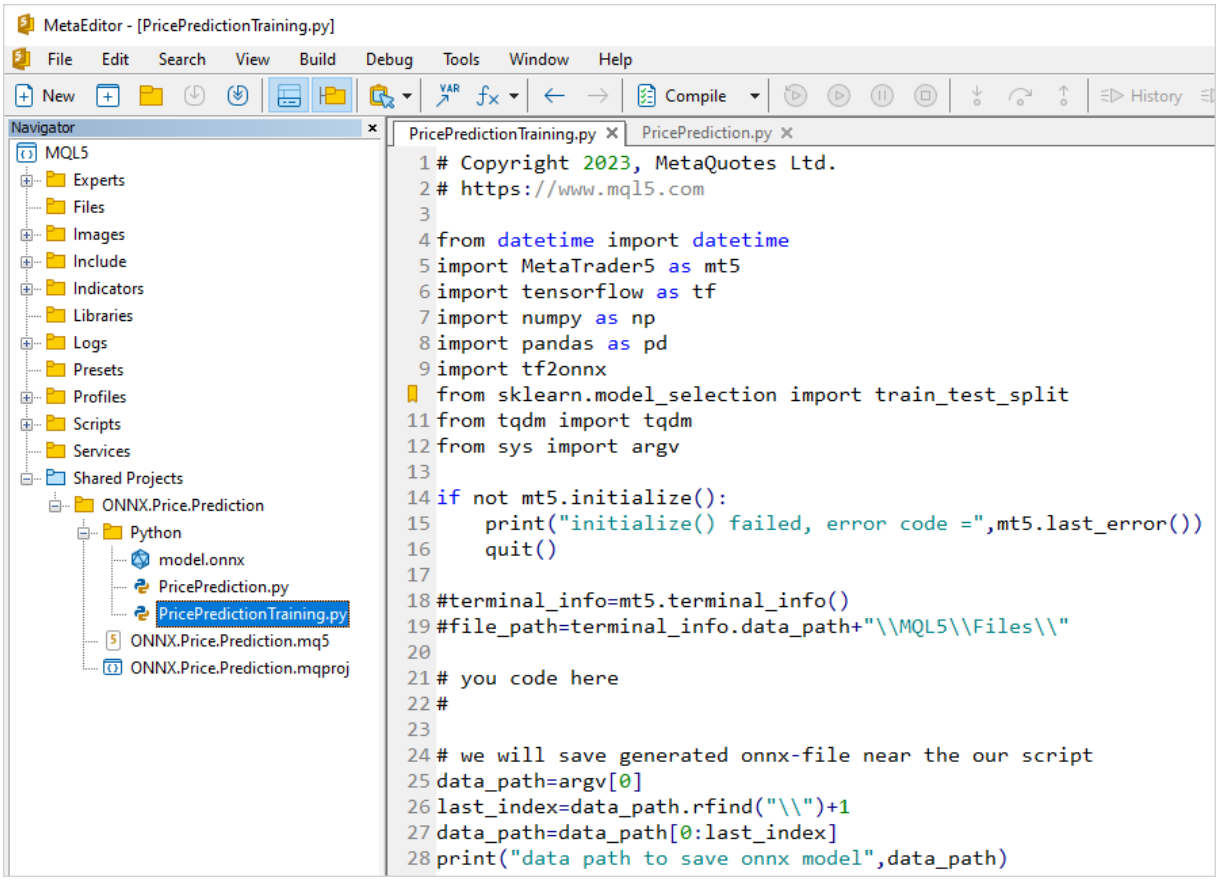

```
#file_path=terminal_info.data_path+"\\MQL5\\Files\\"

# you code here
#

# we will save generated onnx-file near the our script
data_path=argv[0]
last_index=data_path.rfind("\\")+1
data_path=data_path[0:last_index]
print("data path to save onnx model",data_path)

start_date = datetime(2020, 1, 1, 0)
end_date = datetime(2023, 1, 1, 0)

eurusd_rates = mt5.copy_rates_range("EURUSD", mt5.TIMEFRAME_H1, start_date, end_date)
df = pd.DataFrame(eurusd_rates)


def collect_dataset(df: pd.DataFrame, history_size: int):
    """
    Collect dataset for the following regression problem:
    - input: history_size consecutive H1 bars;
    - output: close price for the next bar.

    :param df: H1 bars for a range of dates
    :param history_size: how many bars should be considered for making a prediction
    :return: features and labels
    """
    n = len(df)
    xs = []
    ys = []
```

© 2000-2025, MetaQuotes Ltd.

Sie können das ONNX-Modell selbst erstellen, indem Sie das im Projekt enthaltene Skript PricePredictionTraining.py verwenden. Dazu sollten Sie zunächst die benötigten Module über die Kommandozeile installieren.

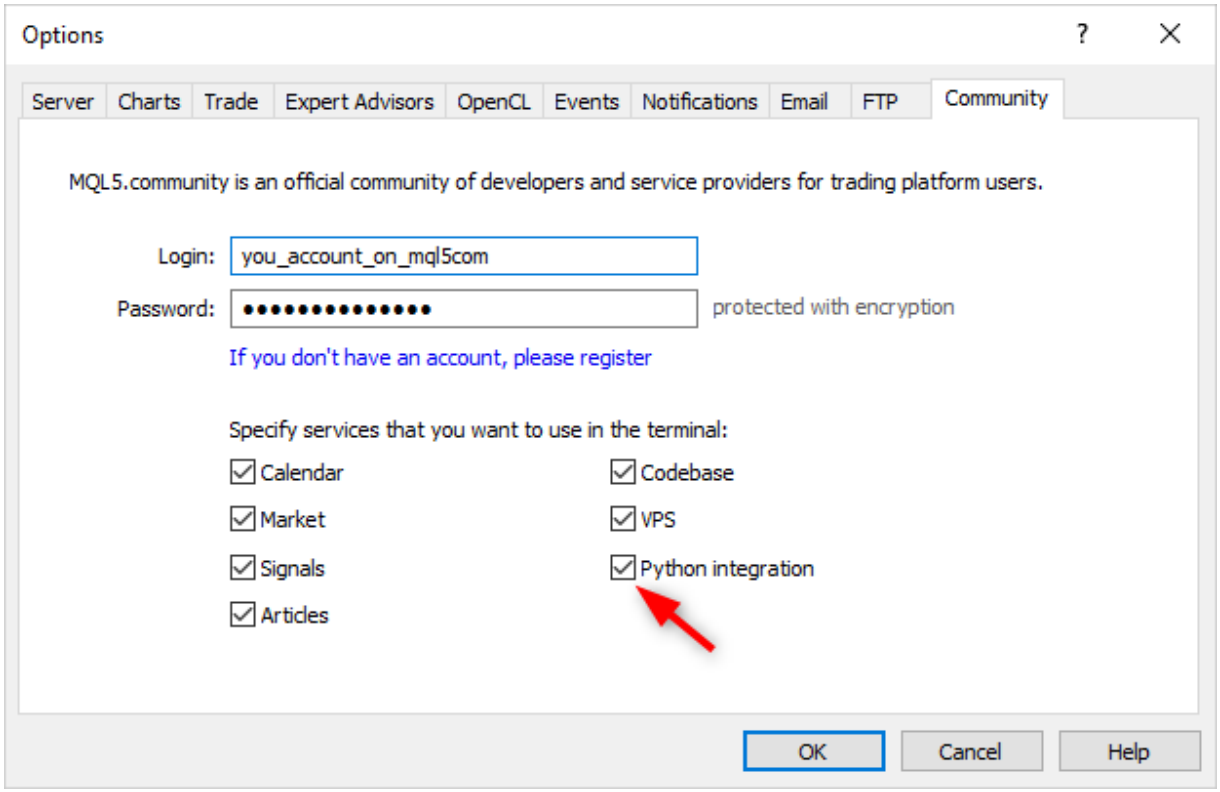
```
python.exe -m pip install --upgrade pip
python -m pip install --upgrade tensorflow
python -m pip install --upgrade pandas
python -m pip install --upgrade scikit-learn
python -m pip install --upgrade matplotlib
python -m pip install --upgrade tqdm
python -m pip install --upgrade metatrader5
python -m pip install --upgrade onnx==1.12
python -m pip install --upgrade tf2onnx
python -m pip install --upgrade numpy
python -m pip install onnxruntime
```

Öffnen Sie nach der Installation der Module das Skript PricePredictionTraining.py im MetaEditor und führen Sie es mit der Schaltfläche „Kompilieren“ oder der Taste F7 aus.



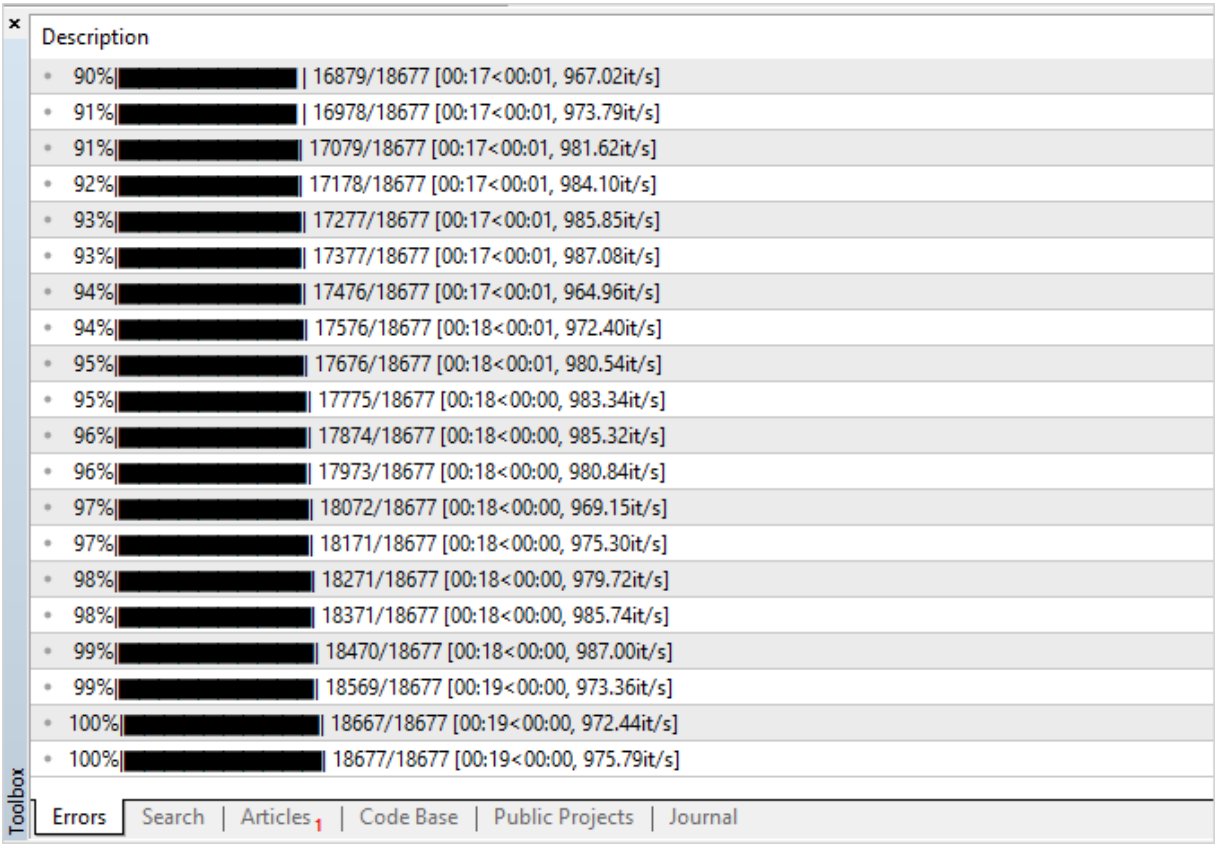
© 2000-2025, MetaQuotes Ltd.

Stellen Sie vor der Ausführung des Python-Skripts sicher, dass das MetaTrader 5-Terminal mit einem Server verbunden ist, auf dem das EURUSD-Symbol verfügbar ist. Stellen Sie beispielsweise eine Verbindung zum MetaQuotes-Demo-Server her und aktivieren Sie „Integration mit Python“ in den Terminal-Einstellungen.



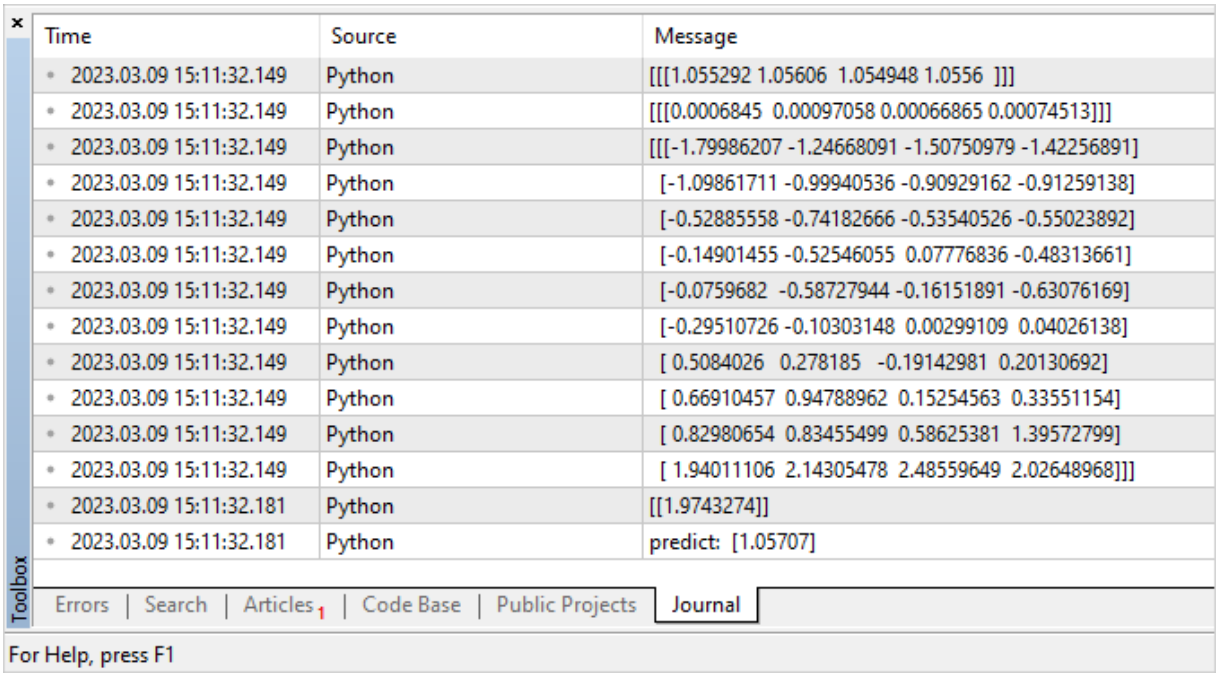
Während das Netzwerk trainiert wird, druckt MetaEditor Nachrichten aus dem Python-Skript, bis das Training abgeschlossen ist.

© 2000-2025, MetaQuotes Ltd.



Wenn das Ergebnis 100 % beträgt, ist das ONNX-Modell fertig und wird im Projektordner unter <Verzeichnis des Terminals>\MQL5\Shared Projects\ONNX.Price.Prediction\Python gespeichert.

Sie können das resultierende Modell überprüfen, indem Sie das zweite Skript PricePrediction.py ausführen und dabei **F7** drücken.



© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Ausführung eines Modells

Um ein ONNX-Modell in MQL5 auszuführen, führen Sie 3 Schritte aus:

1. Laden Sie das Modell aus einer *.onnx-Datei mit der Funktion OnnxCreate oder aus einem Array mit OnnxCreateFromBuffer.
2. Spezifizieren Sie die Formen der Eingangs- und Ausgangsdaten mit den Funktionen OnnxSetInputShape und OnnxSetOutputShape.
3. Starten Sie das Modell mit der Funktion OnnxRun und übergeben Sie ihr die entsprechenden Eingangs- und Ausgangsparameter.
4. Bei Bedarf können Sie den Modellbetrieb mit der Funktion OnnxRelease beenden.

Bei der Erstellung eines ONNX-Modells sollten Sie die bestehenden Grenzen und Einschränkungen berücksichtigen, die unter https://github.com/microsoft/onnxruntime/blob/rel-1.14.0/docs/OperatorKernels.md beschrieben sind.

Einige Beispiele für solche Einschränkungen sind im Folgenden aufgeführt:

Bedienung	Unterstützte Datentypen
ReduceSum	tensor(double), tensor(float), tensor(int32), tensor(int64)
Mul	tensor(bfloat16), tensor(double), tensor(float), tensor(float16), tensor(int32), tensor(int64), tensor(uint32), tensor(uint64)

Nachfolgend ein MQL5-Codebeispiel aus dem öffentlichen Projekt ONNX.Price.Prediction.

```
const long   ExtOutputShape[] = {1,1};     // die Ausgangsform des Modells
const long   ExtInputShape [] = {1,10,4}; // Eingangsform des Modells
#resource "Python/model.onnx" as uchar ExtModel[]// Modell als Ressource
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
int OnStart(void)
  {
   matrix rates;
//--- Abrufen von 10 Balken
   if(!rates.CopyRates("EURUSD",PERIOD_H1,COPY_RATES_OHLC,2,10))
      return(-1);
//--- Eingang einer Reihe von OHLC-Vektoren
   matrix x_norm=rates.Transpose();
   vector m=x_norm.Mean(0);
   vector s=x_norm.Std(0);
   matrix mm(10,4);
   matrix ms(10,4);
//--- die Normalisierungsmatrizen ausfüllen
   for(int i=0; i<10; i++)
     {
      mm.Row(m,i);
```

© 2000-2025, MetaQuotes Ltd.

```
      ms.Row(s,i);
     }
//--- Normalisieren der Eingangsdaten
   x_norm-=mm;
   x_norm/=ms;
//--- Modell erstellen
   long handle=OnnxCreateFromBuffer(ExtModel,ONNX_DEBUG_LOGS);
//--- Die Form der Eingangsdaten spezifizieren
   if(!OnnxSetInputShape(handle,0,ExtInputShape))
     {
      Print("OnnxSetInputShape failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- Spezifikation der Form der Ausgangsdaten
   if(!OnnxSetOutputShape(handle,0,ExtOutputShape))
     {
      Print("OnnxSetOutputShape failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- konvertieren der normalisierten Eingänge in den Type float
   matrixf x_normf;
   x_normf.Assign(x_norm);
//--- hier die Ausgangsdaten des Modells abrufen, d. h. die Preisvorhersage
   vectorf y_norm(1);
//--- Ausführen des Modells
   if(!OnnxRun(handle,ONNX_DEBUG_LOGS | ONNX_NO_CONVERSION,x_normf,y_norm))
     {
      Print("OnnxRun failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- Ausdrucken der Ausgangswerte des Modells in das Protokoll
   Print(y_norm);
//--- die Rücktransformation durchführen, um den voraussichtlichen Preis zu erhalten
   double y_pred=y_norm[0]*s[3]+m[3];
   Print("price predicted:",y_pred);
//--- Operation beenden
   OnnxRelease(handle);
   return(0);
  }
```

Ausführungsbeispiel eines Skripts:

```
ONNX: Creating and using per session threadpools since use_per_session_threads_ is tru
ONNX: Dynamic block base set to 0
ONNX: Initializing session.
ONNX: Adding default CPU execution provider.
```

© 2000-2025, MetaQuotes Ltd.

```
ONNX: Total shared scalar initializer count: 0
ONNX: Total fused reshape node count: 0
ONNX: Total shared scalar initializer count: 0
ONNX: Total fused reshape node count: 0
ONNX: Use DeviceBasedPartition as default
ONNX: Saving initialized tensors.
ONNX: Done saving initialized tensors
ONNX: Session successfully initialized.
[0.28188983]
predicted 1.0559258806393044
```

Das MetaTrader 5 Terminal hat die optimale Ausführung für die Berechnungen ausgewählt — ONNX Runtime Execution Provider. In diesem Beispiel wurde das Modell auf der CPU ausgeführt.

Ändern wir das Skript, um den Prozentsatz erfolgreicher Prognose der Schlusskurse zu berechnen, die auf den Werten der vorangegangenen 10 Balken basieren.

```
#resource "Python/model.onnx" as uchar ExtModel[]// Modell als Ressource

#define TESTS 10000  // Anzahl der Testdatensätze
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
int OnStart()
  {
//--- Modell erstellen
   long session_handle=OnnxCreateFromBuffer(ExtModel,ONNX_DEBUG_LOGS);
   if(session_handle==INVALID_HANDLE)
     {
      Print("Modell konnte nicht erstellt werden. Fehler: ",GetLastError());
      return(-1);
     }

//--- da die Größe des Eingangstensors für das Modell nicht definiert ist, muss das ex
//--- der erste Index ist die Größe des Pakets, der zweite Index ist die Seriengröße u
   const long input_shape[]={1,10,4};
   if(!OnnxSetInputShape(session_handle,0,input_shape))
     {
      Print("OnnxSetInputShape error ",GetLastError());
      return(-2);
     }

//--- da die Größe des Ausgangstensors für das Modell nicht definiert ist, muss sie ex
//--- der erste Index ist die Größe des Pakets, er muss mit der Größe des Pakets im E
//--- der zweite Index ist die Anzahl der vorhergesagten Preise (hier wird nur Close v
   const long output_shape[]={1,1};
   if(!OnnxSetOutputShape(session_handle,0,output_shape))
     {
      Print("OnnxSetOutputShape Fehler: ",GetLastError());
      return(-3);
```

© 2000-2025, MetaQuotes Ltd.

```
     }
//--- Durchführen der Tests
   vector closes(TESTS);       // Vektor zum Speichern von Validierungspreisen
   vector predicts(TESTS);     // Vektor zum Speichern der erhaltenen Vorhersagen
   vector prev_closes(TESTS); // Vektor zum Speichern vorheriger Preise

   matrix rates;               // Matrix zur Ermittlung der OHLC-Reihen
   matrix splitted[2];         // zwei Teilmatrizen, um die Reihen in Test und Validier
   ulong  parts[]={10,1};      // Größen der unterteilten Submatrizen

//--- Start vom vorherigen Balken
   for(int i=1; i<=TESTS; i++)
     {
      //--- Abrufen von 11 Balken
      rates.CopyRates("EURUSD",PERIOD_H1,COPY_RATES_OHLC,i,11);
      //--- Aufteilung der Matrix in Test und Validierung
      rates.Vsplit(parts,splitted);
      //--- Auswählen des Schlusskurses aus der Validierungsmatrix
      closes[i-1]=splitted[1][3][0];
      //--- letzter Schlusskurs in den getesteten Serien
      prev_closes[i-1]=splitted[0][3][9];

      //--- die Testmatrix von 10 Balken für den Test
      predicts[i-1]=PricePredictionTest(session_handle,splitted[0]);
      //--- Laufzeitfehler
      if(predicts[i-1]<=0)
        {
         OnnxRelease(session_handle);
         return(-4);
        }
     }
//--- Operation beenden
   OnnxRelease(session_handle);
//--- Bewerten, ob die Preisentwicklung richtig vorhergesagt wurde
   int    right_directions=0;
   vector delta_predicts=prev_closes-predicts;
   vector delta_actuals=prev_closes-closes;

   for(int i=0; i<TESTS; i++)
      if((delta_predicts[i]>0 && delta_actuals[i]>0) || (delta_predicts[i]<0 && delta_
         right_directions++;
   PrintFormat("right direction predictions = %.2f%%",(right_directions*100.0)/double
//---
   return(0);
  }
//+------------------------------------------------------------------+
//|  Vorbereitung der Daten und Ausführung des Modells               |
//+------------------------------------------------------------------+
double PricePredictionTest(const long session_handle,matrix& rates)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
   static matrixf input_data(10,4); // Matrix für die transformierte Eingangs
   static vectorf output_data(1);   // Vektor zur Aufnahme des Ergebnisses
   static matrix mm(10,4);          // Matrix der Mittelwerte der horizontalen Vektore
   static matrix ms(10,4);          // Matrix der Std der horizontalen Vektoren

//--- eine Reihe von vertikalen OHLC-Vektoren muss in das Modell eingegeben werden
   matrix x_norm=rates.Transpose();
//--- Normalisierung der Preise
   vector m=x_norm.Mean(0);
   vector s=x_norm.Std(0);
   for(int i=0; i<10; i++)
     {
      mm.Row(m,i);
      ms.Row(s,i);
     }
   x_norm-=mm;
   x_norm/=ms;

//--- Ausführen des Modells
   input_data.Assign(x_norm);
   if(!OnnxRun(session_handle,ONNX_DEBUG_LOGS,input_data,output_data))
     {
      Print("OnnxRun Fehler: ",GetLastError());
      return(0);
     }
//--- De-Normalisieren der Preise von den Ausgangswerten
   double y_pred=output_data[0]*s[3]+m[3];

   return(y_pred);
  }
```

Führen Sie das Skript aus: Die Vorhersagegenauigkeit beträgt etwa 51%.

```
ONNX: Creating and using per session threadpools since use_per_session_threads_ is tru
ONNX: Dynamic block base set to 0
ONNX: Initializing session.
ONNX: Adding default CPU execution provider.
ONNX: Total shared scalar initializer count: 0
ONNX: Total fused reshape node count: 0
ONNX: Total shared scalar initializer count: 0
ONNX: Total fused reshape node count: 0
ONNX: Use DeviceBasedPartition as default
ONNX: Saving initialized tensors.
ONNX: Done saving initialized tensors
ONNX: Session successfully initialized.
right direction predictions = 51.34 %
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Modellvalidierung im Strategy Tester

Modelle, die für Operationen auf den Finanzmärkten erstellt wurden, können im MetaTrader 5 Terminal Strategy Tester validiert werden. Dies ist die schnellste und bequemste Option, die eine zusätzliche Emulation der Marktumgebung und der Handelsbedingungen überflüssig macht.

Um das Modell zu testen, erstellen wir einen Expert Advisor, der auf dem Code des öffentlichen Projekts ONNX.Price.Prediction basiert. Dazu sind einige Änderungen erforderlich.

Verschieben Sie die Modellerstellung in die Funktion OnInit und schließen Sie die onnx-Sitzung wieder in OnDeinit. Den Hauptblock der Modelloperationen platzieren Sie in OnTick.

Fügen Sie außerdem den Schlusskurs (close) der beiden vorangegangenen Balken hinzu, der für den Vergleich zwischen dem tatsächlichen Schlusskurs und der Vorhersage benötigt wird.

Der Code des Expert Advisors ist kurz und leicht zu lesen.

```
const long   ExtInputShape [] = {1,10,4}; // Eingangsform des Modells
const long   ExtOutputShape[] = {1,1};    // die Ausgangsform des Modells
#resource "Python/model.onnx" as uchar ExtModel[];// Modell als Resource

long handle;          // Handle des Modells
ulong predictions=0; // Prognosezähler
ulong confirmed=0;   // Prognosezähler der Erfolge
//+------------------------------------------------------------------+
//| Expert Initialisierungsfunktion                                  |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- basic checks
   if(_Symbol!="EURUSD")
     {
      Print("Symbol must be EURUSD, testing aborted");
      return(-1);
     }
   if(_Period!=PERIOD_H1)
     {
      Print("Timeframe must be H1, testing aborted");
      return(-1);
     }
//--- Modell erstellen
   handle=OnnxCreateFromBuffer(ExtModel,ONNX_DEBUG_LOGS);
//--- Die Form der Eingangsdaten spezifizieren
   if(!OnnxSetInputShape(handle,0,ExtInputShape))
     {
      Print("OnnxSetInputShape failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
```

© 2000-2025, MetaQuotes Ltd.

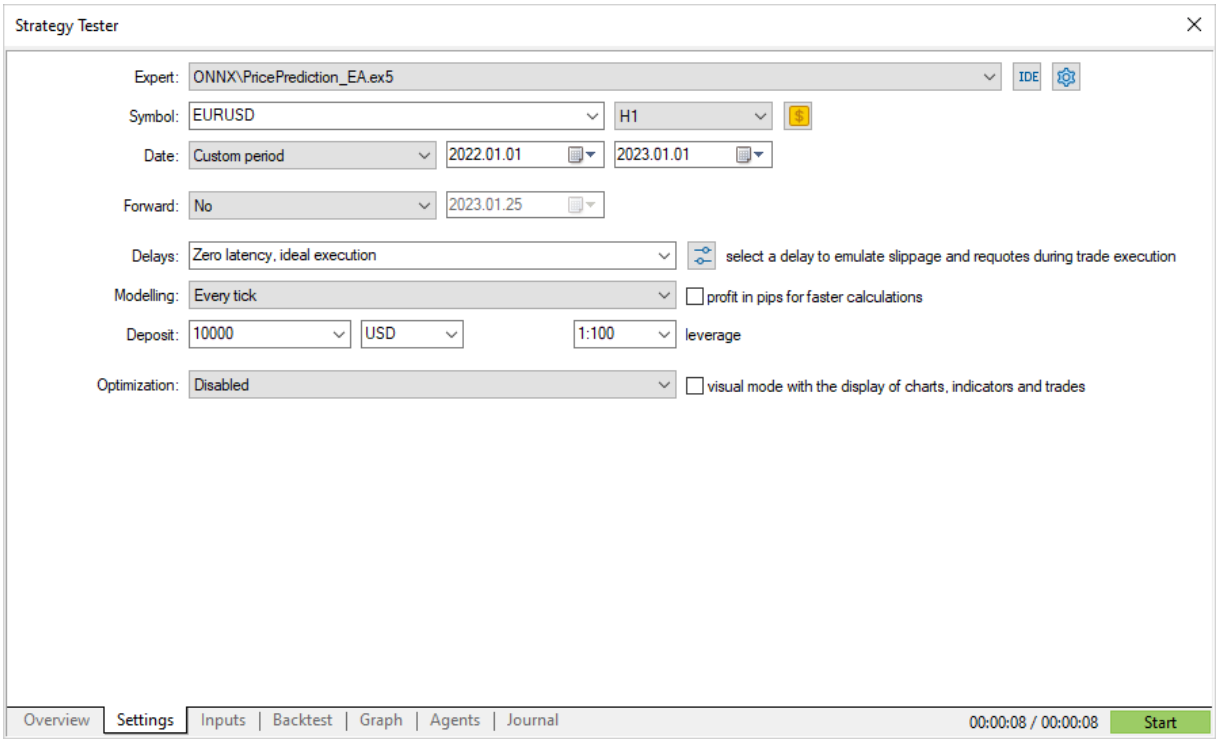
```
//--- Spezifikation der Form der Ausgangsdaten
   if(!OnnxSetOutputShape(handle,0,ExtOutputShape))
     {
      Print("OnnxSetOutputShape failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//---
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Deinitialisierungsfunktion des Experten                          |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Operation des Modells beenden
   OnnxRelease(handle);
//--- Berechnung und Ausgangs von Prognosestatistiken
   PrintFormat("Successfull predictions = %.2f %%",confirmed*100./double(predictions))
  }
//+------------------------------------------------------------------+
//| Tick-Funktion des Experten                                       |
//+------------------------------------------------------------------+
void OnTick()
  {
   static datetime open_time=0;
   static double predict;
//--- die aktuelle Öffnungszeit des Balkens überprüfen
   datetime time=iTime(_Symbol,_Period,0);
   if(time==0)
     {
      PrintFormat("Failed to get Time(0), error %d", GetLastError());
      return;
     }
//--- wenn sich der Eröffnungszeitpunkt nicht geändert hat, OnTick verlassen
   if(time==open_time)
      return;
//--- die Schlusskurse der letzten beiden geschlossenen Balken ermitteln
   double close[];
   int recieved=CopyClose(_Symbol,_Period,1,2,close);
   if(recieved!=2)
     {
      PrintFormat("CopyClose(2 bars) failed, error %d",GetLastError());
      return;
     }
   double delta_predict=predict-close[0]; // prognostizierte Preisbewegung
   double delta_actual=close[1]-close[0]; // aktuelle Preisänderung
   if((delta_predict>0 && delta_actual>0) || (delta_predict<0 && delta_actual<0))
      confirmed++;
```

© 2000-2025, MetaQuotes Ltd.

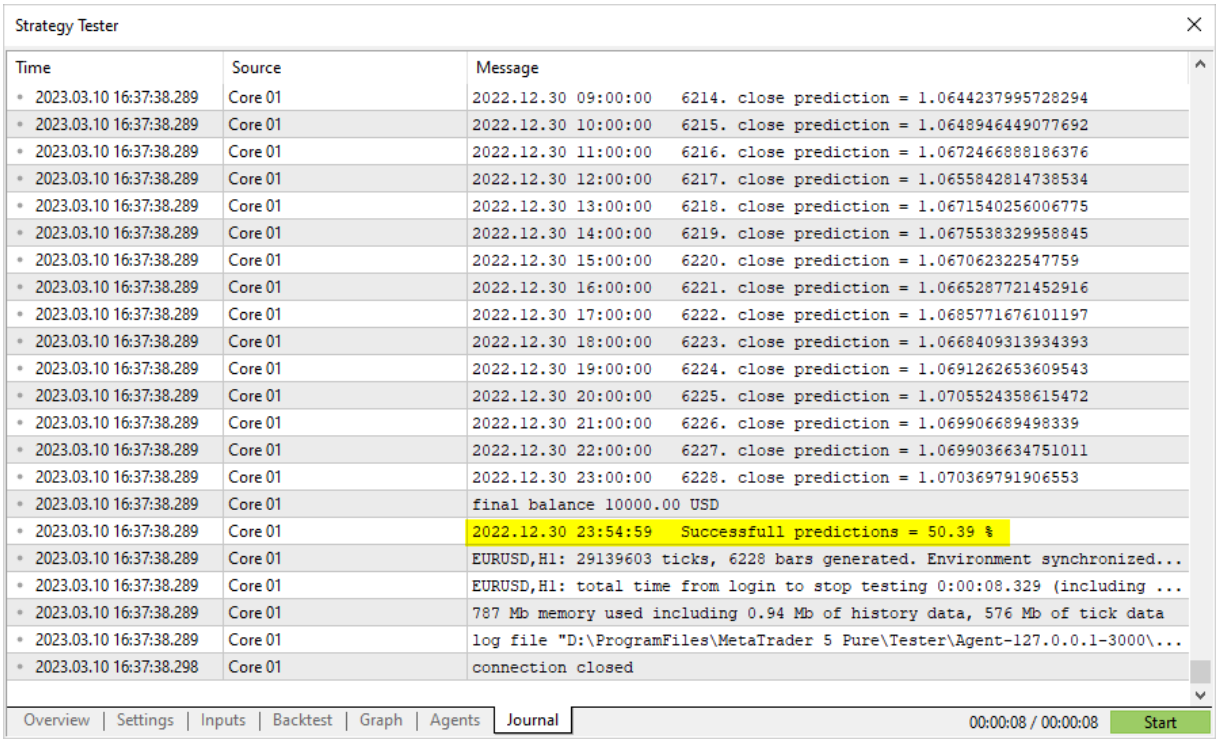
```
//--- den Schlusskurs des neuen Balkens berechnen, um den Kurs des nächsten Balkens zu
   matrix rates;
//--- Abrufen von 10 Balken
   if(!rates.CopyRates("EURUSD",PERIOD_H1,COPY_RATES_OHLC,1,10))
      return;
//--- Eingang einer Reihe von OHLC-Vektoren
   matrix x_norm=rates.Transpose();
   vector m=x_norm.Mean(0);
   vector s=x_norm.Std(0);
   matrix mm(10,4);
   matrix ms(10,4);
//--- die Normalisierungsmatrizen ausfüllen
   for(int i=0; i<10; i++)
     {
      mm.Row(m,i);
      ms.Row(s,i);
     }
//--- Normalisieren der Eingangsdaten
   x_norm-=mm;
   x_norm/=ms;
//--- konvertieren der normalisierten Eingänge in den Type float
   matrixf x_normf;
   x_normf.Assign(x_norm);
//--- hier die Ausgangsdaten des Modells abrufen, d. h. die Preisvorhersage
   vectorf y_norm(1);
//--- Ausführen des Modells
   if(!OnnxRun(handle,ONNX_DEBUG_LOGS | ONNX_NO_CONVERSION,x_normf,y_norm))
     {
      Print("OnnxRun failed, error ",GetLastError());
     }
//--- eine Rücktransformation durchführen, um den vorhergesagten Preis zu erhalten und
   predict=y_norm[0]*s[3]+m[3];
   predictions++;  // Prognosezähler erhöhen
   Print(predictions,". close prediction = ",predict);
//--- die Öffnungszeit des Balkens speichern, um sie beim nächsten Tick zu überprüfen
   open_time=time;
  }
```

Kompilieren Sie den Expert Advisor und führen Sie Tests im Zeitraum des Jahres 2022 durch. Geben Sie EURUSD mit dem H1-Zeitrahmen an, also den Daten, auf denen das Modell trainiert wurde. Der Tick-Modellierungsmodus kann ignoriert werden, da der Code das Erscheinen eines neuen Balkens überprüft.

© 2000-2025, MetaQuotes Ltd.



Ausführen und das Ergebnis im Journal des Testers überprüfen. Es zeigt, dass etwas mehr als 50% der Vorhersagen im Jahr 2022 richtig waren.



Wenn die vorläufigen Modelltests zufriedenstellende Ergebnisse erbracht haben, können Sie mit dem Schreiben einer vollwertigen Handelsstrategie auf der Grundlage dieses Modells beginnen.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

OnnxCreate

Erstellen einer ONNX-Sitzung, laden eines Modells aus einer *.onnx-Datei

```
long  OnnxCreate(
   string  filename,  // Dateipfad
   uint    flags      // Flags für das Erstellen des Modells
   );
```

Parameter

filename

[in] Pfad zur *.onnx-Datei des Modells relativ zum Ordner \MQL5\Files\.

flags

[in] Flags aus ENUM_ONNX_FLAGS, die den Modus der Modellerstellung beschreiben: ONNX_COMMON_FOLDER und ONNX_DEBUG_LOGS.

Rückgabewert

Das Handle der erstellten Sitzung oder INVALID_HANDLE, wenn ein Fehler auftritt. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

Hinweis

Wenn die angegebene Datei auf der Festplatte nicht gefunden wird, versucht das System erneut, die Datei zu öffnen, wobei es die Erweiterung '.onnx' an den Namen anhängt.

© 2000-2025, MetaQuotes Ltd.

OnnxCreateFromBuffer

Erstellen einer ONNX-Sitzung, laden eines Modells aus einem Datenarray.

```
long  OnnxCreateFromBuffer(
   const uchar&  buffer[],   // Array-Referenz
   ulong         flags       // Flags für die Modellerstellung
   );
```

Parameter

buffer

[in] Array mit den Daten des ONNX-Modells.

flags

[in] Flags aus ENUM_ONNX_FLAGS, die den Modus der Modellerstellung beschreiben: ONNX_COMMON_FOLDER und ONNX_DEBUG_LOGS.

Rückgabewert

Das Handle der erstellten Sitzung oder INVALID_HANDLE, wenn ein Fehler auftritt. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxRelease

Schließt eine ONNX-Sitzung.

```
bool  OnnxRelease(
   long   onnx_handle  // Handle der ONNX-Sitzung
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxRun

Ausführen eines ONNX-Modells.

```
bool  OnnxRun(
   long     onnx_handle,  // ONNX-Sitzungshandle
   ulong    flags,        // Flags, die den Laufmodus beschreiben
   ...                    // Eingänge und Ausgänge des Modells
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

flags

[in] Die Flags aus ENUM_ONNX_FLAGS, die den Ausführungsmodus beschreiben: ONNX_DEBUG_LOGS und ONNX_NO_CONVERSION.

...

[in] [out] Eingänge und Ausgänge des Modells.

Gibt bei Erfolg true zurück, andernfalls false. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

ENUM_ONNX_FLAGS

ID	Beschreibung
ONNX_DEBUG_LOGS	Ausgabe der Debugging-Logs
ONNX_NO_CONVERSION	Automatische Konvertierung deaktivieren, Nutzerdaten so verwenden, wie sie sind
ONNX_COMMON_FOLDER	Laden einer Modelldatei aus dem Ordner Common\Files; der Wert ist gleich dem Flag FILE_COMMON

Beispiel:

```
const long                              ExtOutputShape[] = {1,1};     // Form der Modell
const long                              ExtInputShape [] = {1,10,4}; // Form der Modell
#resource "Python/model.onnx" as uchar ExtModel[]                    // Modell als Reso
//+------------------------------------------------------------------+
//| Skript Programm Start Funktion                                   |
//+------------------------------------------------------------------+
int OnStart(void)
  {
   matrix rates;
//--- Abrufen von 10 Balken
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!rates.CopyRates("EURUSD",PERIOD_H1,COPY_RATES_OHLC,2,10))
      return(-1);
//--- Eingang einer Reihe von OHLC-Vektoren
   matrix x_norm=rates.Transpose();
   vector m=x_norm.Mean(0);
   vector s=x_norm.Std(0);
   matrix mm(10,4);
   matrix ms(10,4);
//--- die Normalisierungsmatrizen ausfüllen
   for(int i=0; i<10; i++)
     {
      mm.Row(m,i);
      ms.Row(s,i);
     }
//--- Normalisieren der Eingangsdaten
   x_norm-=mm;
   x_norm/=ms;
//--- Modell erstellen
   long handle=OnnxCreateFromBuffer(ExtModel,ONNX_DEBUG_LOGS);
//--- Die Form der Eingangsdaten spezifizieren
   if(!OnnxSetInputShape(handle,0,ExtInputShape))
     {
      Print("OnnxSetInputShape failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- Spezifikation der Form der Ausgangsdaten
   if(!OnnxSetOutputShape(handle,0,ExtOutputShape))
     {
      Print("OnnxSetOutputShape failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- konvertieren der normalisierten Eingänge in den Type float
   matrixf x_normf;
   x_normf.Assign(x_norm);
//--- hier die Ausgangsdaten des Modells abrufen, d. h. die Preisvorhersage
   vectorf y_norm(1);
//--- Ausführen des Modells
   if(!OnnxRun(handle,ONNX_DEBUG_LOGS | ONNX_NO_CONVERSION,x_normf,y_norm))
     {
      Print("OnnxRun failed, error ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- Ausdrucken der Ausgangswerte des Modells in das Protokoll
   Print(y_norm);
//--- die Rücktransformation durchführen, um den voraussichtlichen Preis zu erhalten
   double y_pred=y_norm[0]*s[3]+m[3];
```

© 2000-2025, MetaQuotes Ltd.

```
   Print("price predicted:",y_pred);
//--- abgeschlossene Operation
   OnnxRelease(handle);
   return(0);
  };
```

Siehe auch

OnnxSetInputShape, OnnxSetOutputShape

© 2000-2025, MetaQuotes Ltd.

OnnxGetInputCount

Ermitteln der Anzahl der Eingänge eines ONNX-Modells.

```
long  OnnxGetInputCount(
   long   onnx_handle  // Handle der ONNX-Sitzung
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

Rückgabewert

Gibt bei Erfolg die Anzahl der Eingangsparameter zurück, andernfalls -1. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxGetOutputCount

Ermitteln der Anzahl der Ausgänge eines ONNX-Modells

```
long  OnnxGetOutputCount(
   long   onnx_handle  // Handle der ONNX-Sitzung
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

Rückgabewert

Gibt bei Erfolg die Anzahl der Ausgangsparameter zurück, andernfalls -1. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxGetInputName

Abrufen des Namens des Eingangs eines Modells anhand des Index.

```
string  OnnxGetInputName(
   long   onnx_handle,  // Handle der ONNX-Sizung
   long   index         // Parameterindex
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

index

[in] Index des Eingangsparameters, beginnend mit 0.

Rückgabewert

Gibt bei Erfolg den Namen des Eingangsparameters zurück, andernfalls NULL. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxGetOutputName

Abrufen des Namens des Ausgangs eines Modells anhand des Index.

```
string  OnnxGetOutputName(
   long   onnx_handle,  // Handle der ONNX-Sizung
   long   index         // Parameterindex
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

index

[in] Index des Ausgangsparameters, beginnend mit 0.

Rückgabewert

Gibt bei Erfolg den Namen des Ausgangsparameters zurück, andernfalls NULL. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxGetInputTypeInfo

Abrufen der Beschreibung des Eingangstyps des Modells.

```
bool  OnnxGetInputTypeInfo(
   long           onnx_handle,  // ONNX-Sitzungshandle
   long           index,        // Parameterindex
   OnnxTypeInfo&  typeinfo      // Parametertypbeschreibung
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

index

[in] Index des Eingangsparameters, beginnend mit 0.

typeinfo

[out] Die Struktur OnnxTypeInfo beschreibt den Typ des Eingangsparameters.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxGetOutputTypeInfo

Holt die Beschreibung des Ausgangstyps aus dem Modell.

```
bool  OnnxGetOutputTypeInfo(
   long           onnx_handle,  // ONNX-Sitzungshandle
   long           index,        // Parameterindex
   OnnxTypeInfo&  typeinfo      // Parametertypbeschreibung
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

index

[in] Index des Ausgangsparameters, beginnend mit 0.

typeinfo

[out] Die Struktur OnnxTypeInfo beschreibt den Typ des Ausgangsparameters.

Rückgabewert

Gibt bei Erfolg true zurück, andernfalls false. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

© 2000-2025, MetaQuotes Ltd.

OnnxSetInputShape

Festlegen der Form der Eingangsdaten eines Modells anhand des Index.

```
bool  ChartSetSymbolPeriod(
   long          onnx_handle,  // Handle der ONNX-Sitzung
   long          input_index,  // Eingangsparameterindex
   const ulong&  shape[]       // Array zur Beschreibung der Form der Eingangsdaten
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

input_index

[in] Index des Eingangsparameters, beginnend bei 0.

shape

[in] Array, das die Form der Eingangsdaten des Modells beschreibt.

Rückgabewert

Gibt bei Erfolg den Namen des Eingangsparameters zurück, andernfalls NULL. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

Beispiel:

```
//---- Beschreibung der Formen der Eingangs- und Ausgangsdaten des Modells
   const long  ExtOutputShape[] = {1,1};
   const long  ExtInputShape [] = {1,10,4};
//--- Modell erstellen
   long handle=OnnxCreateFromBuffer(model,ONNX_DEBUG_LOGS);
//--- Die Form der Eingangsdaten spezifizieren
   if(!OnnxSetInputShape(handle,0,ExtInputShape))
     {
      Print("fehlgeschlagen, OnnxSetInputShape-Fehler: ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- Spezifikation der Form der Ausgangsdaten
   if(!OnnxSetOutputShape(handle,0,ExtOutputShape))
     {
      Print("fehlgeschlagen, OnnxSetOutputShape-Fehler: ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
```

Siehe auch

OnnxSetOutputShape

© 2000-2025, MetaQuotes Ltd.

OnnxSetOutputShape

Legt die Form der Ausgangsdaten eines Modells nach dem Index fest.

```
bool  OnnxSetOutputShape(
   long           onnx_handle,    // Handle der ONNX-Sitzung
   long           output_index,   // Ausgangsparameterindex
   const ulong&   shape[]         // Array zur Beschreibung der Form der Ausgangsdaten
   );
```

Parameter

onnx_handle

[in] Handle des Objekts der ONNX-Sitzung von OnnxCreate oder OnnxCreateFromBuffer.

output_index

[in] Index des Ausgangsparameters, beginnend bei 0.

shape

[in] Array, das die Form der Ausgangsdaten des Modells beschreibt.

Rückgabewert

Gibt bei Erfolg den Namen des Eingangsparameters zurück, andernfalls NULL. Um den Fehler-Code zu erhalten, rufen Sie die Funktion GetLastError auf.

Beispiel:

```
//---- Beschreibung der Formen der Eingangs- und Ausgangsdaten des Modells
   const long  ExtOutputShape[] = {1,1};
   const long  ExtInputShape [] = {1,10,4};
//--- Modell erstellen
   long handle=OnnxCreateFromBuffer(model,ONNX_DEBUG_LOGS);
//--- Die Form der Eingangsdaten spezifizieren
   if(!OnnxSetInputShape(handle,0,ExtInputShape))
     {
      Print("fehlgeschlagen, OnnxSetInputShape-Fehler: ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
//--- Spezifikation der Form der Ausgangsdaten
   if(!OnnxSetOutputShape(handle,0,ExtOutputShape))
     {
      Print("fehlgeschlagen, OnnxSetOutputShape-Fehler: ",GetLastError());
      OnnxRelease(handle);
      return(-1);
     }
```

Siehe auch

OnnxSetInputShape

© 2000-2025, MetaQuotes Ltd.

Datenstrukturen

Die folgenden Datenstrukturen werden für die Operationen der ONNX-Modelle verwendet:

OnnxTypeInfo

Die Struktur beschreibt den Typ eines Eingangs-oder Ausgangsparameter eines ONNX-Modells

```
struct OnnxTypeInfo
 {
  ENUM_ONNX_TYPE         type;          // Parametertyp
  OnnxTensorTypeInfo     tensor;        // Tensorbeschreibung
  OnnxMapTypeInfo        map;           // Map-Beschreibung
  OnnxSequenceTypeInfo   sequence;      // Sequenzbeschreibung
 };
```

Es kann nur ein Tensor (ONNX_TYPE_TENSOR) als Eingang verwendet werden. In diesem Fall wird nur das Feld OnnxTypeInfo::tensor mit Werten gefüllt, während die anderen Felder (Map und Sequenz) nicht definiert sind.

Es kann nur einer der drei OnnxTypeInfo-Typen (ONNX_TYPE_TENSOR, ONNX_TYPE_MAP oder ONNX_TYPE_SEQUENCE) als Eingang verwendet werden. Je nach Typ wird die entsprechende untergeordnete Struktur (OnnxTypeInfo::tensor, OnnxTypeInfo::map oder OnnxTypeInfo::sequence) gefüllt.

OnnxTensorTypeInfo

Die Struktur beschreibt tensor durch die Eingangs-oder Ausgangsparameter eines ONNX-Modells

```
struct OnnxTensorTypeInfo
 {
  const ENUM_ONNX_DATA_TYPE  data_type;      // Datentyp im Tensor
  const long                 dimensions[];   // Anzahl der Elemente im Tensor
 };
```

OnnxMapTypeInfo

Die Struktur beschreibt die im Ausgangsparameter eines ONNX-Modells erhaltene Map

```
struct OnnxMapTypeInfo
 {
  const ENUM_ONNX_DATA_TYPE  key_type;       // Typ des Schlüssels
  const OnnxTypeInfo&        value_type;     // Typ der Werte
 };
```

OnnxSequenceTypeInfo

Die Struktur beschreibt die im Ausgangsparameter eines ONNX-Modells erhaltene Sequenz

© 2000-2025, MetaQuotes Ltd.

```
struct OnnxSequenceTypeInfo
 {
  const OnnxTypeInfo&       value_type;      // Datentyp in der Sequenz
 };
```

ENUM_ONNX_TYPE

Die Enumeration **ENUM_ONNX_TYPE** beschreibt den Typ eines Modellparameters

ID	Beschreibung
ONNX_TYPE_UNKNOWN	Unbekannt
ONNX_TYPE_TENSOR	Tensor
ONNX_TYPE_SEQUENCE	Sequence
ONNX_TYPE_MAP	Map
ONNX_TYPE_OPAQUE	Abstrakt (opaque)
ONNX_TYPE_SPARSETENSOR	Sparse tensor

ENUM_ONNX_DATA_TYPE

Die Enumeration **ENUM_ONNX_DATA_TYPE** beschreibt den Typ der verwendeten Daten

ID	Beschreibung
ONNX_DATA_TYPE_UNDEFINED	Undefiniert
ONNX_DATA_TYPE_FLOAT	float
ONNX_DATA_TYPE_INT8	8-bit int
ONNX_DATA_TYPE_UINT16	16-bit uint
ONNX_DATA_TYPE_INT16	16-bit int
ONNX_DATA_TYPE_INT32	32-bit int
ONNX_DATA_TYPE_INT64	64-bit int
ONNX_DATA_TYPE_STRING	string
ONNX_DATA_TYPE_BOOL	Bool
ONNX_DATA_TYPE_FLOAT16	16-bit float
ONNX_DATA_TYPE_DOUBLE	double
ONNX_DATA_TYPE_UINT32	32-bit uint
ONNX_DATA_TYPE_UINT64	64-bit uint

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
ONNX_DATA_TYPE_COMPLEX64	64-bit komplexe Zahl
ONNX_DATA_TYPE_COMPLEX128	128-bit komplexe Zahl
ONNX_DATA_TYPE_BFLOAT16	16-bit bfloat (Brain Floating Point)

ENUM_ONNX_FLAGs

Die Enumeration **ENUM_ONNX_FLAGS** beschreibt den Ausführungsmodus des Modells

ID	Beschreibung
ONNX_DEBUG_LOGS	Ausgabe der Debugging-Logs
ONNX_NO_CONVERSION	Automatische Konvertierung deaktivieren, Nutzerdaten so verwenden, wie sie sind
ONNX_COMMON_FOLDER	Laden einer Modelldatei aus dem Ordner Common\Files; der Wert ist gleich dem Flag FILE_COMMON

Konvertierung von Arrays bei der Arbeit mit ONNX-Modellen

Aufgaben des maschinellen Lernens erfordern nicht immer eine höhere Rechengenauigkeit. Um die Berechnungen zu beschleunigen, verwenden einige Modelle Datentypen mit geringerer Genauigkeit wie Float16 und sogar Float8. Damit Nutzer die entsprechenden Daten in Modelle eingeben können, bietet MQL5 vier spezielle Funktionen, die die Standardtypen in MQL5 in die speziellen FP16- und FP8-Typen umwandeln.

Funktion	Aktion
ArrayToFP16	Kopiert ein Array vom Typ float oder double in ein Array vom Typ ushort mit dem angegebenen Format
ArrayToFP8	Kopiert ein Array vom Typ float oder double in ein Array vom Typ uchar mit dem angegebenen Format
ArrayFromFP16	Kopiert ein Array vom Typ ushort in ein Array vom Typ float oder double mit dem angegebenen Format
ArrayFromFP8	Kopiert ein Array vom Typ uchar in ein Array vom Typ float oder double mit dem angegebenen Format

Diese Array-Konvertierungsfunktionen verwenden spezielle Formate, die in den folgenden Enumerationen angegeben sind.

ENUM_FLOAT16_FORMAT

Die Enumeration **ENUM_FLOAT16_FORMAT** beschreibt zwei FP16-Typenformate.

© 2000-2025, MetaQuotes Ltd.

ID	Beschreibung
FLOAT_FP16	Standard 16-Bit-Format, auch bekannt als half (halb)
FLOAT_BFP16	Das spezielle Format brain float point

Jedes dieser Formate hat seine Vorteile und Einschränkungen. FLOAT16 bietet eine höhere Genauigkeit, erfordert aber mehr Speicher- und Rechenressourcen. BFLOAT16 hingegen bietet eine höhere Leistung und Effizienz bei der Datenverarbeitung, kann aber weniger genau sein.

ENUM_FLOAT8_FORMAT

Die Enumeration **ENUM_FLOAT8_FORMAT** beschreibt vier FP8-Typenformate.

FP8 (8-Bit-floating point) ist einer der Datentypen, die zur Darstellung von Gleitkommazahlen verwendet werden. In FP8 wird jede Zahl durch 8 Datenbits dargestellt, die typischerweise in drei Komponenten unterteilt sind: Vorzeichen, Exponent und Mantisse. Dieses Format bietet ein Gleichgewicht zwischen Genauigkeit und Speichereffizienz und ist daher attraktiv für Anwendungen, die Speicher und Recheneffizienz erfordern.

ID	Beschreibung
FLOAT_FP8_E4M3FN	8-Bit-Gleitkommazahl, 4 Bits für den Exponenten und 3 Bits für die Mantisse. Wird in der Regel als Koeffizient verwendet.
FLOAT_FP8_E4M3FNUZ	8-Bit-Gleitkommazahl, 4 Bits für den Exponenten und 3 Bits für die Mantisse. Unterstützt NaN, nicht unterstützt werden Negativ-Nullen und Inf. Wird in der Regel als Koeffizient verwendet.
FLOAT_FP8_E5M2FN	8-Bit-Gleitkommazahl, 5 Bits für den Exponenten und 2 Bits für die Mantisse. Unterstützt NaN und Inf. Wird in der Regel für Gradienten verwendet.
FLOAT_FP8_E5M2FNUZ	8-Bit-Gleitkommazahl, 5 Bits für den Exponenten und 2 Bits für die Mantisse. Unterstützt NaN, nicht unterstützt werden Negativ-Nullen und Inf. Wird auch für Gradienten verwendet.

Einer der Hauptvorteile von FP8 ist seine Effizienz bei der Verarbeitung großer Datenmengen. Durch die Verwendung einer kompakten Zahlendarstellung reduziert FP8 den Speicherbedarf und beschleunigt die Berechnungen. Durch die Verwendung einer kompakten Zahlendarstellung reduziert FP8 den Speicherbedarf und beschleunigt Berechnungen.

© 2000-2025, MetaQuotes Ltd.

Standardbibliothek

Diese Gruppe von Abschnitte enthält technische Details der Arbeit mit der Standardbibliothek MQL5 und Beschreibungen aller ihrer Schlüsselkomponenten.

Die Standardbibliothek MQL5 ist in der Sprache MQL5 geschrieben. Sie vereinfacht Entwicklung von Benutzerprogrammen (Indikatoren, Skripts und Expert Advisors). Die Bibliothek bietet einfachen Zugriff auf die meisten der internen MQL5-Funktionen.

Die Standardbibliothek MQL5 befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include.

Abschnitt	Ort
Mathematik	Include\Math\
OpenCL	Include\OpenCL\
Basisklasse CObject	Include\
Datensammlungen	Include\Arrays\
Template-Sammlungen von Daten	Include\Generic\
Dateien	Include\Files\
Strings	Include\Strings\
Grafische Objekte	Include\Objects\
Benutzerdefinierte Grafiken	Include\Canvas\
3D Grafik	Include\Canvas\
Preischarts	Include\Charts\
Wissenschaftliche Grafiken	Include\Graphics\
Indikatoren	Include\Indicators\
Handelsklassen	Include\Trade\
Module von Strategien	Include\Expert\
Panels und Dialoge	Include\Controls\

© 2000-2025, MetaQuotes Ltd.

Mathematik

Für die Durchführung von Berechnungen in verschiedenen Bereichen der Mathematik werden mehrere Bibliotheken angeboten:

- Statistik – Funktionen für das Arbeiten mit verschiedenen Verteilungen der Wahrscheinlichkeitstheorie
- Fuzzy-Logik – Bibliothek der Fuzzy-Logik, in der Systeme der Inferenzmethoden nach Mamdani und Sugeno implementiert sind
- ALGLIB – Datenanalyse (Clustering, Solution Trees, lineare Regression, neuronale Netze), Lösungen von Differentialgleichugnen, Fourier-Transformation, numerische Integration, Optimierungsaufgaben, statistische Analyse und vieles mehr.

© 2000-2025, MetaQuotes Ltd.

Statistik

Die statistische Bibliothek ist für die Berechnung der grundlegenden statistischen Verteilungen vorhergesehen.

Für jede Verteilung sind in der Bibliothek fünf Funktionen vorhanden:

1. Berechnung der Dichte der Verteilung - MathProbabilityDensityX() Funktionen
2. Berechnung der Wahrscheinlichkeiten - MathCumulativeDistributionX() Funktionen
3. Berechnung der Verteilung der Quantile - MathQuantileX() Funktionen
4. Erzeugung von Zufallszahlen aus einer gegebenen Verteilung - MathRandomX() Funktionen
5. Berechnung der theoretischen Momente - MathMomentsX() Funktionen

Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

- Statistische Eigenschaften
- Normalverteilung
- Log-Normalverteilung
- Beta-Verteilung
- Nichtzentrale Betaverteilung
- Gamma-Verteilung
- Chi-Quadrat Verteilung
- Nichtzentrale Chi-Quadrat Verteilung
- Exponentielle Verteilung
- F-Verteilung
- Nichtzentrale F-Verteilung
- t-Verteilung
- Nichtzentrale t-Verteilung
- Logistische Verteilung
- Cauchy-Verteilung
- Gleichverteilung
- Weibull-Verteilung
- Binomial-Verteilung
- Negative Binomial-Verteilung
- Geometrische Verteilung
- Hypergeometrische Verteilung
- Poisson-Verteilung
- Hilfsfunktionen

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                     NormalDistributionExample.mq5 |
//|                        Copyright 2016, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
//--- fügen wir Funktionen für die Berechnung der Normalverteilung hinzu
#include <Math\Stat\Normal.mqh>
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- setzen wir Parameter der Normalverteilung
   double mu=5.0;
   double sigma=1.0;
   PrintFormat("Normalverteilung mit den Parametern mu=%G und sigma=%G, Beispiele der
//--- Interval setzen
   double x1=mu-sigma;
   double x2=mu+sigma;
//--- Variablen für die Berechnung der Wahrscheinlichkeit
   double cdf1,cdf2,probability;
//--- Variablen für den Fehlercode
   int error_code1,error_code2;
//--- berechnen wir den Wert der Funktion der Verteilung
   cdf1=MathCumulativeDistributionNormal(x1,mu,sigma,error_code1);
   cdf2=MathCumulativeDistributionNormal(x2,mu,sigma,error_code2);
//--- überprüfen wir den Fehlercode
   if(error_code1==ERR_OK && error_code2==ERR_OK)
     {
      //--- berechnen wir die Wahrscheinlichkeit der Zufallsvariablen im angebebenen I
      probability=cdf2-cdf1;
      //--- Ergebnis ausgeben
      PrintFormat("1. Die Wahrscheinlichkeit des Auftretens der Zufallszahl im Bereich
      PrintFormat("  Antwort: Probability = %5.8f",probability);
     }

//--- Finden wir den Wertebereich der Zufallsvariablen x, der mit dem Konfidenzinterva
   probability=0.95; //  setzen wir das Konfidentzintervall
//--- setzen wir Wahrscheinlichkeiten an den Grenzen des Intervalls
   double p1=(1.0-probability)*0.5;
   double p2=probability+(1.0-probability)*0.5;
//--- berechnen wir die Grenzen des Berecihs
   x1=MathQuantileNormal(p1,mu,sigma,error_code1);
   x2=MathQuantileNormal(p2,mu,sigma,error_code2);
//--- überprüfen wir den Fehlercode
   if(error_code1==ERR_OK && error_code2==ERR_OK)
     {
      //--- geben wir das Ergebnis aus
      PrintFormat("2. Für das Konfidenzintervall = %.2f den Wertebereich der Zufallsva
      PrintFormat("  Anwort: Wertebereich  %5.8f <= x <=%5.8f",x1,x2);
     }

   PrintFormat("3. Berechnete und theoretische 4 Momente der Verteilung berechnen");
//--- Generieren wir das Array der Zufallsvariablen, berechnen wir die ersten 4 Moment
   int data_count=1000000;  // setzen wir die Anzahl der Werte und bereiten ein Array
   double data[];
   ArrayResize(data,data_count);
//--- erzeugen wir Zufallszahlen und speichern diese im Array
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<data_count; i++)
     {
      data[i]=MathRandomNormal(mu,sigma,error_code1);
     }
//--- setzen wir den Index des Anfangswertes und die Anzahl der Daten für die Berechnu
   int start=0;
   int count=data_count;
//--- berechnen wir die ersten 4 Momente der erzeugten Werte
   double mean=MathMean(data,start,count);
   double variance=MathVariance(data,start,count);
   double skewness=MathSkewness(data,start,count);
   double kurtosis=MathKurtosis(data,start,count);
//--- Variablen für theoretische Momente
   double normal_mean=0;
   double normal_variance=0;
   double normal_skewness=0;
   double normal_kurtosis=0;
//--- geben wir die Werte der berechneten Momente aus
   PrintFormat("            Mean           Variance          Skewness           Kurtos
   PrintFormat("Calculated  %.10f   %.10f      %.10f      %.10f",mean,variance,skewnes
//--- berechnen wir theoretische Werte der Momente und vergleichen diese mit den erhal
   if(MathMomentsNormal(mu,sigma,normal_mean,normal_variance,normal_skewness,normal_ku
     {
      PrintFormat("Theoretical %.10f   %.10f      %.10f       %.10f",normal_mean,norma
      PrintFormat("Difference  %.10f   %.10f      %.10f       %.10f",mean-normal_mean,
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Statistische Eigenschaften

Diese Gruppe von Funktionen berechnet die Standardmerkmale der Elemente eines Arrays:

- Mittelwert,
- Varianz,
- Schiefe,
- Kurtosis,
- Median,
- durchschnittliche Abweichung und
- Standardabweichung.

Funktion	Beschreibung
MathMean	Berechnet den Mittelwert (erstes Moment) der Elemente eines Arrays
MathVariance	Berechnet die Varianz (zweites Moment) der Elemente eines Arrays
MathSkewness	Berechnet die Schiefe (drittes Moment) der Elemente eines Arrays
MathKurtosis	Berechnet die Kurtosis (viertes Moment) der Elemente eines Arrays
MathMoments	Berechnet die ersten 4 Momente (Mittelwert, Varianz, Schiefe, Kurtosis) der Elemente eines Arrays
MathMedian	Berechnet den Medianwert der Elemente eines Arrays
MathStandardDeviation	Berechnet die Standardabweichung der Elemente eines Arrays
MathAverageDeviation	Berechnet die durchschnittliche Abweichung der Elemente eines Arrays

© 2000-2025, MetaQuotes Ltd.

MathMean

Berechnet den Mittelwert (erstes Moment) der Elemente eines Arrays. Analog zu mean() in R.

```
double  MathMean(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung des Mittelwertes.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Mittelwert der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

© 2000-2025, MetaQuotes Ltd.

MathVariance

Berechnet die Varianz (zweites Moment) der Elemente eines Arrays. Analog zu var() in R.

```
double  MathVariance(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Varianz der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

© 2000-2025, MetaQuotes Ltd.

MathSkewness

Berechnet die Schiefe (drittes Moment) der Elemente eines Arrays. Analog zu skewness() in R (Bibliothek e1071).

```
double  MathSkewness(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Schiefe der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

© 2000-2025, MetaQuotes Ltd.

MathKurtosis

Berechnet die Kurtosis (viertes Moment) der Elemente eines Arrays. Analog zu kurtosis() in R (Bibliothek e1071).

```
double  MathKurtosis(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Kurtosis der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

Hinweis

Die Exzesswölbung wird relativ zur Normalverteilung berechnet (excess kurtosis=kurtosis-3), d. h. die Exzesswölbung einer Normalverteilung beträgt Null.

Sie ist positiv, wenn die Spitze der Verteilung um den Erwartungswert steil ist und negativ bei einer flachen Wölbung.

© 2000-2025, MetaQuotes Ltd.

MathMoments

Berechnet die ersten 4 Momente (Mittelwert, Varianz, Schiefe, Kurtosis) der Elemente eines Arrays.

```
double  MathMoments(
   const double&  array[],              // Array mit Daten
   double&        mean,                 // Mittelwert (erstes Moment)
   double&        variance,             // Varianz (zweites Moment)
   double&        skewness,             // Schiefe (drittes Moment)
   double&        kurtosis,             // Kurtosis (viertes Moment)
   const int      start=0,              // Anfangsindex
   const int      count=WHOLE_ARRAY     // Anzahl der Elemente
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

mean

[out] Variable des Mittelwerts (1. Moment)

variance

[out] Variable der Varianz (2. Moment)

skewness

[out] Variable der Schiefe (3. Moment)

kurtosis

[out] Variable der Kurtosis (4. Moment)

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Gibt nach erfolgreicher Berechnung 'true' zurück, andernfalls 'false'.

Hinweis

Die Exzesswölbung wird relativ zur Normalverteilung berechnet (excess kurtosis=kurtosis-3), d. h. die Exzesswölbung einer Normalverteilung beträgt Null.

Sie ist positiv, wenn die Spitze der Verteilung um den Erwartungswert steil ist und negativ bei einer flachen Wölbung.

© 2000-2025, MetaQuotes Ltd.

MathMedian

Berechnet den Medianwert der Elemente eines Arrays. Analog zu median() in R.

```
double  MathMedian(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Medianwert der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

© 2000-2025, MetaQuotes Ltd.

MathStandardDeviation

Berechnet die Standardabweichung der Elemente eines Arrays. Analog zu sd() in R.

```
double  MathStandardDeviation(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Standardabweichung der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

© 2000-2025, MetaQuotes Ltd.

MathAverageDeviation

Berechnet die durchschnittliche Abweichung der Elemente eines Arrays. Analog zu aad() in R.

```
double  MathAverageDeviation(
   const double&  array[]                // Array mit Daten
   );
```

Parameter

array

[in] Array mit Daten für die Berechnung.

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Durchschnittliche Abweichung der Elemente eines Arrays. Im Fehlerfall retourniert NaN (keine Zahl).

© 2000-2025, MetaQuotes Ltd.

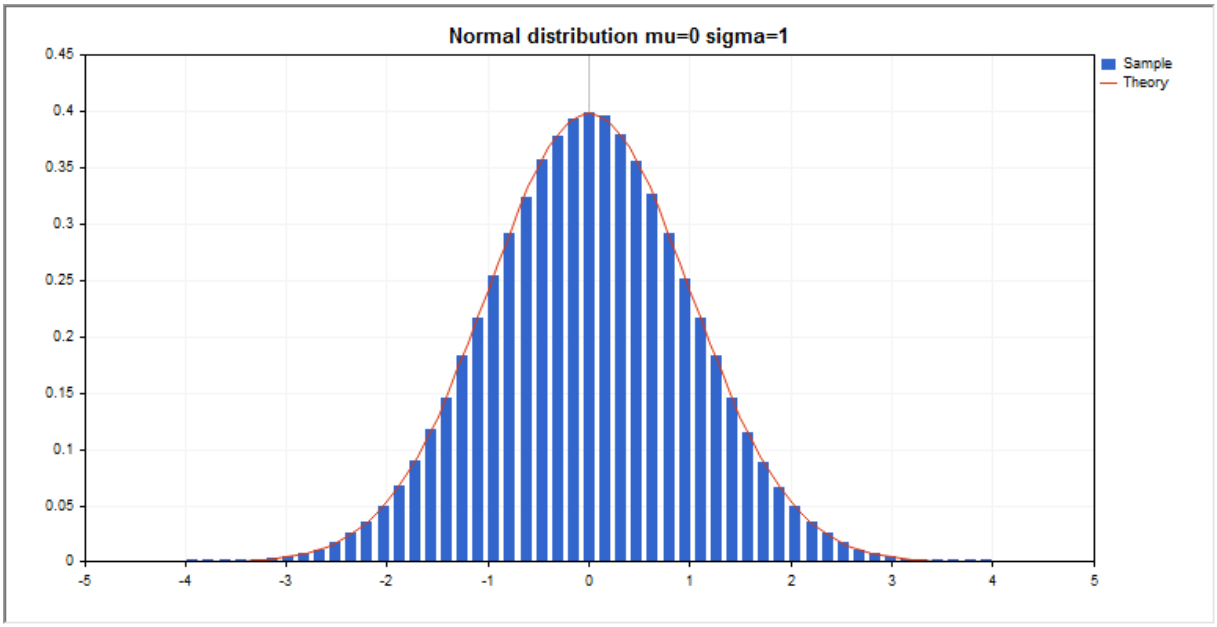
Normalverteilung

In diesem Abschnitt sind Funktionen der Normalverteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Normalverteilung erzeugt werden. Die Normalverteilung wird mit der folgenden Formel berechnet:

$$f_{Normal}(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

wobei:

- x — der Wert der Zufallsvariablen
- μ — mathematische Erwartung
- σ — Standardabweichung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityNormal	Berechnet die Werte der Wahrscheinlichkeitsverteilung einer Normalverteilung
MathCumulativeDistributionNormal	Berechnet den Wert der Normalverteilung
MathQuantileNormal	Berechnet den Wert der Umkehrfunktion der Normalverteilung für eine angegebene Wahrscheinlichkeit
MathRandomNormal	Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Normalverteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsNormal	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Normalverteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Normal.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double mean_value=0;  // Erwartung (mean)
input double std_dev=1;     // Standardabweichung (standard deviation)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=51;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Normalverteilung erhalten
   MathRandomNormal(mean_value,std_dev,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityNormal(x2,mean_value,std_dev,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Normal distribution mu=%G sigma=%G",mean_value
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
//--- plot all curves
   graphic.CurvePlotAll();
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityNormal

Die Funktion berechnet die Werte der Wahrscheinlichkeitsverteilung einer Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung (Erwartung)
   const double  sigma,          // Parameter der Verteilung (Standardabweichung)
   const bool    log_mode,       // Logarithmusberechnung
   int&          error_code      // Variable für den Fehlercode
   );
```

Die Funktion berechnet die Werte der Wahrscheinlichkeitsverteilung einer Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung (Erwartung)
   const double  sigma,          // Parameter der Verteilung (Standardabweichung)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer Log-Normalverteilung mit den Parametern mu und sigma für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pnorm() in R.

```
bool  MathProbabilityDensityNormal(
   const double&  x[],           // Array mit den Werten der Zufallsvariable
   const double   mu,            // mean Parameter der Verteilung (Erwartung)
   const double   sigma,         // sigma Parameter der Verteilung (Standardabweichur
   const bool     log_mode,      // Logarithmusberechnung
   double&        result[]       // Array für die Werte der Funktion der Wahrscheinli
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Normalverteilung mit den Parametern mu und sigma für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityNormal(
   const double&  x[],           // Array mit den Werten der Zufallsvariable
   const double   mu,            // mean Parameter der Verteilung (Erwartung)
   const double   sigma,         // sigma Parameter der Verteilung (Standardabweichur
   double&        result[]       // Array für die Werte der Funktion der Wahrscheinli
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] mean Parameter der Verteilung (mathematische Erwartung.

sigma

[in] sigma Parameter der Verteilung (Standardabweichung).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array der Werte der Funktion der Wahrscheinlichkeitsdichte.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionNormal

Berechnet den Wert der Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   const bool    tail,           // Flag der Berechnung (tail)
   const bool    log_mode,       // Logarithmusberechnung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Funktion der Normalverteilung mit den Parametern mu und sigma für das Array der Zufallvariablen x[]. Im Fehlerfall retourniert false. Analog zu pnorm() in R.

```
bool  MathCumulativeDistributionNormal(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   const bool    tail,           // Flag der Berechnung (tail)
   const bool    log_mode,       // Logarithmusberechnung
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Funktion der Normalverteilung mit den Parametern mu und sigma für das Array der Zufallvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionNormal(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] mean Parameter der Verteilung (mathematische Erwartung.

sigma

[in] sigma Parameter der Verteilung (Standardabweichung).

tail

[in] Flag der Berechnung, wenn tail=true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileNormal

Berechnet den Wert der Umkehrfunktion der Normalverteilung mit den Parametern mu und sigma für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNormal(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   const bool    tail,           // Flag der Berechnung (tail)
   const bool    log_mode,       // Logarithmusberechnung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Normalverteilung mit den Parametern mu und sigma für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNormal(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Normalverteilung mit den Parametern mu und sigma für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false. Analog zu qnorm() in R.

```
bool  MathQuantileNormal(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   const bool    tail,           // Flag der Berechnung (tail)
   const bool    log_mode,       // Logarithmusberechnung
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Normalverteilung mit den Parametern mu und sigma für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileNormal(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Erwartungswert
   const double  sigma,          // Standardabweichung
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

mu

[in] mean Parameter der Verteilung (mathematische Erwartung.

sigma

[in] sigma Parameter der Verteilung (Standardabweichung).

tail

[in] Flag der Berechnung, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomNormal

Erzeugt eine normalverteilte Pseudozufallszahl mit den Parametern mu und sigma. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomNormal(
   const double  mu,              // Erwartungswert
   const double  sigma,           // Standardabweichung
   int&          error_code       // Variable für den Fehlercode
   );
```

Erzeugt normalverteilte Pseudozufallszahlen gemäß den Parametern mu und sigma. Im Fehlerfall retourniert false. Analog rnorm() in R.

```
bool  MathRandomNormal(
   const double  mu,              // Erwartungswert
   const double  sigma,           // Standardabweichung
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

mu

[in] mean Parameter der Verteilung (mathematische Erwartung.

sigma

[in] sigma Parameter der Verteilung (Standardabweichung).

data_count

[in] Anzahl der benötigten Pseudozufallszahlen.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsNormal

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Betaverteilung.

```
double  MathMomentsNormal(
   const double  mu,              // Erwartungswert
   const double  sigma,           // Standardabweichung
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

mu

[in] mean Parameter der Verteilung (mathematische Erwartung.

sigma

[in] sigma Parameter der Verteilung (Standardabweichung).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt nach erfolgreicher Berechnung 'true' zurück, andernfalls 'false'.

© 2000-2025, MetaQuotes Ltd.

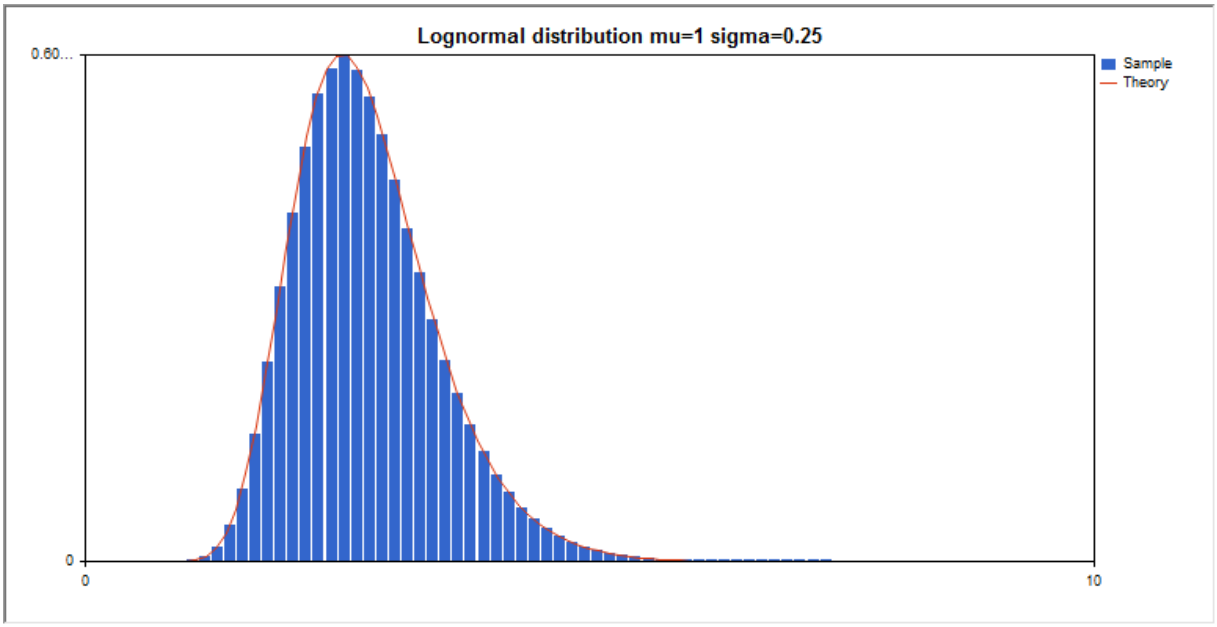
Log-Normalverteilung

In diesem Abschnitt sind Funktionen für die Log-Normalverteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Log-Normalverteilung erzeugt werden. Die Log-Normalverteilung wird mit der folgenden Formel berechnet:

$$f_{Lognormal}(x|\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}}$$

wobei:

- x — der Wert der Zufallsvariablen
- μ — Logarithmus des Erwartungswertes
- σ — Logarithmus der Standardabweichung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityLognormal	Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion der Log-Normalverteilung
MathCumulativeDistributionLognormal	Berechnet den Wert der Log-Normalverteilung
MathQuantileLognormal	Berechnet den Wert der Umkehrfunktion der Log-Normalverteilung für eine angegebene Wahrscheinlichkeit
MathRandomLognormal	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Log-Normalverteilung
MathMomentsLognormal	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Log-Normalverteilung.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Lognormal.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double mean_value=1.0;  // Logarithmus des Erwartungswertes (log mean)
input double std_dev=0.25;    // Logarithmus der Standardabweichung (log standard devi
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=51;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Log-Normalverteilung erhalten
   MathRandomLognormal(mean_value,std_dev,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityLognormal(x2,mean_value,std_dev,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
```

© 2000-2025, MetaQuotes Ltd.

```
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Lognormal distribution mu=%G sigma=%G",mean_va
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der Y-Achse deaktivieren
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Max(theor_max);
   graphic.YAxis().Min(0);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityLognormal

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer Log-Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityLognormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,          // Logarithmus der Standardabweichung (log standard d
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer Log-Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityLognormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,          // Logarithmus der Standardabweichung (log standard d
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer Log-Normalverteilung mit den Parametern mu und sigma für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert sie NaN. Analog zu dlnorm() in R.

```
bool  MathProbabilityDensityLognormal(
   const double&  x[],           // Array mit den Werten der Zufallsvariable
   const double   mu,            // Logarithmus des Erwartungswertes (log mean)
   const double   sigma,         // Logarithmus der Standardabweichung (log standard
   const bool     log_mode,      // Parameter der Logarithmusberechnung, wenn log_mod
   double&        result[]       // Array für die Werte der Funktion der Wahrscheinli
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer Log-Normalverteilung mit den Parametern mu und sigma für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityLognormal(
   const double&  x[],           // Array mit den Werten der Zufallsvariable
   const double   mu,            // Logarithmus des Erwartungswertes (log mean)
   const double   sigma,         // Logarithmus der Standardabweichung (log standard
   double&        result[]       // Array für die Werte der Funktion der Wahrscheinli
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] Logarithmus des Erwartungswertes (log_mean).

sigma

[in] Logarithmus der Standardabweichung (log standard deviation).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array der Werte der Funktion der Wahrscheinlichkeitsdichte.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionLognormal

Berechnet den Wert der Log-Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionLognormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,          // Logarithmus der Standardabweichung (log standard d
   const bool    tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Log-Normalverteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionLognormal(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,          // Logarithmus der Standardabweichung (log standard d
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Log-Normalverteilung mit den Parametern mu und sigma für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert false. Analog zu plnorm() in R.

```
bool  MathCumulativeDistributionLognormal(
   const double&  x[],           // Array mit den Werten der Zufallsvariable
   const double   mu,            // Logarithmus des Erwartungswertes (log mean)
   const double   sigma,         // Logarithmus der Standardabweichung (log standard
   const bool     tail,          // Flag der Berechnung, wenn true, wird die Wahrsche
   const bool     log_mode,      // Parameter der Logarithmusberechnung, wenn log_mod
   double&        result[]       // Array mit den Werten der Wahrscheinlichkeitsfunkt
   );
```

Berechnet den Wert der Log-Normalverteilung mit den Parametern mu und sigma für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionLognormal(
   const double&  x[],           // Array mit den Werten der Zufallsvariable
   const double   mu,            // Logarithmus des Erwartungswertes (log mean)
   const double   sigma,         // Logarithmus der Standardabweichung (log standard
   double&        result[]       // Array mit den Werten der Wahrscheinlichkeitsfunkt
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] Logarithmus des Erwartungswertes (log_mean).

sigma

[in] Logarithmus der Standardabweichung (log standard deviation).

tail

[in] Parameter, wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileLognormal

Berechnet den Wert der Umkehrfunktion der Log-Normalverteilung mit den Parametern mu und sigma für die Wahrscheinlichkeit probability. Im Fehlerfall retourniert sie NaN.

```
double  MathQuantileLognormal(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  mu,             // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,          // Logarithmus der Standardabweichung (log standard d
   const bool    tail,           // Flag, wenn false, wird für eine '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird für eine 'Exp(proba
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Log-Normalverteilung mit den Parametern mu und sigma für die Wahrscheinlichkeit probability. Im Fehlerfall retourniert sie NaN.

```
double  MathQuantileLognormal(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  mu,             // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,          // Logarithmus der Standardabweichung (log standard d
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Umkehrfunktion der Lognormalenverteilung mit den Parametern mu und sigma für den Array der Wahrscheinlichkeitswerten probability[]. Im Fehlerfall retourniert false. Analog zu qlnorm() in R.

```
bool  MathQuantileLognormal(
   const double&  probability[],  // Array mit den Wahrscheinlichkeitswerten der Zufal
   const double   mu,             // Logarithmus des Erwartungswertes (log mean)
   const double   sigma,          // Logarithmus der Standardabweichung (log standard
   const bool     tail,           // Flag, wenn false, wird für eine '1.0-probability'
   const bool     log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(pro
   double&        result[]        // Array mit den Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Log-Normalverteilung mit den Parametern mu und sigma für das Array mit den Wahrscheinlichkeitswerten probability[]. Im Fehlerfall retourniert false.

```
bool  MathQuantileLognormal(
   const double&  probability[],  // Array mit den Wahrscheinlichkeitswerten der Zufal
   const double   mu,             // Logarithmus des Erwartungswertes (log mean)
   const double   sigma,          // Logarithmus der Standardabweichung (log standard
   double&        result[]        // Array mit den Werten der Quantile
   );
```

Parameter

probability

[in] Der Wert der Wahrscheinlichkeit des Auftretens einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

mu

[in] Logarithmus des Erwartungswertes (log_mean).

sigma

[in] Logarithmus der Standardabweichung (log standard deviation).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomLognormal

Erzeugt eine Pseudozufallszahl auf Basis der Log-Normalverteilung mit den Parametern mu und sigma. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomLognormal(
   const double  mu,              // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,           // Logarithmus der Standardabweichung (log standard d
   int&          error_code       // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Log-Normalverteilung mit den Parametern mu und sigma. Im Fehlerfall retourniert false. Analog zu rlnorm() in R.

```
double  MathRandomLognormal(
   const double  mu,              // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,           // Logarithmus der Standardabweichung (log standard d
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Wahrscheinlichkeitswerten der Zufall
   );
```

Parameter

mu

[in] Logarithmus des Erwartungswertes (log_mean).

sigma

[in] Logarithmus der Standardabweichung (log standard deviation).

data_count

[in] Anzahl der benötigten Daten.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsLognormal

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Log-Normalverteilung. Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

```
double  MathMomentsLognormal(
   const double  mu,              // Logarithmus des Erwartungswertes (log mean)
   const double  sigma,           // Logarithmus der Standardabweichung (log standard d
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

mu

[in] Logarithmus des Erwartungswertes (log_mean).

sigma

[in] Logarithmus der Standardabweichung (log standard deviation).

mean

[in] Variable des Mittelwerts.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt nach erfolgreicher Berechnung 'true' zurück, andernfalls 'false'.

© 2000-2025, MetaQuotes Ltd.

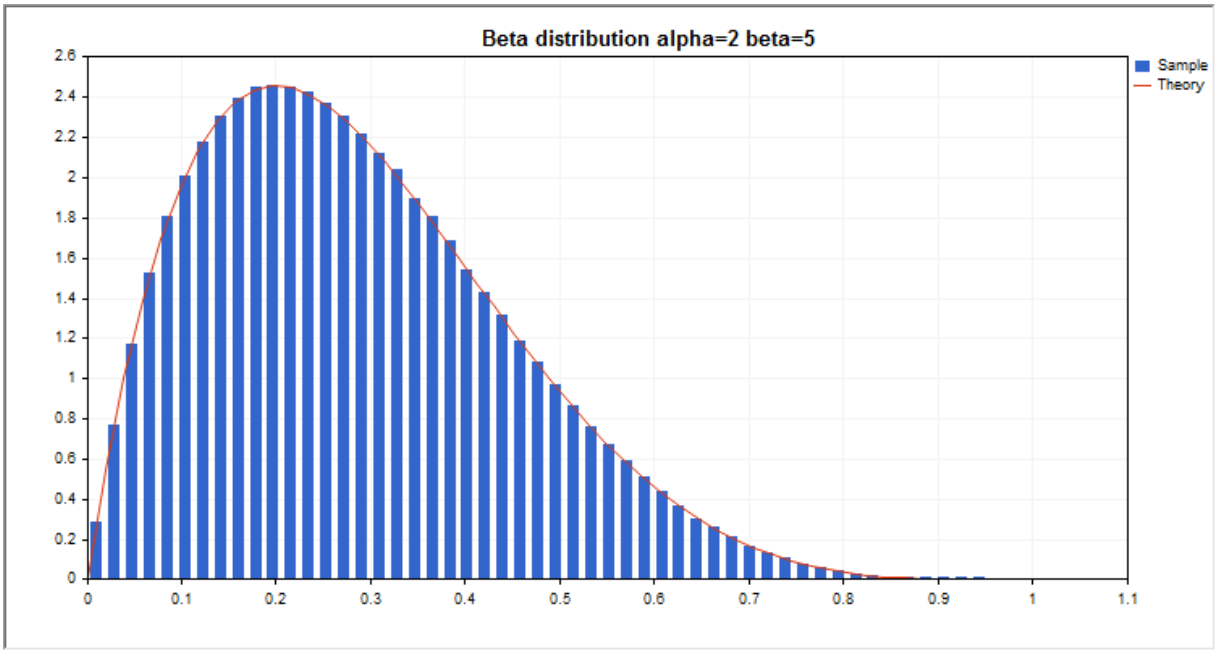
Beta-Verteilung

In diesem Abschnitt sind Funktionen für die Beta-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Betaverteilung erzeugt werden. Die Beta-Verteilung wird mit der folgenden Formel berechnet:

$$f_{Beta}(x \mid a,b) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}$$

wobei:

- x — der Wert der Zufallsvariablen
- a — der erste Parameter der Betaverteilung
- b — der zweite Parameter der Betaverteilung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityBeta	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Betaverteilung
MathCumulativeDistributionBeta	Berechnet den Wert der Wahrscheinlichkeitsverteilung der Betaverteilung
MathQuantileBeta	Berechnet den Wert der Umkehrfunktion der Wahrscheinlichkeitsverteilung der Betaverteilung für eine angegebene Wahrscheinlichkeit
MathRandomBeta	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Betaverteilung
MathMomentsBeta	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Betaverteilung

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Beta.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double alpha=2;   // der erste Parameter der Beta-Verteilung (shape1)
input double beta=5;    // der zweite Parameter der Beta-Verteilung (shape2)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=51;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Beta-Verteilung erhalten
   MathRandomBeta(alpha,beta,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityBeta(x2,alpha,beta,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
```

© 2000-2025, MetaQuotes Ltd.

```
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Beta distribution alpha=%G beta=%G",alpha,beta
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityBeta

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Betaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Betaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Betaverteilung mit den Parametern a und b für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dbeta() in R.

```
bool  MathProbabilityDensityBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Betaverteilung mit den Parametern a und b für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter der Betaverteilung (shape 2)

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array der Werte der Funktion der Wahrscheinlichkeitsdichte.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionBeta

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Betaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const bool    tail,           // Flag der Berechnung, wenn true, dann wird die Wahrs
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_mod
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Betaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Betaverteilung mit den Parametern a und b für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pbeta() in R.

```
bool  MathCumulativeDistributionBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const bool    tail,           // Parameter, wenn true, wird die Wahrscheinlichkeit
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Berechnet die Wahrscheinlichkeitsverteilung der Betaverteilung mit den Parametern a und b für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter der Betaverteilung (shape 2)

tail

[in] Parameter, wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileBeta

Berechnet den Wert der Umkehrfunktion der Betaverteilung mit den Parametern a und b für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileBeta(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Betaverteilung mit den Parametern a und b für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileBeta(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Betaverteilung mit den Parametern a und b für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false. Analog zu qbeta() in R.

```
double  MathQuantileBeta(
   const double&  probability[], // Array mit den Werten der Wahrscheinlichkeitsfunkti
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Betaverteilung mit den Parametern a und b für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false.

```
bool  MathQuantileBeta(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape1).

b

[in] Der zweite Parameter der Betaverteilung (shape2).

tail

[in] Flag, wenn lower_tail=false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomBeta

Erzeugt eine Pseudozufallszahl auf Basis der Betaverteilung mit den Parametern a und b. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomBeta(
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Betaverteilung mit den Parametern a und b. Im Fehlerfall retourniert false. Analog zu rbeta() in R.

```
bool  MathRandomBeta(
   const double  a,               // der erste Parameter der Betaverteilung (shape1)
   const double  b,               // der zweite Parameter der Betaverteilung (shape2)
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

a

[in] Der erste Parameter der Betaverteilung (shape1)

b

[in] Der zweite Parameter der Betaverteilung (shape2).

data_count

[in] Anzahl der benötigten Pseudozufallszahlen.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsBeta

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Betaverteilung.

```
double  MathMomentsBeta(
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

a

[in] Der erste Parameter der Betaverteilung (shape1).

b

[in] Der zweite Parameter der Betaverteilung (shape2).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt nach erfolgreicher Berechnung 'true' zurück, andernfalls 'false'.

© 2000-2025, MetaQuotes Ltd.

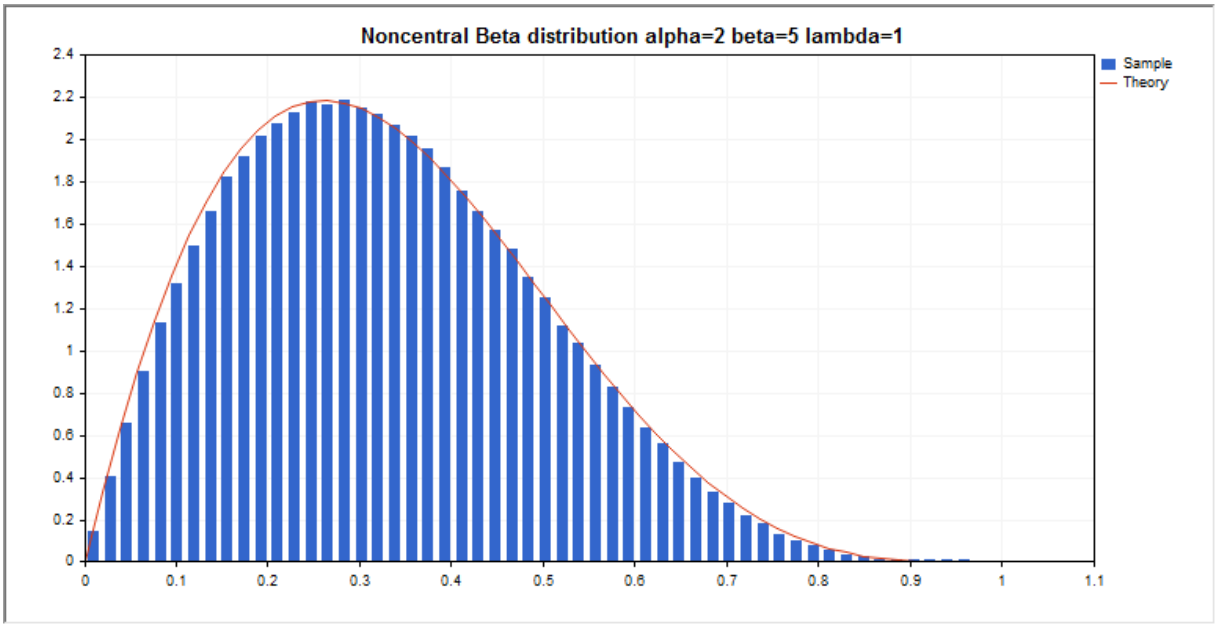
Nichtzentrale Betaverteilung

In diesem Abschnitt sind Funktionen für die nichtzentrale Betaverteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Betaverteilung erzeugt werden. Die nichtzentrale Betaverteilung wird mit der folgenden Formel berechnet:

$$f_{NoncentralBeta}(x|a,b,\lambda)=\sum_{r=0}^{\infty}e^{-\frac{\lambda}{2}}\frac{\left(\frac{\lambda}{2}\right)^{r}}{r!}\frac{x^{a+r-1}(1-x)^{b-1}}{B(a+r,b)}$$

wobei:

- x — der Wert der Zufallsvariablen
- a — der erste Parameter der Betaverteilung
- b — der zweite Parameter der Betaverteilung
- λ — Parameter der Nichtzentralität



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityNoncentralBeta	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Betaverteilung
MathCumulativeDistributionNoncentralBeta	Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung
MathQuantileNoncentralBeta	Berechnet den Wert der Umkehrfunktion der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung für eine angegebene Wahrscheinlichkeit

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathRandomNoncentralBeta	Erzeugt ein Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der nichtzentrale Betaverteilung
MathMomentsNoncentralBeta	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen Betaverteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\NoncentralBeta.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double a_par=2;     // der erste Parameter der Beta-Verteilung (shape1)
input double b_par=5;     // der zweite Parameter der Beta-Verteilung (shape2)
input double l_par=1;     // Parameter der Nichtzentralität (lambda)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=53;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der nichtzentralen Beta-Verteilung erhalten
   MathRandomNoncentralBeta(a_par,b_par,l_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityNoncentralBeta(x2,a_par,b_par,l_par,false,y2);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Noncentral Beta distribution alpha=%G beta=%G
                          a_par,b_par,l_par));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
```

© 2000-2025, MetaQuotes Ltd.

```
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityNoncentralBeta

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dbeta() in R.

```
bool  MathProbabilityDensityNoncentralBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityNoncentralBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

© 2000-2025, MetaQuotes Ltd.

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter der Betaverteilung (shape 2)

lambda

[in] Parameter der Nichtzentralität

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionNoncentralBeta

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   const bool    tail,           // lag der Berechnung, wenn true, dann wird die Wahrsc
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_moc
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralBeta(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pbeta() in R.

```
bool  MathCumulativeDistributionNoncentralBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   const bool    tail,           // Flag der Berechnung, wenn true, dann wird die Wahr
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionNoncentralBeta(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

© 2000-2025, MetaQuotes Ltd.

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter der Betaverteilung (shape 2)

lambda

[in] Parameter der Nichtzentralität

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileNoncentralBeta

Berechnet den Wert der Umkehrfunktion der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die Zufallsvariable *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNoncentralBeta(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die Zufallsvariable *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNoncentralBeta(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für die im Array 'probability[]' angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false. Analog zu qbeta() in R.

```
double  MathQuantileNoncentralBeta(
   const double&  probability[],  // Array mit den Wahrscheinlichkeitswerten der Zufal
   const double  a,               // der erste Parameter der Betaverteilung (shape1)
   const double  b,               // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,          // Parameter der Nichtzentralität
   const bool    tail,            // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,        // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]         // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Wahrscheinlichkeitsverteilung der nichtzentralen Betaverteilung mit den Parametern a, b und lambda für das Array der Wahrscheinlichkeitswerte probability[]. Im Fehlerfall retourniert false.

```
bool  MathQuantileNoncentralBeta(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
```

© 2000-2025, MetaQuotes Ltd.

```
   const double   lambda,          // Parameter der Nichtzentralität
   double&        result[]         // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape1).

b

[in] Der zweite Parameter der Betaverteilung (shape2).

lambda

[in] Parameter der Nichtzentralität.

tail

[in] Flag der Berechnung, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag der Berechnung, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomNoncentralBeta

Erzeugt eine Pseudozufallszahl auf Basis der nichtzentralen Betaverteilung mit den Parametern a, b und lambda. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomNoncentralBeta(
   const double  a,             // der erste Parameter der Betaverteilung (shape1)
   const double  b,             // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,        // Parameter der Nichtzentralität
   int&          error_code     // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der nichtzentralen Betaverteilung mit den Parametern a, b und lambda. Im Fehlerfall retourniert false. Analog zu rbeta() in R.

```
bool  MathRandomNoncentralBeta(
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

a

[in] Der erste Parameter der Betaverteilung (shape1)

b

[in] Der zweite Parameter der Betaverteilung (shape2).

lambda

[in] Parameter der Nichtzentralität

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsNoncentralBeta

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen Betaverteilung mit den Parametern a, b und lambda.

```
double  MathMomentsNoncentralBeta(
   const double  a,              // der erste Parameter der Betaverteilung (shape1)
   const double  b,              // der zweite Parameter der Betaverteilung (shape2)
   const double  lambda,         // Parameter der Nichtzentralität
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

a

[in] Der erste Parameter der Betaverteilung (shape1).

b

[in] Der zweite Parameter der Betaverteilung (shape2).

lambda

[in] Parameter der Nichtzentralität

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

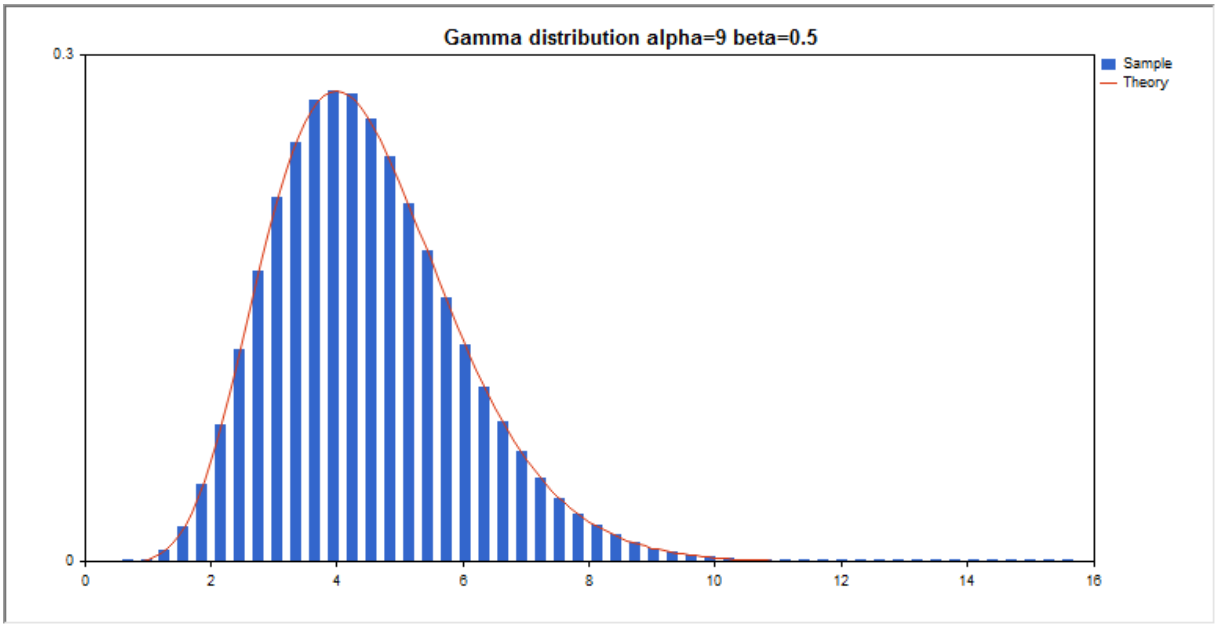
Gamma-Verteilung

In diesem Abschnitt sind Funktionen der Gamma-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Betaverteilung erzeugt werden. Die Gamma-Verteilung wird mit der folgenden Formel berechnet:

$$f_{Gamma}(x|a,b) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-\frac{x}{b}}$$

wobei:

- x — der Wert der Zufallsvariablen
- a — der erste Parameter der Verteilung
- b — der zweite Parameter der Verteilung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityGamma	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gammaverteilung
MathCumulativeDistributionGamma	Berechnet den Wert der Gammaverteilung
MathQuantileGamma	Berechnet den Wert der Umkehrfunktion der Gammaverteilung für eine angegebene Wahrscheinlichkeit
MathRandomGamma	Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Gammaverteilung
MathMomentsGamma	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Gammaverteilung

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Gamma.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double alpha=9;   // der erste Parameter der Gamma-Verteilung (shape)
input double beta=0.5;  // der zweite Parameter der Gamma-Verteilung (scale)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Gamma-Verteilung erhalten
   MathRandomGamma(alpha,beta,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityGamma(x2,alpha,beta,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
```

© 2000-2025, MetaQuotes Ltd.

```
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Gamma distribution alpha=%G beta=%G",alpha,bet
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der Y-Achse deaktivieren
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Max(NormalizeDouble(theor_max,1));
   graphic.YAxis().Min(0);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityGamma

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gammaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityGamma(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gammaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityGamma(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gammaverteilung mit den Parametern a und b für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dgamma() in R.

```
bool  MathProbabilityDensityGamma(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gammaverteilung mit den Parametern a und b für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityGamma(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter (scale).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionGamma

Berechnet den Wert der Gammaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionGamma(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_mo
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Gammaverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionGamma(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Gammaverteilung mit den Parametern a und b für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pgamma() in R.

```
bool  MathCumulativeDistributionGamma(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrsche
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mod
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkt
   );
```

Berechnet den Wert der Gammaverteilung mit den Parametern a und b für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionGamma(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkt
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter der Betaverteilung (scale)

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileGamma

Berechnet den Wert der Umkehrfunktion der Gammaverteilung mit den Parametern a und b für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileGamma(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Gammaverteilung mit den Parametern a und b für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileGamma(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Gammaverteilung mit den Parametern a und b für ein Array mit den Werten der Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false. Analog zu qgamma() in R.

```
double  MathQuantileGamma(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Gammaverteilung mit den Parametern a und b für ein Array mit den Werten der Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert sie false.

```
bool  MathQuantileGamma(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter (scale).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomGamma

Erzeugt eine Pseudozufallszahl auf Basis der Gammaverteilung mit den Parametern a und b. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomGamma(
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt eine Pseudozufallszahlen auf Basis der Gammaverteilung mit den Parametern a und b. Im Fehlerfall retourniert false. Analog zu rgamma() in R.

```
bool  MathRandomGamma(
   const double  a,               // der erste Parameter der Verteilung (shape)
   const double  b,               // der zweite Parameter der Verteilung (scale)
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

a

[in] Der erste Parameter der Betaverteilung (shape 1).

b

[in] Der zweite Parameter (scale).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsGamma

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Gammaverteilung mit den Parametern a und b.

```
double  MathMomentsGamma(
   const double  a,              // der erste Parameter der Verteilung (shape)
   const double  b,              // der zweite Parameter der Verteilung (scale)
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

a

[in] Der erste Parameter der Verteilung (shape).

b

[in] Der zweite Parameter der Verteilung (scale).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

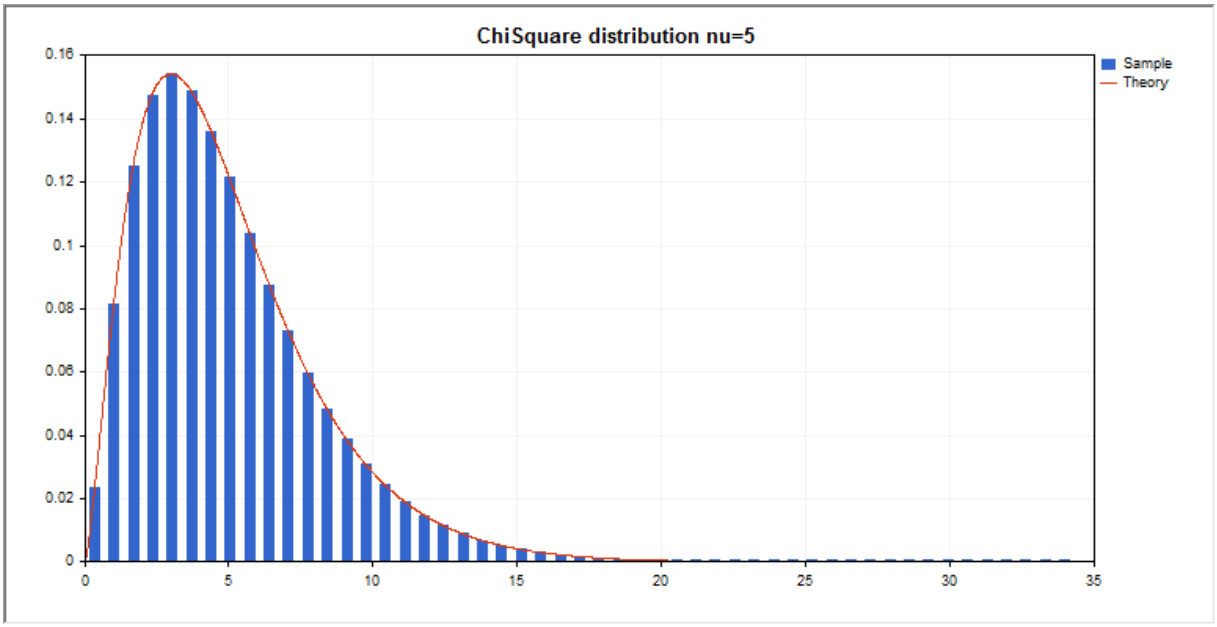
Chi-Quadrat Verteilung

In diesem Abschnitt sind Funktionen für die Chi-Quadrat-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Betaverteilung erzeugt werden. Die Chi-Quadrat-Verteilung wird mit der folgenden Formel berechnet:

$$f_{Chi-Square}(x|\nu) = \frac{x^{\frac{(\nu-2)}{2}} e^{-\frac{x}{2}}}{2^{\frac{\nu}{2}}\Gamma\left(\frac{\nu}{2}\right)}$$

wobei:

- x — der Wert der Zufallsvariablen
- ν — Anzahl der Freiheitsgrade



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityChiSquare	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Chi-Quadrat-Verteilung
MathCumulativeDistributionChiSquare	Berechnet den Wert der Wahrscheinlichkeitsverteilung der Chi-Quadrat-Verteilung
MathQuantileChiSquare	Berechnet den Wert der Umkehrfunktion der Chi-Quadrat-Verteilung für eine angegebene Wahrscheinlichkeit
MathRandomChiSquare	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Chi-Quadrat-Verteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsChiSquare	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Chi-Quadrat-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\ChiSquare.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double nu_par=5;     // Zahl der Freiheitsgrade
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Chi-Quadrat-Verteilung erhalten
+   MathRandomChiSquare(nu_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityChiSquare(x2,nu_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("ChiSquare distribution nu=%G ",nu_par));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityChiSquare

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Chi-Quadrat-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Chi-Quadrat-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Chi-Quadrat-Verteilung mit dem Parameter nu für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dchisq() in R.

```
bool  MathProbabilityDensityChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Chi-Quadrat-Verteilung mit dem Parameter nu für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade)

log_mode

© 2000-2025, MetaQuotes Ltd.

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionChiSquare

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Chi-Quadrat-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_mod
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Chi-Quadrat-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall gibt NaN zurück

```
double  MathCumulativeDistributionChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Chi-Quadrat-Verteilung mit dem Parameter nu für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pchisq() in R.

```
bool  MathCumulativeDistributionChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Chi-Quadrat-Verteilung mit dem Parameter nu für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileChiSquare

Berechnet den Wert der Umkehrfunktion der Chi-Quadrat-Verteilung für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileChiSquare(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Chi-Quadrat-Verteilung für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileChiSquare(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Chi-Quadrat-Verteilung für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false. Analog zu qchisq() in R.

```
double  MathQuantileChiSquare(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Chi-Quadrat-Verteilung für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileChiSquare(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomChiSquare

Erzeugt eine Pseudozufallszahl auf Basis der Chi-Quadrat-Verteilung mit dem Parameter nu. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomChiSquare(
   const double  nu,            // Parameter der Verteilung (Anzahl der Freiheitsgrade
   int&          error_code     // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Chi-Quadrat-Verteilung mit dem Parameter nu. Im Fehlerfall retourniert false. Analog zu rchisq() in R.

```
bool  MathRandomChiSquare(
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsChiSquare

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Chi-Quadrat-Verteilung mit dem Parameter nu.

```
double  MathMomentsChiSquare(
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgrad
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

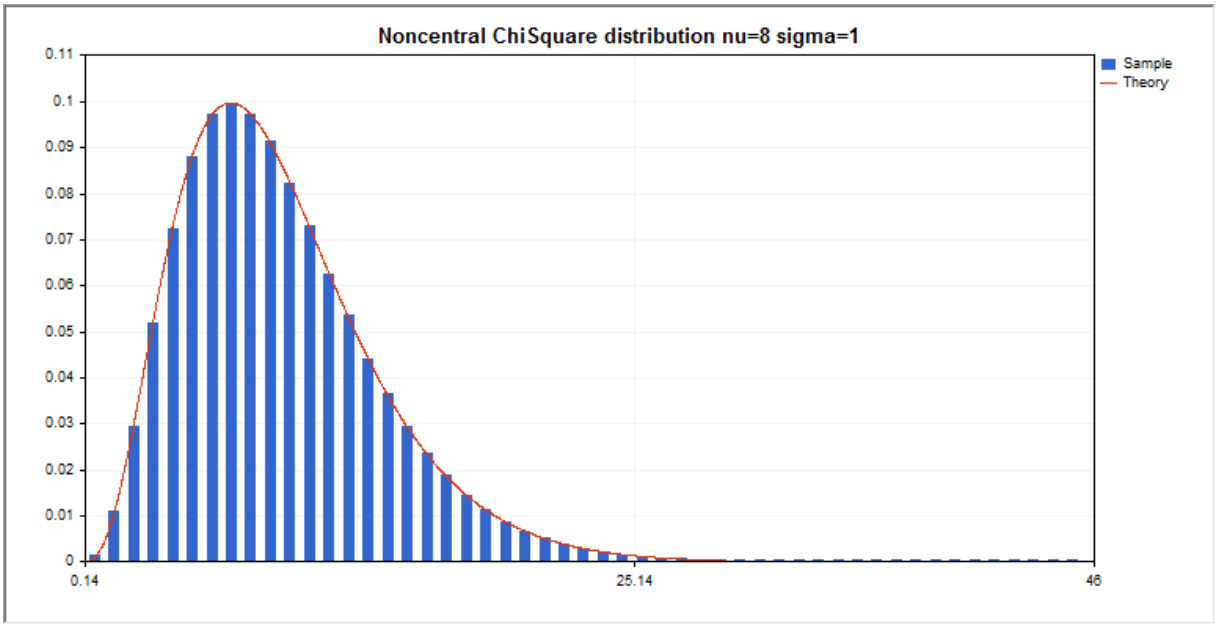
Nichtzentrale Chi-Quadrat Verteilung

In diesem Abschnitt sind Funktionen für die nicht nichtzentrale Chi-Quadrat-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Betaverteilung erzeugt werden. Die nichtzentrale Chi-Quadrat-Verteilung wird mit der folgenden Formel berechnet:

$$f_{NoncentralChiSquare}(x|\nu,\sigma)=\frac{1}{2^{\frac{\nu}{2}}\Gamma\left(\frac{1}{2}\right)}x^{\frac{\nu}{2}-1}e^{-\frac{(x+\sigma)}{2}}\sum_{r=0}^{\infty}\frac{(\lambda x)^r}{(2r)!}\frac{\Gamma\left(\frac{1}{2}+r\right)}{\Gamma\left(\frac{\nu}{2}+r\right)}$$

wobei:

- x — der Wert der Zufallsvariablen
- ν — Anzahl der Freiheitsgrade
- σ — Parameter der Nichtzentralität



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityNoncentralChiSquare	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Chi-Quadrat-Verteilung
MathCumulativeDistributionNoncentralChiSquare	Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Chi-Quadrat-Verteilung
MathQuantileNoncentralChiSquare	Berechnet den Wert der Umkehrfunktion der nichtzentralen Chi-Quadrat-Verteilung für eine angegebene Wahrscheinlichkeit

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathRandomNoncentralChiSquare	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der nichtzentralen Chi-Quadrat-Verteilung
MathMomentsNoncentralChiSquare	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen Chi-Quadrat-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\NoncentralChiSquare.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double nu_par=8;     // Zahl der Freiheitsgrade
input double si_par=1;     // Parameter der Nichtzentralität
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der nichtzentralen Chi-Quadrat-Verteilung erhalten
   MathRandomNoncentralChiSquare(nu_par,si_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityNoncentralChiSquare(x2,nu_par,si_par,false,y2);
//--- skalieren
```

© 2000-2025, MetaQuotes Ltd.

```
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Noncentral ChiSquare distribution nu=%G sigma=
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der X-Achse deaktivieren
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Max(NormalizeDouble(max,0));
   graphic.XAxis().Min(min);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       int ind=int((data[i]-minv)/width);
       if(ind>=cells) ind=cells-1;
       frequency[ind]++;
      }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityNoncentralChiSquare

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Chi-Quadrat-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Chi-Quadrat-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Chi-Quadrat-Verteilung mit dem Parameter nu für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert false. Analog zu dchisq() in R.

```
bool  MathProbabilityDensityNoncentralChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Chi-Quadrat-Verteilung mit dem Parameter nu für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityNoncentralChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionNoncentralChiSquare

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag der Berechnung, wenn lower_tail=true, wird die
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_mod
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralChiSquare(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert false. Analog zu pchisq() in R.

```
bool  MathCumulativeDistributionNoncentralChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag der Berechnung, wenn lower_tail=true, wird di
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für einen Array der Zufallsvariable x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionNoncentralChiSquare(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileNoncentralChiSquare

Berechnet den Wert der Umkehrfunktion der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNoncentralChiSquare(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNoncentralChiSquare(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma für das Array mit Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false. Analog zu qchisq() in R.

```
double  MathQuantileNoncentralChiSquare(
   const double&  probability[],  // Array mit den Wahrscheinlichkeitswerten der Zufal
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgra
   const double  sigma,           // Parameter der Nichtzentralität
   const bool    tail,            // Flag, wenn false, wird mit einer '1.0-probability
   const bool    log_mode,        // Flag der Berechnung, wenn log_mode=true, wird mit
   double&       result[]         // Array mit den Werten der Quatile
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen Chi-Quadrat-Verteilung für das Array mit Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileNoncentralChiSquare(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

© 2000-2025, MetaQuotes Ltd.

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomNoncentralChiSquare

Erzeugt eine Pseudozufallszahl auf Basis der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomNoncentralChiSquare(
   const double  nu,            // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  sigma,         // Parameter der Nichtzentralität
   int&          error_code     // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der nichtzentralen Chi-Quadrat-Verteilung mit den Parametern nu und sigma. Im Fehlerfall retourniert false. Analog zu rchisq() in R.

```
bool  MathRandomNoncentralChiSquare(
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsNoncentralChiSquare

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen Chi-Quadrat-Verteilung mit dem Parameter nu und sigma.

```
double  MathMomentsNoncentralChiSquare(
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  sigma,          // Parameter der Nichtzentralität
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

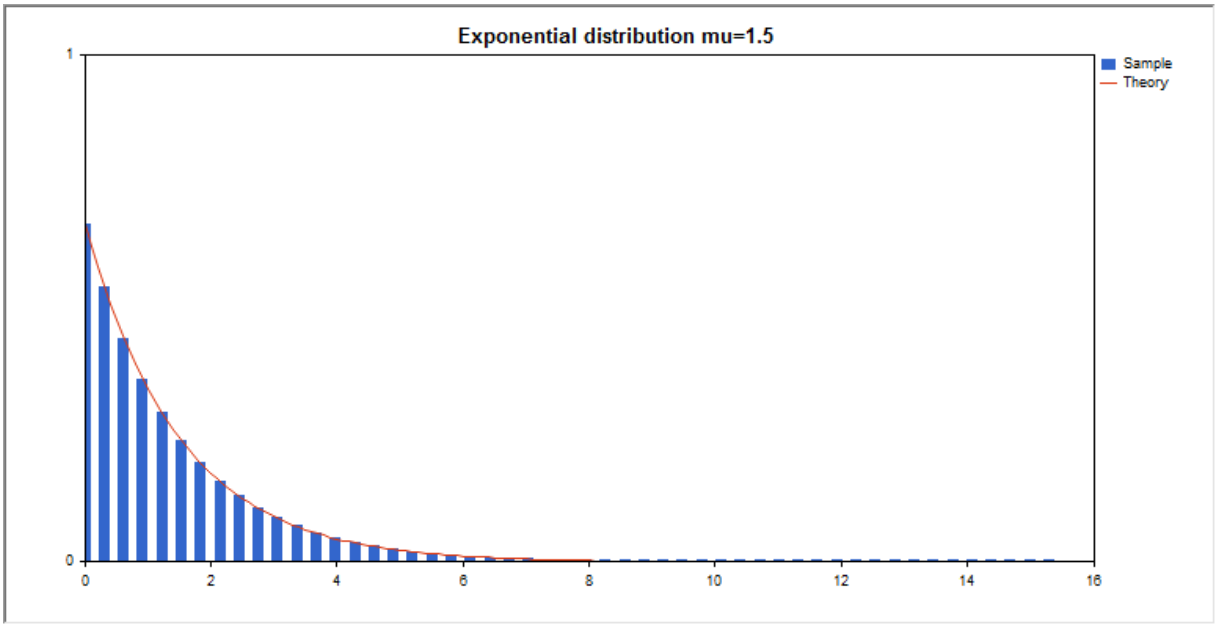
Exponentielle Verteilung

In diesem Abschnitt sind Funktionen für die exponentielle Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der exponentiellen Verteilung erzeugt werden. Die exponentielle Verteilung wird mit der folgenden Formel berechnet:

$$f_{Exponential}(x|\mu) = \frac{1}{\mu} e^{-\frac{x}{\mu}}$$

wobei:

- x — der Wert der Zufallsvariablen
- μ — mathematische Erwartung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityExponential	Berechnet die Wahrscheinlichkeitsdichtefunktion der exponentiellen Verteilung
MathCumulativeDistributionExponential	Berechnet den Wert der Exponentialverteilung
MathQuantileExponential	Berechnet den Wert der Umkehrfunktion der Exponentialverteilung für eine angegebene Wahrscheinlichkeit
MathRandomExponential	Erzeugt eine Pseudozufallszahl auf Basis der Exponentialverteilung
MathMomentsExponential	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Exponentialverteilung

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Exponential.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double mu_par=1.5;     // Zahl der Freiheitsgrade
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=51;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Exponentialverteilung erhalten
   MathRandomExponential(mu_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityExponential(x2,mu_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
```

© 2000-2025, MetaQuotes Ltd.

```
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Exponential distribution mu=%G ",mu_par));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+i*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
```

© 2000-2025, MetaQuotes Ltd.

```
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityExponential

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion der exponentiellen Verteilung mit dem Parameter mu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityExponential(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion der exponentiellen Verteilung mit dem Parameter mu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityExponential(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion der exponentiellen Verteilung mit dem Parameter mu für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dexp() in R.

```
bool  MathProbabilityDensityExponential(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion der exponentiellen Verteilung mit dem Parameter mu für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityExponential(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] Parameter der Verteilung (mathematische Erwartung).

log_mode

© 2000-2025, MetaQuotes Ltd.

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionExponential

Berechnet den Wert der exponentiellen Verteilung mit dem Parameter mu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionExponential(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_mo
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der exponentiellen Verteilung mit dem Parameter mu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionExponential(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der exponentiellen Verteilung mit dem Parameter mu für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pexp() in R.

```
bool  MathCumulativeDistributionExponential(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der exponentiellen Verteilung mit dem Parameter mu für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionExponential(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] Parameter der Verteilung (mathematische Erwartung).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileExponential

Berechnet den Wert der Umkehrfunktion der exponentiellen Verteilung mit dem Parameter mu für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileExponential(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der exponentiellen Verteilung mit dem Parameter mu für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileExponential(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der exponentiellen Verteilung mit dem Parameter mu für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false. Analog zu qexp() in R.

```
double  MathQuantileExponential(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der exponentiellen Verteilung mit dem Parameter mu für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false.

```
bool  MathQuantileExponential(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

mu

© 2000-2025, MetaQuotes Ltd.

[in] Parameter der Verteilung (mathematische Erwartung).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomExponential

Erzeugt eine Pseudozufallszahl auf Basis der Exponentialverteilung mit dem Parameter mu. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomExponential(
   const double  mu,            // Parameter der Verteilung (mathematische Erwartung)
   int&          error_code     // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Exponentialverteilung mit dem Parameter mu. Im Fehlerfall retourniert false. Analog zu rexp() in R.

```
bool  MathRandomExponential(
   const double  mu,             // Parameter der Verteilung (mathematische Erwartung)
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

mu

[in] Parameter der Verteilung (mathematische Erwartung).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsExponential

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Exponentialverteilung mit dem Parameter mu.

```
double  MathMomentsExponential(
   const double  mu,              // Parameter der Verteilung (mathematische Erwartung)
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

mu

[in] Parameter der Verteilung (mathematische Erwartung).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

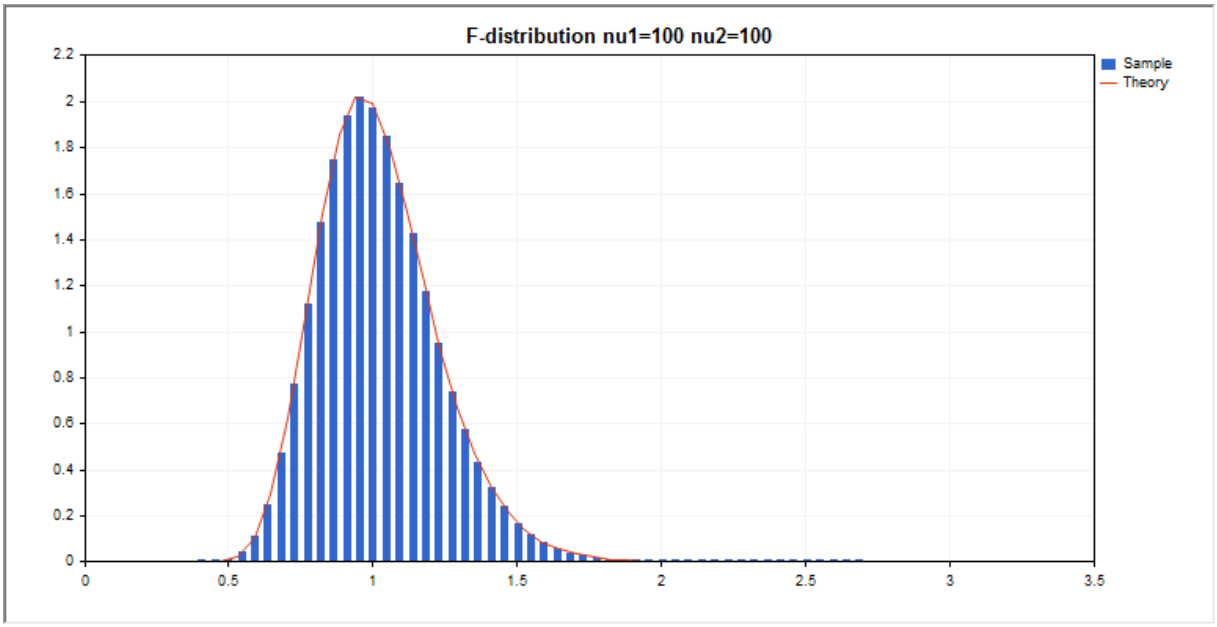
F-Verteilung

Hier sind Funktionen der F-Verteilung. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet werden und Pseudozufallszahlen auf Basis der F-Verteilung erzeugt werden. Die F-Verteilung wird mit der folgenden Formel berechnet:

$$f_F(x \mid \nu_1, \nu_2) = \frac{\Gamma\left(\frac{\nu_1+\nu_2}{2}\right)}{\Gamma\left(\frac{\nu_1}{2}\right)\Gamma\left(\frac{\nu_2}{2}\right)}\left(\frac{\nu_1}{\nu_2}\right)^{\frac{\nu_1}{2}} \frac{x^{\frac{\nu_1-2}{2}}}{\left(1+\left(\frac{\nu_1}{\nu_2}\right)x\right)^{\frac{\nu_1+\nu_2}{2}}}$$

wobei:

- x — der Wert der Zufallsvariablen
- ν1 — der erste Parameter der Verteilung (Anzahl der Freiheitsgrade)
- ν2 — der zweite Parameter der Verteilung (Anzahl der Freiheitsgrade)



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityF	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der F-Verteilung
MathCumulativeDistributionF	Berechnet den Wert der Wahrscheinlichkeitsverteilung der F-Verteilung
MathQuantileF	Berechnet den Wert der Umkehrfunktion der F-Verteilung für eine angegebene Wahrscheinlichkeit
MathRandomF	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der F-Verteilung von Fisher

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsF	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der F-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\F.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double nu_1=100;     // erste Zahl der Freiheitsgrade
input double nu_2=100;     // zweite Zahl der Freiheitsgrade
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der F-Verteilung erhalten
   MathRandomF(nu_1,nu_2,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityF(x2,nu_1,nu_2,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("F-distribution nu1=%G nu2=%G",nu_1,nu_2));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(4);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityF

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu df() in R.

```
bool  MathProbabilityDensityF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionF

Berechnet den Wert der Wahrscheinlichkeitsverteilung der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrscheir
   const bool    log_mode,       // Berechnung den Logarithmus des Wertes, wenn log_mod
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pf() in R.

```
bool  MathCumulativeDistributionF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der F-Verteilung von Fisher mit den Parametern nu1 und nu2 für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileF

Berechnet den Wert der Umkehrfunktion der F-Verteilung mit den Parametern nu1 und nu2 für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileF(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der F-Verteilung mit den Parametern nu1 und nu2 für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileF(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der F-Verteilung mit den Parametern nu1 und nu2 für ein Array der angegebenen Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false. Analog zu qf() in R.

```
double  MathQuantileF(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der F-Verteilung mit den Parametern nu1 und nu2 für ein Array der angegebenen Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileF(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

tail

[in] Flag, wenn lower_tail=false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomF

Erzeugt eine Pseudozufallszahl auf Basis der F-Verteilung von Fisher mit den Parametern nu1 und nu2. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomF(
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   int&          error_code     // Variable für den Fehlercode
   );
```

Erzeugt eine Pseudozufallszahl auf Basis der F-Verteilung von Fisher mit den Parametern nu1 und nu2. Im Fehlerfall retourniert false. Analog rf() in R.

```
bool  MathRandomF(
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsF

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der F-Verteilung von Fisher mit den Parametern nu1 und nu2.

```
double  MathMomentsF(
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

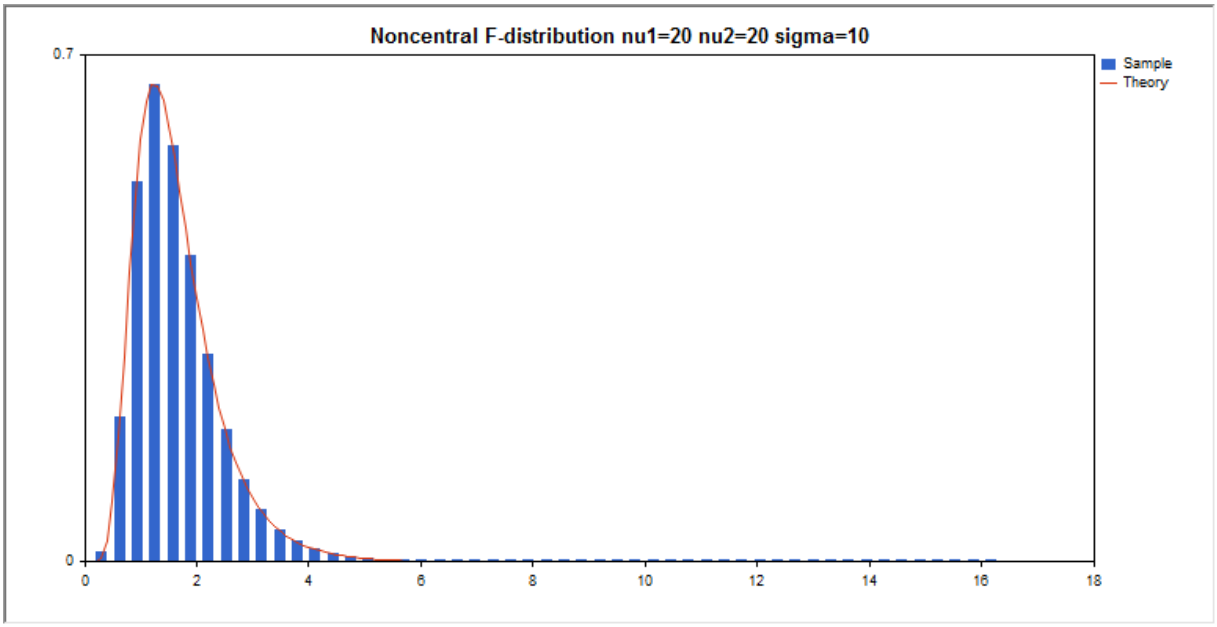
Nichtzentrale F-Verteilung

In diesem Abschnitt sind Funktionen für die nichtzentrale F-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der nichtzentralen F-Verteilung erzeugt werden. Die nichtzentrale F-Verteilung wird mit der folgenden Formel berechnet:

$$f_{NoncentralF}(x|\nu_1,\nu_2,\sigma) = e^{-\frac{\sigma}{2}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{\sigma}{2}\right)^r \frac{\Gamma\left(\frac{\nu_1+\nu_2}{2}+r\right)}{\Gamma\left(\frac{\nu_2}{2}+r\right)\Gamma\left(\frac{\nu_2}{2}\right)} \left(\frac{\nu_1}{\nu_2}\right)^{\frac{\nu_2}{2}+r} \frac{x^{\frac{\nu_2}{2}-1+r}}{\left(1+\frac{\nu_1}{\nu_2}x\right)^{\frac{\nu_1+\nu_2}{2}+r}}$$

wobei:

- x — der Wert der Zufallsvariablen
- ν1 — der erste Parameter der Verteilung (Anzahl der Freiheitsgrade)
- ν2 — der zweite Parameter der Verteilung (Anzahl der Freiheitsgrade)
- σ — Parameter der Nichtzentralität



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityNoncentralF	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen F-Verteilung
MathCumulativeDistributionNoncentralF	Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen F-Verteilung
MathQuantileNoncentralF	Berechnet den Wert der Umkehrfunktion der nichtzentralen F-Verteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathRandomNoncentralF	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der nichtzentralen Verteilung von Fisher
MathMomentsNoncentralF	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen F-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\NoncentralF.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double nu_1=20;     // die erste Zahl der Freiheitsgrade
input double nu_2=20;     // die zweite Zahl der Freiheitsgrade
input double sig=10;      // Parameter der Nichtzentralität
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der nichtzentralen F-Verteilung erhalten
   MathRandomNoncentralF(nu_1,nu_2,sig,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityNoncentralF(x2,nu_1,nu_2,sig,false,y2);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Noncentral F-distribution nu1=%G nu2=%G sigma=
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
```

© 2000-2025, MetaQuotes Ltd.

```
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityNoncentralF

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Die Funktion berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu df() in R.

```
bool  MathProbabilityDensityNoncentralF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityNoncentralF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

© 2000-2025, MetaQuotes Ltd.

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionNoncentralF

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   const double  sigma,          // Parameter der Nichtzentralität
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralF(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Frei
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fre
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Wahrscheinlichkeitsverteilung von Fisher mit debn Parametern nu1, nu2 und sigma für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pf() in R.

```
bool  MathCumulativeDistributionNoncentralF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet die Wahrscheinlichkeitsverteilung der nichtzentralen Verteilung von Fisher mit den Parametern nu1, nu2 und sigma für das Array der Zuifallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionNoncentralF(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

© 2000-2025, MetaQuotes Ltd.

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileF

Berechnet den Wert der Umkehrfunktion der nichtzentralen F-Verteilung mit den Parametern nu1 und nu2 für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileF(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fr
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der F
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen F-Verteilung mit den Parametern nu1 und nu2 für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileF(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fr
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der F
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen F-Verteilung mit den Parametern nu1 und nu2 für das Array mit Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false. Analog zu qf() in R.

```
double  MathQuantileF(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fr
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der F
   const double  sigma,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen F-Verteilung mit den Parametern nu1 und nu2 für das Array mit Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileF(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fr
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der F
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

© 2000-2025, MetaQuotes Ltd.

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomNoncentralF

Erzeugt eine Pseudozufallszahl auf Basis der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomNoncentralF(
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma. Im Fehlerfall retourniert false. Analog rf() in R.

```
bool  MathRandomNoncentralF(
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsNoncentralF

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen F-Verteilung von Fisher mit den Parametern nu1, nu2 und sigma.

```
double  MathMomentsNoncentralF(
   const double  nu1,            // Der erste Parameter der Verteilung (Anzahl der Fre
   const double  nu2,            // Der zweite Parameter der Verteilung (Anzahl der Fr
   const double  sigma,          // Parameter der Nichtzentralität
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

nu1

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

nu2

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

sigma

[in] Parameter der Nichtzentralität.

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

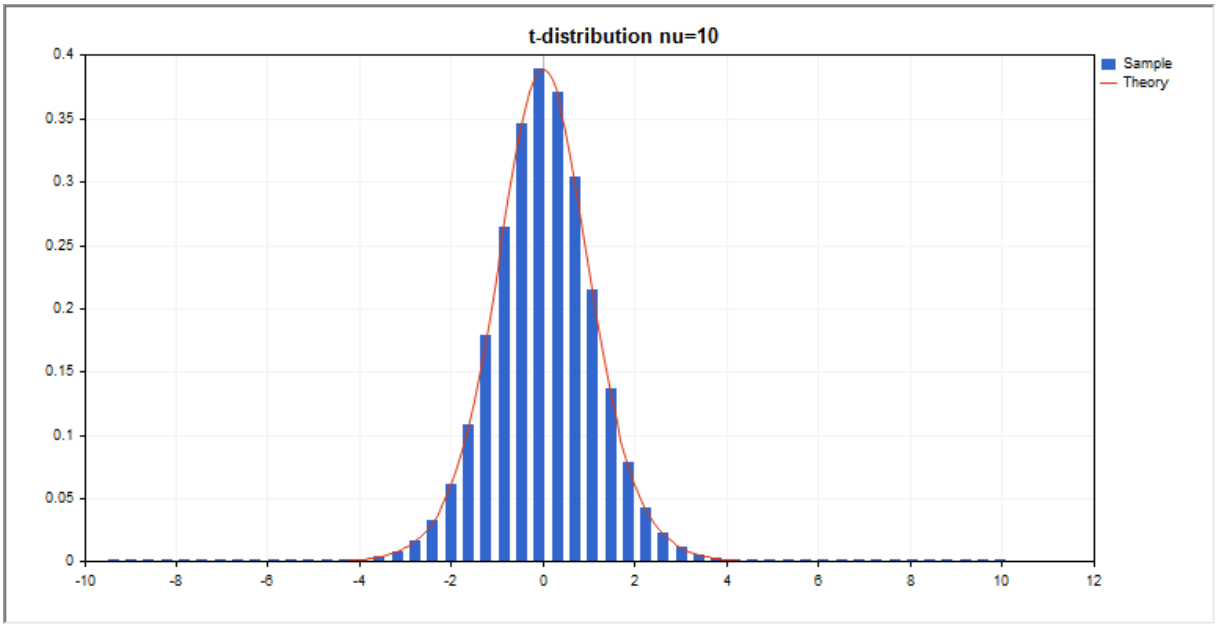
T-Verteilung

In diesem Abschnitt sind Funktionen für die t-Verteilung von Student beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der t-Verteilung von Student erzeugt werden. Die t-Verteilung wird mit der folgenden Formel berechnet:

$$f_T(x|\nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\sqrt{\pi\nu}} \frac{1}{\left(1+\frac{x^2}{\nu}\right)^{\frac{\nu+1}{2}}}$$

wobei:

- x — der Wert der Zufallsvariablen
- ν — Parameter der Verteilung (Anzahl der Freiheitsgrade)



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityT	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der t-Verteilung
MathCumulativeDistributionT	Berechnet den Wert der t-Verteilung
MathQuantileT	Berechnet den Wert der Umkehrfunktion der t-Verteilung für eine angegebene Wahrscheinlichkeit
MathRandomT	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der t-Verteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsT	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Studentschen t-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\T.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double nu_par=10;     // Zahl der Freiheistgrade
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der T-Verteilung erhalten
   MathRandomT(nu_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityT(x2,nu_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("t-distribution nu=%G",nu_par));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityT

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Studentschen t-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Studentschen t-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Studentsche t-Verteilung mit dem Parameter nu für den Array der Zufallsvariable x[]. Im Fehlerfall retourniert false. Analog zu dt() in R.

```
bool  MathProbabilityDensityT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Studentsche t-Verteilung mit dem Parameter nu für den Array der Zufallsvariable x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

log_mode

© 2000-2025, MetaQuotes Ltd.

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionT

Berechnet den Wert der Studentschen t-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Studentschen t-Verteilung mit dem Parameter nu für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Studentschen t-Verteilung mit dem Parameter nu für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pt() in R.

```
bool  MathCumulativeDistributionT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet die Studentsche Verteilung mit dem Parameter nu für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileT

Berechnet den Wert der Umkehrfunktion der Studentschen t-Verteilung mit dem Parameter nu für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileT(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Studentschen t-Verteilung mit dem Parameter nu für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileT(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Studentschen t-Verteilung mit dem Parameter nu für das Array der Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false. Analog zu qt() in R.

```
double  MathQuantileT(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Studentschen t-Verteilung mit dem Parameter nu für das Array der Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileT(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomT

Erzeugt eine Pseudozufallszahl auf Basis der Studentschen t-Verteilung mit dem Parameter nu. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomT(
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgrad
   int&          error_code       // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Studentschen t-Verteilung mit dem Parameter nu. Im Fehlerfall retourniert false. Analog zu rt() in R.

```
bool  MathRandomT(
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsT

Die Funktion berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Studentschen t-Verteilung mit dem Parameter nu.

```
double  MathMomentsT(
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgrad
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

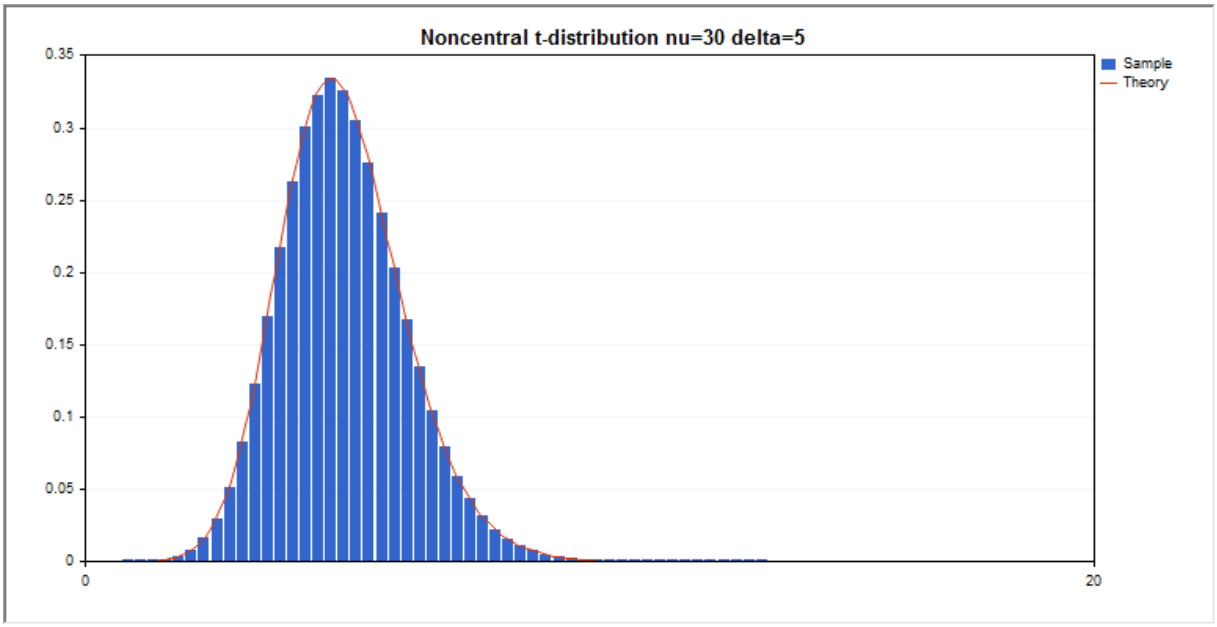
Nichtzentrale t-Verteilung

In diesem Abschnitt finden Sie Funktionen der nichtzentralen Studentschen t-Verteilung. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der nichtzentralen t-Verteilung erzeugt werden. Die nichtzentrale t-Verteilung wird mit der folgenden Formel berechnet:

$$f_{NoncentralT}(x \mid \nu, \delta) = \frac{\nu^{\frac{\nu}{2}} e^{-\frac{\sigma^2}{2}}}{\Gamma\left(\frac{\nu}{2}\right)\sqrt{\pi}\left(\nu+x^2\right)^{\frac{\nu+1}{2}}} \sum_{r=0}^{\infty} \frac{(x\delta)^r}{r!}\left(\frac{2}{\nu+x^2}\right)^{\frac{r}{2}} \Gamma\left(\frac{\nu+r+1}{2}\right)$$

wobei:

- x — der Wert der Zufallsvariablen
- ν — Parameter der Verteilung (Anzahl der Freiheitsgrade)
- σ — Parameter der Nichtzentralität



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityNoncentralT	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen t-Verteilung
MathCumulativeDistributionNoncentralT	Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen t-Verteilung
MathQuantileNoncentralT	Berechnet den Wert der Umkehrfunktion der nichtzentralen t-Verteilung für die angegebene Wahrscheinlichkeit
MathRandomNoncentralT	Erzeugt eine Pseudozufallszahl/ein Array von Pseudozufallszahlen auf Basis der nichtzentralen Studentschen t-Verteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsNoncentralT	Die Funktion berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen Studentschen t-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\NoncentralT.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double nu_par=30;        // Zahl der Freiheitsgrade
input double delta_par=5;     // Parameter der Nichtzentralität
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der nichtzentralen T-Verteilung erhalten
   MathRandomNoncentralT(nu_par,delta_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityNoncentralT(x2,nu_par,delta_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Noncentral t-distribution nu=%G delta=%G",nu_p
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityNoncentralT

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Studentschen t-Verteilung mit den Parametern nu und delta für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  delta,          // Parameter der Nichtzentralität
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Studentschen t-Verteilung mit den Parametern nu und delta für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNoncentralT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  delta,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Studentschen t-Verteilung mit den Parametern nu und delta für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dt() in R.

```
bool  MathProbabilityDensityNoncentralT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrac
   const double  delta,          // Parameter der Nichtzentralität
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der nichtzentralen Studentschen t-Verteilung mit den Parametern nu und delta für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityNoncentralT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrac
   const double  delta,          // Parameter der Nichtzentralität
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

delta

[in] Parameter der Nichtzentralität.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionNoncentralT

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen t-Verteilung mit den Parametern nu und delta für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  delta,          // Parameter der Nichtzentralität
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen t-Verteilung mit den Parametern nu und delta für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNoncentralT(
   const double  x,              // Wert der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrade
   const double  delta,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen t-Verteilung mit den Parametern nu und delta für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pt() in R.

```
bool  MathCumulativeDistributionNoncentralT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der nichtzentralen t-Verteilung mit den Parametern nu und delta für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionNoncentralT(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

x[]

[in] Array mit den Werten der Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

delta

[in] Parameter der Nichtzentralität.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileNoncentralT

Berechnet den Wert der Umkehrfunktion der nichtzentralen t-Verteilung mit den Parametern nu und delta für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNoncentralT(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag, lower_tail=false, wird für eine 1.0-Wahrsche
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen t-Verteilung mit den Parametern nu und delta für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNoncentralT(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen t-Verteilung mit den Parametern nu und delta für das Array mit Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false. Analog zu qt() in R.

```
double  MathQuantileNoncentralT(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
   const bool    tail,           // Flag, lower_tail=false, wird für eine 1.0-Wahrsche
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der nichtzentralen t-Verteilung mit den Parametern nu und delta für das Array mit Wahrscheinlichkeitswerten *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileNoncentralT(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

delta

[in] Parameter der Nichtzentralität.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomNoncentralT

Erzeugt eine Pseudozufallszahl auf Basis der nichtzentralen Studentschen t-Verteilung mit den Parametern nu und delta. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomNoncentralT(
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,           // Parameter der Nichtzentralität
   int&          error_code       // Variable für den Fehlercode
   );
```

Erzeugt eine Pseudozufallszahl auf Basis der nichtzentralen t-Verteilung mit den Parametern nu und delta. Im Fehlerfall retourniert false. Analog zu rt() in R.

```
bool  MathRandomNoncentralT(
   const double  nu,              // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,           // Parameter der Nichtzentralität
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

delta

[in] Parameter der Nichtzentralität.

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsNoncentralT

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der nichtzentralen Studentschen t-Verteilung mit dem Parameter nu und delta.

```
double  MathMomentsNoncentralT(
   const double  nu,             // Parameter der Verteilung (Anzahl der Freiheitsgrad
   const double  delta,          // Parameter der Nichtzentralität
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

nu

[in] Parameter der Verteilung (Anzahl der Freiheitsgrade).

delta

[in] Parameter der Nichtzentralität.

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

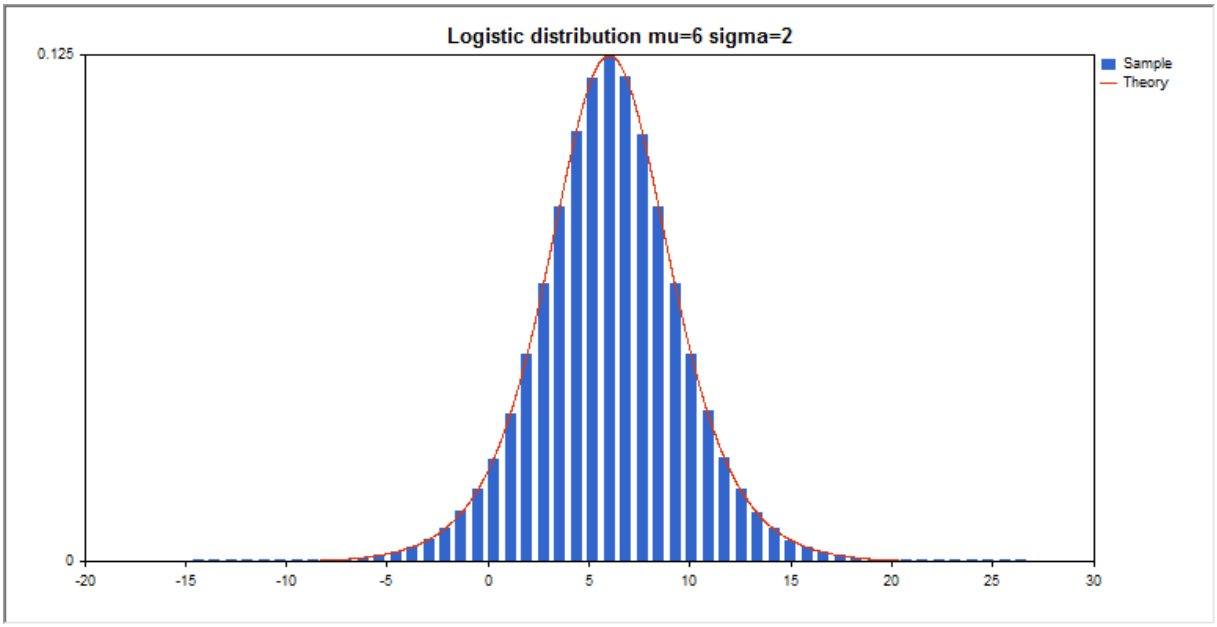
Logistische Verteilung

In diesem Abschnitt sind Funktionen für die logistische Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der logistischen Verteilung erzeugt werden. Die logistische Verteilung wird mit der folgenden Formel berechnet:

$$f_{Logistic}(x|\mu,\sigma) = \frac{e^{\frac{x-\mu}{\sigma}}}{\sigma\left(1+e^{\frac{x-\mu}{\sigma}}\right)^2}$$

wobei:

- x — der Wert der Zufallsvariablen
- μ — mean Parameter der Verteilung
- σ — scale Parameter der Verteilung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityLogistic	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der logistischen Verteilung
MathCumulativeDistributionLogistic	Berechnet den Wert der logistischen Verteilung
MathQuantileLogistic	Berechnet den Wert der Umkehrfunktion der logistischen Verteilung für die angegebene Wahrscheinlichkeit
MathRandomLogistic	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der logistischen Verteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsLogistic	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der logistischen Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Logistic.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double mu_par=6;         // mean Parameter der Verteilung
input double sigma_par=2;      // scale Parameter der Verteilung
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der logistischen Verteilung erhalten
   MathRandomLogistic(mu_par,sigma_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityLogistic(x2,mu_par,sigma_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Logistic distribution mu=%G sigma=%G",mu_par,s
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der Y-Achse deaktivieren
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Max(theor_max);
   graphic.YAxis().Min(0);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
```

© 2000-2025, MetaQuotes Ltd.

```
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityLogistic

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der logistischen Verteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityLogistic(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der logistischen Verteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityLogistic(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer logistischen Verteilung mit den Parametern mu und sigma für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dlogis() in R.

```
bool  MathProbabilityDensityLogistic(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet die Werte der Wahrscheinlichkeitsdichtefunktion einer logistischen Verteilung mit den Parametern mu und sigma für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityLogistic(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] mean Parameter der Verteilung.

sigma

[in] scale Parameter der Verteilung.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionLogistic

Berechnet den Wert der logistischen Verteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionLogistic(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der logistischen Verteilung mit den Parametern mu und sigma für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionLogistic(
   const double  x,              // Wert der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der logistischen Verteilung mit den Parametern mu und sigma für den Array mit Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionLogistic(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der logistischen Verteilung mit den Parametern mu und sigma für den Array mit Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog plogis() in R.

```
bool  MathCumulativeDistributionLogistic(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

mu

[in] mean Parameter der Verteilung.

sigma

[in] scale Parameter der Verteilung.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileLogistic

Berechnet den Wert der Umkehrfunktion der logistischen Verteilung mit den Parametern mu und sigma für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileLogistic(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der logistischen Verteilung mit den Parametern mu und sigma für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileLogistic(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der logistischen Verteilung mit den Parametern mu und sigma für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false. Analog zu qlogis() in R.

```
double  MathQuantileLogistic(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der logistischen Verteilung mit den Parametern mu und sigma für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false.

```
bool  MathQuantileLogistic(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  mu,             // mean Parameter der Verteilung
   const double  sigma,          // scale Parameter der Verteilung
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

mu

[in] mean Parameter der Verteilung.

sigma

[in] scale Parameter der Verteilung.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomLogistic

Erzeugt eine Pseudozufallszahl auf Basis der logistischen Verteilung mit den Parametern mu und sigma. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomLogistic(
   const double  mu,              // mean Parameter der Verteilung
   const double  sigma,           // scale Parameter der Verteilung
   int&          error_code       // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der logistischen Verteilung mit den Parametern mu und sigma. Im Fehlerfall retourniert false. Analog zu rlogis() in R.

```
bool  MathRandomLogistic(
   const double  mu,              // mean Parameter der Verteilung
   const double  sigma,           // scale Parameter der Verteilung
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

mu

[in] mean Parameter der Verteilung.

sigma

[in] scale Parameter der Verteilung.

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsLogistic

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der logistischen Verteilung mit dem Parameter mu und sigma.

```
double  MathMomentsLogistic(
   const double  mu,              // mean Parameter der Verteilung
   const double  sigma,           // scale Parameter der Verteilung
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

mu

[in] mean Parameter der Verteilung.

sigma

[in] scale Parameter der Verteilung.

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

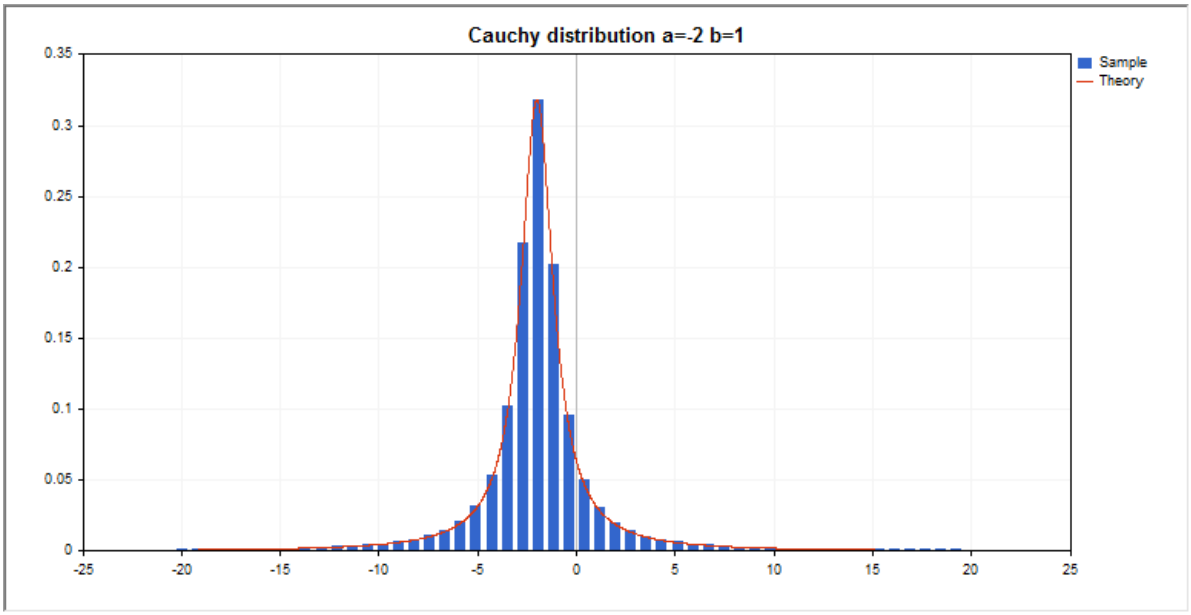
Cauchy-Verteilung

In diesem Abschnitt sind Funktionen der Cauchy-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Cauchy-Verteilung erzeugt werden. Die Cauchy-Verteilung wird mit der folgenden Formal beschrieben:

$$f_{Cauchy}(x|a,b) = \frac{1}{\pi}\frac{b}{\left(b^2+(x-a)^2\right)}$$

wobei:

- x — der Wert der Zufallsvariablen
- a — mean Parameter der Verteilung
- b — scale Parameter der Verteilung



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityCauchy	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Cauchy-Verteilung
MathCumulativeDistributionCauchy	Berechnet den Wert der Wahrscheinlichkeitsverteilung der Cauchy-Verteilung
MathQuantileCauchy	Berechnet den Wert der Umkehrfunktion der Cauchy-Verteilung für eine angegebene Wahrscheinlichkeit
MathRandomCauchy	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Cauchy-Verteilung
MathMomentsCauchy	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Cauchy-Verteilung

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Cauchy.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double a_par=-2;      // Parameter der Verteilung mean
input double b_par=1;       // Parameter der Verteilung scale
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Cauchy-Verteilung erhalten
   MathRandomCauchy(a_par,b_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityCauchy(x2,a_par,b_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
```

© 2000-2025, MetaQuotes Ltd.

```
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Cauchy distribution a=%G b=%G",a_par,b_par));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1)
      return(false);
   int size=ArraySize(data);
   if(size<cells*10)
      return(false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   Print("min=",minv," max=",maxv);
   minv=-20;
   maxv=20;
   double range=maxv-minv;
   double width=range/cells;
   if(width==0)
      return(false);
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+i*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=(int)MathRound((data[i]-minv)/width);
      if(ind>=0 && ind<cells)
         frequency[ind]++;
     }
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityCauchy

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Cauchy-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityCauchy(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Cauchy-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityCauchy(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Cauchy-Verteilung mit den Parametern a und b für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dcauchy() in R.

```
bool  MathProbabilityDensityCauchy(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Cauchy-Verteilung mit den Parametern a und b für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityCauchy(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] mean Parameter der Verteilung.

b

[in] scale Parameter der Verteilung.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionCauchy

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Cauchy-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionCauchy(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Cauchy-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionCauchy(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Cauchy-Verteilung mit den Parametern a und b für die Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionCauchy(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Cauchy-Verteilung mit den Parametern a und b für ein Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog plogis() in R.

```
bool  MathCumulativeDistributionCauchy(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] mean Parameter der Verteilung.

b

[in] scale Parameter der Verteilung.

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileCauchy

Berechnet den Wert der Umkehrfunktion der Cauchy-Verteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileCauchy(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Cauchy-Verteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileCauchy(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Cauchy-Verteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false. Analog zu qcauschy() in R.

```
double  MathQuantileCauchy(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Cauchy-Verteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileCauchy(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

a

[in] mean Parameter der Verteilung.

b

[in] scale Parameter der Verteilung.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomCauchy

Erzeugt eine Pseudozufallszahl auf Basis der Cauchy-Verteilung mit den Parametern a und b. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomCauchy(
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Cauchy-Verteilung mit den Parametern a und b. Im Fehlerfall retourniert false. Analog zu rcauchy() in R.

```
bool  MathRandomCauchy(
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

a

[in] mean Parameter der Verteilung.

b

[in] scale Parameter der Verteilung.

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsCauchy

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Cauchy-Verteilung.

```
double  MathMomentsCauchy(
   const double  a,              // mean Parameter der Verteilung
   const double  b,              // scale Parameter der Verteilung
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

a

[in] mean Parameter der Verteilung.

b

[in] scale Parameter der Verteilung.

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

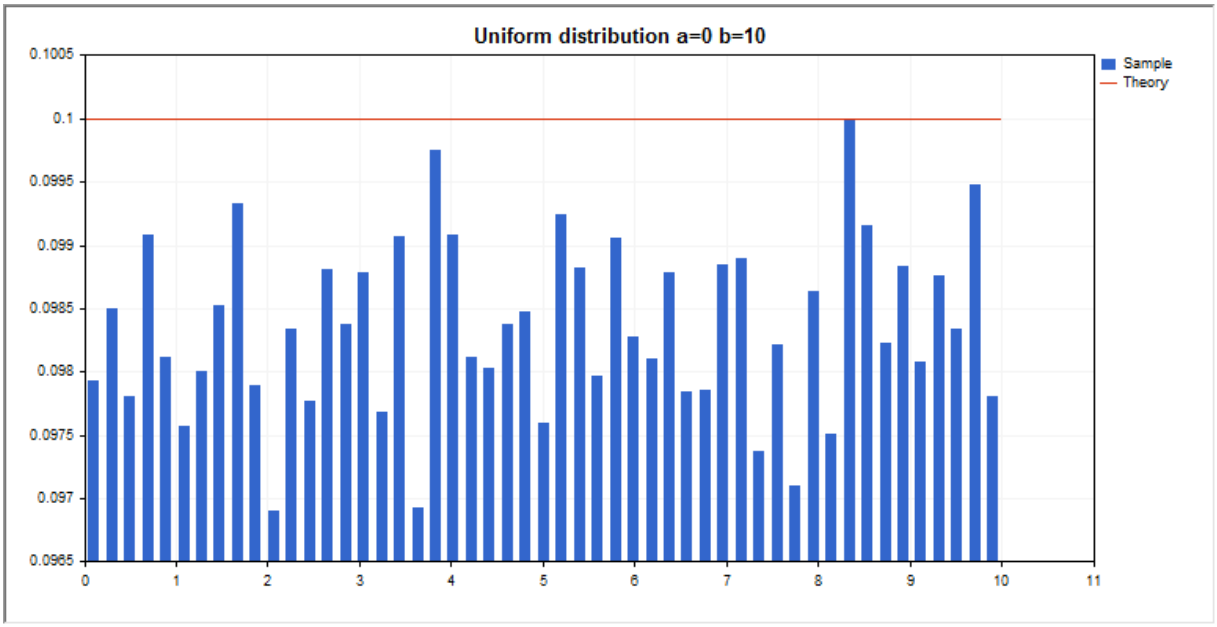
Gleichverteilung

In diesem Abschnitt sind Funktionen für die Gleichverteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Gleichverteilung erzeugt werden. Die Gleichverteilung wird mit der folgenden Formel berechnet:

$$f_{Uniform}(x|\,a,b) = \frac{1}{b-a}$$

wobei:

- x — der Wert der Zufallsvariablen
- a — Parameter der Verteilung (untere Grenze)
- b — Parameter der Verteilung (obere Grenze)



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityUniform	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gleichverteilung
MathCumulativeDistributionUniform	Berechnet den Wert der Funktion der Gleichverteilung
MathQuantileUniform	Berechnet den Wert der Umkehrfunktion der Gleichverteilung für eine angegebene Wahrscheinlichkeit
MathRandomUniform	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Gleichverteilung
MathMomentsUniform	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Gleichverteilung.

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Uniform.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double a_par=0;      // a Parameter der Verteilung (untere Grenze)
input double b_par=10;     // b Parameter der Verteilung (obere Grenze)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=51;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Gleichverteilung erhalten
   MathRandomUniform(a_par,b_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityUniform(x2,a_par,b_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
```

© 2000-2025, MetaQuotes Ltd.

```
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Uniform distribution a=%G b=%G",a_par,b_par));
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityUniform

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gleichverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityUniform(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // a Parameter der Verteilung (untere Grenze)
   const double  b,              // b Parameter der Verteilung (obere Grenze)
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gleichverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityUniform(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // a Parameter der Verteilung (untere Grenze)
   const double  b,              // b Parameter der Verteilung (obere Grenze)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gleichverteilung mit den Parametern a und b für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dunif() in R.

```
bool  MathProbabilityDensityUniform(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Gleichverteilung mit den Parametern a und b für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityUniform(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

a

[in] a-Parameter der Verteilung (untere Grenze).

b

[in] b-Parameter der Verteilung (obere Grenze).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionUniform

Berechnet den Wert der Gleichverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionUniform(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // a Parameter der Verteilung (untere Grenze)
   const double  b,              // b Parameter der Verteilung (obere Grenze)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Gleichverteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionUniform(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // a Parameter der Verteilung (untere Grenze)
   const double  b,              // b Parameter der Verteilung (obere Grenze)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Gleichverteilung mit den Parametern a und b für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionUniform(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Gleichverteilung mit den Parametern a und b für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu punif() in R.

```
bool  MathCumulativeDistributionUniform(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] a-Parameter der Verteilung (untere Grenze).

b

[in] b-Parameter der Verteilung (obere Grenze).

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileUniform

Berechnet den Wert der Umkehrfunktion der Gleichverteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileUniform(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Gleichverteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileUniform(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Gleichverteilung mit den Parametern a und b für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false. Analog zu qcauschy() in R.

```
double  MathQuantileUniform(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Gleichverteilung mit den Parametern a und b für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileUniform(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

a

[in] a-Parameter der Verteilung (untere Grenze).

b

[in] b-Parameter der Verteilung (obere Grenze).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomUniform

Erzeugt eine Pseudozufallszahl auf Basis der Gleichverteilung mit den Parametern a und b. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomUniform(
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Gleichverteilung mit den Parametern a und b. Im Fehlerfall retourniert false. Analog zu runif() in R.

```
bool  MathRandomUniform(
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

a

[in] a-Parameter der Verteilung (untere Grenze).

b

[in] b-Parameter der Verteilung (obere Grenze).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsUniform

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Gleichverteilung mit den Parametern a und b.

```
double  MathMomentsUniform(
   const double  a,              // a-Parameter der Verteilung (untere Grenze)
   const double  b,              // b-Parameter der Verteilung (obere Grenze)
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

a

[in] a-Parameter der Verteilung (untere Grenze).

b

[in] b-Parameter der Verteilung (obere Grenze).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

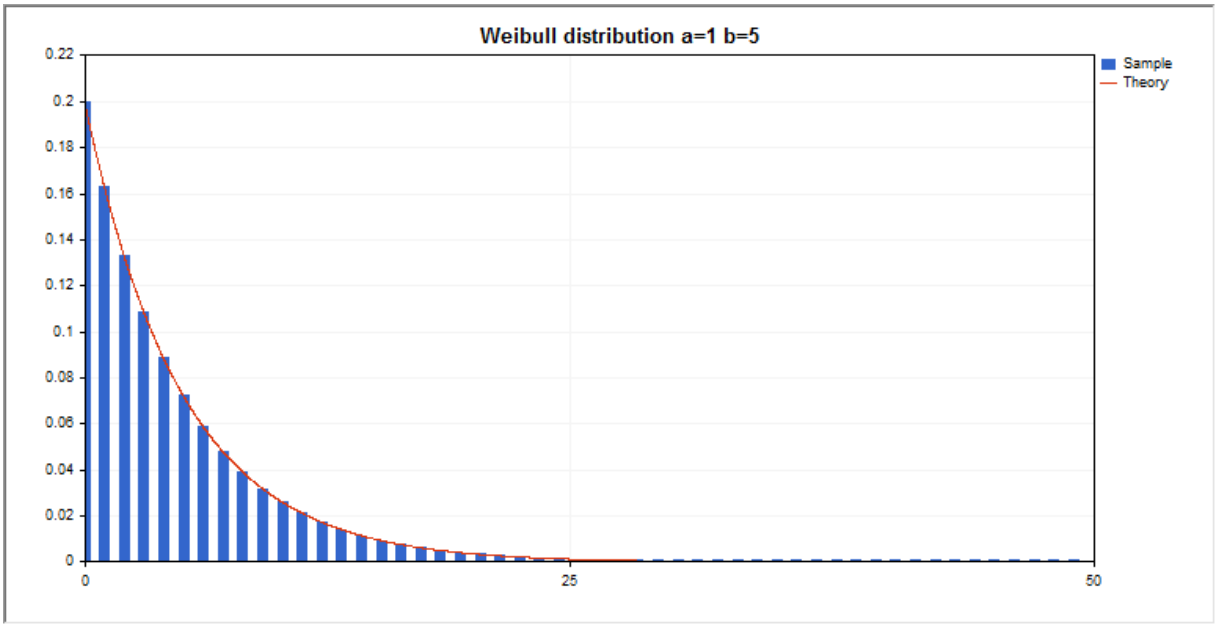
Weibull-Verteilung

In diesem Abschnitt sind Funktionen der Weibull-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet werden und Pseudozufallszahlen auf Basis der Weibull-Verteilung erzeugt werden. Die Weibull-Verteilung wird mit der folgenden Funktion beschrieben:

$$f_{Weibull}(x|a,b) = \frac{a}{b}\left(\frac{x}{b}\right)^{a-1} e^{-\left(\frac{x}{b}\right)^{a}}$$

wobei:

- x — der Wert der Zufallsvariablen
- a — Parameter der Verteilung (shape)
- b — Parameter der Verteilung (scale)



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityWeibull	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Weibull-Verteilung
MathCumulativeDistributionWeibull	Berechnet den Wert der Weibull-Verteilung
MathQuantileWeibull	Berechnet den Wert der Umkehrfunktion der Weibull-Verteilung für die angegebene Wahrscheinlichkeit
MathRandomWeibull	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Weibull-Verteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsWeibull	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Weibull-Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Weibull.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double a_par=1;      // Parameter der Verteilung (shape)
input double b_par=5;      // Parameter der Verteilung (scale)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=51;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Weibull-Verteilung erhalten
   MathRandomWeibull(a_par,b_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   step=MathMin(step,(max-min)/ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(min,max,step,x2);
   MathProbabilityDensityWeibull(x2,a_par,b_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Weibull distribution a=%G b=%G",a_par,b_par));
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der X-Achse deaktivieren
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Max(max);
   graphic.XAxis().Min(min);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+i*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
```

© 2000-2025, MetaQuotes Ltd.

```
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityWeibull

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Weibull-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityWeibull(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Parameter der Verteilung (scale)
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Weibull-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityWeibull(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Parameter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Weibull-Verteilung mit den Parametern a und b für einen Array der Zufallsvariablen x[. Im Fehlerfall retourniert false. Analog zu dweibull() in R.

```
bool  MathProbabilityDensityWeibull(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Weibull-Verteilung mit den Parametern a und b für einen Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityWeibull(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Parameter der Verteilung (scale).

b

[in] Parameter der Verteilung (shape).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionWeibull

Berechnet den Wert der Weibull-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionWeibull(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Parameter der Verteilung (scale)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Weibull-Verteilung mit den Parametern a und b für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionWeibull(
   const double  x,              // Wert der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Parameter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Weibull-Verteilung mit den Parametern a und b für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pweibull() in R.

```
bool  MathCumulativeDistributionWeibull(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Weibull-Verteilung mit den Parametern a und b für das Array der Zufallsvariablen x[] Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionWeibull(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

a

[in] Parameter der Verteilung (scale).

b

[in] Parameter der Verteilung (shape).

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileWeibull

Berechnet den Wert der Umkehrfunktion der Weibull-Verteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileWeibull(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Weibull-Verteilung mit den Parametern a und b für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileWeibull(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Weibull-Verteilung mit den Parametern a und b für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false. Analog zuqweibull() in R.

```
double  MathQuantileWeibull(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Weibull-Verteilung mit den Parametern a und b für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileWeibull(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

a

[in] Parameter der Verteilung (scale).

b

[in] Parameter der Verteilung (shape).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomWeibull

Die Funktion erzeugt eine Pseudozufallszahl auf Basis der Weibull-Verteilung mit den Parametern a und b. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomWeibull(
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Weibull-Verteilung mit den Parametern a und b. Im Fehlerfall retourniert false. Analog zu rweibull() in R.

```
bool  MathRandomWeibull(
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

a

[in] Parameter der Verteilung (scale).

b

[in] Parameter der Verteilung (shape).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsWeibull

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Weibull-Verteilung.

```
double  MathMomentsWeibull(
   const double  a,              // Parameter der Verteilung (shape)
   const double  b,              // Paramerter der Verteilung (scale)
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

a

[in] Parameter der Verteilung (scale).

b

[in] Parameter der Verteilung (shape).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

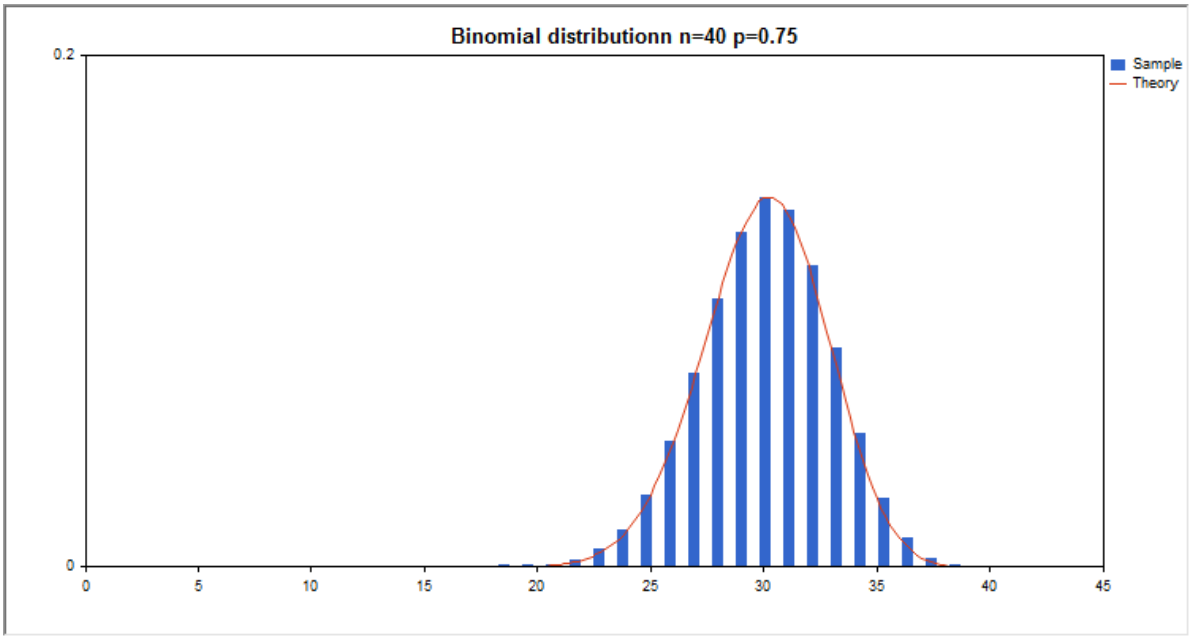
Binomial-Verteilung

Hier sind Funktionen für die Binomialverteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Binomialverteilung erzeugt werden. Die Binomialverteilung wird mit der folgenden Formel berechnet:

$$f_{Binomial}(x \mid n, p) = \binom{n}{x} p^x (1-p)^{n-x}$$

wobei:

- x — der Wert der Zufallsvariablen
- n — Anzahl der Tests
- p — Erfolgswahrscheinlichkeit für jeden Test



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityBinomial	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Binomialverteilung
MathCumulativeDistributionBinomial	Berechnet den Wert der Wahrscheinlichkeitsverteilung der Binomialverteilung
MathQuantileBinomial	Berechnet den Wert der Umkehrfunktion der Binominalverteilung für die angegebene Wahrscheinlichkeit
MathRandomBinomial	Erzeugt Pseudozufallszahlen auf Basis der Binomialverteilung
MathMomentsBinomial	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Binomialverteilung

© 2000-2025, MetaQuotes Ltd.

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Binomial.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double n_par=40;         // Anzahl der Tests
input double p_par=0.75;       // Erfolgswahrscheinlichkeit für jeden Test
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=20;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Binomialverteilung erhalten
   MathRandomBinomial(n_par,p_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(0,n_par,1,x2);
   MathProbabilityDensityBinomial(x2,n_par,p_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Binomial distributionn n=%G p=%G",n_par,p_par)
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory").LinesSmooth(true);
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityBinomial

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Binomialverteilung mit den Parametern n und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityBinomial(
   const double  x,              // Wert der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichkeit
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Binomialverteilung mit den Parametern n und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityBinomial(
   const double  x,              // Wert der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichkeit
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion (probability mass function) der Binomialverteilung mit den Parametern n und p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dbinom() in R.

```
bool  MathProbabilityDensityBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion (probability mass function) der Binomialverteilung mit den Parametern n und p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichke
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

© 2000-2025, MetaQuotes Ltd.

[in] Array mit den Werten der Zufallsvariablen.

n

[in] Parameter der Verteilung (Anzahl der Tests).

p

[in] Parameter der Verteilung (Erfolgswahrscheinlichkeit für jeden Test).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionBinomial

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Binomialverteilung mit den Parametern n und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionBinomial(
   const double  x,              // Wert der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichkeit
   const bool    tail,           // lag der Berechnung, wenn true, dann wird die Wahrsc
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Binomialverteilung mit den Parametern n und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionBinomial(
   const double  x,              // Wert der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichkeit
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der negativen Binomialverteilung mit den Parametern r und p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pweibull() in R.

```
bool  MathCumulativeDistributionBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichkei
   const bool    tail,           // Flag der Berechnung, wenn true, dann wird die Wahr
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der Binomialverteilung mit den Parametern n und p für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichkei
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

x[]

[in] Array mit den Werten der Zufallsvariablen.

n

[in] Parameter der Verteilung (Anzahl der Tests).

p

[in] Parameter der Verteilung (Erfolgswahrscheinlichkeit für jeden Test).

tail

[in] Flag der Berechnung, Wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, Wenn log_mode=true, dann wird der natürliche Logarithmus der Wahrscheinlichkeit zurückgegeben.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileBinomial

Berechnet die Umkehrfunktion der Binomialverteilung mit den Parametern n und p für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileBinomial(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Umkehrfunktion der Binomialverteilung mit den Parametern n und p für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileBinomial(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Umkehrfunktion der Binomialverteilung mit den Parametern n und p für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false. Analog zu qbinom() in R.

```
double  MathQuantileBinomial(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet die Umkehrfunktion der Binomialverteilung mit den Parametern n und p für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte. Im Fehlerfall retourniert false.

```
bool  MathQuantileBinomial(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

n

[in] Parameter der Verteilung (Anzahl der Tests).

p

[in] Parameter der Verteilung (Erfolgswahrscheinlichkeit für jeden Test).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomBinomial

Erzeugt eine Pseudozufallszahl auf Basis der Binomialverteilung mit den Parametern n und p. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomBinomial(
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Binomialverteilung mit den Parametern n und p. Im Fehlerfall retourniert false. Analog zu rweibull() in R.

```
bool  MathRandomBinomial(
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

n

[in] Parameter der Verteilung (Anzahl der Tests).

p

[in] Parameter der Verteilung (Erfolgswahrscheinlichkeit für jeden Test).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsBinomial

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Binomialverteilung mit den Parametern n und p.

```
double  MathMomentsBinomial(
   const double  n,              // Parameter der Verteilung (Anzahl der Tests)
   const double  p,              // Parameter der Verteilung (Erfolgswahrscheinlichke
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

n

[in] Parameter der Verteilung (Anzahl der Tests).

p

[in] Parameter der Verteilung (Erfolgswahrscheinlichkeit für jeden Test).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

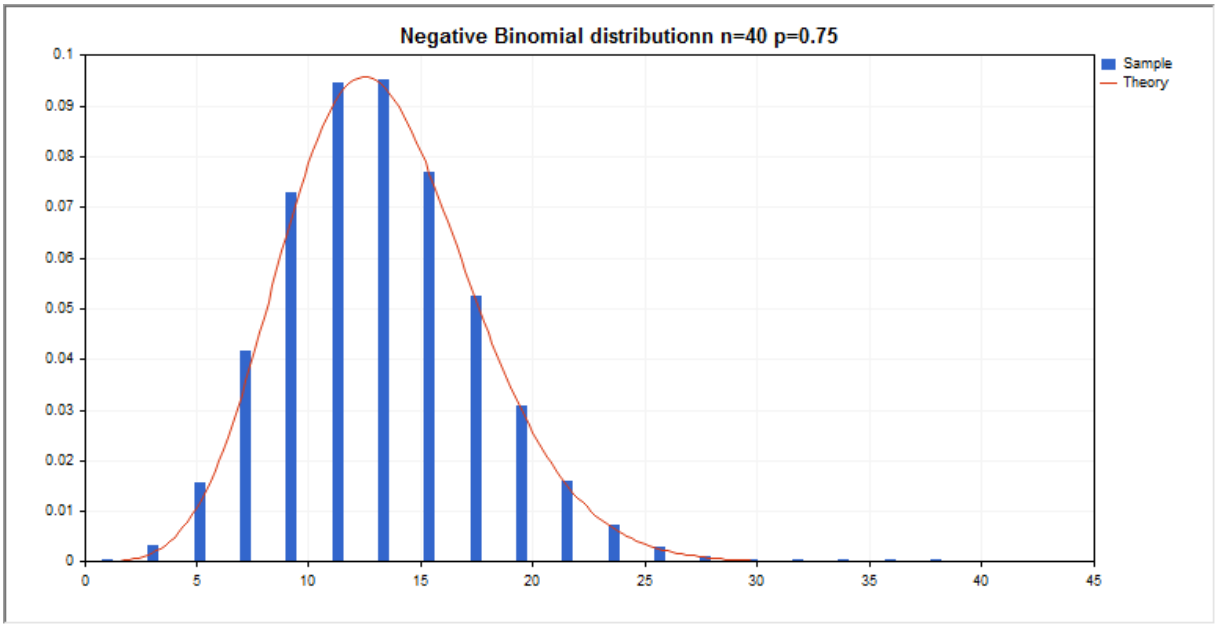
Negative Binomial-Verteilung

In diesem Abschnitt sind Funktionen für die negative Binomialverteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der negativen Binomialverteilung erzeugt werden. Die negative Binomialverteilung wird mit der folgenden Formel berechnet:

$$f_{NegativeBinomial}(x \mid r, p) = \frac{\Gamma(r+x)}{\Gamma(r)\Gamma(x+1)} p^r (1-p)^x$$

wobei:

- x — der Wert der Zufallsvariablen
- r — Anzahl erfolgreicher Tests
- p — Erfolgswahrscheinlichkeit



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityNegativeBinomial	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der negativen Binomialverteilung
MathCumulativeDistributionNegativeBinomial	Berechnet den Wert der Wahrscheinlichkeitsverteilung der negativen Binomialverteilung
MathQuantileNegativeBinomial	Berechnet den Wert der Umkehrfunktion der negativen Binomialverteilung für eine angegebene Wahrscheinlichkeit
MathRandomNegativeBinomial	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der negativen Binomialverteilung

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathMomentsNegativeBinomial	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der negativen Binomialverteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\NegativeBinomial.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double n_par=40;         // Anzahl der Tests
input double p_par=0.75;       // Erfolgswahrscheinlichkeit für jeden Test
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;        // Anzahl der Werte in der Stichprobe
   int ncells=19;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der negativen Binomialverteilung erhalten
   MathRandomNegativeBinomial(n_par,p_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(0,n_par,1,x2);
   MathProbabilityDensityNegativeBinomial(x2,n_par,p_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
```

© 2000-2025, MetaQuotes Ltd.

```
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Negative Binomial distributionn n=%G p=%G",n_p
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory").LinesSmooth(true);
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityNegativeBinomial

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion (probability mass function) der negativen Binomialverteilung mit den Parametern r und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNegativeBinomial(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion (probability mass function) der negativen Binomialverteilung mit den Parametern r und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityNegativeBinomial(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der negativen Binomialverteilung mit den Parametern r und p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dnbinom() in R.

```
bool  MathProbabilityDensityNegativeBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der Binomalverteilung mit den Parametern r und p für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityNegativeBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

x[]

[in] Array mit den Werten der Zufallsvariablen.

r

[in] Anzahl erfolgreicher Tests

p

[in] Erfolgswahrscheinlichkeit.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionNegativeBinomial

Berechnet den Wert der Wahrscheinlichkeitsverteilung der negativen Binomialverteilung mit den Parametern r und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNegativeBinomial(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der negativen Binomialverteilung mit den Parametern r und p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionNegativeBinomial(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der negativen Binomialverteilung mit den Parametern r und p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pweibull() in R.

```
bool  MathCumulativeDistributionNegativeBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der negativen Binomialverteilung mit den Parametern r und p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionNegativeBinomial(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

x[]

[in] Array mit den Werten der Zufallsvariablen.

r

[in] Anzahl erfolgreicher Tests.

p

[in] Erfolgswahrscheinlichkeit.

tail

[in] Flag der Berechnung, wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeit berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileNegativeBinomial

Berechnet den Wert der Umkehrfunktion der negativen Binomialverteilung mit den Parametern r und p für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNegativeBinomial(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der negativen Binomialverteilung mit den Parametern r und p für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileNegativeBinomial(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ei
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der negativen Binomialverteilung mit den Parametern r und p für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false. Analog zu qnbinom() in R.

```
double  MathQuantileNegativeBinomial(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der negativen Binomialverteilung mit den Parametern r und p für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileNegativeBinomial(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

© 2000-2025, MetaQuotes Ltd.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

r

[in] Anzahl erfolgreicher Tests.

p

[in] Erfolgswahrscheinlichkeit.

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomNegativeBinomial

Erzeugt eine Pseudozufallszahl auf Basis der negativen Binomialverteilung mit den Parametern r und p. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomNegativeBinomial(
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der negativen Binomialverteilung mit den Parametern r und p. Im Fehlerfall retourniert false. Analog zu rweibull() in R.

```
bool  MathRandomNegativeBinomial(
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

r

[in] Anzahl erfolgreicher Tests.

p

[in] Erfolgswahrscheinlichkeit.

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsNegativeBinomial

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der negativen Binomialverteilung mit den Parametern r und p.

```
double  MathMomentsNegativeBinomial(
   const double  r,              // Anzahl erfolgreicher Tests
   const double  p,              // Erfolgswahrscheinlichkeit
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

r

[in] Anzahl erfolgreicher Tests.

p

[in] Erfolgswahrscheinlichkeit.

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

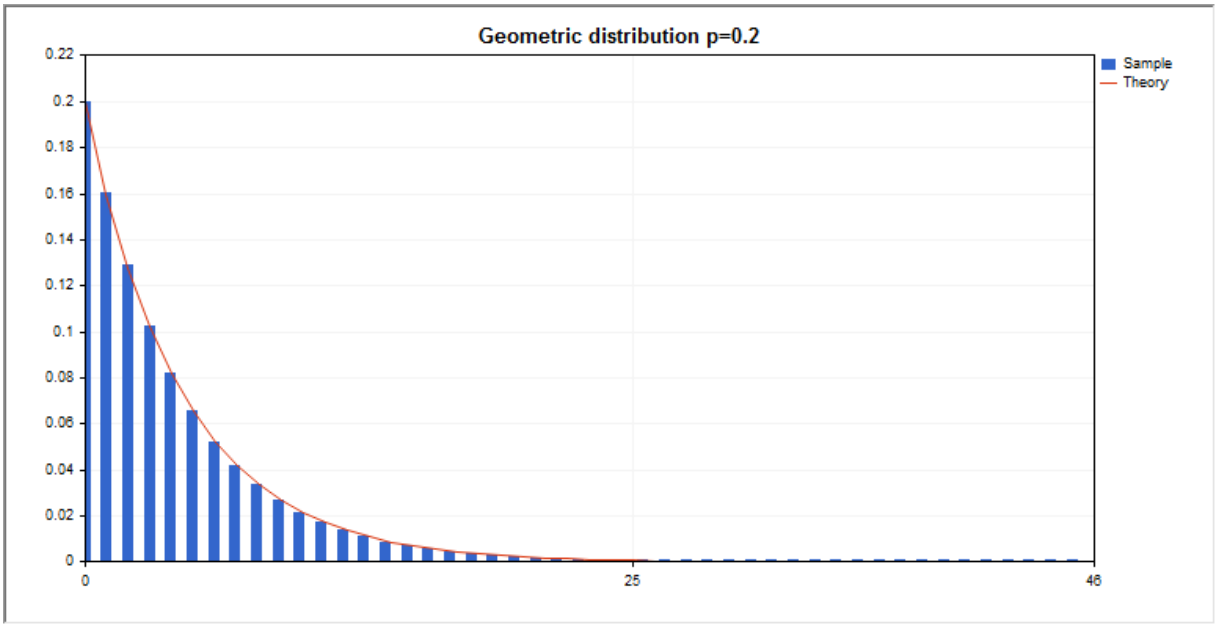
Geometrische Verteilung

In diesem Abschnitt sind Funktionen für die geometrische Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der geometrischen Verteilung erzeugt werden. Die geometrische Verteilung wird mit der folgenden Funktion beschrieben:

$$f_{Geometric}(x|p) = p(1-p)^x$$

wobei:

- x — Wert der Zufallsvariablen (integer)
- p — Wahrscheinlichkeit des Auftretens in einem Test



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityGeometric	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der geometrische Verteilung
MathCumulativeDistributionGeometric	Berechnet den Wert der Wahrscheinlichkeitsverteilung der geometrische Verteilung
MathQuantileGeometric	Berechnet den Wert der Umkehrfunktion der geometrischen Verteilung für eine angegebene Wahrscheinlichkeit
MathRandomGeometric	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der geometrische Verteilung
MathMomentsGeometric	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der geometrische Verteilung

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Geometric.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double p_par=0.2;      // Wahrscheinlichkeit des Auftretens des Ereignisses in e
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=47;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der geometrischen Verteilung erhalten
   MathRandomGeometric(p_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   PrintFormat("max=%G min=%G",max,min);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(0,ncells,1,x2);
   MathProbabilityDensityGeometric(x2,p_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.BackgroundMain(StringFormat("Geometric distribution p=%G",p_par));
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der X-Achse deaktivieren
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Max(max);
   graphic.XAxis().Min(min);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory");
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+i*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityGeometric

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion (probability mass function) der geometrische Verteilung mit dem Parameter p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityGeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des Au
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion (probability mass function) der geometrische Verteilung mit dem Parameter p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityGeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des Au
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der geometrischen Verteilung mit dem Parameter p für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dgeom() in R.

```
bool  MathProbabilityDensityGeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der geometrischen Verteilung mit dem Parameter p für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityGeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

p

© 2000-2025, MetaQuotes Ltd.

[in] Parameter der Verteilung (Wahrscheinlichkeit des Auftretens in einem Test).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionGeometric

Berechnet den Wert der Wahrscheinlichkeitsverteilung der geometrische Verteilung mit dem Parameter p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionGeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des Au
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der geometrische Verteilung mit dem Parameter p für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionGeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des Au
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der geometrische Verteilung mit dem Parameter p für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu pgeom() in R.

```
bool  MathCumulativeDistributionGeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Berechnet den Wert der Wahrscheinlichkeitsverteilung der geometrische Verteilung mit dem Parameter p für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionGeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
#   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunkti
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

p

[in] Parameter der Verteilung (Wahrscheinlichkeit des Auftretens in einem Test).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag der Berechnung, wenn tail=true, dann wird die Wahrscheinlichkeit berechnet, dass die Zuvallsvariable nicht größer als x ist.

log_mode

[in] Parameter der Berechnung des Logarithmus, log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeit berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileGeometric

Berechnet die Umkehrfunktion der geometrische Verteilung für die angegebene Wahrscheinlichkeit *probability* mit dem Parameter p. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileGeometric(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Umkehrfunktion der geometrische Verteilung für die angegebene Wahrscheinlichkeit *probability* mit dem Parameter p. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileGeometric(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet die Umkehrfunktion der geometrischen Verteilung für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte mit dem Parameter p. Im Fehlerfall retourniert false. Analog zu qgeom() in R.

```
double  MathQuantileGeometric(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet die Umkehrfunktion der geometrischen Verteilung für die im Array *probability[]* angegebenen Wahrscheinlichkeitswerte mit dem Parameter p. Im Fehlerfall retourniert false.

```
bool  MathQuantileGeometric(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

p

© 2000-2025, MetaQuotes Ltd.

[in] Parameter der Verteilung (Wahrscheinlichkeit des Auftretens in einem Test).

tail

[in] Flag, wenn false, wird mit einer '1.0-probability' gerechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomGeometric

Erzeugt eine Pseudozufallszahl auf Basis der geometrische Verteilung mit dem Parameter p. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomGeometric(
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der geometrischen Verteilung mit dem Parameter p. Im Fehlerfall retourniert false. Analog zu rgeom() in R.

```
bool  MathRandomGeometric(
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

p

[in] Parameter der Verteilung (Wahrscheinlichkeit des Auftretens in einem Test).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsGeometric

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der geometrische Verteilung mit dem Parameter p.

```
double  MathMomentsGeometric(
   const double  p,              // Parameter der Verteilung (Wahrscheinlichkeit des A
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

p

[in] Parameter der Verteilung (Wahrscheinlichkeit des Auftretens in einem Test).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

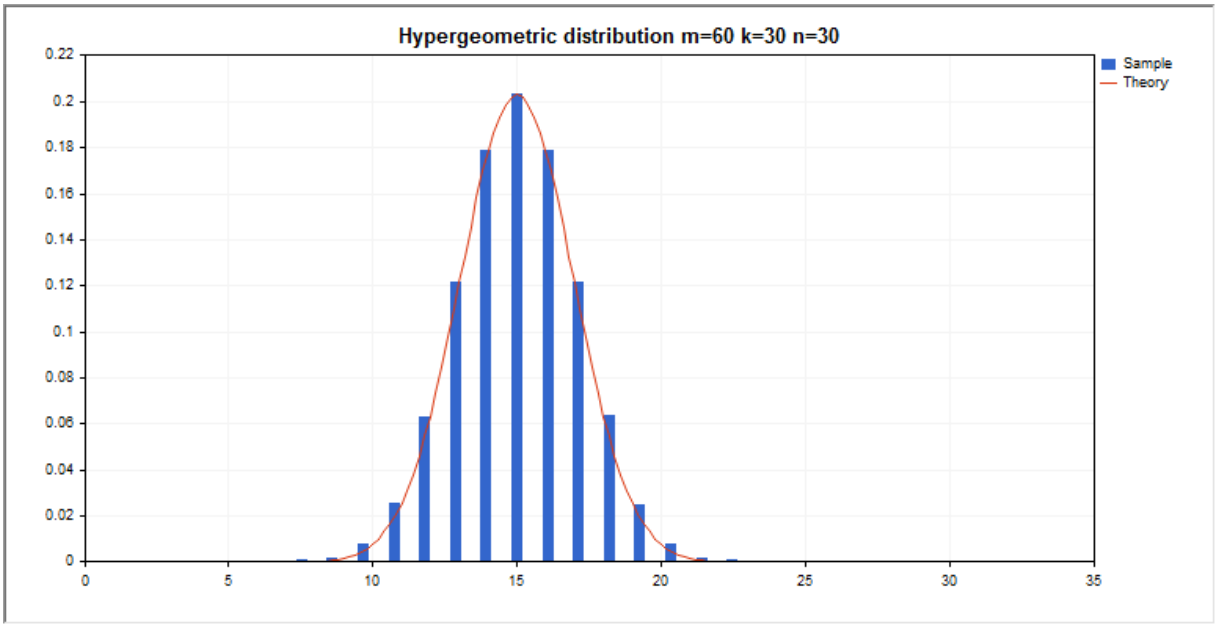
Hypergeometrische Verteilung

In diesem Abschnitt sind Funktionen für die hypergeometrische Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der hypergeometrischen Verteilung erzeugt werden. Die hypergeometrische Verteilung wird mit der folgenden Formel berechnet:

$$f_{Hypergeometric}(x \mid m, k, n) = \frac{\binom{k}{x}\binom{m-k}{n-x}}{\binom{m}{n}}$$

wobei:

- x — Wert der Zufallsvariablen (integer)
- m — Gesamtzahl der Objekte
- k — Anzahl der Objekte mit den gewünschten Eigenschaften
- n — Anzahl der gezogenen Objekte



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityHypergeometric	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der hypergeometrische Verteilung
MathCumulativeDistributionHypergeometric	Berechnet den Wert der Wahrscheinlichkeitsfunktion der hypergeometrische Verteilung
MathQuantileHypergeometric	Berechnet den Wert der Umkehrfunktion der hypergeometrische Verteilung für eine angegebene Wahrscheinlichkeit

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathRandomHypergeometric	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der hypergeometrischen Verteilung
MathMomentsHypergeometric	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der hypergeometrischen Verteilung

Beispiel:

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Hypergeometric.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double m_par=60;      // Gesamtzahl der Objekte
input double k_par=30;      // Anzahl der Objekte mit der gewünschten Eigenschaft
input double n_par=30;      // Anzahl der gezogenen Objekte
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=1000000;       // Anzahl der Werte in der Stichprobe
   int ncells=15;       // Anzahl der Intervalle im Histogramm
   double x[];          // Zentren der Intervalle des Histogramms
   double y[];          // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];       // Stichprobe
   double max,min;      // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der hypergeometrischen Verteilung erhalten
   MathRandomHypergeometric(m_par,k_par,n_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   PrintFormat("max=%G min=%G",max,min);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(0,n_par,1,x2);
   MathProbabilityDensityHypergeometric(x2,m_par,k_par,n_par,false,y2);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
   graphic.BackgroundMain(StringFormat("Hypergeometric distribution m=%G k=%G n=%G",m_
   graphic.BackgroundMainSize(16);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory").LinesSmooth(true);
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
```

© 2000-2025, MetaQuotes Ltd.

```
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityHypergeometric

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der hypergeometrischen Verteilung mit den Parametern m, k und n für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityHypergeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschafte
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der hypergeometrischen Verteilung mit den Parametern m, k und n für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityHypergeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschafte
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion(probability mass function) der hypergeometrischen Verteilung mit den Parametern m, k und n für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dhyper() in R.

```
bool  MathProbabilityDensityHypergeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const bool    log_mode,       // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der hypergeometrischen Verteilung mit den Parametern m, k und n für den Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityHypergeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   double&       result[]        // Array der Werte der Funktion der Wahrscheinlichkei
```

© 2000-2025, MetaQuotes Ltd.

```
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

m

[in] Gesamtzahl der Objekte (integer).

k

[in] Anzahl der Objekte mit den gewünschten Eigenschaften (integer).

n

[in] Anzahl der gezogenen Objekte (integer).

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionHypergeometric

Berechnet den Wert der Wahrscheinlichkeitsfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionHypergeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschafte
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrscheir
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionHypergeometric(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschafte
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dhyper() in R.

```
bool  MathCumulativeDistributionHypergeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionHypergeometric(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

© 2000-2025, MetaQuotes Ltd.

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

m

[in] Gesamtzahl der Objekte (integer).

k

[in] Anzahl der Objekte mit den gewünschten Eigenschaften (integer).

n

[in] Anzahl der gezogenen Objekte (integer).

tail

[in] Flag der Berechnung, wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeit berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantileHypergeometric

Berechnet den Wert der Umkehrfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileHypergeometric(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für die angegebene Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantileHypergeometric(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für das Array mit den Werten der Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false. Analog zu qhyper() in R.

```
double  MathQuantileHypergeometric(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der hypergeometrischen Verteilung mit den Parametern m, k und n für das Array mit den Werten der Wahrscheinlichkeit *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantileHypergeometric(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   double&       result[]        // Array mit der Werten der Quantile
```

© 2000-2025, MetaQuotes Ltd.

```
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

m

[in] Gesamtzahl der Objekte (integer).

k

[in] Anzahl der Objekte mit den gewünschten Eigenschaften (integer).

n

[in] Anzahl der gezogenen Objekte (integer).

tail

[in] Flag der Berechnung, wenn tail=false, wird für eine '1.0-probability' berechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomHypergeometric

Erzeugt eine Pseudozufallszahl auf Basis der hypergeometrischen Verteilung mit den Parametern m, n und k. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomHypergeometric(
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   int&          error_code      // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der hypergeometrischen Verteilung mit den Parametern m, n und k. Im Fehlerfall retourniert false. Analog zu rgeom() in R.

```
bool  MathRandomHypergeometric(
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   const int     data_count,     // Anzahl der benötigten Daten
   double&       result[]        // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

m

[in] Gesamtzahl der Objekte (integer).

k

[in] Anzahl der Objekte mit den gewünschten Eigenschaften (integer).

n

[in] Anzahl der gezogenen Objekte (integer).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsHypergeometric

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der hypergeometrischen Verteilung mit den Parametern m, n und k.

```
double  MathMomentsHypergeometric(
   const double  m,              // Gesamtzahl der Objekte (integer)
   const double  k,              // Anzahl der Objekte mit den gewünschten Eigenschaft
   const double  n,              // Anzahl der gezogenen Objekte (integer)
   double&       mean,           // Variable des Mittelwerts
   double&       variance,       // Variable der Varianz
   double&       skewness,       // Variable der Schiefe
   double&       kurtosis,       // Variable der Kurtosis
   int&          error_code      // Variable für den Fehlercode
   );
```

Parameter

m

[in] Gesamtzahl der Objekte (integer).

k

[in] Anzahl der Objekte mit den gewünschten Eigenschaften (integer).

n

[in] Anzahl der gezogenen Objekte (integer).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

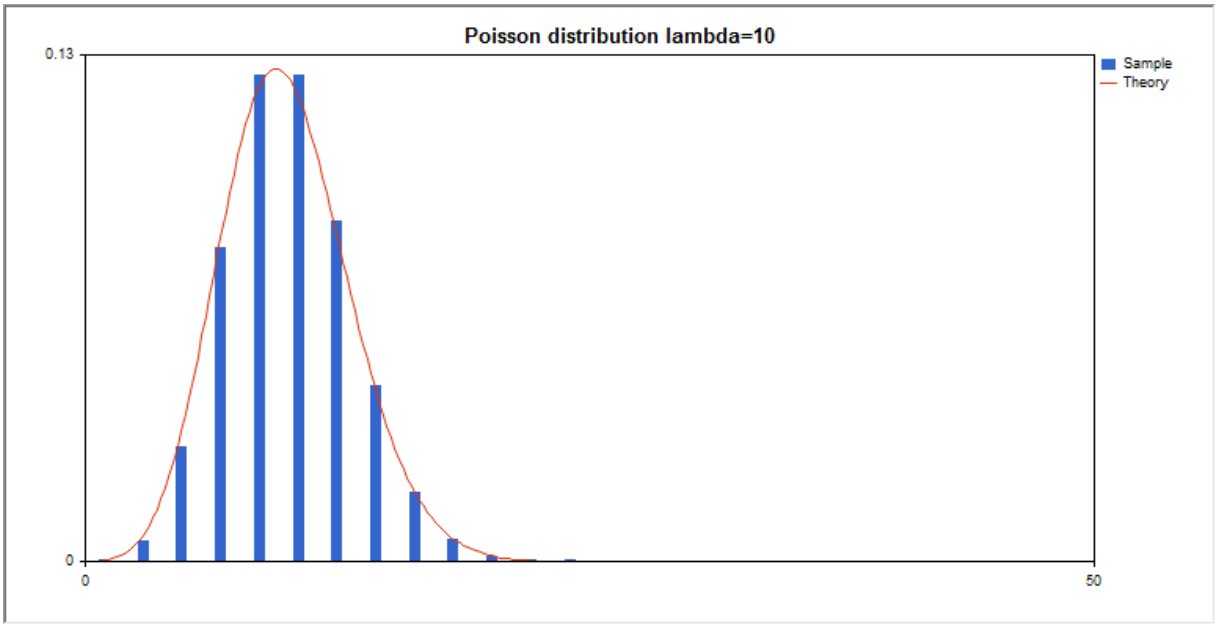
Poisson-Verteilung

In diesem Abschnitt sind Funktionen für die Poisson-Verteilung beschrieben. Mit diesen Funktionen können Dichte, Wahrscheinlichkeit und Quantile berechnet und Pseudozufallszahlen auf Basis der Poisson-Verteilung erzeugt werden. Die Poisson-Verteilung wird mit der folgenden Formel berechnet:

$$f_{Poisson}(x|\lambda) = \frac{\lambda^x}{x!}e^{-\lambda}$$

wobei:

- x — der Wert der Zufallsvariablen
- λ — Parameter der Verteilung (mean)



Man kann sowohl einzelne Zufallsvariablen berechnen, als auch mit Arrays von Zufallsvariablen arbeiten.

Funktion	Beschreibung
MathProbabilityDensityPoisson	Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Poisson-Verteilung
MathCumulativeDistributionPoisson	Berech Poisson-Verteilung
MathQuantilePoisson	Berechnet den Wert der Umkehrfunktion der Poisson-Verteilung für eine angegebene Wahrscheinlichkeit
MathRandomPoisson	Erzeugt eine Pseudozufallszahl/ein Array für Pseudozufallszahlen auf Basis der Poisson-Verteilung
MathMomentsPoisson	Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der Poisson-Verteilung

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
#include <Graphics\Graphic.mqh>
#include <Math\Stat\Poisson.mqh>
#include <Math\Stat\Math.mqh>
#property script_show_inputs
//--- input parameters
input double lambda_par=10;      // Parameter der Verteilung (mean)
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- Anzeige des Preischarts deaktivieren
   ChartSetInteger(0,CHART_SHOW,false);
//---
   MathSrand(GetTickCount());
//--- Stichprobe einer zufälligen Größe erzeugen
   long chart=0;
   string name="GraphicNormal";
   int n=100000;        // Anzahl der Werte in der Stichprobe
   int ncells=13;        // Anzahl der Intervalle im Histogramm
   double x[];           // Zentren der Intervalle des Histogramms
   double y[];           // Anzahl der Werte aus der Stichprobe, die innerhalb des Inte
   double data[];        // Stichprobe
   double max,min;       // der höchste und der niedrigste Werte in der Stichprobe
//--- Stichprobe aus der Poisson-Verteilung erhalten
   MathRandomPoisson(lambda_par,n,data);
//--- Daten für das Zeichnen des Histogramms berechnen
   CalculateHistogramArray(data,x,y,max,min,ncells);
//--- Grenzen der Sequenz und Schritt für das Zeichnen einer theoretischen Kurve erhal
   double step;
   GetMaxMinStepValues(max,min,step);
   PrintFormat("max=%G min=%G",max,min);
//--- theoretisch berechnete Daten im Intervall [min,max] erhalten
   double x2[];
   double y2[];
   MathSequence(0,int(MathCeil(max)),1,x2);
   MathProbabilityDensityPoisson(x2,lambda_par,false,y2);
//--- skalieren
   double theor_max=y2[ArrayMaximum(y2)];
   double sample_max=y[ArrayMaximum(y)];
   double k=sample_max/theor_max;
   for(int i=0; i<ncells; i++)
      y[i]/=k;
//--- Charts ausgeben
   CGraphic graphic;
   if(ObjectFind(chart,name)<0)
      graphic.Create(chart,name,0,0,0,780,380);
   else
      graphic.Attach(chart,name);
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.BackgroundMain(StringFormat("Poisson distribution lambda=%G",lambda_par));
   graphic.BackgroundMainSize(16);
//--- Autoskalierung der Y-Achse deaktivieren
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Max(NormalizeDouble(theor_max,2));
   graphic.YAxis().Min(0);
//--- plot all curves
   graphic.CurveAdd(x,y,CURVE_HISTOGRAM,"Sample").HistogramWidth(6);
//--- und nun die theoretische Kurve der Verteilungsdichte zeichnen
   graphic.CurveAdd(x2,y2,CURVE_LINES,"Theory").LinesSmooth(true);
   graphic.CurvePlotAll();
//--- plot all curves
   graphic.Update();
  }
//+------------------------------------------------------------------+
//|  Calculate frequencies for data set                              |
//+------------------------------------------------------------------+
bool CalculateHistogramArray(const double &data[],double &intervals[],double &frequenc
                             double &maxv,double &minv,const int cells=10)
  {
   if(cells<=1) return (false);
   int size=ArraySize(data);
   if(size<cells*10) return (false);
   minv=data[ArrayMinimum(data)];
   maxv=data[ArrayMaximum(data)];
   double range=maxv-minv;
   double width=range/cells;
   if(width==0) return false;
   ArrayResize(intervals,cells);
   ArrayResize(frequency,cells);
//--- Zentren der Intervalle setzen
   for(int i=0; i<cells; i++)
     {
      intervals[i]=minv+(i+0.5)*width;
      frequency[i]=0;
     }
//--- Frequenzen des Auftretens innerhalb des Intervalls füllen
   for(int i=0; i<size; i++)
     {
      int ind=int((data[i]-minv)/width);
      if(ind>=cells) ind=cells-1;
      frequency[ind]++;
     }
   return (true);
  }
//+------------------------------------------------------------------+
//|  Calculates values for sequence generation                       |
//+------------------------------------------------------------------+
void GetMaxMinStepValues(double &maxv,double &minv,double &stepv)
```

© 2000-2025, MetaQuotes Ltd.

```
  {
//--- die absolute Spannweite der Sequenz berechnen, um die Genauigkeit der Normalisie
   double range=MathAbs(maxv-minv);
   int degree=(int)MathRound(MathLog10(range));
//--- den höchsten und den niedrigsten Wert mit der angegebenen Genauigkeit normalisie
   maxv=NormalizeDouble(maxv,degree);
   minv=NormalizeDouble(minv,degree);
//--- den Schritt der Erzeugung einer Sequenz auch basierend auf der angegebenen Genau
   stepv=NormalizeDouble(MathPow(10,-degree),degree);
   if((maxv-minv)/stepv<10)
      stepv/=10.;
  }
```

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityPoisson

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der Poisson-Verteilung mit dem Parameter lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityPoisson(
   const double  x,             // Wert der Zufallsvariablen (integer)
   const double  lambda,        // Parameter der Verteilung (mean)
   const bool    log_mode,      // Parameter der Logarithmusberechnung, wenn log_mode=
   int&          error_code     // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der Poisson-Verteilung mit dem Parameter lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathProbabilityDensityPoisson(
   const double  x,             // Wert der Zufallsvariablen (integer)
   const double  lambda,        // Parameter der Verteilung (mean)
   int&          error_code     // Variable für den Fehlercode
   );
```

Berechnet den Wert der Wahrscheinlichkeitsdichtefunktion der Poisson-Verteilung mit dem Parameter lambda für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dhyper() in R.

```
bool  MathProbabilityDensityPoisson(
   const double& x[],           // Array mit den Werten der Zufallsvariablen
   const double  lambda,        // Parameter der Verteilung (mean)
   const bool    log_mode,      // Flag der Berechnung des Logarithmus, wenn log_mode
   double&       result[]       // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Berechnet den Wert der Wahrscheinlichkeitsfunktion (probability mass function) der Poisson-Verteilung mit dem Parameter lambda für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathProbabilityDensityPoisson(
   const double& x[],           // Array mit den Werten der Zufallsvariablen
   const double  lambda,        // Parameter der Verteilung (mean)
   double&       result[]       // Array der Werte der Funktion der Wahrscheinlichkei
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

lambda

[in] Parameter der Verteilung (mean).

log_mode

© 2000-2025, MetaQuotes Ltd.

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeitsdichte berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array für die Werte der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionPoisson

Berechnet den Wert der Poisson-Verteilung mit den Parametern lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionPoisson(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  lambda,         // Parameter der Verteilung (mean)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschein
   const bool    log_mode,       // Flag der Logarithmusberechnung, wenn log_mode=true,
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Poisson-Verteilung mit den Parametern lambda für die Zufallsvariable x. Im Fehlerfall retourniert sie NaN.

```
double  MathCumulativeDistributionPoisson(
   const double  x,              // Wert der Zufallsvariablen (integer)
   const double  lambda,         // Parameter der Verteilung (mean)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Poisson-Verteilung mit den Parametern lambda für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false. Analog zu dhyper() in R.

```
bool  MathCumulativeDistributionPoisson(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  lambda,         // Parameter der Verteilung (mean)
   const double  tail,           // Flag der Berechnung, wenn true, wird die Wahrschei
   const bool    log_mode,       // Parameter der Logarithmusberechnung, wenn log_mode
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Berechnet den Wert der Poisson-Verteilung mit den Parametern lambda für das Array der Zufallsvariablen x[]. Im Fehlerfall retourniert false.

```
bool  MathCumulativeDistributionPoisson(
   const double& x[],            // Array mit den Werten der Zufallsvariablen
   const double  lambda,         // Parameter der Verteilung (mean)
   double&       result[]        // Array für die Werte der Wahrscheinlichkeitsfunktio
   );
```

Parameter

x

[in] Wert der Zufallsvariablen.

x[]

[in] Array mit den Werten der Zufallsvariablen.

lambda

[in] Parameter der Verteilung (mean).

© 2000-2025, MetaQuotes Ltd.

tail

[in] Flag der Berechnung, wenn true, wird die Wahrscheinlichkeit einer Zufallsvariablen, die nicht größer als x ist, berechnet.

log_mode

[in] Flag der Logarithmusberechnung, wenn log_mode=true, wird der natürliche Logarithmus der Wahrscheinlichkeit berechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit den Werten der Wahrscheinlichkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

MathQuantilePoisson

Berechnet den Wert der Umkehrfunktion der Poisson-Verteilung mit dem Parameter lambda für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantilePoisson(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  lambda,         // Parameter der Verteilung (mean)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Poisson-Verteilung mit dem Parameter lambda für die Wahrscheinlichkeit *probability*. Im Fehlerfall gibt NaN zurück.

```
double  MathQuantilePoisson(
   const double  probability,    // Der Wert der Wahrscheinlichkeit des Auftretens ein
   const double  lambda,         // Parameter der Verteilung (mean)
   int&          error_code      // Variable für den Fehlercode
   );
```

Berechnet den Wert der Umkehrfunktion der Poisson-Verteilung mit dem Parameter lambda für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false. Analog zu qhyper() in R.

```
double  MathQuantilePoisson(
   const double& probability[],  // Array mit den Wahrscheinlichkeitswerten einer Zufa
   const double  lambda,         // Parameter der Verteilung (mean)
   const bool    tail,           // Flag, wenn false, wird mit einer '1.0-probability'
   const bool    log_mode,       // Flag, wenn log_mode=true, wird mit einer 'Exp(prob
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Berechnet den Wert der Umkehrfunktion der Poisson-Verteilung mit dem Parameter lambda für das Array der Wahrscheinlichkeitswerte *probability[]*. Im Fehlerfall retourniert false.

```
bool  MathQuantilePoisson(
   const double& probability[],  // Array mit den Werten der Zufallsvariablen
   const double  lambda,         // Parameter der Verteilung (mean)
   double&       result[]        // Array mit der Werten der Quantile
   );
```

Parameter

probability

[in] Wahrscheinlichkeitswert einer Zufallsvariablen.

probability[]

[in] Array mit den Wahrscheinlichkeitswerten einer Zufallsvariablen.

lambda

© 2000-2025, MetaQuotes Ltd.

[in] Parameter der Verteilung (mean).

tail

[in] Flag der Berechnung, wenn tail=false, wird für eine '1.0-probability' berechnet.

log_mode

[in] Flag, wenn log_mode=true, wird mit einer 'Exp(probability)' gerechnet.

error_code

[out] Variable für den Fehlercode.

result[]

[out] Array mit der Werten der Quantile.

© 2000-2025, MetaQuotes Ltd.

MathRandomPoisson

Erzeugt eine Pseudozufallszahl auf Basis der Poisson-Verteilung mit dem Parameter lambda. Im Fehlerfall retourniert sie NaN.

```
double  MathRandomPoisson(
   const double  lambda,          // Parameter der Verteilung (mean)
   int&          error_code       // Variable für den Fehlercode
   );
```

Erzeugt Pseudozufallszahlen auf Basis der Poisson-Verteilung mit dem Parameter lambda. Im Fehlerfall retourniert false. Analog zu rgeom() in R.

```
bool  MathRandomPoisson(
   const double  lambda,          // Parameter der Verteilung (mean)
   const int     data_count,      // Anzahl der benötigten Daten
   double&       result[]         // Array mit den Werten der Pseudozufallszahlen
   );
```

Parameter

lambda

[in] Parameter der Verteilung (mean).

error_code

[out] Variable für den Fehlercode.

data_count

[out] Anzahl der benötigten Daten.

result[]

[out] Array mit den Werten der Pseudozufallszahlen.

© 2000-2025, MetaQuotes Ltd.

MathMomentsPoisson

Berechnet die theoretischen, numerischen Werte der ersten 4 Momente der hypergeometrischen Verteilung mit den Parametern m, n und k.

```
double  MathMomentsPoisson(
   const double  lambda,          // Parameter der Verteilung (mean)
   double&       mean,            // Variable des Mittelwerts
   double&       variance,        // Variable der Varianz
   double&       skewness,        // Variable der Schiefe
   double&       kurtosis,        // Variable der Kurtosis
   int&          error_code       // Variable für den Fehlercode
   );
```

Parameter

lambda

[in] Parameter der Verteilung (mean).

mean

[out] Variable des Mittelwertes.

variance

[out] Variable der Varianz.

skewness

[out] Variable der Schiefe.

kurtosis

[out] Variable der Kurtosis.

error_code

[out] Variable für den Fehlercode.

Rückgabewert

Gibt true zurück, wenn die Momente erfolgreich berechnet wurden, sonst false.

© 2000-2025, MetaQuotes Ltd.

Hilfsfunktionen

Eine Gruppe der Funktionen, die die grundlegenden mathematischen Operationen ausführen: Berechnung der Gamma-Funktion, Beta-Funktion, Fakultät, Exponentialfunktion, Logarithmen mit unterschiedlicher Basis, Quadratwurzel u.Ä.

Es ist möglich, sowohl einzelne Werte (reelle und ganzzahlige), als auch Arrays dieser Werte zu verarbeiten (und die Ergebnisse in ein separates oder in das Quell-Array auszugeben).

Funktion	Beschreibung
MathRandomNonZero	Gibt eine zufällige Gleitkommazahl im Bereich von 0.0 bis 1.0 zurück.
MathMoments	Berechnet die ersten 4 Momente der Elemente eines Arrays: Mittelwert, Varianz, Schiefe und Kurtosis.
MathPowInt	Erhebt eine Zahl in eine angegebene Potenz.
MathFactorial	Berechnet die Fakultät einer ganzen Zahl.
MathTrunc	Berechnet den ganzzahligen Teil der angegeben Zahl oder der Elemente eines Arrays.
MathRound	Rundet eine Zahl oder ein Array auf die angegebene kommastelle ab.
MathArctan2	Berechnet den Winkel, dessen Tangens der Relation zwei angegebener Zahlen aus dem Bereich [-pi, pi] gleich ist.
MathGamma	Berechnet den Wert der Gammafunktion.
MathGammaLog	Berechnet den Logarithmus der Gammafunktion.
MathBeta	Berechnet den Wert der Betafunktion.
MathBetaLog	Berechnet den Logarithmus der Betafunktion.
MathBetaIncomplete	Berechnet den Wert der unvollständigen Betafunktion.
MathGammaIncomplete	Berechnet den Wert der unvollständigen Gammafunktion.
MathBinomialCoefficient	Berechnet den Binomialkoeffizienten.
MathBinomialCoefficientLog	Berechnet den Logarithmus des Binomialkoeffizienten.
MathHypergeometric2F2	Berechnet den Wert der hypergeometrischen Funktion.

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathSequence	Bildet eine Sequenz von den folgenden Werten: erstes Element, letztes Element, Schritt der Sequenz.
MathSequenceByCount	Bildet eine Sequenz von den folgenden Werten: erstes Element, letztes Element, Anzahl der Elemente in der Sequenz.
MathReplicate	Berechnet eine sich wiederholende Sequenz von Werten.
MathReverse	Bildet ein Array der Werte mit der umgekehrten Reihenfolge der Elemente.
MathIdentical	Vergleicht zwei Arrays der Werte und gibt true zurück, wenn alle Elemente miteinander übereinstimmen.
MathUnique	Bildet ein Array nur mit sich nicht wiederholenden Werten.
MathQuickSortAscending	Funktion für die aufsteigende Sortierung.
MathQuickSortDescending	Funktion für die absteigende Sortierung.
MathQuickSort	Funktion für Sortierung.
MathOrder	Bildet ein Array mit der Umordnung entsprechend der Reihenfolge der Elemente des Arrays nach Sortierung.
MathBitwiseNot	Berechnet das Ergebnis der binären Operation NOT für die Elemente eines Arrays.
MathBitwiseAnd	Berechnet das Ergebnis der binären Operation AND für die Elemente eines Arrays.
MathBitwiseOr	Berechnet das Ergebnis der binären Operation OR für die Elemente eines Arrays.
MathBitwiseXor	Berechnet das Ergebnis der binären Operation OR für die Elemente eines Arrays.
MathBitwiseShiftL	Berechnet das Ergebnis der binären Operation SHL für die Elemente eines Arrays.
MathBitwiseShiftR	Berechnet das Ergebnis der binären Operation SHR für die Elemente eines Arrays.
MathCumulativeSum	Bildet ein Array mit kumulierter Summe.
MathCumulativeProduct	Bildet ein Array mit kumuliertem Produkt.
MathCumulativeMin	Bildet ein Array mit kumulierten Minima.
MathCumulativeMax	Bildet ein Array mit kumulierten Maxima.

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathSin	Berechnet den Wert der sin(x) Funktion für die Elemente eines Arrays.
MathCos	Berechnet den Wert der cos(x) Funktion für die Elemente eines Arrays.
MathTan	Berechnet den Wert der tan(x) Funktion für die Elemente eines Arrays.
MathArcsin	Berechnet den Wert der arcsin(x) Funktion für Elemente eines Arrays.
MathArccos	Berechnet den Wert der arccos(x) Funktion für Elemente eines Arrays.
MathArctan	Berechnet den Wert der arctan(x) Funktion für die Elemente eines Arrays.
MathSinPi	Berechnet den Wert der sin(pi*x) Funktion für die Elemente eines Arrays.
MathCosPi	Berechnet den Wert der cos(pi*x) Funktion für die Elemente eines Arrays.
MathTanPi	Berechent den Wert der tan(pi*x) Funktion für die Elemente eines Arrays.
MathAbs	Berechnet den absoluten Wert der Elemente eines Arrays.
MathCeil	Gibt den nächsten größten ganzzahligen Wert für die Elemente des Arrays.
MathFloor	Gibt den nächsten kleinsten ganzzahligen Wert für die Elemente eines Arrays.
MathSqrt	Berechnet Quadratwurzel für die Elemente eines Arrays.
MathExp	Berechnet die Werte der exp(x) Funktion für die Elemente eines Arrays.
MathPow	Berechnet den Wert der pow(x, power) Funktion für die Elemente eines Arrays.
MathLog	Berechnet den Wert der log(x) Funktion für die Elemente eines Arrays.
MathLog2	Berechnet den Wert des Logarithmus mit der Basis 2 für die Elemente eines Arrays.
MathLog10	Berechnet den Wert des Logarithmus mit der Basis 10 für die Elemente eines Arrays.
MathDifference	Bildet ein Array mit der Differenz der Elemente y[i]=x[i+lag]-x[i].

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathSample	Macht eine zufällige Stichprobe der Elemente eines Arrays.
MathTukeySummary	Berechnet die Fünf-Punkte-Zusammenfassung nach Tukey für die Elemente eines Arrays.
MathRange	Berechnet minimale und maximale Werte der Elemente eines Arrays.
MathMin	Gibt den minimalen Wert unter allen Elementen eines Arrays zurück.
MathMax	Gibt den maximalen Wert unter allen Elementen eines Arrays zurück.
MathSum	Gibt die Summe der Elemente eines Arrays zurück.
MathProduct	Gibt das Produkt der Elemente eines Arrays zurück.
MathStandardDeviation	Berechnet die Standardabweichung für die Elemente eines Arrays.
MathAverageDeviation	Berechnet die durchschnittliche Abweichung der Elemente eines Arrays.
MathMedian	Berechnet den Medianwert der Elemente eines Arrays.
MathMean	Berechnet den Mittelwert der Elemente eines Arrays.
MathVariance	Berechnet die Varianz der Elemente eines Arrays.
MathSkewness	Berechnet die Schiefe der Elemente eines Arrays.
MathKurtosis	Berechnet die Kurtosis der Elemente eines Arrays.
MathLog1p	Berechnet den Wert der log(1+x) Funktion für die Elemente eines Arrays.
MathExpm1	Berechnet den Wert der exp(x)-1 Funktion für die Elemente eines Arrays.
MathSinh	Berechnet den Wert der sinh(x) Funktion für die Elemente eines Arrays.
MathCosh	Berechnet den Wert der cosh(x) Funktion für die Elemente eines Arrays.
MathTanh	Berechnet den Wert der tanh(x) Funktion für die Elemente eines Arrays.

© 2000-2025, MetaQuotes Ltd.

Funktion	Beschreibung
MathArcsinh	Berechnet den Wert der arcsinh(x) Funktion für die Elemente массива.
MathArccosh	Berechnet den Wert der arccosh(x) Funktion für die Elemente eines Arrays.
MathArctanh	Berechnet den Wert der arctanh(x) Funktion für die Elemente eines Arrays.
MathSignif	Rundet den Wert auf die angegebene Anzahl der Kommastellen in der Mantisse ab.
MathRank	Berechnet Ränge der Elemente eines Arrays.
MathCorrelationPearson	Berechnet den Korrelationskoeffizienten nach Pearson.
MathCorrelationSpearman	Berechnet den Korrelationskoeffizienten nach Spearman.
MathCorrelationKendall	Berechnet den Korrelationskoeffizienten nach Kendall.
MathQuantile	Berechnet Stichprobenquantile, die den angegebenen Wahrscheinlichkeiten entsprechen.
MathProbabilityDensityEmpirical	Berechnet die empirische Dichte für zufällige Werte.
MathCumulativeDistributionEmpirical	Berechnet die empirische kumulative Verteilung für zufällige Werte.

© 2000-2025, MetaQuotes Ltd.

MathRandomNonZero

Gibt eine zufällige Gleitkommazahl im Bereich von 0.0 bis 1.0 zurück.

```
double  MathRandomNonZero()
```

Rückgabewert

Eine zufällige Gleitkommazahl innerhalb des Bereichs von 0.0 bis 1.0.

© 2000-2025, MetaQuotes Ltd.

MathMoments

Berechnet die ersten 4 Momente der Elemente eines Arrays: Mittelwert, Varianz, Schiefe und Kurtosis.

```
bool  MathMoments(
   const double&  array[],            // Array der Werte
   double&        mean,               // Variable des Mittelwerts
   double&        variance,           // Variable der Varianz
   double&        skewness,           // Variable der Schiefe
   double&        kurtosis,           // Variable der Kurtosis
   const int      start=0,            // Anfangsindex
   const int      count=WHOLE_ARRAY   // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Array der Werte.

mean

[out] Variable des Mittelwerts (1. Moment).

variance

[out] Variable der Varianz (2. Moment).

skewness

[out] Variable der Schiefe (3. Moment).

kurtosis

[out] Variable der Kurtosis (4. Moment).

start=0

[in] Anfangsindex der Berechnung.

count=WHOLE_ARRAY

[in] Anzahl der Elemente für die Berechnung.

Rückgabewert

Gibt nach erfolgreicher Berechnung 'true' zurück, andernfalls 'false'.

© 2000-2025, MetaQuotes Ltd.

MathPowInt

Erhebt eine Zahl in eine angegebene Potenz.

```
double  MathPowInt(
   const double  x,       // die Zahl
   const int     power    // die Potenz
   )
```

Parameter

x

[in] Gleitkommazahl doppelter Genauigkeit, die in Potenz erhoben wird.

power

[in] Ganze Zahl, die die Potenz definiert.

Rückgabewert

Zahl *x*, erhoben in die angegebene Potenz.

© 2000-2025, MetaQuotes Ltd.

MathFactorial

Berechnet die Fakultät einer ganzen Zahl.

```
double  MathFactorial(
   const int  n   // die Zahl
   )
```

Parameter

n

[in] Ganze Zahl, deren Fakultät berechnet werden muss.

Rückgabewert

Fakultät einer Zahl.

© 2000-2025, MetaQuotes Ltd.

MathTrunc

Berechnet den ganzzahligen Teil der angegeben Zahl oder der Elemente eines Arrays.

Version für das Arbeiten mit dem Array für Gleitkommazahlen doppelter Genauigkeit:

```
double  MathTrunc(
   const double   x          // die Zahl
   )
```

Rückgabewert

Ganzzahliger Teil der angegebenen Zahl.

Version für das Arbeiten mit dem Array für Gleitkommazahlen mit doppelter Genauigkeit. Die Ergebnisse werden in ein neues Array geschrieben:

```
bool  MathTrunc(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Version für das Arbeiten mit dem Array für Gleitkommazahlen mit doppelter Genauigkeit. Die Ergebnisse werden in dasselbe Array ausgegeben:

```
bool  MathTrunc(
   double&        array[]    // Array der Werte
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Parameter

x

[in] Gleitkommazahl mit doppelter Genauigkeit, deren ganzzahliger Teil erhalten werden muss.

array[]

[in] Array für Gleitkommazahlen mit doppelter Genauigkeit, deren ganzzahliger Teil erhalten werden muss.

array[]

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

© 2000-2025, MetaQuotes Ltd.

MathRound

Rundet eine Gleitkommazahl mit doppelter Genauigkeit oder ein Array solcher Zahlen auf die angegebene Kommastellen ab.

Version für das Abrunden einer Gleitkommazahl doppelter Genauigkeit auf auf die angegeben Anzahl der Kommastellen:

```
double  MathRound(
   const double   x,          // Zahl
   const int      digits      // Anzahl der Stellen nach dem Komma
   )
```

Rückgabewert

Zahl, die am nächsten zum x Parameter liegt, Anzahl der Ziffern des Bruchteils ist gleich digits.

Version für das Abrunden des Array der Gleitkommazahlen doppelter Genauigkeit auf die angegebene Anzahl der Bruchteile. Die Ergebnisse werden in ein neues Array geschrieben.

```
bool  MathRound(
   const double&  array[],    // Array der Werte
   int            digits,     // Anzahl der Stellen nach dem Komma
   double&        result[]    // Array der Ergebnisse
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Version für das Abrunden des Array der Gleitkommazahlen doppelter Genauigkeit auf die angegebene Anzahl der Bruchteile. Die Ergebnisse werden in dasselbe Array geschrieben.

```
bool  MathRound(
   double&        array[],    // Array der Werte
   int            digits      // Anzahl der Stellen nach dem Komma
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Parameter

x

[in] Gleitkommazahl doppelter Genauigkeit, die abgerundet wird.

digits

[in] Anzahl der Ziffern des Bruchteils im Rückgabewert.

array[]

[in] Array der Gleitkommazahlen doppelter Genauigkeit, die abgerundet werden.

array[]

© 2000-2025, MetaQuotes Ltd.

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

© 2000-2025, MetaQuotes Ltd.

MathArctan2

Gibt den Arktangens vom Quotient zweier Argumente (x, y).

Version für das Arbeiten mit der Relation der zwei angegebenen Zahlen (x, y):

```
double  MathArctan2(
   const double       y,          // y Koordinate
   const double       x           // x Koordinate
   )
```

Rückgabewert

Der Winkel, θ, in Radianten, so dass -π≤θ≤π, und tan (θ) = y oder x, wobei (x, y) einen Punkt im Kartesischen Koordinatensystem darstellt.

Version für das Arbeiten mit der Relation von zwei Paaren der Elemente aus den x und y Arrays:

```
bool  MathArctan2(
   const double&      x[],        // Array der x Werte
   const double&      y[],        // Array der y Werte
   double&            result[]    // Array der Ergebnisse
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Parameter

y

[in] y Koordinate des Punktes.

x

[in] x Koordinate des Punktes.

x[]

[in] Array der x Koordinaten der Werte.

y[]

[in] Array der y Koordinaten der Werte.

result[]

[out] Array für die Ausgabe der Ergebnisse

Hinweise

Bitte beachten Sie:

- Für (x, y) im Quadrant 1 ist der Rückgabewert: 0 < θ < π/2.
- Für (x, y) im Quadrant 2 ist der Rückgabewert: π/2 < θ≤π.
- Für (x, y) im Quadrant 3 ist der Rückgabewert: -π < θ <-π/2.

© 2000-2025, MetaQuotes Ltd.

- Für (x, y) im Quadrant 4 ist der Rückgabewert: $-\pi/2 < \theta < 0$.

Für Punkte außerhalb der angegebenen Quadranten ist der Rückgabewert unten angeführt.

- Wenn y gleich 0 und x nicht negativ ist, $\theta = 0$.
- Wenn y gleich 0 und x negativ ist, $\theta = \pi$.
- Wenn y eine positive Zahl ist, und x gleich 0 ist, $\theta = \pi/2$.
- Wenn y eine negative Zahl und x gleich 0 ist, $\theta = -\pi/2$.
- Wenn y gleich 0 und x gleich 0 sind, beträgt $\theta = -\pi/2$.

Wenn der x- oder y-Wert gleich NaN ist, oder wenn die x- und y-Werte gleich PositiveInfinity oder NegativeInfinity sind, gibt die Methode NaN zurück.

© 2000-2025, MetaQuotes Ltd.

MathGamma

Berechnet den Wert der Gammafunktion für das reelle Argument *x*.

```
double  MathGamma(
   const double  x       // Argument der Funktion
   )
```

Parameter

x

[in] Reelles Argument der Funktion.

Rückgabewert

Wert der Gammafunktion.

© 2000-2025, MetaQuotes Ltd.

MathGammaLog

Berechnet den Wert des Logarithmus der Gammafunktion für das reelle Argument x.

```
double  MathGammaLog(
   const double  x      // Argument der Funktion
   )
```

Parameter

x

[in] Reelles Argument der Funktion.

Rückgabewert

Wert des Logarithmus der Funktion

© 2000-2025, MetaQuotes Ltd.

MathBeta

Berechnet den Wert der Betafunktion für die reellen Argumente a und b.

```
double  MathBeta(
   const double  a,      // erstes Argument der Funktion
   const double  b       // zweites Argument der Funktion
   )
```

Parameter

a

[in] Argument der Funktion a

b

[in] Argument der Funktion b.

Rückgabewert

Wert der Funktion.

© 2000-2025, MetaQuotes Ltd.

MathBetaLog

Berechnet den Wert des Logarithmus der Betafunktion für die reellen Argumente a und b.

```
double  MathBetaLog(
   const double  a,      // erstes Argument der Funktion
   const double  b       // zweites Argument der Funktion
   )
```

Parameter

a

[in] Argument der Funktion a

b

[in] Argument der Funktion b.

Rückgabewert

Wert des Logarithmus der Funktion

© 2000-2025, MetaQuotes Ltd.

MathBetaIncomplete

Berechnet den Wert der unvollständigen Betafunktion.

```
double  MathBetaIncomplete(
   const double  x,      // Argument der Funktion
   const double  p,      // erster Parameter der Funktion
   const double  q       // zweiter Parameter der Funktion
   )
```

Parameter

x

[in] Argument der Funktion.

p

[in] Der erste Parameter der Beta-Funktion muss >0.0 sein.

q

[in] Der zweite Parameter der Beta-Funktion muss >0.0 sein.

Rückgabewert

Wert der Funktion.

© 2000-2025, MetaQuotes Ltd.

MathGammaIncomplete

Berechnet den Wert der unvollständigen Gammafunktion.

```
double  MathGammaIncomplete(
   double  x,           // Argument der Funktion
   double  alpha        // Parameter der Funktion
   )
```

Parameter

x

[in] Argument der Funktion.

alpha

[in] Parameter der unvollständigen Gammafunktion.

Rückgabewert

Wert der Funktion.

© 2000-2025, MetaQuotes Ltd.

MathBinomialCoefficient

Berechnet den Binomialkoeffizienten: C(n,k)=n!/(k!*(n-k)!).

```
long  MathBinomialCoefficient(
   const int  n,      // Gesamtzahl der Elemente
   const int  k       // Anzahl der Elemente in einer Kombination
   )
```

Parameter

n

[in] Anzahl der Elemente.

k

[in] Anzahl der Elemente für jede Kombination.

Rückgabewert

Anzahl der Kombinationen aus N über K.

© 2000-2025, MetaQuotes Ltd.

MathBinomialCoefficientLog

Berechnet den Logarithmus des Binomialkoeffizienten: Log(C(n,k))=Log(n!/(k!*(n-k)!))

Version für ganzzahlige Argumente:

```
double  MathBinomialCoefficientLog(
   const int      n,       // Gesamtzahl der Elemente
   const int      k        // Anzahl der Elemente in einer Kombination
   )
```

Version für reelle Koeffizienten:

```
double  MathBinomialCoefficientLog(
   const double   n,       // Gesamtzahl der Elemente
   const double   k        // Anzahl der Elemente in einer Kombination
   )
```

Parameter

n

[in] Anzahl der Elemente.

k

[in] Anzahl der Elemente für jede Kombination.

Rückgabewert

Logarithmus von C(n,k).

© 2000-2025, MetaQuotes Ltd.

MathHypergeometric2F2

Berechnet den Wert der Funktion Hypergeometric_2F2 (a, b, c, d, z) anhand der Taylor-Methode.

```
double  MathHypergeometric2F2(
   const double  a,      // erster Parameter der Funktion
   const double  b,      // zweiter Parameter der Funktion
   const double  c,      // dritter Parameter der Funktion
   const double  d,      // vierter Parameter der Funktion
   const double  z       // fünfter Parameter der Funktion
   )
```

Parameter

a

[in] Erster Parameter der Funktion.

b

[in] Zweiter Parameter der Funktion.

c

[in] Dritter Parameter der Funktion.

d

[in] Vierter Parameter der Funktion.

z

[in] Fünfter Parameter der Funktion.

Rückgabewert

Wert der Funktion.

© 2000-2025, MetaQuotes Ltd.

MathSequence

Bildet eine Sequenz der Werte basierend auf den gegebenen Werten: erstes Element, letztes Element, Schritt der Sequenz.

Version für das Arbeiten mit reellen Werten:

```
bool  MathSequence(
   const double  from,        // Anfangswert
   const double  to,          // Endwert
   const double  step,        // Schritt
   double&       result[]     // Array der Ergebnisse
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathSequence(
   const int     from,        // Anfangswert
   const int     to,          // Endwert
   const int     step,        // Schritt
   int&          result[]     // Array der Ergebnisse
   )
```

Parameter

from

[in] Erster Wert der Sequenz

to

[in] Letzter Wert der Sequenz

step

[in] Schritt der Sequenz.

result[]

[out] Array für die Ausgabe der Sequenz.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSequenceByCount

Bildet eine Sequenz basierend auf den gegebenen Werten: erstes Element, letztes Element, Anzahl der Elemente in der Sequenz.

Version für das Arbeiten mit reellen Werten:

```
bool  MathSequenceByCount(
   const double  from,        // Anfangswert
   const double  to,          // Endwert
   const int     count,       // Anzahl
   double&       result[]     // Array der Ergebnisse
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathSequenceByCount(
   const int     from,        // Anfangswert
   const int     to,          // Endwert
   const int     count,       // Anzahl
   int&          result[]     // Array der Ergebnisse
   )
```

Parameter

from

[in] Erster Wert der Sequenz.

to

[in] Letzter Wert der Sequenz.

count

[in] Anzahl der Elemente in der Sequenz.

result[]

[out] Array für die Ausgabe der Sequenz.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathReplicate

Berechnet eine sich wiederholende Sequenz von Werten.

Version für das Arbeiten mit reellen Werten:

```
bool  MathReplicate(
   const double&  array[],    // Array der Werte
   const int      count,      // Anzahl der Wiederholungen
   double&        result[]    // Array der Ergebnisse
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathReplicate(
   const int&     array[],    // Array der Werte
   const int      count,      // Anzahl der Wiederholungen
   int&           result[]    // Array der Ergebnisse
   )
```

Parameter

array[]

[in] Array für die Bildung einer Sequenz.

count

[in] Anzahl der Wiederholungen des Arrays in der Sequenz.

result[]

[out] Array für die Ausgabe der Sequenz.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathReverse

Bildet ein Array der Werte mit der umgekehrten Reihenfolge der Elemente.

Version für das Arbeiten mit reellen Werten mit dem Speichern des Ergebnisses in ein neues Array:

```
bool  MathReverse(
   const double&  array[],      // Array der Werte
   double&        result[]      // Array der Ergebnisse
   )
```

Version für das Arbeiten mit ganzzahligen Werten mit dem Speichern der Ergebnisse in ein neues Array:

```
bool  MathReverse(
   const int&     array[],      // Array der Werte
   int&           result[]      // Array der Ergebnisse
   )
```

Version für das Arbeiten mit reellen Werten mit dem Speichern der Ergebnisse in dasselbe Array.

```
bool  MathReverse(
   double&        array[]       // Array der Werte
   )
```

Version für das Arbeiten mit ganzzahligen Werten mit dem Speichern der Ergebnisse in dasselbe Array.

```
bool  MathReverse(
   int&           array[]       // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

array[]

[out] Output Array mit der umgekehrten Reihenfolge der Werte.

result[]

[out] Output Array mit der umgekehrten Reihenfolge der Werte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathIdentical

Vergleicht zwei Arrays der Werte und gibt true zurück, wenn alle Elemente miteinander übereinstimmen.

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathIdentical(
   const double&  array1[],      // erstes Array der Werte
   const double&  array2[]       // zweites Array der Werte
   )
```

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathIdentical(
   const int&     array1[],      // erstes Array der Werte
   const int&     array2[]       // zweites Array der Werte
   )
```

Parameter

array1[]

[in] Erstes Array für den Vergleich.

array2[]

[in] Zweites Array für den Vergleich.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathUnique

Bildet ein Array nur mit sich nicht wiederholenden Werten.

Version für das Arbeiten mit reellen Werten:

```
bool  MathUnique(
   const double&   array[],   // Array der Werte
   double&         result[]   // Array der Ergebnisse
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathUnique(
   const int&     array[],      // Array der Werte
   int&           result[]      // Array der Ergebnisse
   )
```

Parameter

array[]

[in] Quell-Array.

result[]

[out] Array für die Ausgabe einmaliger Werte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathQuickSortAscending

Funktion für eine gleichzeitige aufsteigende Sortierung der Arrays array[] und indices[] unter Verwendung des QuickSort Algorithmus.

```
void  MathQuickSortAscending(
   double&  array[],      // Array der Werte
   int&     indices[],    // Array der Indizes
   int      first,        // Anfangswert
   int      last          // Endwert
   )
```

Parameter

array[]

[in][out] Array für Sortierung.

indices[]

[in][out] Array für das Speichern der Indizes des Quell-Arrays.

first

[in] Index des Elements, mit welchem man die Sortierung beginnt.

last

[in] Index des Elements, mit welchem man die Sortierung beendet.

© 2000-2025, MetaQuotes Ltd.

MathQuickSortDescending

Funktion für eine gleichzeitige absteigende Sortierung der Arrays y[] und indices[] unter Verwendung des QuickSort Algorithmus.

```
void  MathQuickSortDescending(
   double&  array[],      // Array der Werte
   int&     indices[],    // Array der Indizes
   int      first,        // Anfangswert
   int      last          // Endwert
   )
```

Parameter

array[]

[in][out] Array für Sortierung.

indices[]

[in][out] Array für das Speichern der Indizes des Quell-Arrays.

first

[in] Index des Elements, mit welchem man die Sortierung beginnt.

last

[in] Index des Elements, mit welchem man die Sortierung beendet.

© 2000-2025, MetaQuotes Ltd.

MathQuickSort

Funktion für eine gleichzeitige Sortierung der Arrays array[] und indices[] unter Verwendung des QuickSort Algorithmus.

```
void  MathQuickSort(
   double&  array[],      // Array der Werte
   int&     indices[],    // Array der Indizes
   int      first,        // Anfangswert
   int      last,         // Endwert
   int      mode          // Richtung
   )
```

Parameter

array[]

[in][out] Array für Sortierung.

indices[]

[in][out] Array für das Speichern der Indizes des Quell-Arrays.

first

[in] Index des Elements, mit welchem man die Sortierung beginnt.

last

[in] Index des Elements, mit welchem man die Sortierung beendet.

mode

[in] Richtung der Sortierung (>0 aufsteigend, andernfalls absteigend).

© 2000-2025, MetaQuotes Ltd.

MathOrder

Bildet ein ganzzahliges Array mit einer Umordnung entsprechend der Reihenfolge der Elemente eines Arrays nach Sortierung.

Version für das Arbeiten mit dem Array reeller Werte:

```
bool  MathOrder(
   const double&  array[],   // Array der Werte
   int&           result[]   // Array der Ergebnisse
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathOrder(
   const int&     array[],   // Array der Werte
   int&           result[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array für die Ausgabe aussortierter Indizes.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathBitwiseNot

Berechnet das Ergebnis der binären Operation NOT für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathBitwiseNot(
   const int&  array[],   // Array der Werte
   int&        result[]   // Array der Werte
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathBitwiseNot(
   int&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

array[]

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathBitwiseAnd

Berechnet das Ergebnis der binären Operation AND für die angegebenen Arrays.

```
bool  MathBitwiseAnd(
   const int&  array1[],  // erstes Array der Werte
   const int&  array2[],  // zweites Array der Werte
   int&        result[]   // Array der Werte
   )
```

Parameter

array1[]

[in] Erstes Array der Werte.

array2[]

[in] Zweites Array der Werte.

result[]

[out] Array für die Ausgabe der Ergebnisse.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathBitwiseOr

Berechnet das Ergebnis der binären Operation OR für die angegebenen Arrays.

```
bool  MathBitwiseOr(
   const int&  array1[],  // erstes Array der Werte
   const int&  array2[],  // zweites Array der Werte
   int&        result[]   // Array der Werte
   )
```

Parameter

array1[]

[in] Erstes Array der Werte.

array2[]

[in] Zweites Array der Werte.

result[]

[out] Array für die Ausgabe der Ergebnisse.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathBitwiseXor

Berechnet das Ergebnis der binären Operation XOR für die angegebenen Arrays.

```
bool  MathBitwiseXor(
   const int&  array1[],  // erstes Array der Werte
   const int&  array2[],  // zweites Array der Werte
   int&        result[]   // Array der Werte
   )
```

Parameter

array1[]

[in] Erstes Array der Werte.

array2[]

[in] Zweites Array der Werte.

result[]

[out] Array für die Ausgabe der Ergebnisse.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathBitwiseShiftL

Berechnet das Ergebnis der binären Operation SHL (bitweise Verschiebung nach links) für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathBitwiseShiftL(
   const int&  array[],     // Array der Werte
   const int   n,           // Wert der Verschiebung
   int&        result[]     // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathBitwiseShiftL(
   int&        array[],     // Array der Werte
   const int   n            // Wert der Verschiebung
   )
```

Parameter

array[]

[in] Array der Werte.

n

[in] Anzahl der Bits für die Verschiebung.

array[]

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathBitwiseShiftR

Berechnet das Ergebnis der binären Operation SHL (bitweise Verschiebung nach links) für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathBitwiseShiftR(
   const int&  array[],     // Array der Werte
   const int   n,           // Wert der Verschiebung
   int&        result[]     // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathBitwiseShiftR(
   int&        array[],     // Array der Werte
   const int   n            // Wert der Verschiebung
   )
```

Parameter

array[]

[in] Array der Werte.

n

[in] Anzahl der Bits für die Verschiebung.

array[]

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeSum

Bildet ein Array mit kumulierter Summe.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCumulativeSum(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCumulativeSum(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

array[]

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeProduct

Bildet ein Array mit kumuliertem Produkt.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCumulativeProduct(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCumulativeProduct(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeMin

Bildet ein Array mit kumulierten Minima.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCumulativeMin(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCumulativeMin(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeMax

Bildet ein Array mit kumulierten Maxima.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCumulativeMax(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCumulativeMax(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSin

Berechnet den Wert der sin(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathSin(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathSin(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCos

Berechnet den Wert der cos(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCos(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCos(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathTan

Berechnet den Wert der tan(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathTan(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathTan(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathArcsin

Berechnet den Wert der arcsin(x) Funktion für Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathArcsin(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathArcsin(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathArccos

Berechnet den Wert der arccos(x) Funktion für Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathArccos(
   const double&   array[],    // Array der Werte
   double&         result[]    // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathArccos(
   double&         array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathArctan

Berechnet den Wert der arctan(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathArctan(
   const double&   array[],   // Array der Werte
   double&         result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathArctan(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSinPi

Berechnet den Wert der sin(pi*x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathSinPi(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathSinPi(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCosPi

Berechnet den Wert der cos(pi*x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCosPi(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCosPi(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathTanPi

Berechent den Wert der tan(pi*x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathTanPi(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathTanPi(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathAbs

Berechnet den absoluten Wert der Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathAbs(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathAbs(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCeil

Gibt den nächsten größten ganzzahligen Wert für die Elemente des Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCeil(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCeil(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathFloor

Gibt den nächsten kleinsten ganzzahligen Wert für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathFloor(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathFloor(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSqrt

Berechnet Quadratwurzel für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathSqrt(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathSqrt(
   double&         array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathExp

Berechnet die Werte der exp(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathExp(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathExp(
   double&       array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathPow

Berechnet den Wert der pow(x, power) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathPow(
   const double&  array[],    // Array der Werte
   const double   power,      // Potenz
   double&        result[]    // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathPow(
   double&        array[],    // Array der Werte
   const double   power       // Potenz
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathLog

Berechnet den Wert der log(x) Funktion für die Elemente eines Arrays.

Version für die Berechnung des natürlichen Logarithmus mit der Ausgabe der Ergebnisse in ein neues Array.

```
bool  MathLog(
   const double&  array[],      // Array der Werte
   double&        result[]      // Array der Ergebnisse
   )
```

Version für die Berechnung des natürlichen Logarithmus mit der Ausgabe der Ergebnisse in das Quell-Array.

```
bool  MathLog(
   double&        array[]       // Array der Werte
   )
```

Version für de Berechnung des Logarithmus mit der angegebenen Basis mit der Ausgabe der Ergebnisse in ein neues Array.

```
bool  MathLog(
   const double&   array[],     // Array der Werte
   const double    base,        // Basis des Logarithmus
   double&         result[]     // Array der Ergebnisse
   )
```

Version für de Berechnung des Logarithmus mit der angegebenen Basis mit der Ausgabe der Ergebnisse in das Quell-Array.

```
bool  MathLog(
   double&         array[],     // Array der Werte
   const double    base         // Basis des Logarithmus
   )
```

Parameter

array[]

[in] Array der Werte.

base

[in] Basis des Logarithmus.

array[]

[out] Array der Ausgabewerte.

result[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

MathLog2

Berechnet den Wert des Logarithmus mit der Basis 2 für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathLog2(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathLog2(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathLog10

Berechnet den Wert des Logarithmus mit der Basis 10 für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathLog10(
   const double&   array[],   // Array der Werte
   double&         result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathLog10(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathLog1p

Berechnet den Wert der log(1+x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathLog1p(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathLog1p(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathDifference

Bildet ein Array mit der Differenz der Elemente y[i]=x[i+lag]-x[i].

Version für eine einmalige Bildung des Arrays reeller Werte:

```
bool  MathDifference(
   const double  &array[],     // Array der Werte
   const int     lag,          // Verzögerung
   double        &result[]     // Array der Ergebnisse
   )
```

Version für eine einmalige Bildung des Arrays ganzzahliger Werte:

```
bool  MathDifference(
   const int     &array[],     // Array der Werte
   const int     lag,          // Verzögerung
   int           &result[]     // Array der Ergebnisse
   )
```

Version für eine mehrfache Bildung des Arrays reeller Werte (Anzahl der Iterationen wird in den Eingabeparametern gesetzt):

```
bool  MathDifference(
   const double  &array[],     // Array der Werte
   const int     lag,          // Verzögerung
   const int     differences,  // Anzahl der Iterationen
   double        &result[]     // Array der Ergebnisse
   )
```

Version für eine mehrfache Bildung des Arrays reeller Werte (Anzahl der Iterationen wird in den Eingabeparametern gesetzt):

```
bool  MathDifference(
   const int&    array[],      // Array der Werte
   const int     lag,          // Verzögerung
   const int     differences,  // Anzahl der Iterationen
   int&          result[]      // Array der Ergebnisse
   )
```

Parameter

array[]

[in] Array der Werte.

lag

[in] Parameter der Verzögerung.

differences

[in] Anzahl der Iterationen.

result[]

© 2000-2025, MetaQuotes Ltd.

[out] Array für die Ausgabe der Ergebnisse.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSample

Macht eine zufällige Stichprobe der Elemente eines Arrays.

Version für das Arbeiten mit dem Array reeller Werte:

```
bool  MathSample(
   const double&  array[],          // Array der Werte
   const int      count,            // Anzahl
   double&        result[]          // Array der Ergebnisse
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathSample(
   const int&     array[],          // Array der Werte
   const int      count,            // Anzahl
   int&           result[]          // Array der Ergebnisse
   )
```

Version für das Arbeiten mit dem Array reeller Werte. Es gibt die Möglichkeit, eine Stichprobe mit Zurücklegen vorzunehmen:

```
bool  MathSample(
   const double&  array[],          // Array der Werte
   const int      count,            // Anzahl
   const bool     replace,          // Flag
   double&        result[],         // Array der Ergebnisse
   )
```

Version für das Arbeiten mit dem Array reeller Werte. Es gibt die Möglichkeit, eine Stichprobe mit Zurücklegen vorzunehmen:

```
bool  MathSample(
   const int&     array[],          // Array der Werte
   const int      count,            // Anzahl
   const bool     replace,          // Flag
   int&           result[]          // Array der Ergebnisse
   )
```

Version für das Arbeiten mit dem Array reeller Werte, für welche die Wahrscheinlichkeiten in der Stichprobe definiert sind.

```
bool  MathSample(
   const double&  array[],          // Array der Werte
   double&        probabilities[],  // Array der Wahrscheinlichkeiten
   const int      count,            // Anzahl
   double&        result[]          // Array der Ergebnisse
   )
```

Version für das Arbeiten mit dem Array ganzzahliger Werte, für welche die Wahrscheinlichkeiten in der Stichprobe definiert sind.

© 2000-2025, MetaQuotes Ltd.

```
bool  MathSample(
   const int&     array[],           // Array der Werte
   double&        probabilities[],   // Array der Wahrscheinlichkeiten
   const int      count,             // Anzahl
   int&           result[]           // Array der Ergebnisse
   )
```

Version für das Arbeiten mit dem Array reeller Werte, für welche die Wahrscheinlichkeiten in der Stichprobe definiert sind. Es besteht die Möglichkeit, eine Stichprobe mit Zurücklegen vorzunehmen:

```
bool  MathSample(
   const double&  array[],           // Array der Werte
   double&        probabilities[],   // Array der Wahrscheinlichkeiten
   const int      count,             // Anzahl
   const bool     replace,           // Flag
   double&        result[]           // Array der Ergebnisse
   )
```

Version für das Arbeiten mit dem Array ganzzahliger Werte, für welche die Wahrscheinlichkeiten in der Stichprobe definiert sind. Es besteht die Möglichkeit, eine Stichprobe mit Zurücklegen vorzunehmen:

```
bool  MathSample(
   const int&     array[],           // Array der Werte
   double&        probabilities[],   // Array der Wahrscheinlichkeiten
   const int      count,             // Anzahl
   const bool     replace,           // Flag
   int&           result[]           // Array der Ergebnisse
   )
```

Parameter

array[]

[in] Array ganzzahliger Werte.

probabilities[]

[in] Array der Wahrscheinlichkeiten, mit welchen die Stichprobe vorgenommen wird.

count

[in] Anzahl der Elemente.

replace

[in] Parameter, der Stichprobe mit Zurücklegen ermöglicht.

result[]

[out] Array für die Ausgabe der Ergebnisse.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Hinweis

Das replace=true Argument ermöglicht eine zufällige Stichprobe der Elemente mit Zurücklegen.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

MathTukeySummary

Berechnet die Fünf-Punkte-Zusammenfassung nach John Tukey (Minimum, unteres Quartil, Mittelwert, oberes Quartil, Maximum) für die Elemente eines Arrays.

```
bool  MathTukeySummary(
   const double&  array[],       // Array der Werte
   const bool     removeNAN,     // Flag
   double&        minimum,       // minimaler Wert
   double&        lower_hinge,   // unteres Quartil
   double&        median,        // Mittelwert
   double&        upper_hinge,   // oberes Quartil
   double&        maximum        // maximaler Wert
   )
```

Parameter

array[]

[in] Array reeller Werte.

removeNAN

[in] Flag, das angibt, ob nicht numerische Werte gelöscht werden müssen.

minimum

[out] Variable für die Ausgabe des minimalen Wertes.

lower_hinge

[out] Variable für die Ausgabe des unteren Quatils.

median

[out] Variable für die Ausgabe des Mittelwertes.

upper_hinge

[out] Variable für die Ausgabe des oberen Quartils.

maximum

[out] Variable für die Ausgabe des maximalen Wertes.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathRange

Berechnet minimale und maximale Werte der Elemente eines Arrays.

```
bool  MathRange(
   const double&  array[],  // Array der Werte
   double&        min,      // minimaler Wert
   double&        max       // maximaler Wert
   )
```

Parameter

array[]

[in] Array der Werte.

min

[out] Variable für die Ausgabe des minimalen Wertes.

max

[out] Variable für die Ausgabe des maximalen Wertes.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathMin

Gibt den minimalen Wert unter allen Elementen eines Arrays zurück.

```
double  MathMin(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Minimaler Wert.

© 2000-2025, MetaQuotes Ltd.

MathMax

Gibt den maximalen Wert unter allen Elementen eines Arrays zurück.

```
double  MathMax(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Maximaler Wert.

© 2000-2025, MetaQuotes Ltd.

MathSum

Gibt die Summe der Elemente eines Arrays zurück.

```
double  MathSum(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Summe der Elemente.

© 2000-2025, MetaQuotes Ltd.

MathProduct

Gibt das Produkt der Elemente eines Arrays zurück.

```
double  MathProduct(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Produkt der Elemente.

© 2000-2025, MetaQuotes Ltd.

MathStandardDeviation

Berechnet die Standardabweichung für die Elemente eines Arrays.

```
double  MathStandardDeviation(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Standardabweichung.

© 2000-2025, MetaQuotes Ltd.

MathAverageDeviation

Die Funktion berechnet die durchschnittliche Abweichung der Elemente eines Arrays.

```
double  MathAverageDeviation(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Durchschnittliche Abweichung der Elemente eines Arrays.

© 2000-2025, MetaQuotes Ltd.

MathMedian

Berechnet den Medianwert der Elemente eines Arrays.

```
double  MathMedian(
   double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Medianwert.

© 2000-2025, MetaQuotes Ltd.

MathMean

Berechnet den Mittelwert der Elemente eines Arrays.

```
double  MathMean(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Mittelwert.

© 2000-2025, MetaQuotes Ltd.

MathVariance

Die Funktion berechnet die Varianz (zweites Moment) der Elemente eines Arrays.

```
double  MathVariance(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Varianz.

© 2000-2025, MetaQuotes Ltd.

MathSkewness

Die Funktion berechnet die Schiefe (drittes Moment) der Elemente eines Arrays.

```
double  MathSkewness(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Schiefe.

© 2000-2025, MetaQuotes Ltd.

MathKurtosis

Die Funktion berechnet die Kurtosis (viertes Moment) der Elemente eines Arrays.

```
double  MathKurtosis(
   const double&  array[]   // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

Rückgabewert

Kurtosis.

© 2000-2025, MetaQuotes Ltd.

MathExpm1

Berechnet den Wert der exp(x)-1 Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathExpm1(
   const double&   array[],   // Array der Werte
   double&         result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathExpm1(
   double&         array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSinh

Berechnet den Wert der sinh(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathSinh(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathSinh(
   double&       array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCosh

Berechnet den Wert der cosh(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathCosh(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathCosh(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathTanh

Berechnet den Wert der tanh(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathTanh(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathTanh(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathArcsinh

Berechnet den Wert der arcsinh(x) Funktion für die Elemente массива.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathArcsinh(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathArcsinh(
   double&         array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathArccosh

Berechnet den Wert der arccosh(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathArccosh(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathArccosh(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathArctanh

Berechnet den Wert der arctanh(x) Funktion für die Elemente eines Arrays.

Version mit der Ausgabe der Ergebnisse in ein neues Array:

```
bool  MathArctanh(
   const double&  array[],   // Array der Werte
   double&        result[]   // Array der Ergebnisse
   )
```

Version mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathArctanh(
   double&        array[]    // Array der Werte
   )
```

Parameter

array[]

[in] Array der Werte.

result[]

[out] Array der Ausgabewerte.

array[]

[out] Array der Ausgabewerte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathSignif

Rundet den Wert auf die angegebene Anzahl der Kommastellen in der Mantisse ab.

Version für das Arbeiten mit reellem Wert:

```
double  MathSignif(
   const double   x,          // Wert
   const int      digits      // Anzahl der Kommastellen
   )
```

Rückgabewert

Abgerundeter Wert.

Version für das Arbeiten mit dem Array reeller Werte mit der Ausgabe der Ergebnisse in ein separates Array:

```
bool  MathSignif(
   const double&  array[],    // Array der Werte
   int            digits,     // Anzahl der Stellen
   double         result[]    // Array der Ergebnisse
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Version für das Arbeiten mit dem Array reeller Werte mit der Ausgabe der Ergebnisse in das Quell-Array:

```
bool  MathSignif(
   double&        array[],    // Array der Werte
   int            digits      // Anzahl der Kommastellen
   )
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

Parameter

x

[in] Reeller Wert für das Abrunden.

digits

[in] Anzahl der Stellen.

array[]

[in] Array reeller Werte.

array[]

[out] Array der Ausgabewerte.

result[]

© 2000-2025, MetaQuotes Ltd.

[out] Array der Ausgabewerte.

© 2000-2025, MetaQuotes Ltd.

MathRank

Berechnet Ränge der Elemente eines Arrays.

Version für das Arbeiten mit dem Array reeller Werte:

```
bool  MathRank(
   const double&  array[],  // Array der Werte
   double&        rank[]    // Array der Ränge
   )
```

Version für das Arbeiten mit den Arrays ganzzahliger Werte:

```
bool  MathRank(
   const int&     array[],  // Array der Werte
   double&        rank[]    // Array der Ränge
   )
```

Parameter

array[]

[in] Array der Werte.

rank[]

[out] Array für die Ausgabe der Ränge.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCorrelationPearson

Berechnet den Korrelationskoeffizienten nach Pearson.

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathCorrelationPearson(
   const double&  array1[],  // erstes Array
   const double&  array2[],  // erstes Array
   double&        r          // Korrelationskoeffizient
   )
```

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathCorrelationPearson(
   const int&     array1[],  // erstes Array der Werte
   const int&     array2[],  // zweites Array der Werte
   double&        r          // Korrelationskoeffizient
   )
```

Parameter

array1[]

[in] Erster Array der Werte.

array2[]

[in] Zweites Array der Werte.

r

[out] Variable für die Ausgabe des Korrelationskoeffizienten.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCorrelationSpearman

Berechnet den Korrelationskoeffizienten nach Spearman.

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathCorrelationSpearman(
   const double&  array1[],  // erstes Array
   const double&  array2[],  // erstes Array
   double&        r          // Korrelationskoeffizient
   )
```

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathCorrelationSpearman(
   const int&     array1[],   // erstes Array der Werte
   const int&     array2[],   // zweites Array der Werte
   double&        r           // Korrelationskoeffizient
   )
```

Parameter

array1[]

[in] Erster Array der Werte.

array2[]

[in] Zweites Array der Werte.

r

[out] Variable für den Korrelationskoeffizienten.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCorrelationKendall

Berechnet den Korrelationskoeffizienten nach Kendall.

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathCorrelationKendall(
   const double&  array1[],  // erstes Array
   const double&  array2[],  // erstes Array
   double&        tau        // Korrelationskoeffizient
   )
```

Version für das Arbeiten mit den Arrays reeller Werte:

```
bool  MathCorrelationKendall(
   const int&     array1[],  // erstes Array der Werte
   const int&     array2[],  // zweites Array der Werte
   double&        tau        // Korrelationskoeffizient
   )
```

Parameter

array1[]

[in] Erster Array der Werte.

array2[]

[in] Zweites Array der Werte.

tau

[out] Variable für die Ausgabe des Korrelationskoeffizienten.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathQuantile

Berechnet zufällige Quantile, die den angegebenen Wahrscheinlichkeiten entsprechen: Q[i](p) = (1 - gamma)*x[j] + gamma*x[j+1]

```
bool  MathQuantile(
   const double&  array[],     // Array der Werte
   const double&  probs[],     // Array der Wahrscheinlichkeiten
   double&        quantile[]   // Array für die Ausgabe der Quantile
   )
```

Parameter

array[]

[in] Array der Werte.

probs[]

[in] Array der Wahrscheinlichkeiten.

quantile[]

[out] Array für die Ausgabe der Quantile.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathProbabilityDensityEmpirical

Die Funktion berechnet die empirische Wahrscheinlichketisdichte(pdf) für zufällige Werte eines Arrays.

```
bool  MathProbabilityDensityEmpirical(
   const double&  array[],   // Array zufälliger Werte
   const int      count,     // Anzahl der Paare
   double&        x[],       // Array der x Werte
   double&        pdf[]      // Array der pdf Werte
   )
```

Parameter

array[]

[in] Array zufälliger Werte.

count

[in] Anzahl der Paare (x, pdf(x)).

x[]

[out] Array für die Ausgabe der x Werte.

pdf[]

[out] Array für die Ausgabe der pdf(x) Werte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

MathCumulativeDistributionEmpirical

Die Funktion berechnet die empirische kumulative Verteilung (cdf) für zufällige Werte aus einem Array.

```
bool  MathCumulativeDistributionEmpirical(
   const double&  array[],   // Array zufälliger Werte
   const int      count,     // Anzahl der Paare
   double&        x[],       // Array der x Werte
   double&        cdf[]      // Array der cdf Werte
   )
```

Parameter

array[]

[in] Array zufälliger Werte.

count

[in] Anzahl der Paare (x, cdf(x)).

x[]

[out] Array für die Ausgabe der x Werte.

cdf[]

[out] Array für die Ausgabe der cdf(x) Werte.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Fuzzy — Bibliothek für das Arbeiten mit der Fuzzy-Logik

Fuzzy-Logik stellt eine Verallgemeinerung der traditionellen aristotelischen Logik dar für den Fall, wenn die Wahrheit als linguistische Variable betrachtet wird. Wie in der klassischen Logik, gibt es auch in der Fuzzy-Logik unscharfe logische Operationen mit Fuzzy-Mengen. Operationen für Fuzzy-Mengen unterscheiden sich nicht von den Operationen für gewöhnliche Mengen, aber ihre Berechnung ist um vieles komplexer. Es ist zu betonen, dass eine Komposition unscharfer Mengen selbst eine Fuzzy-Menge ist.

Zu den wichtigsten Besonderheiten der Fuzzy-Logik, durch welche sie sich von der klassischen Logik unterscheidet, gehören maximale Annäherung an die Realität und eine hohe Subjektivität, was zu Fehlern bei den Berechnungen führen kann.

Ein Fuzzy-Modell (oder -System) stellt ein mathematisches Modell dar, dem die Fuzzy-Logik zugrunde liegt. Solche Modelle werden in dem Fall erstellt, wenn das Untersuchungsobjekt eine schwache Formalisierung hat, und seine genaue mathematische Beschreibung zu kompliziert oder unbekannt ist. Die Qualität von Ausgabedaten solcher Modelle (Ungenauigkeit des Modells) hängt nur von dem Expert Advisor ab, welcher das Modell einstellte. Für die Minimierung von Fehlern wird die Erstellung eines ausführlichen Modells und ihre konsequente Anpassung mithilfe maschinellen Lernens anhand eines großen Trainigssets die beste Lösung sein.

Die Erstellung eines Modells kann in drei Schritte geteilt werden:

1. Definition von Ein- und Ausgabeparametern des Modells.
2. Erstellung einer Wissensdatenbank.
3. Auswahl einer der Methoden der Fuzzy-Inferenz (Mamdani oder Sugeno).

Von dem ersten Schritt hängen unmittelbar die zwei weiteren Schritte ab und gerade er bestimmt das weitere Funktionieren des Modells. Als Wissensdatenbank oder mit anderen Worten Regeldatenbank wird eine Gesamtheit Fuzzy-Regeln vom Typ "wenn, dann" bezeichnet, die den Zusammenhang zwischen Inputs und Outputs des Untersuchungsobjekts bestimmen. Die Anzahl der Regel im System ist unbegrenzt und wird vom Experten festgelegt. Das verallgemeinerte Format von Fuzzy-Regeln sieht wie folgt aus:

Wenn Bedingung der Regel, dann die Konklusion der Regel.

Die Bedingung der Regel charakterisiert den aktuellen Zustand des Objekts und die Konklusion — wie die Bedingung das Objekt beeinflussen wird. Man kann keine allgemeine Art von Bedingungen und Konklusionen definieren, denn sie werden durch Fuzzy-Inferenz festgelegt.

Jede Regel im System hat ein Gewicht. Dieser Parameter charakterisiert die Wichtigkeit der Regel im Modell. Die Gewichtskoeffizienten nehmen Werte aus dem Bereich [0, 1] an. In vielen Beispielen unscharfer Modelle, die man in der Fachliteratur finden kann, sind Gewichte nicht angegeben, dies bedeutet allerdings nicht, dass es sie nicht gibt. In diesem Fall haben wir mit einem festgelegten Gewicht für jede Regel der Wissensdatenbank zu tun, das gleich eins ist. Bedingungen und Konklusionen für jede Regel können von zwei Arten sein:

1. einfach — eine Fuzzy-Variable;
2. zusammengesetzt — mehrere Fuzzy-Variablen.

Je nach der erstellten Wissensdatenbank wird für das Modell eine Inferenz festgelegt. Als Fuzzy-Inferenz wird das Erhalten einer Konklusion in Form einer Fuzzy-Menge bezeichnet, die aktuellen Input-

© 2000-2025, MetaQuotes Ltd.

Werten entspricht, unter Verwendung einer Fuzzy-Wissensdatenbank und Fuzzy-Operationen. Zu den zwei grundlegenden Typen der Inferenz gehören die Inferenz nach Mamdani und Inferenz nach Sugeno.

© 2000-2025, MetaQuotes Ltd.

Zugehörigkeitsfunktionen

Als **Zugehörigkeitsfunktion** *(membership function)* wird die Funktion bezeichnet, mit der man den Zugehörigkeitsgrad eines zufälligen Elements einer universellen Menge der Fuzzy-Menge berechnen kann. Daraus folgt, dass die Werte der Zugehörigkeitsfunktion im Bereich [0, 1] liegen.

In den meisten Fällen ist die Zugehörigkeitsfunktion monoton und stetig.

Klasse der Zugehörigkeitsfunktionen	Beschreibung
CConstantMembershipFunction	Klasse für die Implementierung der Zugehörigkeitsfunktion als eine Gerade parallel zur Koordinatenachse
CCompositeMembershipFunction	Klasse für die Implementierung einer Komposition von Zugehörigkeitsfunktionen
CDifferencTwoSigmoidalMembershipFunction	Implementierung der Zugehörigkeitsfunktion als Differenz zwischen zwei Sigmoidfunktionen mit den Parametern A1, A2, C1 und C2
CGeneralizedBellShapedMembershipFunction	Klasse für die Implementierung einer verallgemeinerten glockenförmigen Zugehörigkeitsfunktion mit den Parametern A, B und C.
CNormalCombinationMembershipFunction	Klasse für die Implementierung einer doppelseitigen Gauß'schen Zugehörigkeitsfunktion mit den Parametern B1, B2, Sigma1 und Sigma2
CNormalMembershipFunction	Klasse für die Implementierung einer symmetrischen Gauß'schen Zugehörigkeitsfunktion mit den Parametern B und Sigma
CP_ShapedMembershipFunction	Klasse für die Implementierung einer P-förmigen Zugehörigkeitsfunktion mit den Parametern A, B, C und D
CProductTwoSigmoidalMembershipFunction	Klasse für die Implementierung der Zugehörigkeitsfunktion als Produkt zweier Sigmoidfunktionen mit den Parametern A1, A2, C1 und C2
CS_ShapedMembershipFunction	Klasse für die Implementierung einer S-förmigen Zugehörigkeitsfunktion mit den Parametern A und B
CTrapezoidMembershipFunction	Klasse für die Implementierung einer trapezförmigen Zugehörigkeitsfunktion mit den Parametern X1, X2, X3 und X4

© 2000-2025, MetaQuotes Ltd.

Klasse der Zugehörigkeitsfunktionen	Beschreibung
CTriangularMembershipFunction	Klasse für die Implementierung einer Dreiecksfunktion mit den Parametern X1, X2 und X3
CSigmoidalMembershipFunction	Klasse für die Implementierung einer sigmoiden Zugehörigkeitsfunktion mit den Parametern A und C
CZ_ShapedMembershipFunction	Klasse für die Implementierung einer z-förmigen Zugehörigkeitsfunktion mit den Parametern A und B.
IMembershipFunction	Basisklasse für alle Klassen der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

CConstantMembershipFunction

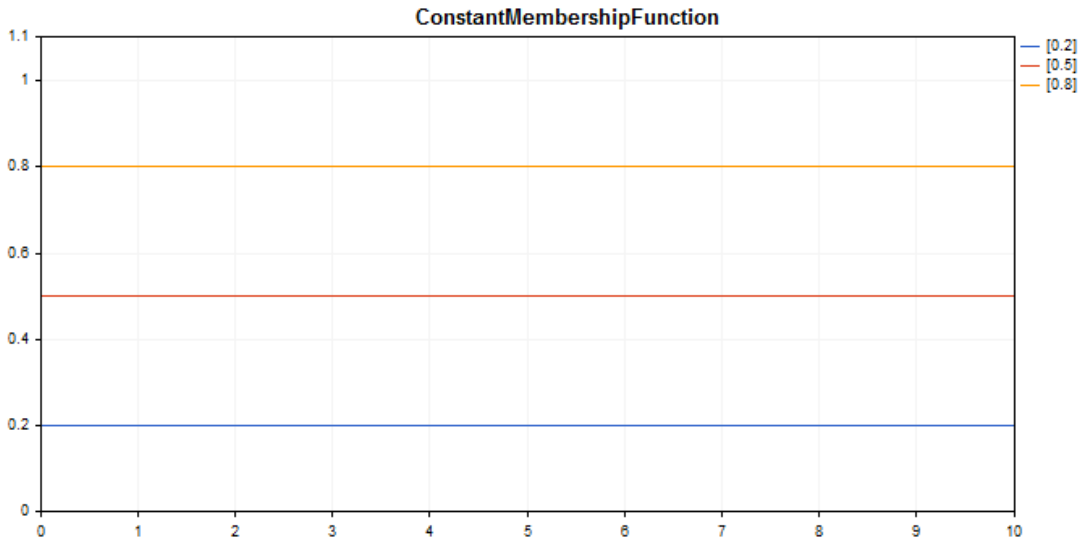
Klasse für die Implementierung der Zugehörigkeitsfunktion als eine Gerade parallel zur Koordinatenachse.

Beschreibung

Die Funktion wird durch die folgende Gleichung beschrieben:

y(x)=c

Dementsprechend ist der Zugehörigkeitsgrad für diese Funktion auf der ganzen Achse gleich und macht den Parameter aus, der im Konstruktor angegeben wurde.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CConstantMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject
IMembershipFunction
CConstantMembershipFunction

Methoden der Klasse

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                  ConstantMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CConstantMembershipFunction func1(0.2);
CConstantMembershipFunction func2(0.5);
CConstantMembershipFunction func3(0.8);
//--- Create wrappers for membership functions
double ConstantMembershipFunction1(double x) { return(func1.GetValue(x)); }
double ConstantMembershipFunction2(double x) { return(func2.GetValue(x)); }
double ConstantMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"ConstantMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"ConstantMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("ConstantMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(ConstantMembershipFunction1,0.0,10.0,1.0,CURVE_LINES,"[0.2]");
   graphic.CurveAdd(ConstantMembershipFunction2,0.0,10.0,1.0,CURVE_LINES,"[0.5]");
   graphic.CurveAdd(ConstantMembershipFunction3,0.0,10.0,1.0,CURVE_LINES,"[0.8]");
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const double  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

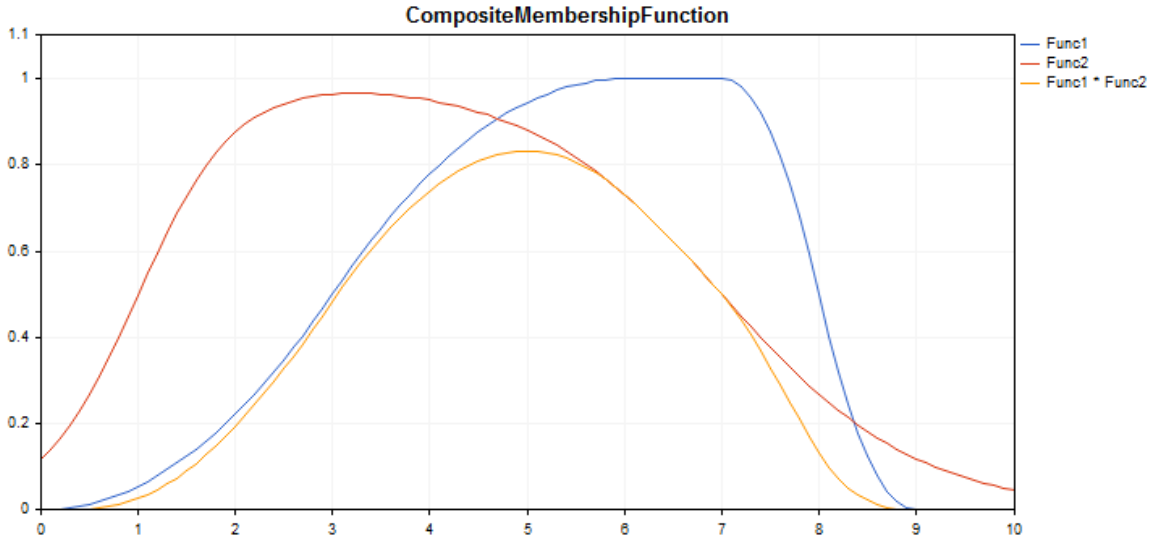
© 2000-2025, MetaQuotes Ltd.

CCompositeMembershipFunction

Klasse für die Implementierung einer Komposition von Zugehörigkeitsfunktionen.

Beschreibung

Die Komposition von Zugehörigkeitsfunktionen stellt eine Zusammensetzung aus einer oder mehrerer Zugehörigkeitsfunktionen mithilfe des angegebenen Operators dar.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CCompositeMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject
  IMembershipFunction
    CCompositeMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
CompositionType	Setzt den Operator der Komposition.
MembershipFunctions	Gibt die Liste der Zugehörigkeitsfunktionen zurück.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                  CompositeMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CProductTwoSigmoidalMembershipFunctions func1(2,1,-1,7);
CP_ShapedMembershipFunction func2(0,6,7,9);
CCompositeMembershipFunction composite(ProdMF,GetPointer(func1),GetPointer(func2));
//--- Create wrappers for membership functions
double ProductTwoSigmoidalMembershipFunctions(double x) { return(func1.GetValue(x)); }
double P_ShapedMembershipFunction(double x) { return(func2.GetValue(x)); }
double CompositeMembershipFunction(double x) { return(composite.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"CompositeMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"CompositeMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("CompositeMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(P_ShapedMembershipFunction,0.0,10.0,0.1,CURVE_LINES,"Func1");
   graphic.CurveAdd(ProductTwoSigmoidalMembershipFunctions,0.0,10.0,0.1,CURVE_LINES,"F
   graphic.CurveAdd(CompositeMembershipFunction,0.0,10.0,0.1,CURVE_LINES,"Func1 * Func
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

CompositionType

Setzt den Operator der Komposition.

```
void  CompositionType(
   MfCompositionType  value      // Typ des Operators
   )
```

Parameter

value

[in] Typ des Operators einer Komposition.

Hinweis

Es stehen die folgenden Typen von Operatoren zur Verfügung:

- MinMF (Minimum der Funktionen)
- MaxMF (Maximum der Funktionen)
- ProdMF (Produkt der Funktionen)
- SumMF (Summe der Funktionen)

MembershipFunctions

Gibt die Liste der Zugehörigkeitsfunktionen in der Komposition zurück.

```
CList*  MembershipFunctions(
   void        // Liste der Zugehörigkeitsfunktionen
   )
```

Rückgabewert

Liste der Zugehörigkeitsfunktionen.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

© 2000-2025, MetaQuotes Ltd.

Wert der Zugehörigkeitsfunktion.

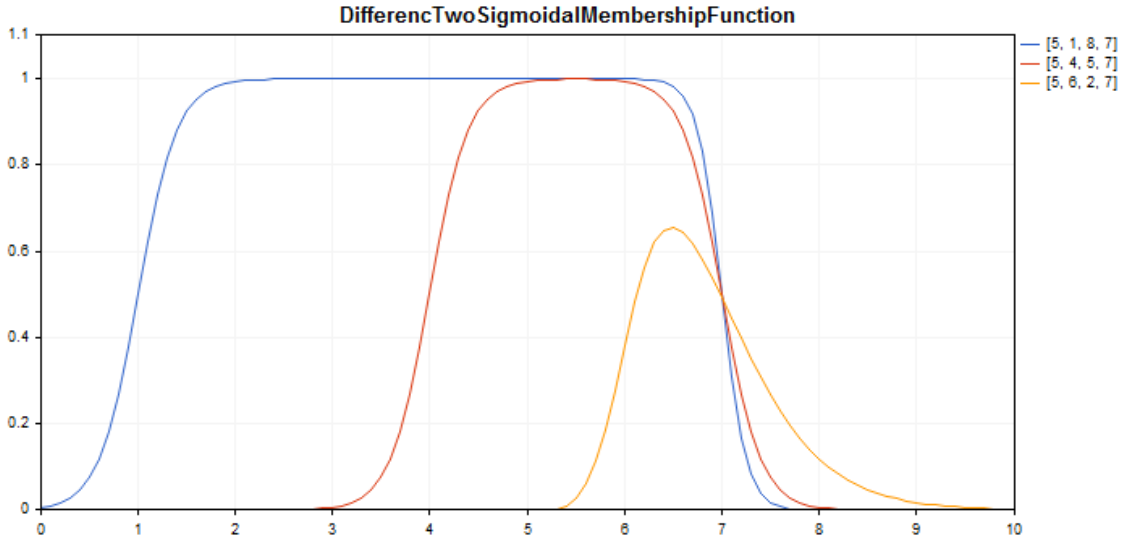
© 2000-2025, MetaQuotes Ltd.

CDifferencTwoSigmoidalMembershipFunction

Klasse für die Implementierung der Zugehörigkeitsfunktion als Differenz zwischen zwei Sigmoidfunktionen mit den Parametern A1, A2, C1 und C2.

Beschreibung

Die Funktion basiert auf einer Sigmoidkurve. Mit der Funktion kann man Zugehörigkeitsfunktionen mit den Werten erstellen, die gleich 1 sind, beginnend mit einem bestimmten Wert des Arguments. Solche Funktionen passen sehr gut, wenn man linguistische Begriffe wie "Short" oder "Long" angeben muss.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CDifferencTwoSigmoidalMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CDifferencTwoSigmoidalMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
A1	Liefert und setzt den Steigungskoeffizienten der ersten Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
A2	Liefert und setzt den Steigungskoeffizienten der zweiten Zugehörigkeitsfunktion.
C1	Liefert und setzt den Parameter des Wendepunktes der ersten Zugehörigkeitsfunktion.
C2	Liefert und setzt den Parameter des Wendepunktes der zweiten Zugehörigkeitsfunktion.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                      DifferencTwoSigmoidalMembershipFunction.mq5 |
//|                         Copyright 2000-2024, MetaQuotes Ltd.     |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CDifferencTwoSigmoidalMembershipFunction func1(5,1,8,7);
CDifferencTwoSigmoidalMembershipFunction func2(5,4,5,7);
CDifferencTwoSigmoidalMembershipFunction func3(5,6,2,7);
//--- Create wrappers for membership functions
double DifferencTwoSigmoidalMembershipFunction1(double x) { return(func1.GetValue(x));
double DifferencTwoSigmoidalMembershipFunction2(double x) { return(func2.GetValue(x));
double DifferencTwoSigmoidalMembershipFunction3(double x) { return(func3.GetValue(x));
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"DifferencTwoSigmoidalMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"DifferencTwoSigmoidalMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("DifferencTwoSigmoidalMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(DifferencTwoSigmoidalMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.CurveAdd(DifferencTwoSigmoidalMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,
   graphic.CurveAdd(DifferencTwoSigmoidalMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A1 (Get-Methode)

Gibt den Steigungskoeffizienten der ersten Zugehörigkeitsfunktion zurück.

```
double  A1()
```

Rückgabewert

Wert des Steigungskoeffizienten.

A1 (Set-Methode)

Setzt den Steigungskoeffizienten der ersten Zugehörigkeitsfunktion.

```
void  A1(
   const double  a1      // Wert des Steigungskoeffizienten
   )
```

Parameter

a1

[in] Wert des Steigungskoeffizienten.

A2 (Get-Methode)

Gibt den Steigungskoeffizienten der zweiten Zugehörigkeitsfunktion zurück.

```
double  A2()
```

Rückgabewert

Wert des Steigungskoeffizienten.

A2 (Set-Methode)

© 2000-2025, MetaQuotes Ltd.

Setzt den Steigungskoeffizienten der zweiten Zugehörigkeitsfunktion.

```
void  A2(
   const double  a2      // Wert des Steigungskoeffizienten
   )
```

Parameter

a2

[in] Wert des Steigungskoeffizienten.

C1 (Get-Methode)

Gibt den Parameter der Koordinate des Wendepunktes der ersten Zugehörigkeitsfunktion zurück.

```
double  C1()
```

Rückgabewert

Koordinate des Wendepunktes.

C1 (Set-Methode)

Liefert und setzt den Parameter des Wendepunktes der zweiten Zugehörigkeitsfunktion.

```
void  C1(
   const double  c1      // Koordinate des Wendepunktes
   )
```

Parameter

c1

[in] Koordinate des Wendepunktes.

C2 (Get-Methode)

Gibt den Parameter der Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion zurück.

```
double  C2()
```

Rückgabewert

Koordinate des Wendepunktes.

C2 (Set-Methode)

Setzt den Parameter der Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion

```
void  C2(
   const double  c2      // Koordinate des Wendepunktes
   )
```

© 2000-2025, MetaQuotes Ltd.

Parameter

c2

[in] Koordinate des Wendepunktes.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const double  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

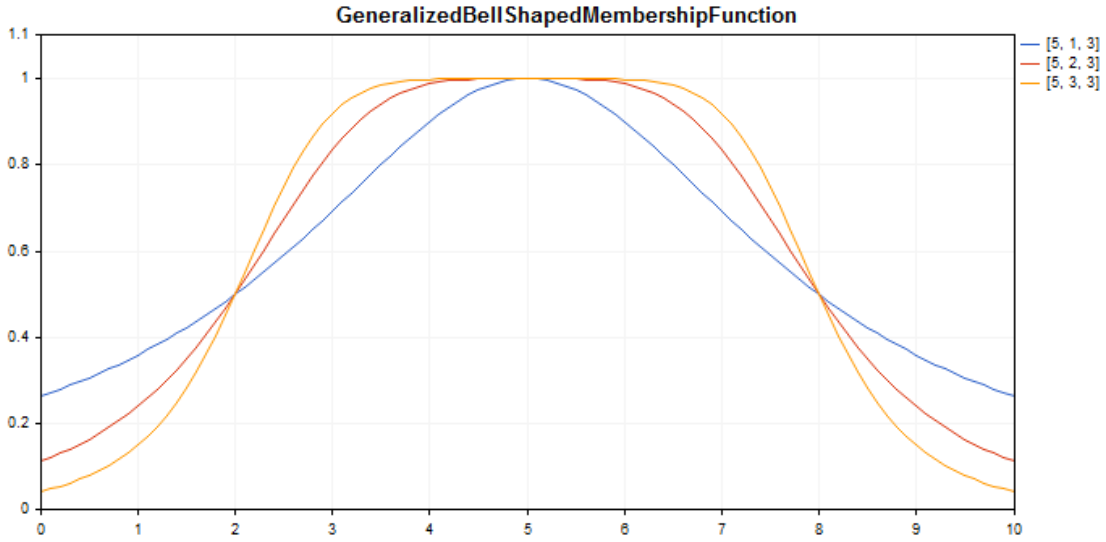
© 2000-2025, MetaQuotes Ltd.

CGeneralizedBellShapedMembershipFunction

Klasse für die Implementierung einer verallgemeinerten glockenförmigen Zugehörigkeitsfunktion mit den Parametern A, B und C.

Beschreibung

Die verallgemeinerte glockenförmige Zugehörigkeitsfunktion sieht der Form nach wie Gauß'sche Funktionen aus. Im ganzen Definitionsbereich ist die Funktion glatt und nimmt non zero Werte an.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CGeneralizedBellShapedMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CGeneralizedBellShapedMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
A	Liefert und setzt den Koeffizienten der Konzentration der Zugehörigkeitsfunktion.
B	Liefert und setzt den Steigungskoeffizienten der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
C	Liefert und setzt die Koordinate des Maximums der Zugehörigkeitsfunktion.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                      GeneralizedBellShapedMembershipFunction.mq5 |
//|                         Copyright 2000-2024, MetaQuotes Ltd.     |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CGeneralizedBellShapedMembershipFunction func1(5, 1, 3);
CGeneralizedBellShapedMembershipFunction func2(5, 2, 3);
CGeneralizedBellShapedMembershipFunction func3(5, 3, 3);
//--- Create wrappers for membership functions
double GeneralizedBellShapedMembershipFunction1(double x) { return(func1.GetValue(x));
double GeneralizedBellShapedMembershipFunction2(double x) { return(func2.GetValue(x));
double GeneralizedBellShapedMembershipFunction3(double x) { return(func3.GetValue(x));
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"GeneralizedBellShapedMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"GeneralizedBellShapedMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("GeneralizedBellShapedMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(GeneralizedBellShapedMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,
   graphic.CurveAdd(GeneralizedBellShapedMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,
   graphic.CurveAdd(GeneralizedBellShapedMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A (Get-Methode)

Gibt den Konzentrationkoeffizienten der Zugehörigkeitsfunktion zurück.

```
double  A()
```

Rückgabewert

Wert des Konzentrationkoeffizienten.

A (Set-Methode)

Setzt den Konzentrationkoeffizienten der Zugehörigkeitsfunktion.

```
void  A(
   const double  a      // Konzentrationkoeffizient
   )
```

Parameter

a

[in] Konzentrationkoeffizient der Zugehörigkeitsfunktion.

B (Get-Methode)

Gibt den Steigungskoeffizienten der Zugehörigkeitsfunktion zurück.

```
double  B()
```

Rückgabewert

Wert des Steigungskoeffizienten.

B (Set-Methode)

Setzt den Steigungskoeffizienten der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

```
void  B(
   const double  b      // Steigungskoeffizient
   )
```

Parameter

b

[in] Steigungskoeffizient der Zugehörigkeitsfunktion.

C (Get-Methode)

Gibt die Koordinate des Maximums der Zugehörigkeitsfunktion zurück.

```
double  C()
```

Rückgabewert

Koordinate des Maximums der Zugehörigkeitsfunktion.

C (Set-Methode)

Setzt die Koordinate des Maximums der Zugehörigkeitsfunktion.

```
void  C(
   const double  c      // Wert der Koordinate des Maximums
   )
```

Parameter

c

[in] Koordinate des Maximums der Zugehörigkeitsfunktion.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

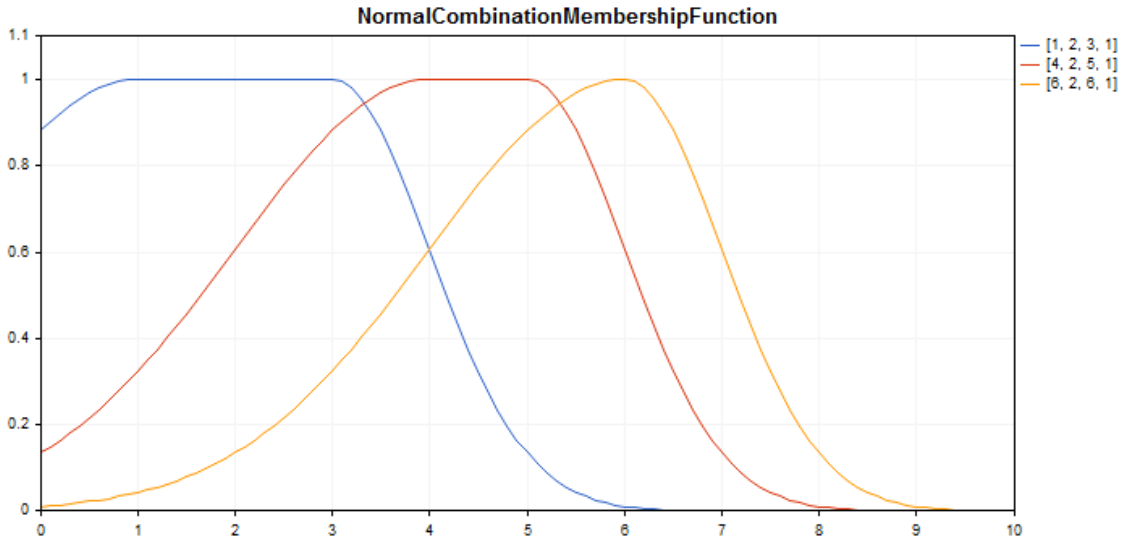
© 2000-2025, MetaQuotes Ltd.

CNormalCombinationMembershipFunction

Klasse für die Implementierung einer doppelseitigen Gauß'schen Zugehörigkeitsfunktion mit den Parametern B1, B2, Sigma1 und Sigma2.

Beschreibung

Eine doppelseitige Gauß'sche Zugehörigkeitsfunktion wird unter Verwendung der Gauß-Verteilung gebildet. Sie ermöglicht es, asymmetrische Zugehörigkeitsfunktionen zu setzen. Im ganzen Definitionsbereich ist die Funktion glatt und nimmt non zero Werte an.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CNormalCombinationMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CNormalCombinationMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
B1	Liefert und setzt den Wert des ersten Zentrums der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
B2	Liefert und setzt den Wert des zweiten Zentrums der Zugehörigkeitsfunktion.
Sigma1	Liefert und setzt den Wert des ersten Parameter der Krümmung der Zugehörigkeitsfunktion.
Sigma2	Liefert und setzt den Wert des zweiten Parameter der Krümmung der Zugehörigkeitsfunktion.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                          NormalCombinationMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CNormalCombinationMembershipFunction func1(1,2,3,1);
CNormalCombinationMembershipFunction func2(4,2,5,1);
CNormalCombinationMembershipFunction func3(6,2,6,1);
//--- Create wrappers for membership functions
double NormalCombinationMembershipFunction1(double x) { return(func1.GetValue(x)); }
double NormalCombinationMembershipFunction2(double x) { return(func2.GetValue(x)); }
double NormalCombinationMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"NormalCombinationMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"NormalCombinationMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("NormalCombinationMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(NormalCombinationMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[1,
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.CurveAdd(NormalCombinationMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[4,
   graphic.CurveAdd(NormalCombinationMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[6,
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

B1 (Get-Methode)

Gibt den Wert des ersten Zentrums der Zugehörigkeitsfunktion zurück.

```
double  B1()
```

Rückgabewert

Wert des ersten Zentrums der Zugehörigkeitsfunktion.

B1 (Set-Methode)

Setzt den Wert des ersten Zentrums der Zugehörigkeitsfunktion.

```
void  B1(
   const double  b1      // Wert des ersten Zentrums
   )
```

Parameter

b

[in] Wert des ersten Zentrums der Zugehörigkeitsfunktion.

B2 (Get-Methode)

Gibt den Wert des zweiten Zentrums der Zugehörigkeitsfunktion zurück.

```
double  B2()
```

Rückgabewert

Wert des zweiten Zentrums der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

B2 (Set-Methode)

Setzt den Wert des zweiten Zentrums der Zugehörigkeitsfunktion.

```
void  B2(
   const double  b2      // Wert des zweiten Zentrums
   )
```

Parameter

b2

[in] Wert des zweiten Zentrums der Zugehörigkeitsfunktion.

Sigma1 (Get-Methode)

Gibt den ersten Krümmungsparameter der Zugehörigkeitsfunktion zurück.

```
double  Sigma1()
```

Rückgabewert

Wert des ersten Krümmungsparameters der Zugehörigkeitsfunktion.

Sigma1 (Set-Methode)

Setzt den Wert des ersten Krümmungsparameters der Zugehörigkeitsfunktion.

```
void  Sigma1(
   const double  sigma1      // Wert des ersten Krümmungsparameters
   )
```

Parameter

sigma1

[in] Erster Krümmungsparameter der Zugehörigkeitsfunktion.

Sigma2 (Get-Methode)

Gibt den zweiten Krümmungsparameter der Zugehörigkeitsfunktion zurück.

```
double  Sigma2()
```

Rückgabewert

Wert des zweiten Krümmungsparameters der Zugehörigkeitsfunktion.

Sigma2 (Set-Methode)

© 2000-2025, MetaQuotes Ltd.

Setzt den Wert des zweiten Krümmungsparameters der Zugehörigkeitsfunktion.

```
void  Sigma2(
   const double  sigma2      // Wert des zweiten Krümmungsparameters
   )
```

Parameter

sigma2

[in] Zweiter Krümmungsparameter der Zugehörigkeitsfunktion.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

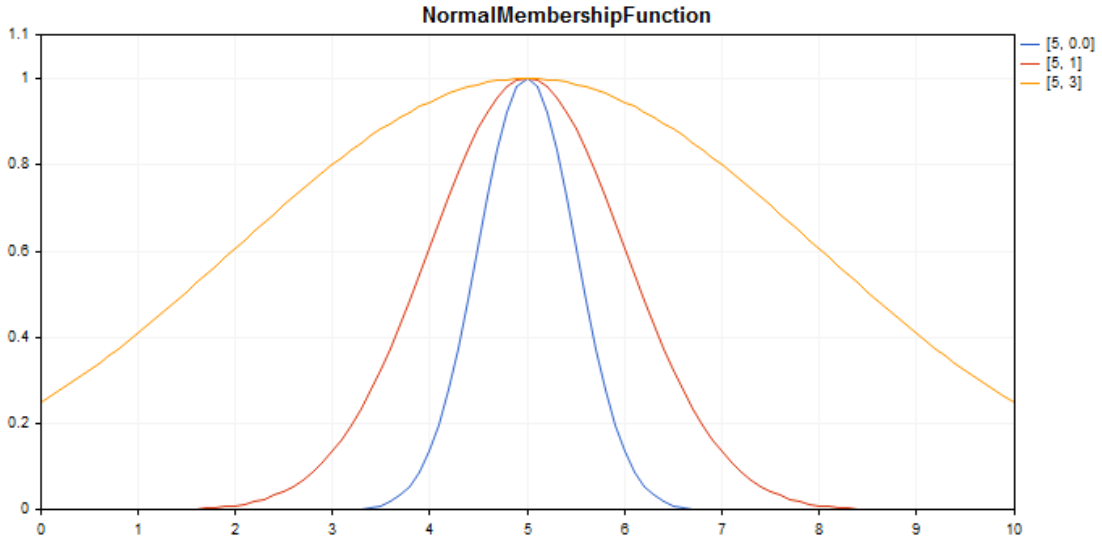
© 2000-2025, MetaQuotes Ltd.

CNormalMembershipFunction

Klasse für die Implementierung einer symmetrischen Gauß'schen Zugehörigkeitsfunktion mit den Parametern B und Sigma.

Beschreibung

Eine symmetrische Gauß'sche Zugehörigkeitsfunktion wird mithilfe der Gaußschen Verteilung gebildet. Im ganzen Definitionsbereich ist die Funktion glatt und nimmt non zero Werte an.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CNormalMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject
  IMembershipFunction
    CNormalMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
B	Liefert und setzt das Zentrum der Zugehörigkeitsfunktion.
Sigma	Liefert und setzt den Parameter der Steigung der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                     NormalMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CNormalMembershipFunction func1(5,0.5);
CNormalMembershipFunction func2(5,1);
CNormalMembershipFunction func3(5,3);
//--- Create wrappers for membership functions
double NormalMembershipFunction1(double x) { return(func1.GetValue(x)); }
double NormalMembershipFunction2(double x) { return(func2.GetValue(x)); }
double NormalMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"NormalMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"NormalMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("NormalMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(NormalMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[5, 0.0]");
   graphic.CurveAdd(NormalMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[5, 1]");
   graphic.CurveAdd(NormalMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[5, 3]");
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

B (Get-Methode)

Gibt das Zentrum der Zugehörigkeitsfunktion zurück.

```
double  B()
```

Rückgabewert

Wert des Zentrums der Zugehörigkeitsfunktion.

B (Set-Methode)

Setzt den Wert des Zentrums der Zugehörigkeitsfunktion.

```
void  B(
   const double  b      // Wert des Zentrums der Zugehörigkeitsfunktion
   )
```

Parameter

b

[in] Wert des Zentrums der Zugehörigkeitsfunktion.

Sigma (Get-Methode)

Gibt den Krümmungsparameter der Zugehörigkeitsfunktion zurück

```
double  Sigma()
```

Rückgabewert

Krümmungsparameter der Zugehörigkeitsfunktion

Sigma (Set-Methode)

Setzt den Wert des Krümmungsparameters der Zugehörigkeitsfunktion.

```
void  Sigma(
   const double  sigma      // Wert des Krümmungsparameters
   )
```

Parameter

sigma

© 2000-2025, MetaQuotes Ltd.

[in] Krümmungsparameter der Zugehörigkeitsfunktion.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const double  x      // Argument
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

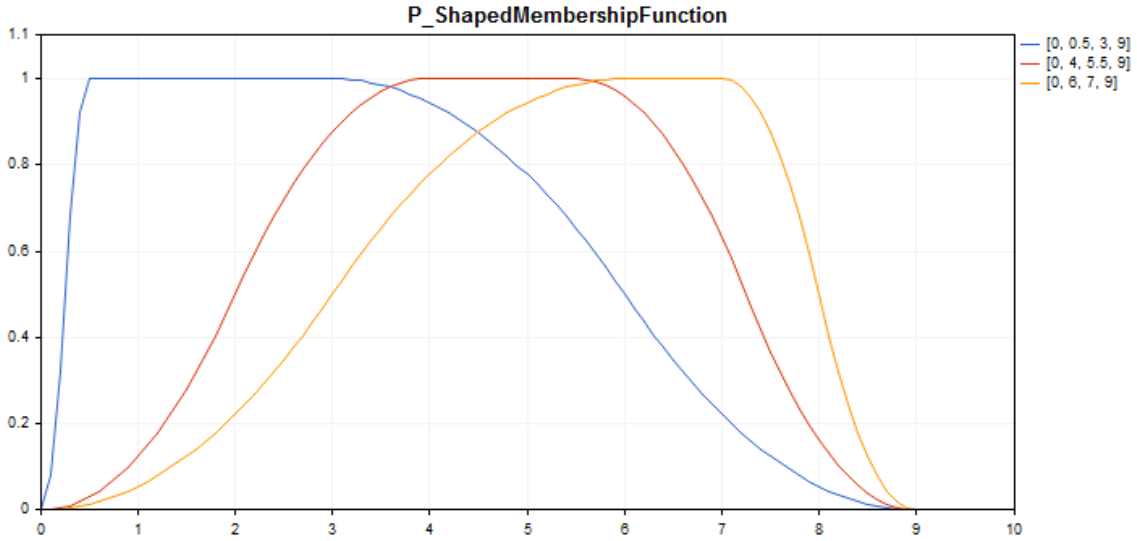
© 2000-2025, MetaQuotes Ltd.

CP_ShapedMembershipFunction

Klasse für die Implementierung einer P-förmigen Zugehörigkeitsfunktion mit den Parametern A, B, C und D.

Beschreibung

Die P-förmige Zugehörigkeitsfunktion stellt ein krummliniges Trapez dar. Die Funktion wird für das Setzen asymmetrischer Zugehörigkeitsfunktionen mit einem reibungslosen Übergang von einer pessimistischen zu einer optimistischen Schätzung einer Fuzzy-Zahl verwendet.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CP_ShapedMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CP_ShapedMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
A	Liefert und setzt den Parameter des Anfangs einer Fuzzy-Menge.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
B	Liefert und setzt den ersten Parameter des Kernes einer Fuzzy-Menge.
C	Liefert und setzt den zweiten Parameter des Kernes einer Fuzzy-Menge.
D	Liefert und setzt den Parameter des Endes einer Fuzzy-Menge.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                  P_ShapedMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CP_ShapedMembershipFunction func1(0,0.5,3,9);
CP_ShapedMembershipFunction func2(0,4,5.5,9);
CP_ShapedMembershipFunction func3(0,6,7,9);
//--- Create wrappers for membership functions
double P_ShapedMembershipFunction1(double x) { return(func1.GetValue(x)); }
double P_ShapedMembershipFunction2(double x) { return(func2.GetValue(x)); }
double P_ShapedMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"P_ShapedMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"P_ShapedMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("P_ShapedMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(P_ShapedMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[0, 0.5, 3,
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.CurveAdd(P_ShapedMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[0, 4, 5.5,
   graphic.CurveAdd(P_ShapedMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[0, 6, 7, 9]
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A (Get-Methode)

Gibt den Parameter des Anfangs einer Fuzzy-Menge zurück.

```
double  A()
```

Rückgabewert

Parameter des Anfangs einer Fuzzy-Menge.

A (Set-Methode)

Setzt den Parameter des Anfangs einer Fuzzy-Menge.

```
void  A(
   const double  a      // Parameter des Anfangs einer Fuzzy-Menge
   )
```

Parameter

a

[in] Parameter des Anfangs einer Fuzzy-Menge.

B (Get-Methode)

Gibt den ersten Parameter des Kernes einer Fuzzy-Menge zurück.

```
double  B()
```

Rückgabewert

Erster Parameter des Kernes einer Fuzzy-Menge.

B (Set-Methode)

© 2000-2025, MetaQuotes Ltd.

Setzt den ersten Parameter des Kernes einer Fuzzy-Menge.

```
void  B(
   const double  b      // Wert des ersten Parameters des Kernes der Fuzzy-Menge
   )
```

Parameter

b

[in] Erster Parameter einer Fuzzy-Menge.

C (Get-Methode)

Gibt den zweiten Parameter des Kernes einer Fuzzy-Menge zurück.

```
double  C()
```

Rückgabewert

Zweiter Parameter des Kernes einer Fuzzy-Menge.

C (Set-Methode)

Setzt den zweiten Parameter des Kernes einer Fuzzy-Menge.

```
void  C(
   const double  c      // Wert des zweiten Parameters des Kernes einer Fuzzy-Menge
   )
```

Parameter

c

[in] Zweiter Parameter des Kernes einer Fuzzy-Menge.

D (Get-Methode)

Gibt den Parameter des Endes einer Fuzzy-Menge zurück.

```
double  D()
```

Rückgabewert

Wert des Parameter des Endes einer Fuzzy-Menge.

D (Set-Methode)

Setzt den Parameter des Endes einer Fuzzy-Menge.

```
void  D(
   const double  d      // Wert des Parameters des Endes eines Fuzzy-Menge
   )
```

Parameter

d

© 2000-2025, MetaQuotes Ltd.

[in] Wert des Parameters des Endes einer Fuzzy-Menge.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument

```
double  GetValue(
   const double  x
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion

Rückgabewert

Wert der Zugehörigkeitsfunktion

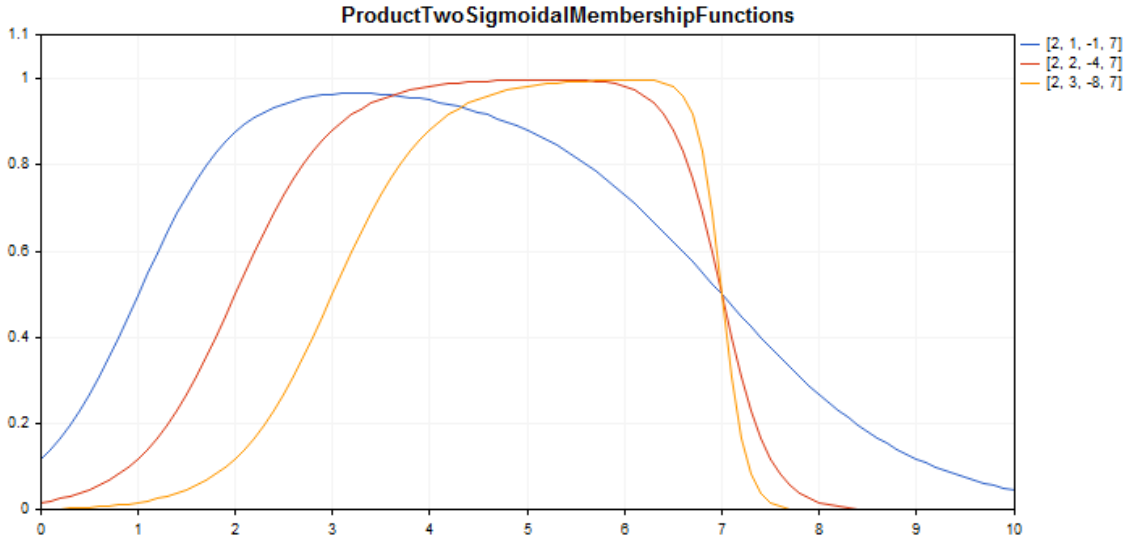
© 2000-2025, MetaQuotes Ltd.

CProductTwoSigmoidalMembershipFunction

Klasse für die Implementierung der Zugehörigkeitsfunktion als Produkt zweier Sigmoidfunktionen mit den Parametern A1, A2, C1 und C2.

Beschreibung

Das Produkt zweier Sigmoidfunktion wird für die Setzung glatter asymmetrischer Funktionen verwendet. Mit dem Produkt kann man Zugehörigkeitsfunktionen mit den Werten erstellen, die gleich 1 sind, beginnend mit einem bestimmten Wert des Arguments. Solche Funktionen passen sehr gut, wenn man linguistische Begriffe wie "Short" oder "Long" angeben muss.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CProductTwoSigmoidalMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CProductTwoSigmoidalMembershipFunctions

Methoden der Klasse

Methode der Klasse	Beschreibung
A1	Liefert und setzt den Steigungskoeffizienten der ersten Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
A2	Liefert und setzt den Steigungskoeffizienten der zweiten Zugehörigkeitsfunktion.
C1	Gibt den Parameter der Koordinate des Wendepunktes der ersten Zugehörigkeitsfunktion zurück.
C2	Gibt den Parameter der Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion zurück.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                      ProductTwoSigmoidalMembershipFunctions.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CProductTwoSigmoidalMembershipFunctions func1(2,1,-1,7);
CProductTwoSigmoidalMembershipFunctions func2(2,2,-4,7);
CProductTwoSigmoidalMembershipFunctions func3(2,3,-8,7);
//--- Create wrappers for membership functions
double ProductTwoSigmoidalMembershipFunctions1(double x) { return(func1.GetValue(x));
double ProductTwoSigmoidalMembershipFunctions2(double x) { return(func2.GetValue(x));
double ProductTwoSigmoidalMembershipFunctions3(double x) { return(func3.GetValue(x));
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"ProductTwoSigmoidalMembershipFunctions",0,30,30,780,380))
     {
      graphic.Attach(0,"ProductTwoSigmoidalMembershipFunctions");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("ProductTwoSigmoidalMembershipFunctions");
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(ProductTwoSigmoidalMembershipFunctions1,0.0,10.0,0.1,CURVE_LINES,'
   graphic.CurveAdd(ProductTwoSigmoidalMembershipFunctions2,0.0,10.0,0.1,CURVE_LINES,'
   graphic.CurveAdd(ProductTwoSigmoidalMembershipFunctions3,0.0,10.0,0.1,CURVE_LINES,'
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A1 (Get-Methode)

Gibt den Steigungskoeffizienten der ersten Zugehörigkeitsfunktion zurück.

```
double  A1()
```

Rückgabewert

Steigungskoeffizient der ersten Zugehörigkeitsfunktion.

A1 (Set-Methode)

Setzt den Steigungskoeffizienten der ersten Zugehörigkeitsfunktion.

```
void  A1(
   const double  a1      // Steigungskoeffizient der ersten Zugehörigkeitsfunktion
   )
```

Parameter

a1

[in] Steigungskoeffizient der ersten Zugehörigkeitsfunktion.

A2 (Get-Methode)

Gibt den Steigungskoeffizienten der zweiten Zugehörigkeitsfunktion zurück.

```
double  A2()
```

Rückgabewert

Steigungskoeffizient der zweiten Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

A2 (Set-Methode)

Setzt den Steigungskoeffizienten der zweiten Zugehörigkeitsfunktion.

```
void  A2(
   const double  a2      // Steigungskoeffizient der zweiten Zugehörigkeitsfunktion
   )
```

Parameter

a2

[in] Steigungskoeffizient der zweiten Zugehörigkeitsfunktion.

C1 (Get-Methode)

Gibt den Parameter der Koordinate des Wendepunktes der ersten Zugehörigkeitsfunktion zurück.

```
double  C1()
```

Rückgabewert

Koordinate des Wendepunktes der ersten Zugehörigkeitsfunktion.

C1 (Set-Methode)

Setzt die Koordinate des Wendepunktes der ersten Zugehörigkeitsfunktion.

```
void  C1(
   const double  c1      // Koordinate des Wendepunktes der ersten Zugehörigkeitsfunkt
   )
```

Parameter

c1

[in] Koordinate des Wendepunktes der ersten Zugehörigkeitsfunktion.

C2 (Метод Get)

Gibt den Parameter der Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion zurück.

```
double  C2()
```

Rückgabewert

Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion.

C2 (Set-Methode)

Setzt die Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion.

```
void  C2(
   const double  c2      // Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunk
   )
```

© 2000-2025, MetaQuotes Ltd.

Parameter

c2

[in] Koordinate des Wendepunktes der zweiten Zugehörigkeitsfunktion.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

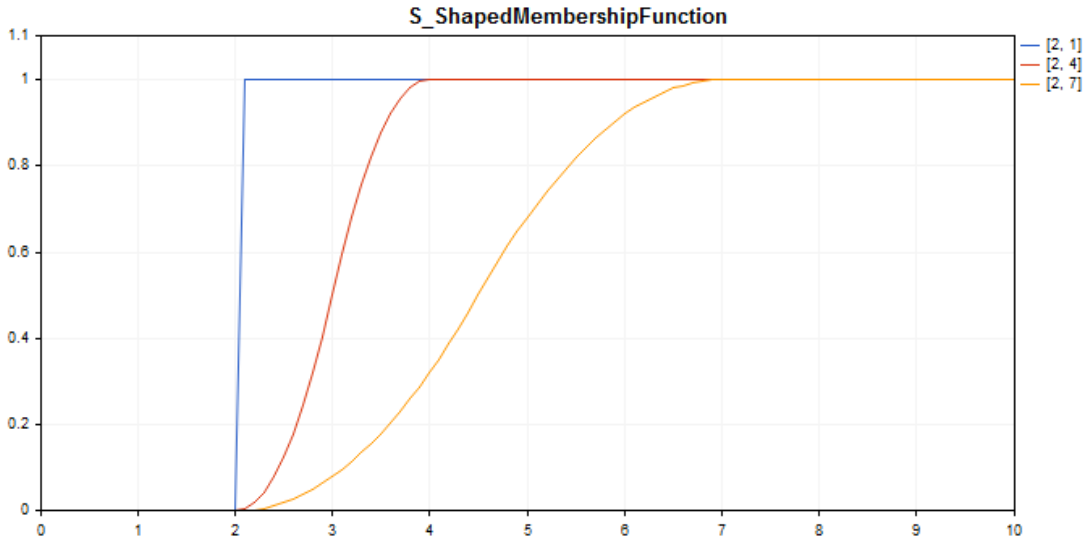
CS_ShapedMembershipFunction

Klasse für die Implementierung einer S-förmigen Zugehörigkeitsfunktion mit den Parametern A und B.

Beschreibung

Die Funktion gibt eine S-förmige zweiparametrige Zugehörigkeitsfunktion. Das ist eine steigende Funktion, welche Werte von 0 bis 1 annimmt. Die Parameter A und B bestimmen das Intervall, innerhalb dessen die Funktion nach der nicht linearen Bahn von 0 bis 1 steigt.

Diese Funktion stellt Fuzzy-Mengen vom Typ "sehr hoch" dar (d.h., es werden steigende Zugehörigkeitsfunktionen mit Sättigung).



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CS_ShapedMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject
  IMembershipFunction
    CS_ShapedMembershipFunction

Methoden der Klasse

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
A	Liefert und setzt den Parameter des Anfangs des Intervalls, in welchem die Funktion steigt.
B	Liefert und setzt den ersten Parameter des Kernes einer Fuzzy-Menge.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                   S_ShapedMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CS_ShapedMembershipFunction func1(2,1);
CS_ShapedMembershipFunction func2(2,4);
CS_ShapedMembershipFunction func3(2,7);
//--- Create wrappers for membership functions
double S_ShapedMembershipFunction1(double x) { return(func1.GetValue(x)); }
double S_ShapedMembershipFunction2(double x) { return(func2.GetValue(x)); }
double S_ShapedMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"S_ShapedMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"S_ShapedMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("S_ShapedMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(S_ShapedMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[2, 1]");
   graphic.CurveAdd(S_ShapedMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[2, 4]");
   graphic.CurveAdd(S_ShapedMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[2, 7]");
//--- sets the X-axis properties
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A (Get-Methode)

Gibt den Parameter des Anfangs des Intervalls zurück, in welchem die Funktion steigt.

```
double  A()
```

Rückgabewert

Parameter des Anfangs des Intervalls, in welchem die Funktion steigt.

A (Set-Methode)

Setzt den Parameter des Anfangs des Intervalls, in welchem die Funktion steigt.

```
void  A(
   const double  a      // Parameter des Anfangs des Intervalls, in welchem die Funkti
   )
```

Parameter

a

[in] Parameter des Anfangs des Intervalls, in welchem die Funktion steigt.

B (Get-Methode)

Gibt den ersten Parameter des Kernes einer Fuzzy-Menge zurück.

```
double  B()
```

Rückgabewert

Erster Parameter des Kernes einer Fuzzy-Menge.

B (Set-Methode)

Setzt den ersten Parameter des Kernes einer Fuzzy-Menge.

```
void  B(
   const double  b      // der erste Parameter des Kernes einer Fuzzy-Menge
```

© 2000-2025, MetaQuotes Ltd.

```
   )
```

Parameter

b

[in] Erster Parameter einer Fuzzy-Menge.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

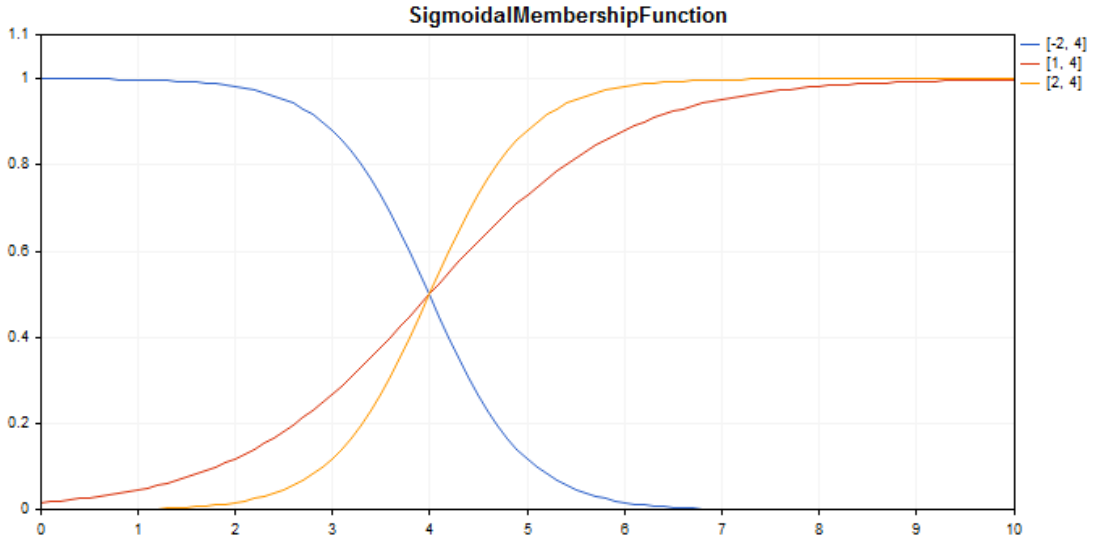
© 2000-2025, MetaQuotes Ltd.

CSigmoidalMembershipFunction

Klasse für die Implementierung einer sigmoiden Zugehörigkeitsfunktion mit den Parametern A und C.

Beschreibung

Eine Sigmoidfunktion wird für das Setzen monotoner Zugehörigkeitsfunktionen verwendet. Mit der Funktion kann man Zugehörigkeitsfunktionen mit den Werten erstellen, die gleich 1 sind, beginnend mit einem bestimmten Wert des Arguments. Solche Funktionen passen sehr gut, wenn man linguistische Begriffe wie "Short" oder "Long" angeben muss.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CSigmoidalMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CSigmoidalMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
A	Liefert und setzt den Steigungskoeffizienten der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
C	Liefert und setzt den Parameter der Koordinate des Wendepunktes der Zugehörigkeitsfunktion.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                  SigmoidalMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CSigmoidalMembershipFunction func1(-2, 4);
CSigmoidalMembershipFunction func2(1, 4);
CSigmoidalMembershipFunction func3(2, 4);
//--- Create wrappers for membership functions
double SigmoidalMembershipFunction1(double x) { return(func1.GetValue(x)); }
double SigmoidalMembershipFunction2(double x) { return(func2.GetValue(x)); }
double SigmoidalMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"SigmoidalMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"SigmoidalMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("SigmoidalMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(SigmoidalMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[-2, 4]");
   graphic.CurveAdd(SigmoidalMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[1, 4]");
   graphic.CurveAdd(SigmoidalMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[2, 4]");
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A (Get-Methode)

Gibt den Steigungskoeffizienten der Zugehörigkeitsfunktion zurück.

```
double  A()
```

Rückgabewert

Steigungskoeffizient der Zugehörigkeitsfunktion.

A (Set-Methode)

Setzt den Steigungskoeffizienten der Zugehörigkeitsfunktion.

```
void  A(
   const double  a      // Steigungskoeffizient der Zugehörigkeitsfunktion
   )
```

Parameter

a

[in] Steigungskoeffizient der Zugehörigkeitsfunktion.

C (Get-Methode)

Gibt den Parameter der Koordinate des Wendepunktes der Zugehörigkeitsfunktion zurück.

```
double  C()
```

Rückgabewert

Koordinate des Wendepunktes der Zigehörigkeitsfunktion.

C (Set-Methode)

Setzt die Koordinate des Wendepunktes der Zigehörigkeitsfunktion.

```
void  C(
   const double  c      // Koordinate des Wendepunktes der Zugehörigkeitsfunktion
   )
```

© 2000-2025, MetaQuotes Ltd.

Parameter

c

[in] Koordinate des Wendepunktes der Zigehörigkeitsfunktion.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

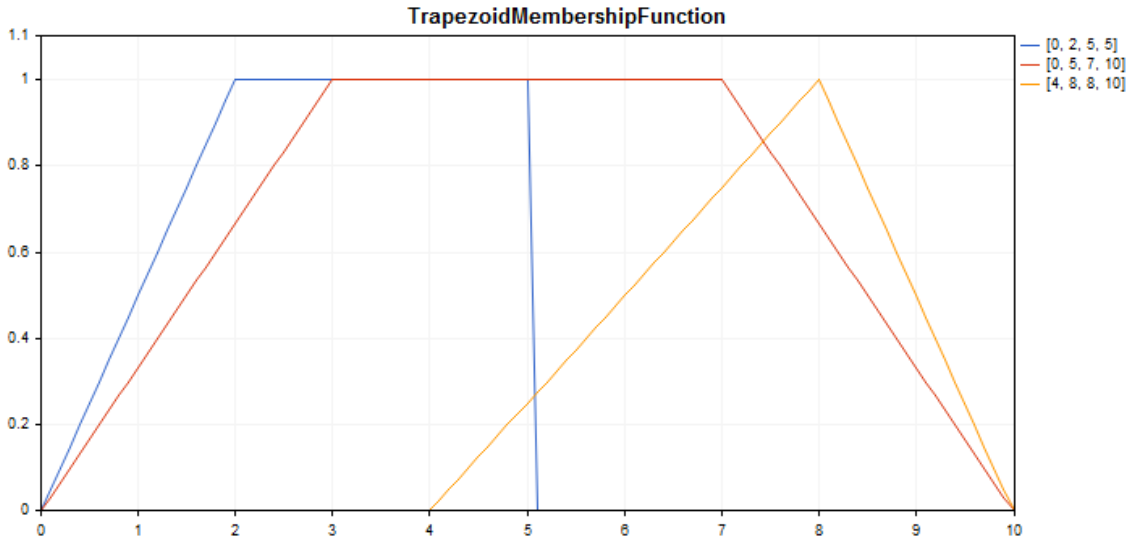
CTrapezoidMembershipFunction

Klasse für die Implementierung einer trapezförmigen Zugehörigkeitsfunktion mit den Parametern X1, X2, X3 und X4.

Beschreibung

Die Funktion wird unter Verwendung der stückweise linearen Approximation gebildet. Das ist eine Verallgemeinerung der Dreiecksfunktion, sie erlaubt es, den Kern einer Fuzzy-Menge als Intervall zu setzen. Eine solche Zugehörigkeitsfunktion ermöglicht die Interpretation einer optimistischen/pessimistischen Schätzung.

Die Funktion wird für das Setzen asymmetrischer Zugehörigkeitsfunktionen der Variablen verwendet, deren besonders mögliche Werte innerhalb eines bestimmten Bereichs definiert werden.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CTrapezoidMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

CTrapezoidMembershipFunction

Methoden der Klasse

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
X1	Liefert und setzt den Wert des ersten Punktes auf der X-Achse.
X2	Liefert und setzt den Wert des zweiten Punktes auf der X-Achse.
X3	Liefert und setzt den Wert des dritten Punktes auf der X-Achse.
X4	Liefert und setzt den Wert des vierten Puntkes auf der X-Achse.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                  TrapezoidMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CTrapezoidMembershipFunction func1(0,2,5,5);
CTrapezoidMembershipFunction func2(0,3,7,10);
CTrapezoidMembershipFunction func3(4,8,8,10);
//--- Create wrappers for membership functions
double TrapezoidMembershipFunction1(double x) { return(func1.GetValue(x)); }
double TrapezoidMembershipFunction2(double x) { return(func2.GetValue(x)); }
double TrapezoidMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"TrapezoidMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"TrapezoidMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("TrapezoidMembershipFunction");
   graphic.BackgroundMainSize(16);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- create curve
   graphic.CurveAdd(TrapezoidMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[0, 2, 5, 5
   graphic.CurveAdd(TrapezoidMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[0, 5, 7, 1
   graphic.CurveAdd(TrapezoidMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[4, 8, 8, 1
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

X1 (Get-Methode)

Gibt den Wert des ersten Punktes auf der X-Achse zurück.

```
double  X1()
```

Rückgabewert

Wert des ersten Punktes auf der X-Achse.

X1 (Set-Methode)

Setzt den Wert des ersten Punktes auf der X-Achse.

```
void  X1(
   const double  x1      // Wert des ersten Punktes auf der X-Achse
   )
```

Parameter

x1

[in] Wert des ersten Punktes auf der X-Achse.

X2 (Get-Methode)

Gibt den Wert des zweiten Punktes auf der X-Achse zurück.

```
double  X2()
```

Rückgabewert

Wert des zweiten Punktes auf der X-Achse.

© 2000-2025, MetaQuotes Ltd.

X2 (Set-Methode)

Setzt den Wert des zweiten Punktes auf der X-Achse.

```
void  X2(
   const double  x2      // Wert des zweiten Punktes auf der X-Achse.
   )
```

Parameter

x2

[in] Wert des zweiten Punktes auf der X-Achse.

X3 (Get-Methode)

Gibt den Wert des dritten Punktes auf der X-Achse zurück.

```
double  X3()
```

Rückgabewert

Wert des dritten Punktes auf der X-Achse.

X3 (Set-Methode)

Setzt den Wert des ersten Punktes auf der X-Achse.

```
void  X3(
   const double  x3      // Wert des ersten Punktes auf der X-Achse
   )
```

Parameter

x3

[in] Wert des ersten Punktes auf der X-Achse.

X4 (Get-Methode)

Gibt den Wert des vierten Punktes auf der X-Achse.

```
double  X4()
```

Rückgabewert

Wert des vierten Punktes auf der X-Achse.

X4 (Set-Methode)

Setzt den Wert des vierten Punktes auf der X-Achse.

```
void  X4(
   const double  x4      // Wert des vierten Punktes auf der X-Achse
   )
```

© 2000-2025, MetaQuotes Ltd.

Parameter

x4

[in] Wert des vierten Punktes auf der X-Achse.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

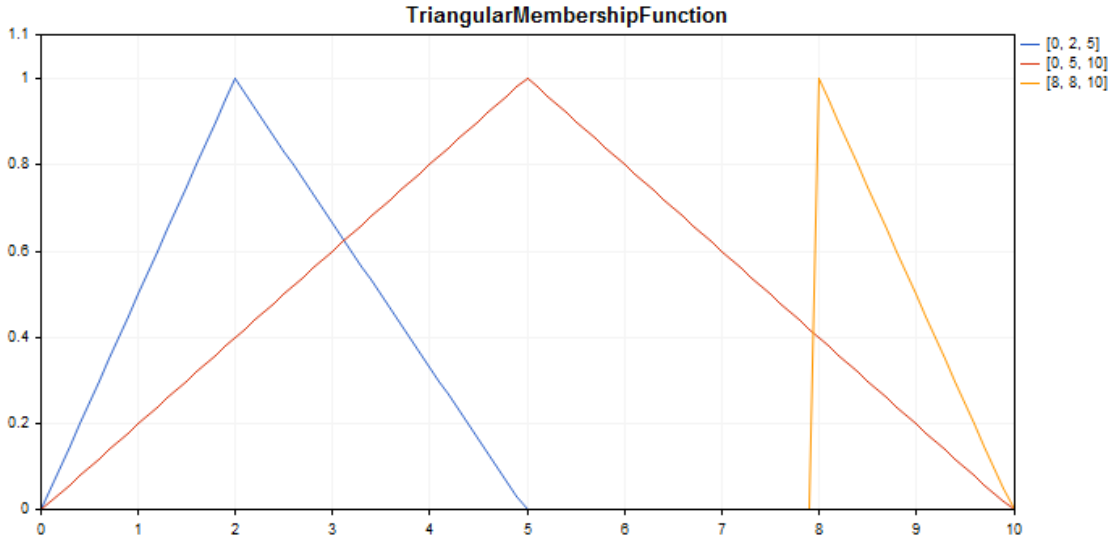
© 2000-2025, MetaQuotes Ltd.

CTriangularMembershipFunction

Klasse für die Implementierung einer Dreicksfunktion mit den Parametern X1, X2 und X3.

Beschreibung

Die Funktion setzt die Zugehörigkeitsfunktion in Form eines Dreiecks. Eine einfache und meist verwendete Zugehörigkeitsfunktion.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CTriangularMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject
  IMembershipFunction
    CTriangularMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
X1	Gibt den Wert des ersten Punktes auf der X-Achse zurück.
X2	Gibt den Wert des zweiten Punktes auf der X-Achse zurück.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
X3	Gibt den Wert des dritten Punktes auf der X-Achse zurück.
ToNormalMF	Wandelt eine Dreiecksfunktion in die Gauß'sche Zugehörigkeitsfunktion um.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                TriangularMembershipFunction.mq5 |
//|                        Copyright 2016, MetaQuotes Software Corp. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2000-2024, MetaQuotes Ltd."
#property link      "https://www.mql5.com"
#property version   "1.00"
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CTriangularMembershipFunction func1(0,2,5);
CTriangularMembershipFunction func2(0,5,10);
CTriangularMembershipFunction func3(8,8,10);
//--- Create wrappers for membership functions
double TriangularMembershipFunction1(double x) { return(func1.GetValue(x)); }
double TriangularMembershipFunction2(double x) { return(func2.GetValue(x)); }
double TriangularMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"TriangularMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"TriangularMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("TriangularMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(TriangularMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[0, 2, 5]"
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.CurveAdd(TriangularMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[0, 5, 10]
   graphic.CurveAdd(TriangularMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[8, 8, 10]
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

X1 (Get-Methode)

Gibt den Wert des ersten Punktes auf der X-Achse zurück.

```
double  X1()
```

Rückgabewert

Wert des ersten Punktes auf der X-Achse.

X1 (Set-Methode)

Setzt den Wert des ersten Punktes auf der X-Achse.

```
void  X1(
   const double  x1      // Wert des ersten Punktes auf der X-Achse
   )
```

Parameter

x1

[in] Wert des ersten Punktes auf der X-Achse.

X2 (Get-Methode)

Gibt den Wert des zweiten Punktes auf der X-Achse zurück.

```
double  X2()
```

Rückgabewert

Wert des zweiten Punktes auf der X-Achse.

X2 (Set-Methode)

© 2000-2025, MetaQuotes Ltd.

Setzt den Wert des zweiten Punktes auf der X-Achse.

```
void  X2(
   const double  x2      // Wert des zweiten Punktes auf der X-Achse.
   )
```

Parameter

x2

[in] Wert des zweiten Punktes auf der X-Achse.

X3 (Get-Methode)

Gibt den Wert des dritten Punktes auf der X-Achse zurück.

```
double  X3()
```

Rückgabewert

Wert des dritten Punktes auf der X-Achse.

X3 (Set-Methode)

Setzt den Wert des ersten Punktes auf der X-Achse.

```
void  X3(
   const double  x3      // Wert des ersten Punktes auf der X-Achse
   )
```

Parameter

x3

[in] Wert des ersten Punktes auf der X-Achse.

ToNormalMF

Wandelt eine Dreiecksfunktion in die Gauß'sche Zugehörigkeitsfunktion um.

```
CNormalMembershipFunction*  ToNormalMF()
```

Rückgabewert

Pointer auf die Gauß'sche Zugehörigkeitsfunktion.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

© 2000-2025, MetaQuotes Ltd.

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

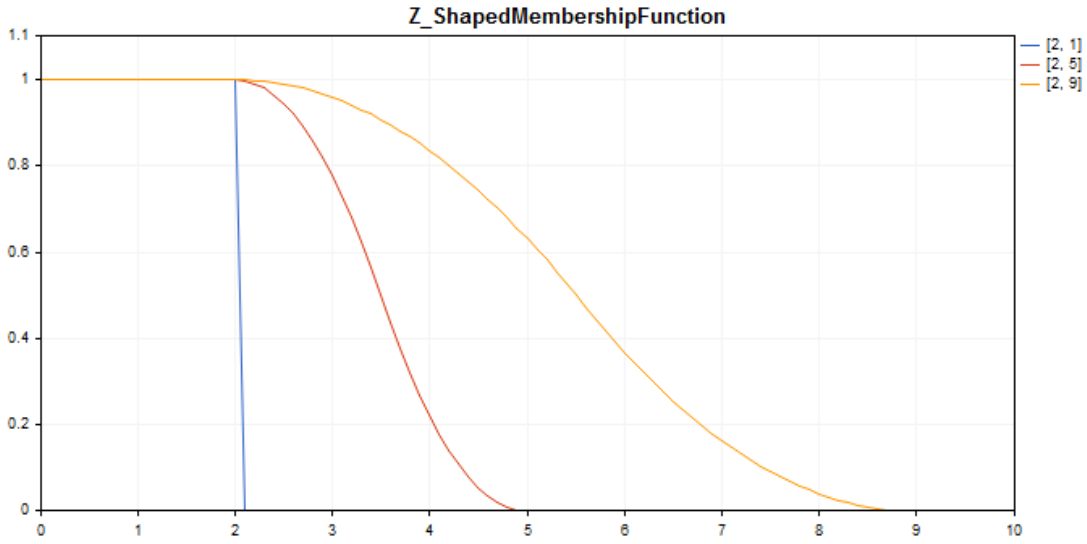
CZ_ShapedMembershipFunction

Klasse für die Implementierung einer z-förmigen Zugehörigkeitsfunktion mit den Parametern A und B.

Beschreibung

Die Funktion setzt eine z-förmige zweiparametrische Zugehörigkeitsfunktion. Das ist eine fallende Zugehörigkeitsfunktion, welche Werte von 1 bis 0 annimmt. Die Parameter der Funktion bestimmen das Intervall, innerhalb dessen die Funktion von 0 bis 1 nichtlinear fällt.

Mithilfe der Funktion können Fuzzy-Mengen vom Typ "sehr niedrig" dargestellt werden. D.h. sie setzt fallende Zugehörigkeitsfunktionen mit Sättigung.



Ein Beispielcode für das Zeichnen dieser Grafik ist unten angeführt.

Deklaration

```
class CZ_ShapedMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject
  IMembershipFunction
    CZ_ShapedMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
A	Liefert und setzt den Parameter des Anfangs des Intervalls, in welchem die Funktion fällt.

© 2000-2025, MetaQuotes Ltd.

Methode der Klasse	Beschreibung
B	Liefert und setzt den Parameter des Endes des Intervalls, in welchem die Funktion fällt.
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Beispiel

```
//+------------------------------------------------------------------+
//|                                  Z_ShapedMembershipFunction.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Math\Fuzzy\membershipfunction.mqh>
#include <Graphics\Graphic.mqh>
//--- Create membership functions
CZ_ShapedMembershipFunction func1(2,1);
CZ_ShapedMembershipFunction func2(2,5);
CZ_ShapedMembershipFunction func3(2,9);
//--- Create wrappers for membership functions
double Z_ShapedMembershipFunction1(double x) { return(func1.GetValue(x)); }
double Z_ShapedMembershipFunction2(double x) { return(func2.GetValue(x)); }
double Z_ShapedMembershipFunction3(double x) { return(func3.GetValue(x)); }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"Z_ShapedMembershipFunction",0,30,30,780,380))
     {
      graphic.Attach(0,"Z_ShapedMembershipFunction");
     }
   graphic.HistoryNameWidth(70);
   graphic.BackgroundMain("Z_ShapedMembershipFunction");
   graphic.BackgroundMainSize(16);
//--- create curve
   graphic.CurveAdd(Z_ShapedMembershipFunction1,0.0,10.0,0.1,CURVE_LINES,"[2, 1]");
   graphic.CurveAdd(Z_ShapedMembershipFunction2,0.0,10.0,0.1,CURVE_LINES,"[2, 5]");
   graphic.CurveAdd(Z_ShapedMembershipFunction3,0.0,10.0,0.1,CURVE_LINES,"[2, 9]");
//--- sets the X-axis properties
   graphic.XAxis().AutoScale(false);
   graphic.XAxis().Min(0.0);
```

© 2000-2025, MetaQuotes Ltd.

```
   graphic.XAxis().Max(10.0);
   graphic.XAxis().DefaultStep(1.0);
//--- sets the Y-axis properties
   graphic.YAxis().AutoScale(false);
   graphic.YAxis().Min(0.0);
   graphic.YAxis().Max(1.1);
   graphic.YAxis().DefaultStep(0.2);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

A (Get-Methode)

Gibt den Parameter des Anfangs des Intervalls zurück, in welchem die Funktion fällt.

```
double  A()
```

Rückgabewert

Parameter des Anfangs des Intervalls, in welchem die Funktion fällt.

A (Set-Methode)

Setzt den Parameter des Anfangs des Intervalls, in welchem die Funktion fällt.

```
void  A(
   const double  a      // Parameter des Anfangs des Intervalls, in welchem die Funkti
   )
```

Parameter

a

[in] Parameter des Anfangs des Intervalls, in welchem die Funktion fällt.

B (Get-Methode)

Gibt den Parameter des Endes des Intervalls, in welchem die Funktion fällt.

```
double  B()
```

Rückgabewert

Parameter des Endes des Intervalls, in welchem die Funktion fällt.

B (Set-Methode)

Setzt den Parameter des Endes des Intervalls, in welchem die Funktion fällt.

```
void  B(
   const double  b      // Parameter des Endes des Intervalls, in welchem die Funktion
   )
```

© 2000-2025, MetaQuotes Ltd.

Parameter

b

[in] Parameter des Endes des Intervalls, in welchem die Funktion fällt.

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

IMembershipFunction

Basisklasse für alle Klassen der Zugehörigkeitsfunktion.

Deklaration

```
class CZ_ShapedMembershipFuncion : public IMembershipFunction
```

Überschrift

```
#include <Math\Fuzzy\membershipfunction.mqh>
```

Vererbungshierarchie

CObject

IMembershipFunction

Direkte Ableitungen

CCompositeMembershipFunction, CConstantMembershipFunction, CDifferencTwoSigmoidalMembershipFunction, CGeneralizedBellShapedMembershipFunction, CNormalCombinationMembershipFunction, CNormalMembershipFunction, CP_ShapedMembershipFunction, CProductTwoSigmoidalMembershipFunctions, CS_ShapedMembershipFunction, CSigmoidalMembershipFunction, CTrapezoidMembershipFunction, CTriangularMembershipFunction, CZ_ShapedMembershipFunction

Methoden der Klasse

Methode der Klasse	Beschreibung
GetValue	Berechnet den Wert der Funktion nach dem angegebenen Argument.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

GetValue

Berechnet den Wert der Funktion nach dem angegebenen Argument.

```
double  GetValue(
   const  x      // Argument der Zugehörigkeitsfunktion
   )
```

Parameter

x

[in] Argument der Zugehörigkeitsfunktion.

Rückgabewert

Wert der Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Regeln für Fuzzy-Systeme

Als **Fuzzy-System** (System der Fuzzy-Inferenz) wird das Erhalten der Konklusion in Form einer Fuzzy-Menge bezeichnet, die den aktuellen Input-Werten entsprechen unter Verwendung **Fuzzy-Regeln** und Fuzzy-Operationen.

Fuzzy-Regeln bestimmen den Zusammenhang zwischen Inputs und Outpts eines Untersuchungsobjekts. Die Anzahl der Regeln im System ist unbegrenzt. Das verallgemeinerte Format unscharfer Regel sieht wie folgt aus:

wenn *Bedingung der Regel*, **dann** *Konklusion der Regel*.

Die Bedingung der *Regel* charakterisiert den aktuellen Zustand des Objekts. Die *Konklusion* charakterisiert, wie die Bedingung das Objekt beeinflussen wird.

Klasse der Regeln für Fuzzy-Systeme	Beschreibung
CMamdaniFuzzyRule	Konklusion einer Fuzzy-Regel für den Mamdani-Algorithmus
CSugenoFuzzyRule	Klasse für die Implementierung einer Fuzzy-Regel für den Sugeno-Algorithmus
CSingleCondition	Die Klasse setzt eine Fuzzy-Bedingung, ausgedrückt durch das Paar "Fuzzy-Variable — Fuzzy-Term".
CConditions	Die Klasse gibt einen Set unscharfer Bedingungen an, die mit dem Operator verbunden sind.
CGenericFuzzyRule	Basisklasse für die Implementierung beider Typen von Fuzzy-Regeln.

© 2000-2025, MetaQuotes Ltd.

CMamdaniFuzzyRule

Fuzzy-Inferenz nach Mamdani — eines der zwei grundlegenden Fuzzy-Systeme. Werte der Ausgabevariablen werden durch Fuzzy-Terme gesetzt.

Beschreibung

Eine unscharfe logische Regel für den Mamdani-Algorithmus kann mit dem folgenden Ausdruck beschrieben werden:

$$if(X_1 \ is \ a_1) \wedge (X_2 \ is \ a_2) \wedge \ldots \wedge (X_n \ is \ a_n) \ then \ (Y \ is \ d)(W)$$

wobei:

- X = (X1, X2, X3 ... Xn) — Vector von Eingabevariablen;
- Y — Eingabevariable;
- a = (a1, a2, a3 ... an) — Vector von Eingabevariablen;
- d — Wert der Ausgabevariablen;
- W — Gewicht der Regel.

Deklaration

```
class CMamdaniFuzzyRule : public CGenericFuzzyRule
```

Überschrift

```
#include <Math\Fuzzy\fuzzyrule.mqh>
```

Vererbungshierarchie

CObject

IParsableRule

CGenericFuzzyRule

CMamdaniFuzzyRule

Methoden der Klasse

Methode der Klasse	Beschreibung
Conclusion	Liefert und setzt die Inferenz einer unscharfen Regel nach Mamdani.
Weight	Liefert und setzt das Gewicht einer unscharfen Regel nach Mamdani.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CGenericFuzzyRule

Condition, Condition, CreateCondition, CreateCondition, CreateCondition

© 2000-2025, MetaQuotes Ltd.

Conclusion (Get-Methode)

Gibt die Konklusion einer Fuzzy-Regel nach Mamdani zurück.

```
CSingleConditon*  Conclusion()
```

Rückgabewert

Konklusion einer Fuzzy-Regel nach Mamdani.

Conclusion (Set-Methode)

Setzt die Konklusion einer Fuzzy-Regel nach Mamdani.

```
void  Conclusion(
   CSingleConditon*  value      // Konklusion einer Fuzzy-Regel nach Mamdani
   )
```

Parameter

value

[in] Konklusion einer Fuzzy-Regel nach Mamdani.

Weight (Get-Methode)

Gibt das Gewicht einer Fuzzy-Regel nach Mamdani zurück.

```
double  Weight()
```

Rückgabewert

Gewicht einer Fuzzy-Regel nach Mamdani.

Weight (Set-Methode)

Setzt das Gewicht einer Fuzzy-Regel nach Mamdani.

```
void  Weight(
   const double  value      // Gewicht einer Fuzzy-Regel nach Mamdani
   )
```

Parameter

value

[in] Gewicht einer Fuzzy-Regel nach Mamdani.

© 2000-2025, MetaQuotes Ltd.

CSugenoFuzzyRule

Fuzzy-Inferenz nach Sugeno — eines der zwei grundlegenden Fuzzy-Systeme. Die Werte der Ausgabevariablen werden als eine lineare Kombination von Eingabevariablen gesetzt.

Beschreibung

Im Vergleich zum Mamdani System wird der Wert der Ausgabevariablen nicht durch einen Fuzzy-Term, sondern durch eine lineare Funktion von Eingabevariablen gesetzt. Eine Fuzzy-Regel für den Sugeno Algorithmus kann mit dem folgenden Ausdruck beschrieben werden:

$$if(X_1\ is\ a_1)\wedge(X_2\ is\ a_2)\wedge\ldots\wedge(X_n\ is\ a_n)\ then\ (Y=b_0+b_1\cdot X_1+b_2\cdot X_2+\ldots+b_n\cdot X_n)(W)$$

wobei:

- X = (X1, X2, X3 ... Xn) — Vector von Eingabevariablen;
- Y — Eingabevariable;
- a = (a1, a2, a3 ... an) — Vector von Eingabevariablen;
- b = (b1, b2, b3 ... bn) — Koeffizient bei einem freien Glied in der linearen Funktion für den Ausgabewert
- W — Gewicht der Regel.

Deklaration

```
class CSugenoFuzzyRule : public CGenericFuzzyRule
```

Überschrift

```
#include <Math\Fuzzy\fuzzyrule.mqh>
```

Vererbungshierarchie

CObject

IParsableRule

CGenericFuzzyRule

CSugenoFuzzyRule

Methoden der Klasse

Methode der Klasse	Beschreibung
Conclusion	Liefert und setzt die Konklusion einer Fuzzy-Regel nach Sugeno.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CGenericFuzzyRule

Condition, Condition, CreateCondition, CreateCondition, CreateCondition

© 2000-2025, MetaQuotes Ltd.

Conclusion (Get-Methode)

Gibt die Konklusion einer Fuzzy-Regel nach Sugeno zurück.

```
CSingleConditon*  Conclusion()
```

Rückgabewert

Konklusion einer Fuzzy-Regel nach Sugeno.

Conclusion (Set-Methode)

Setzt die Konklusion einer Fuzzy-Regel nach Sugeno.

```
void  Conclusion(
   CSingleCondition*  value      // Konklusion einer Fuzzy-Regel nach Sugeno
   )
```

Parameter

value

[in] Konklusion einer Fuzzy-Regel nach Sugeno.

© 2000-2025, MetaQuotes Ltd.

CSingleCondition

Die Klasse setzt eine Fuzzy-Bedingung, ausgedrückt durch das Paar "Fuzzy-Variable — Fuzzy-Term".

Beschreibung

Nach einer Fuzzy-Bedingung entspricht eine Variablen einem Term. Die Fuzzy-Regel kann mit dem folgenden Ausdruck beschrieben werden: *X is a*,

wobei:

- X — eine Fuzzy-Variable;
- a — ein Wert der Fuzzy-Variablen (Fuzzy-Term).

Deklaration

```
class CSingleCondition : public ICondition
```

Überschrift

```
#include <Math\Fuzzy\fuzzyrule.mqh>
```

Vererbungshierarchie

CObject

ICondition

CSingleCondition

Direkte Ableitungen

CFuzzyCondition

Methoden der Klasse

Methode der Klasse	Beschreibung
Not	Liefert und setzt das Flag, das angibt, ob auf diese Regel Negation angewandt werden muss.
Term	Liefert und setzt einen Fuzzy-Term für diese Bedingung.
Var	Liefert und setzt eine Fuzzy-Variable für diese Bedingung.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Not (Get-Methode)

Gibt das Flag zurück, dass angibt, ob auf diese Regel Negation angewandt werden muss.

```
bool Not()
```

© 2000-2025, MetaQuotes Ltd.

Rückgabewert

Wert des Flags.

Not (Set-Methode)

Setzt das Flag zurück, dass angibt, ob auf diese Regel Negation angewandt werden muss.

```
void  Not(
   bool  not      // Wert des Flags
   )
```

Parameter

not

[in] Wert des Flags.

Term (Get-Methode)

Gibt einen Fuzzy-Term für die angegebene Bedingung zurück.

```
INamedValue* Term()
```

Rückgabewert

Fuzzy-Term für die angegebene Bedingung.

Term (Set-Methode)

Setzt einen Fuzzy-Term für die angegebene Bedingung.

```
void  Term(
   INamedValue*&  value      // Fuzzy-Term für die angegebene Bedingung
   )
```

Parameter

value

[in] Fuzzy-Term für die angegebene Bedingung.

Var (Get-Methode)

Gibt eine Fuzzy-Variable für diese Bedingung zurück.

```
INamedVariable* Var()
```

Rückgabewert

Fuzzy-Variable für die angegebene Bedingung.

Var (Set-Methode)

Setzt eine Fuzzy-Variable für die angegebene Bedingung

© 2000-2025, MetaQuotes Ltd.

```
void  Var(
   INamedVariable*&  value      // Fuzzy-Variable für die angegebene Bedingung
   )
```

Parameter

value

[in] Fuzzy-Variable.

© 2000-2025, MetaQuotes Ltd.

CConditions

Die Klasse gibt einen Set unscharfer Bedingungen an, die mit dem Operator verbunden sind.

Beschreibung

Der Set unscharfer Regeln kann zum Beispiel mit dem folgenden Ausdruck beschrieben werden:

$$(X_1 \; is \; a_1) \wedge (X_2 \; is \; a_2) \wedge \ldots \wedge (X_n \; is \; a_n)$$

wobei:

- X = (X1, X2, X3 ... Xn) — Vektor der Eingabevariablen;
- a = (a1, a2, a3 ... an) — Vektor der Werte der Eingabevariablen.

In diesem Beispiel wurde der *and* Operator (und) verwendet. Darüber hinaus ist der *or* Operator (oder) verwendet.

Deklaration

```
class CConditions : public ICondition
```

Überschrift

```
#include <Math\Fuzzy\fuzzyrule.mqh>
```

Vererbungshierarchie

CObject
  ICondition
    CConditions

Methoden der Klasse

Methode der Klasse	Beschreibung
ConditionsList	Gibt die Liste der Zugehörigkeitsfunktionen zurück
Not	Liefert und setzt einen Parameter, der anzeigt, ob auf diese Regeln Negation angewandt werden muss
Op	Liefert und setzt den Typ des Operators.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

ConditionsList

Gibt die Liste aller Bedingungen zurück.

```
CList*  ConditionsList()
```

© 2000-2025, MetaQuotes Ltd.

Rückgabewert

Gibt die Liste aller Bedingungen zurück.

Not (Get-Methode)

Gibt das Flag zurück, der anzeigt, ob auf diese Regeln Negation angewandt werden muss.

```
bool Not()
```

Rückgabewert

Wert des Flags.

Not (Set-Methode)

Setzt das Flag, der anzeigt, ob auf diese Regeln Negation angewandt werden muss

```
void  Not(
   bool  not      // Wert des Flags
   )
```

Parameter

not

[in] Wert des Flags.

Op (Get-Methode)

Gibt den Operatortyp für die Verknüpfung von Bedingungen zurück. Zwei *and* (und) und *or* (oder).

```
OperatorType Op()
```

Rückgabewert

Operatortyp für die Verknüpfung von Bedingungen.

Op (Set-Methode)

Setzt den Operatortyp für die Verknüpfung von Bedingungen. Es sind zwei Oepratoren verfügbar: *and* (und) und *or* (oder).

```
void  Op(
   OperatorType  op     // Operatortyp für die Verknüpfung von Bedingungen
   )
```

Parameter

op

[in] Operatortyp für die Verknüpfung von Bedingungen.

© 2000-2025, MetaQuotes Ltd.

CGenericFuzzyRule

Basisklasse für beide Typen der Fuzzy-Regeln.

Deklaration

```
class CGenericFuzzyRule : public IParsableRule
```

Überschrift

```
#include <Math\Fuzzy\fuzzyrule.mqh>
```

Vererbungshierarchie

CObject

IParsableRule

CGenericFuzzyRule

Direkte Ableitungen

CMamdaniFuzzyRule, CSugenoFuzzyRule

Methoden der Klasse

Methode der Klasse	Beschreibung
Conclusion	Liefert und setzt die Konklusion einer Fuzzy-Regel
Condition	Liefert und setzt die Bedingung (Bedingungen) 'if' für eine Fuzzy-Regel
CreateCondition	Erstellt die Bedingung für eine Fuzzy-Regel nach angegebenen Parametern

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Conclusion (Get-Methode)

Gibt die Konklusion einer Fuzzy-Regel zurück.

```
CSingleConditon*  Conclusion()
```

Rückgabewert

Konklusion einer Fuzzy-Regel.

Conclusion (Set-Methode)

Setzt die Konklusion einer Fuzzy-Regel.

```
virtual void  Conclusion(
   CSingleConditon*  value      // Konklusion einer Fuzzy-Regel
```

© 2000-2025, MetaQuotes Ltd.

```
   )
```

Parameter

value

[in] Konklusion einer Fuzzy-Regel.

Condition (Get-Methode)

Gibt die Bedingung (Bedingungen) 'if' für eine Fuzzy-Regel zurück.

```
CConditons*  Condition()
```

Rückgabewert

Fuzzy-Bedingung (Bedingungen).

Condition (Set-Methode)

Setzt die Bedingung (Bedingungen) 'if' für eine Fuzzy-Regel.

```
void  Condition(
   CConditons*  value      // Bedingung (Bedingungen) "if" für die Fuzzy-Regel
   )
```

Parameter

value

[in] Fuzzy-Bedingung (Bedingungen).

CreateCondition

Erstellt eine Bedingung für eine unscharfe Regel nach angegebenen Parametern.

```
CFuzzyCondition*  CreateCondition(
   CFuzzyVariable*  var,       // Fuzzy-Variable
   CFuzzyTerm*      term,      // Fuzzy-Term
   )
```

Parameter

var

[in] Fuzzy-Variable.

term

[in] Fuzzy-Term.

Rückgabewert

Status einer Fuzzy-Regel.

CreateCondition

© 2000-2025, MetaQuotes Ltd.

Erstellt eine Bedingung für eine unscharfe Regel nach angegebenen Parametern.

```
CFuzzyCondition*  CreateCondition(
   CFuzzyVariable*  var,      // Fuzzy-Variable
   CFuzzyTerm*      term,     // Fuzzy-Term
   bool             not,      // Flag, das anzeigt, ob auf diese Regel Negation angewa
   )
```

Parameter

var

[in] Fuzzy-Variable.

term

[in] Fuzzy-Term.

not

[in] Flag, das anzeigt, ob auf diese Regel Negation angewandt werden muss.

Rückgabewert

Status einer Fuzzy-Regel.

CreateCondition

Erstellt eine Bedingung für eine unscharfe Regel nach angegebenen Parametern.

```
CFuzzyCondition*  CreateCondition(
   CFuzzyVariable* var,      // Fuzzy-Variable
   CFuzzyTerm*     term,     // Fuzzy-Term
   bool            not,      // Flag, das anzeigt, ob auf diese Regel Negation angewar
   HedgeType       hedge     // Typ der Verknüpfung in der Regel
   )
```

Parameter

var

[in] Fuzzy-Variable.

term

[in] Fuzzy-Term.

not

[in] Flag, das anzeigt, ob auf diese Regel Negation angewandt werden muss.

hedge

[in] Typ der Verknüpfung in der Regel.

Rückgabewert

Status einer Fuzzy-Regel.

© 2000-2025, MetaQuotes Ltd.

Variablen für Fuzzy-Systeme

In Fuzzy-Systemen werden **unscharfe (linguistische) Variablen** verwendet. Das sind solche Variablen, die Wörter oder Wortgruppen einer natürlichen oder künstlichen Sprache als Werte annehmen können.

Linguistische Variablen bilden Fuzzy-Mengen. Der Charakter und die Anzahl von Fuzzy-Variablen beim Definieren unscharfer Mengen sind verschieden für jede einzelne Aufgabe.

Klasse	Beschreibung
CFuzzyVariable	Klasse für die Erstellung allgemeiner Fuzzy-Variablen.
CSugenoVariable	Klasse für die Erstellung von Fuzzy-Variablen nach Sugeno.

© 2000-2025, MetaQuotes Ltd.

CFuzzyVariable

Klasse für die Erstellung allgemeiner Fuzzy-Variablen.

Beschreibung

Eine Fuzzy-Variable wird in diesem Fall durch folgende Parameter gesetzt:

- Höchstwert der Variablen;
- Mindestwert der Variablen;
- Name der Fuzzy-Variablen;
- Term-Menge (die Menge aller möglichen Werte, die eine linguistische Variable annehmen kann).

Deklaration

```
class CFuzzyVariable : public CNamedVariableImpl
```

Überschrift

```
#include <Math\Fuzzy\fuzzyvariable.mqh>
```

Vererbungshierarchie

CObject
  INamedValue
    INamedVariable
      CNamedVariableImpl
        CFuzzyVariable

Methoden der Klasse

Methode der Klasse	Beschreibung
#AddTerm	Fügt einer Fuzzy-Variablen einen Fuzzy-Term hinzu.
GetTermByName	Erhält einen Fuzzy-Term nach dem angegebenen Namen.
Max	Liefert und setzt den Höchstwert für eine Fuzzy-Variable.
Min	Liefert und setzt den Mindestwert für eine Fuzzy-Variable.
Terms	Liefert und setzt eine Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable.
Values	Liefert und setzt eine Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable.

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CNamedVariableImpl

Name, Name

#AddTerm

Fügt einer Fuzzy-Variablen einen Fuzzy-Term hinzu.

```
void  AddTerm(
   CFuzzyTerm*&  term      // Fuzzy-Term
   )
```

Parameter

term

[in] Fuzzy-Term.

GetTermByName

Erhält einen Fuzzy-Term nach dem angegebenen Namen.

```
CFuzzyTerm*  GetTermByName(
   const string  name      // Name des Fuzzy-Terms
   )
```

Parameter

name

[in] Name des Fuzzy-Terms.

Rückgabewert

Fuzzy-Term mit dem angegebenen Namen.

Max (Get-Methode)

Gibt den Höchstwert für eine Fuzzy-Varibale zurück.

```
double  Max()
```

Rückgabewert

Höchstwert für eine Fuzzy-Variable.

Max (Set-Methode)

Setzt den Höchstwert für die Fuzzy-Variable.

```
void  Max(
   const double  max      // Höchstwert für die Fuzzy-Variable
```

© 2000-2025, MetaQuotes Ltd.

```
   )
```

Parameter

max

[in] Höchstwert für eine Fuzzy-Variable.

Min (Get-Methode)

Gibt den Mindestwert für eine Fuzzy-Variable zurück.

```
double  Min()
```

Rückgabewert

Mindestwert für eine Fuzzy-Variable.

Max (Set-Methode)

Setzt den Mindestwert für eine Fuzzy-Variable.

```
void  Min(
   const double  min      // Mindestwert für die Fuzzy-Variable
   )
```

Parameter

min

[in] Mindestwert für eine Fuzzy-Variable.

Terms (Get-Methode)

Gibt eine Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable zurück.

```
CList*  Terms()
```

Rückgabewert

Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable.

Terms (Set-Methode)

Setzt eine Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable.

```
void  Terms(
   CList*&  terms      // Liste von Fuzzy-Termen für die gegebene Variable
   )
```

Parameter

terms

[in] Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable.

© 2000-2025, MetaQuotes Ltd.

Values

Gibt eine Liste von Fuzzy-Termen für die gegebene Fuzzy-Variable zurück.

```
CList*  Values()
```

Rückgabewert

Liste von Fuzzy-Termen für die gegebene Variable.

© 2000-2025, MetaQuotes Ltd.

CSugenoVariable

Klasse für die Erstellung von Fuzzy-Variablen nach Sugeno.

Beschreibung

Eine Fuzzy-Variable vom Typ Sugeno unterscheidet sich von einer linguistischen Variablen allgemeinen Typs dadurch, dass sie nicht durch eine Term-Menge, sondern durch einen Set linearer Funktionen gesetzt wird.

Deklaration

```
class CSugenoVariable : public CNamedVariableImpl
```

Überschrift

```
#include <Math\Fuzzy\sugenovariable.mqh>
```

Vererbungshierarchie

CObject
INamedValue
INamedVariable
CNamedVariableImpl
CSugenoVariable

Methoden der Klasse

Methode der Klasse	Beschreibung
Functions	Gibt die Liste linearer Funktionen der Fuzzy-Variablen nach Sugeno.
GetFuncByName	Gibt die lineare Funktion nach dem angegebene Namen zurück.
Values	Gibt die Liste linearer Funktionen der Fuzzy-Variablen nach Sugeno.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CNamedVariableImpl

Name, Name

Functions

Gibt die Liste linearer Funktionen der Fuzzy-Variablen nach Sugeno.

```
CList*  Functions()
```

© 2000-2025, MetaQuotes Ltd.

Rückgabewert

Liste linearer Funktionen.

GetFuncByName

Gibt die lineare Funktion nach dem angegebene Namen zurück.

```
ISugenoFunction*  GetFuncByName(
   const string  name      // Name der linearen Funktion
   )
```

Parameter

name

[in] Name der linearen Funktion.

Rückgabewert

Lineare Funktion mit dem angegebenen Namen.

Values

Gibt die Liste linearer Funktionen der Fuzzy-Variablen nach Sugeno.

```
CList*  Values()
```

Rückgabewert

Liste linearer Funktionen einer Fuzzy-Variablen nach Sugeno.

© 2000-2025, MetaQuotes Ltd.

CFuzzyTerm (Fuzzy-Terme)

Klasse für die Implementierung von Fuzzy-Termen.

Beschreibung

Als **Term (term)** wird jedes Element einer Term-Menge bezeichnet. Ein Term wird durch zwei Komponenten definiert:

- durch den Namen des unscharfen Terms;
- durch die Zugehörigkeitsfunktion.

Deklaration

```
class CFuzzyTerm : public CNamedValueImpl
```

Überschrift

```
#include <Math\Fuzzy\fuzzyterm.mqh>
```

Vererbungshierarchie

CObject
  INamedValue
    CNamedValueImpl
      CFuzzyTerm

Methoden der Klasse

Methode der Klasse	Beschreibung
MembershipFunction	Gibt die Zugehörigkeitsfunktion für einen Fuzzy-Term zurück.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CNamedValueImpl

Name, Name

© 2000-2025, MetaQuotes Ltd.

MembershipFunction

Gibt die Zugehörigkeitsfunktion für einen Fuzzy-Term zurück.

```
IMembershipFunction*  MembershipFunction()
```

Rückgabewert

Zugehörigkeitsfunktion.

© 2000-2025, MetaQuotes Ltd.

Fuzzy-Systeme

Ein **Fuzzy-System** (oder *Fuzzy-Modell*) stellt ein mathematisches Modell dar, dem die Fuzzy-Logik zugrunde liegt. Solche Modelle werden in dem Fall erstellt, wenn das Untersuchungsobjekt eine schwache Formalisierung hat, und seine genaue mathematische Beschreibung zu kompliziert oder unbekannt ist.

Die Erstellung eines Modells kann in drei Schritte geteilt werden:

1. Definition von Ein- und Ausgabeparametern des Modells.
2. Erstellung einer Wissensdatenbank.
3. Auswahl einer Inferenzmethode (Mamdani oder Sugeno).

Von dem ersten Schritt hängen unmittelbar zwei weiteren Schritte ab und gerade er bestimmt das weitere Funktionieren des Modells.

Eine **Wissensdatenbank** (*Regeldatenbank*) stellt eine Gesamtheit unscharfer Regeln vom Typ: "wenn, dann", die den Zusammenhang zwischen Input und Output des Untersuchungsobjektes bestimmt.

Die **Bedingung (Condition)** der Regel charakterisiert den aktuellen Zustand des Objekts und die **Konklusion (Conclusion)** — wie die Bedingung das Objekt beeinflussen wird.

Bedingungen und Konklusionen für jede Regel können von zwei Arten sein:

1. einfach (Verweis zu Csinglcond) — eine Fuzzy-Variable;
2. zusammengesetzt (Verweis zu Cconditions) — mehrere Fuzzy-Variablen.

Jede Regel im System hat ein **Gewicht** — die Wichtigkeit der Regel im Modell. Die Gewichtskoeffizienten nehmen Werte aus dem Bereich [0, 1] an.

Je nach der erstellten Wissensdatenbank wird für des Modell eine Inferenzmethode bestimmt. Als **Fuzzy-Inferenz** wird das Erhalten der Konklusion in Form einer Fuzzy-Menge bezeichnet, die den aktuellen Input-Werten entsprechen, unter Verwendung der Fuzzy-Wissensdatenbank und Fuzzy-Operationen. Zwei grundlegende Inferenz-Typen — nach Mamdani und Sugeno.

© 2000-2025, MetaQuotes Ltd.

Mamdani System

Werte der Ausgabevariablen werden im Mamdani-System durch Fuzzy-Terme gesetzt.

Beschreibung

Eine unscharfe logische Regel für den Mamdani-Algorithmus kann mit dem folgenden Ausdruck beschrieben werden:

$$if(X_1 \ is \ a_1) \wedge (X_2 \ is \ a_2) \wedge \ldots \wedge (X_n \ is \ a_n) \ then \ (Y \ is \ d)(W)$$

wobei:

- X = (X1, X2, X3 ... Xn) — Vector von Eingabevariablen;
- Y — Eingabevariable;
- a = (a1, a2, a3 ... an) — Vector von Eingabevariablen;
- d — Wert der Ausgabevariablen;
- W — Gewicht der Regel.

Methoden der Klasse

Methode der Klasse	Beschreibung
AggregationMethod	Setzt den Typ der Aggregation der Bedingungen
Calculate	Berechnet die Fuzzy-Inferenz für dieses System
DefuzzificationMethod	Setzt den Typ der Defuzzifizierung
EmptyRule	Erstellt eine leere Fuzzy-Regel nach Mamdani basierend auf dem aktuellen System
ImplicationMethod	Setzt den Typ des Operators für die Implikation des Systems
Output	Gibt die Liste unscharfer Ausgabevariablen nach Mamdani zurück.
OutputByName	Gibt die unscharfe Ausgabevariable nach Mamdani nach dem angegebenen Namen zurück.
ParseRule	Erstellt eine Fuzzy-Regel nach Mamdani basierend auf dem angegeben String.
Rules	Gibt die Liste unscharfer Regeln nach Mamdani zurück.

Methoden geerbt von der Klasse CGenericFuzzySystem

Input, AndMethod, AndMethod, OrMethod, OrMethod, InputByName, Fuzzify

AggregationMethod

Setzt den Typ der Aggregationsmethode für Bedingungen.

© 2000-2025, MetaQuotes Ltd.

```
void  AggregationMethod(
   AggregationMethod  value      // Typ der Aggregationsmethode
   )
```

Parameter

value

[in] Typ der Aggregationsmethode für Bedingungen.

Calculate

Berechnet die Fuzzy-Inferenz für dieses System.

```
CList*  Calculate(
   CList*  inputValues      // Inputdaten
   )
```

Parameter

inputValues

[in] Inputdaten für die Berechnung.

Rückgabewert

Berechnungsergebnis.

DefuzzificationMethod

Setzt den Typ der Methode der Defuzzifizierung.

```
void  DefuzzificationMethod(
   DefuzzificationMethod  value      // Typ der Methode der Defuzzifierung.
   )
```

Parameter

value

[in] Typ der Defuzzifizierung.

EmptyRule

Erstellt eine leere Fuzzy-Regel nach Mamdani basierend auf dem aktuellen System.

```
CMamdaniFuzzyRule*  EmptyRule()
```

Rückgabewert

Fuzzy-Regel nach Mamdani.

ImplicationMethod

Setzt den Typ des Implikationsoperators der Bedingungen.

© 2000-2025, MetaQuotes Ltd.

```
void  ImplicationMethod(
   ImplicationMethod  value      // Typ des Implikationsoperators
   )
```

Parameter

value

[in] Typ des Implikationsoperators der Bedingungen.

Output

Gibt die Liste unscharfer Ausgabevariablen nach Mamdani.

```
CList*  Output()
```

Rückgabewert

Liste unscharfer Variablen.

OutputByName

Gibt die unscharfe Ausgabevariable nach Mamdani nach dem angegebenen Namen zurück.

```
CFuzzyVariable*  OutputByName(
   const string  name      // Name unscharfer Variablen
   )
```

Parameter

name

[in] Name unscharfer Variablen.

Rückgabewert

Fuzzy-Variable nach Mamdani mit dem angegebenen Namen.

ParseRule

Erstellt eine Fuzzy-Regel nach Mamdani basierend auf dem angegeben String.

```
CMamdaniFuzzyRule*  ParseRule(
   const string  rule      // Darstellung der Fuzzy-Regel als String
   )
```

Parameter

rule

[in] Darstellung der Fuzzy-Regel nach Mamdani als String.

Rückgabewert

Fuzzy-Regel nach Mamdani.

© 2000-2025, MetaQuotes Ltd.

Rules

Gibt die Liste unscharfer Regeln nach Mamdani zurück.

```
CList*  Rules()
```

Rückgabewert

Liste von Fuzzy-Regeln nach Mamdani.

© 2000-2025, MetaQuotes Ltd.

CSugenoFuzzyRule

Das unscharfe Sugeno System ist eines der grundlegenden Fuzzy-Systeme. Die Werte der Ausgabevariablen werden als eine lineare Kombination von Eingabevariablen gesetzt.

Beschreibung

Im Vergleich zum Mamdani System wird der Wert der Ausgabevariablen nicht durch einen Fuzzy-Term sondern durch eine lineare Funktion von Eingabevariablen gesetzt. Eine Fuzzy-Regel für den Sugeno Algorithmus kann mit dem folgenden Ausdruck beschrieben werden:

$$if(X_1\ is\ a_1)\wedge(X_2\ is\ a_2)\wedge\ldots\wedge(X_n\ is\ a_n)\ then\ (Y=b_0+b_1\cdot X_1+b_2\cdot X_2+\ldots+b_n\cdot X_n)(W)$$

wobei:

- X = (X1, X2, X3 ... Xn) — Vector von Eingabevariablen;
- Y — Eingabevariable;
- a = (a1, a2, a3 ... an) — Vector von Eingabevariablen;
- b = (b1, b2, b3 ... bn) — Koeffizient bei einem freien Glied in der linearen Funktion für den Ausgabewert
- W — Gewicht der Regel.

Methoden der Klasse

Methode der Klasse	Beschreibung
Calculate	Berechnet die Fuzzy-Inferenz für dieses System.
CreateSugenoFunction	Erstellt die lineare Funktion nach Sugeno für dieses System.
EmptyRule	Erstellt eine leere Fuzzy-Regel nach Sugeno basierend auf dem aktuellen System.
Output	Gibt die Liste unscharfer Ausgabevariablen nach Sugeno zurück.
OutputByName	Gibt eine Fuzzy-Variable nach Sugeno nach dem angegebenen Namen zurück.
ParseRule	Erstellt eine Fuzzy-Regel nach Sugeno basierend auf einem vorgegebenen String.
Rules	Gibt die Liste unscharfer Regel zurück.

Methoden geerbt von der Klasse CGenericFuzzySystem

Input, AndMethod, AndMethod, OrMethod, OrMethod, InputByName, Fuzzify

Calculate

Berechnet die Fuzzy-Inferenz für dieses System.

© 2000-2025, MetaQuotes Ltd.

```
CList*  Calculate(
   CList*&  inputValues      // Inputdaten
   )
```

Parameter

inputValues

[in] Inputdaten für die Berechnung.

Rückgabewert

Berechnungsergebnis.

CreateSugenoFunction

Erstellt die lineare Funktion nach Sugeno für dieses System.

```
CLinearSugenoFunction*  CreateSugenoFunction(
   const string  name,          // Name der Funktion
   const double& coeffs[]       // Koeffizienten der Funktion
   )
```

Parameter

name

[in] Name der Funktion.

coeffs[]

[in] Koeffizienten der Funktion.

Rückgabewert

Lineare Funktion nach Sugeno.

Hinweis

Die Größe des Arrays für Koeffizienten kann gleich der Anzahl der Eingabevariablen oder größer um eins sein. Im ersten Fall wird das freie Glied der linearen Funktion nach Sugeno gleich Null sein, und im zweiten — gleich dem letzten Koeffizienten.

CreateSugenoFunction

Erstellt die lineare Funktion nach Sugeno für dieses System.

```
CLinearSugenoFunction*  CreateSugenoFunction(
   const string  name,          // Name der Funktion
   CList*&       coeffs,        // Liste der Paare "Fuzzy-Variable - Koeffizient"
   const double  constValue     // Koeffizient beim freien Glied der Funktion
   )
```

Parameter

name

[in] Name der Funktion.

© 2000-2025, MetaQuotes Ltd.

coeffs[]

[in] Koeffizienten der Funktion.

Rückgabewert

Lineare Funktion nach Sugeno.

EmptyRule

Erstellt eine leere Fuzzy-Regel nach Sugeno basierend auf dem aktuellen System.

```
CSugenoFuzzyRule*  EmptyRule()
```

Rückgabewert

Fuzzy-Regel nach Sugeno.

Output

Gibt die Liste unscharfer Ausgabevariablen nach Sugeno zurück.

```
CList*  Output()
```

Rückgabewert

Liste unscharfer Variablen.

OutputByName

Gibt eine unscharfe Ausgabevariable nach Sugeno nach dem angegebenen Namen zurück.

```
CSugenoVariable*  OutputByName(
   const string  name      // Name der Fuzzy-Varibalen
   )
```

Parameter

name

Name der Fuzzy-Variablen.

Rückgabewert

Fuzzy-Variable nach Sugeno nach dem angegebenen Namen.

ParseRule

Erstellt eine Fuzzy-Regel nach Sugeno basierend auf einem vorgegebenen String.

```
CSugenoFuzzyRule*  ParseRule(
   const string  rule      // Fuzzy-Regel nach Sugeno als String
   )
```

Parameter

© 2000-2025, MetaQuotes Ltd.

rule

[in] Fuzzy-Regel nach Sugeno als String.

Rückgabewert

Fuzzy-Regel nach Sugeno.

Rules

Gibt die Liste von Fuzzy-Regeln zurück.

```
CList*  Rules()
```

Rückgabewert

Liste von Fuzzy-Regeln.

© 2000-2025, MetaQuotes Ltd.

Klasse für das Arbeiten mit OpenCL Programmen

Die Klasse COpenCL ist ein Wrapper ("Umhüllung") für das Arbeiten mit OpenCL Funktionen. Die Verwendung von GPU erlaubt es, in einigen Fällen die Geschwindigkeit der Berechnungen wesentlich zu erhöhen.

Beispiele für die Verwendung der Klasse für Berechnungen anhand der Zahlen float und double kann man im Ordner MQL5\Scripts\Examples\OpenCL\ in den entsprechenden Unterverzeichnissen finden, Quellcodes der OpenCL-Programme sind in den Unterverzeichnissen MQL5\Scripts\Examples\OpenCL\Double\Kernels und MQL5\Scripts\Examples\OpenCL\Float\Kernels zu finden.

- MatrixMult.mq5 - ein Beispiel für die Multiplikation von Matrizes unter Verwendung des globalen und lokalen Speichers
- BitonicSort.mq5 - ein Beispiel für eine parallele Sortierung von Elementen eines Arrays auf GPU
- FFT.mq5 - ein Beispiel für die Berechnung schneller Fourier-Transformation
- Wavelet.mq5 - ein Beispiel für die Berechnung der Wavelet-Transformation mithilfe von Morlet-Wavelet.

Es ist zu empfehlen, den OpenCL-Code in separaten CL-Dateien zu schreiben, die man später mithilfe von Ressourcenvariablen einem MQL5-Programm hinzufügen kann.

Deklaration

```
class COpenCL
```

Überschrift

```
#include <OpenCL\COpenCL.mqh>
```

Methoden der Klasse

Name	Beschreibung
BufferCreate	Erstellt einen OpenCL Puffer nach dem angegebenen Index
BufferFree	Löscht einen Puffer nach dem angegebenen Index
BufferFromArray	Erstellt einen Puffer nach dem angegebenen Index aus dem Array von Werten
BufferRead	Liest einen OpenCL Puffer nach dem angegebenen Index ins Array
BufferWrite	Schreibt ein Array von Daten in den Puffer nach dem angegebenen Index
Execute	Führt ein OpenCL Programm nach dem angegebenen Index aus
GetContext	Gibt den Handle des OpenCL Kontextes zurück
GetKernel	Gibt den Handle eines Kernel-Objekts nach dem angegebenen Index zurück

© 2000-2025, MetaQuotes Ltd.

Name	Beschreibung
GetKernelName	Gibt den Namen eines Kernel-Objekts nach dem angegebenen Index zurück
GetProgram	Gibt den Handle des OpenCL-Programms zurück
Initialize	Initialisiert ein OpenCL Programm
KernelCreate	Erstellt den Einsprungpunkt eines OpenCL Programm nach dem angegebenen Index
KernelFree	Löscht die Startfunktion von OpenCL nach dem angegebenen Index
SetArgument	Setzt einen Parameter für eine OpenCL Funktion nach dem angegebenen Index
SetArgumentBuffer	Setzt einen OpenCL Puffer als Parameter einer OpenCL Funktion nach dem angegebenen Index
SetArgumentLocalMemory	Setzt einen Parameter im lokalen Speicher für eine OpenCL Funktion nach dem angegebenen Index
SetBuffersCount	Setzt die Anzahl von Puffern
SetKernelsCount	Setzt die Anzahl von Kernel-Objekten
Shutdown	Entfernt ein OpenCL Programm aus dem Speicher
SupportDouble	Ermittelt, ob reelle Datentypen auf dem Gerät unterstützt werden

© 2000-2025, MetaQuotes Ltd.

BufferCreate

Erstellt ein OpenCL Puffer nach dem angegebenen Index.

```
bool  BufferCreate(
   const int   buffer_index,      // Index des Puffers
   const uint  size_in_bytes,     // Puffergröße in Bytes
   const uint  flags              // Kombination der Flags, die Puffereigenschaften se
   );
```

Parameter

buffer_index

[in] Pufferindex.

size_in_bytes

[in] Puffergröße in Bytes.

flags

[in] Eigenschaften des Puffers, die über die Kombination der Flags gesetzt werden.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

BufferFree

Löscht einen Puffer nach dem angegebenen Index.

```
bool  BufferFree(
   const int  buffer_index      // Index des Puffers
   );
```

Parameter

buffer_index

[in] Pufferindex.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

BufferFromArray

Erstellt einen Puffer nach dem angegebenen Index aus dem Array von Werten.

```
template<typename T>
bool  BufferFromArray(
   const int   buffer_index,          // Index des Puffers
   T           &data[],               // Array von Werten
   const uint  data_array_offset,     // Verschiebung im Array von Werten in Bytes
   const uint  data_array_count,      // Anzahl der Werte aus dem Array für Schreiben
   const uint  flags                  // Kombination von Flags, die Puffereigenschafte
   );
```

Parameter

buffer_index

[in] Pufferindex.

&data[]

[in] Array von Werten, die in den OpenCL Puffer geschrieben werden müssen.

data_array_offset

[in] Verschiebung im Array von Werten in Bytes, von dem das Schreiben beginnt.

data_array_count

[in] Anzahl von Werten, die geschrieben werden müssen.

flags

[in] Eigenschaften des Puffers, die über die Kombination der Flags gesetzt werden.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

BufferRead

Liest einen OpenCL Puffer nach dem angegebenen Index ins Array.

```
template<typename T>
bool  BufferRead(
   const int   buffer_index,          // Index des Puffers
   T           &data[],               // Array von Werten
   const uint  cl_buffer_offset,      // Verschiebung im OpenCL Puffer in Bytes
   const uint  data_array_offset,     // Verschiebung im Array in Elementen
   const uint  data_array_count       // Anzahl der Werte aus dem Puffer, die gelesen
   );
```

Parameter

buffer_index

[in] Pufferindex.

&data[]

[in] Array für das Erhalten von Werten aus dem OpenCL Puffer.

cl_buffer_offset

[in] Verschiebung im OpenCL Puffer in Bytes, von dem das Lesen beginnt.

data_array_offset

[in] Index des ersten Elements des Arrays für das Schreiben von Werten des OpenCL Puffers.

data_array_count

[in] Anzahl der Werte, die gelesen werden müssen.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

BufferWrite

Schreibt ein Array von Daten in den Puffer nach dem angegebenen Index.

```
template<typename T>
bool  BufferWrite(
   const int   buffer_index,          // Index des Puffers
   T           &data[],               // Array von Werten
   const uint  cl_buffer_offset,      // Verschiebung im OpenCL Puffer in Bytes
   const uint  data_array_offset,     // Verschiebung im Array in Elementen
   const uint  data_array_count       // Anzahl der Werte aus dem Array für das Schrei
   );
```

Parameter

buffer_index

[in] Pufferindex.

&data[]

[in] Array von Werten, die in den OpenCL Puffer geschrieben werden müssen.

cl_buffer_offset

[in] Verschiebung im OpenCL Puffer in Bytes, von dem das Schreiben beginnt.

data_array_offset

[in] Index des ersten Elements des Arrays, von welchem an Werte aus dem Array für das Schreiben in den OpenCL Puffer genommen werden.

data_array_count

[in] Anzahl von Werten, die geschrieben werden müssen.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Execute

Führt ein OpenCL Programm nach dem angegebenen Index aus.

```
bool  Execute(
   const int   kernel_index,           // Index des Kernels
   const int   work_dim,               // Anzahl der Dimensionen im Raum der Aufgaben
   const uint  &work_offset[],         // ursprüngliche Verschiebung im Raum der Aufga
   const uint  &work_size[]            // Gesamtzahl der Aufgaben
   );
```

Führt ein OpenCL Programm nach dem angegebenen Index mit der gesetzten Anzahl der Aufgaben in einer lokalen Gruppe aus.

```
bool  Execute(
   const int   kernel_index,           // Index des Kernels
   const int   work_dim,               // Anzahl der Dimensionen im Raum der Aufgaben
   const uint  &work_offset[],         // ursprüngliche Verschiebung im Raum der Aufga
   const uint  &work_size[],           // Gesamtzahl der Aufgaben
   const uint  &local_work_size[]      // Anzahl der Aufgaben in einer lokalen Gruppe
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

work_dim

[in] Anzahl der Dimensionen im Raum der Aufgaben.

&work_offset[]

[in] Ursprüngliche Verschiebung im Raum der Aufgaben.

&work_size[]

[in] Größe der Untermenge der Aufgaben.

&local_work_size[]

[in] Größe der lokalen Untermenge der Aufgaben in der Gruppe.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

GetContext

Gibt den Handle des OpenCL Kontexts zurück.

```
int  GetContext();
```

Rückgabewert

Handle des OpenCL Kontexts.

© 2000-2025, MetaQuotes Ltd.

GetKernel

Gibt den Handle eines Kernel-Objekts nach dem angegebenen Index zurück.

```
int  GetKernel(
   const int  kernel_index      // Index des Kernels
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

Rückgabewert

Handle des Kernel-Objekts.

© 2000-2025, MetaQuotes Ltd.

GetKernelName

Gibt den Namen eines Kernel-Objekts nach dem angegebenen Index zurück.

```
string  GetKernelName(
   const int  kernel_index      // Index des Kernels
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

Rückgabewert

Name des Kernel-Objekts.

© 2000-2025, MetaQuotes Ltd.

GetProgram

Gibt den Handle eines OpenCL-Programms zurück.

```
int  GetProgram();
```

Rückgabewert

Handle des OpenCL Programms.

© 2000-2025, MetaQuotes Ltd.

Initialize

Initialisiert ein OpenCL-Programm.

```
bool  Initialize(
   const string  program,           // Handle des OpenCL-Programms
   const bool    show_log=true      // im Log anzeigen
   );
```

Parameter

program

[in] Handle des OpenCL-Programms.

show_log=true

[in] Meldungen ins Journal schreiben.

Rückgabewert

Gibt true zurück, wenn die Initialisierung erfolgreich abgeschlossen wurde. Andernfalls gibt false zurück.

© 2000-2025, MetaQuotes Ltd.

KernelCreate

Erstellt den Einsprungpunkt eines OpenCL Programms nach dem angegebenen Index.

```
bool  KernelCreate(
   const int     kernel_index,     // Index des Kernels
   const string  kernel_name       // Name des Kernels
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

kernel_name

[in] Name des Kernel-Objekts

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

KernelFree

Löscht die OpenCL Startfunktion nach dem angegebenen Index.

```
bool  KernelFree(
   const int  kernel_index      // Index des Kernels
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetArgument

Setzt einen Parameter für die OpenCL Funktion nach dem angegebenen Index.

```
template<typename T>
bool  SetArgument(
   const int  kernel_index,     // Index des Kernels
   const int  arg_index,        // Index des Funktionsarguments
   T          value             // Quellcode
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

arg_index

[in] Index des Funktionsarguments.

value

[in] Wert des Funktionsarguments.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetArgumentBuffer

Setzt einen OpenCL Puffer als Parameter einer OpenCL Funktion nach dem angegebenen Index.

```
bool  SetArgumentBuffer(
   const int  kernel_index,     // Index des Kernels
   const int  arg_index,        // Index des Funktionsarguments
   const int  buffer_index      // Index des Puffers
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

arg_index

[in] Index des Funktionsarguments.

buffer_index

[in] Pufferindex.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetArgumentLocalMemory

Setzt einen Parameter im lokalen Speicher für eine OpenCL Funktion nach dem angegebenen Index.

```
bool  SetArgumentLocalMemory(
   const int  kernel_index,          // Index des Kernels
   const int  arg_index,             // Index des Funktionsarguments
   const int  local_memory_size      // Größe des lokalen Speichers
   );
```

Parameter

kernel_index

[in] Index des Kernel-Objekts.

arg_index

[in] Index des Funktionsarguments.

local_memory_size

[in] Größe des lokalen Speichers.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetBuffersCount

Setzt die Anzahl von Puffern.

```
bool  SetBuffersCount(
   const int  total_buffers      // Anzahl von Puffern
   );
```

Parameter

total_buffers

[in] Gesamtzahl von Puffern.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetKernelsCount

Setzt die Anzahl von Kernel-Objekten.

```
bool  SetKernelsCount(
   const int  total_kernels      // Anzahl von Kernel
   );
```

Parameter

total_kernels

[in] Gesamtzahl von Kernel.

Rückgabewert

Wenn erfolgreich, gibt true zurück, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Shutdown

Entfernt ein OpenCL-Programm aus dem Speicher.

```
void  Shutdown();
```

Rückgabewert

Keine.

© 2000-2025, MetaQuotes Ltd.

SupportDouble

Ermittelt, ob reelle Datentypen auf dem Gerät unterstützt werden.

```
bool  SupportDouble();
```

Rückgabewert

Gibt true zurück, wenn reelle Datentypen auf dem Gerät unterstützt werden.

© 2000-2025, MetaQuotes Ltd.

Die Basisklasse CObject

CObject ist eine Basisklasse der Standardbibliothek MQL5.

Beschreibung

Klasse CObject bietet allen ihren abgeleiteten Klassen die Möglichkeit ein Element einer verketteten Liste zu sein. Außerdem werden eine Reihe von virtuellen Methoden für die weitere Umsetzung in abgeleiteten Klassen definiert.

Deklaration

```
class CObject
```

Kopf

```
#include <Object.mqh>
```

Vererbungshierarchie

CObject

Direkte Ableitungen

CAccountInfo, CArray, CChart, CChartObject, CCurve, CDealInfo, CDictionary_Obj_Double, CDictionary_Obj_Obj, CDictionary_String_Obj, CExpertBase, CFile, CHistoryOrderInfo, CList, COrderInfo, CPositionInfo, CString, CSymbolInfo, CTerminalInfo, CTrade, CTreeNode, CWnd, ICondition, IExpression, IMembershipFunction, INamedValue, IParsableRule

Gruppen der Klassenmethode

Attribute	
Prev	Erhält den Wert des vorhergehenden Elements
Prev	Setzt den Wert des vorhergehenden Elements
Next	Erhält den Wert des nachfolgenden Elements
Next	Setzt den Wert des nachfolgenden Elements
Vergleich	
virtual Compare	Gibt das Ergebnis des Vergleichs zum einen anderen Objekt zurück
Eingabe/Ausgabe	
virtual Save	Schreibt ein Objekt in eine Datei
virtual Load	Liest ein Objekt aus einer Datei
virtual Type	Gibt den Typ des Objekts zurück

© 2000-2025, MetaQuotes Ltd.

Prev

Erhält einen Zeiger auf das vorhergehende Element in der Liste.

```
CObject*  Prev()
```

Rückgabewert

Ein Zeiger auf das vorhergehende Element in der Liste. Wenn es das erste Element in der Liste ist, wird NULL zurückgegeben.

Beispiel:

```
//--- example for CObject::Prev()
#include <Object.mqh>
//---
void OnStart()
  {
   CObject *object_first,*object_second;
   //---
   object_first=new CObject;
   if(object_first==NULL)
     {
      printf("Object create error");
      return;
     }
   object_second=new CObject;
   if(object_second==NULL)
     {
      printf("Object create error");
      delete object_first;
      return;
     }
   //--- set interconnect
   object_first.Next(object_second);
   object_second.Prev(object_first);
   //--- use prev object
   CObject *object=object_second.Prev();
   //--- delete objects
   delete object_first;
   delete object_second;
  }
```

© 2000-2025, MetaQuotes Ltd.

Prev

Setzt einen Zeiger auf das vorhergehende Element in der Liste.

```
void  Prev(
   CObject*  object      // Zeiger auf das vorhergehende Element in der Liste
   )
```

Parameter

object

[in] Der neue Wert des Zeigers auf das nachfolgende Element in der Liste.

Beispiel:

```
//--- example for CObject::Prev(CObject*)
#include <Object.mqh>
//---
void OnStart()
  {
   CObject *object_first,*object_second;
   //---
   object_first=new CObject;
   if(object_first==NULL)
     {
      printf("Object create error");
      return;
     }
   object_second=new CObject;
   if(object_second==NULL)
     {
      printf("Object create error");
      delete object_first;
      return;
     }
   //--- set interconnect
   object_first.Next(object_second);
   object_second.Prev(object_first);
   //--- use objects
   //--- ...
   //--- delete objects
   delete object_first;
   delete object_second;
  }
```

© 2000-2025, MetaQuotes Ltd.

Next

Erhält einen Zeiger auf das nachfolgende Element in der Liste.

```
CObject*  Next()
```

Rückgabewert

Ein Zeiger auf das nachfolgende Element in der Liste. Wenn es das letzte Element in der Liste ist, wird NULL zurückgegeben.

Beispiel:

```
//--- example for CObject::Next()
#include <Object.mqh>
//---
void OnStart()
  {
   CObject *object_first,*object_second;
   //---
   object_first=new CObject;
   if(object_first==NULL)
     {
      printf("Object create error");
      return;
     }
   object_second=new CObject;
   if(object_second==NULL)
     {
      printf("Object create error");
      delete object_first;
      return;
     }
   //--- set interconnect
   object_first.Next(object_second);
   object_second.Prev(object_first);
   //--- use next object
   CObject *object=object_first.Next();
   //--- delete objects
   delete object_first;
   delete object_second;
  }
```

© 2000-2025, MetaQuotes Ltd.

Next

Setzt einen Zeiger auf das nächste Element in der Liste.

```
void  Next(
   CObject*  object      // Zeiger auf das nächste Element in der Liste
   )
```

Parameter

object

[in] Der neue Wert des Zeigers auf das nächste Element in der Liste.

Beispiel:

```
//--- example for CObject::Next(CObject*)
#include <Object.mqh>
//---
void OnStart()
  {
   CObject *object_first,*object_second;
   //---
   object_first=new CObject;
   if(object_first==NULL)
     {
      printf("Object create error");
      return;
     }
   object_second=new CObject;
   if(object_second==NULL)
     {
      printf("Object create error");
      delete object_first;
      return;
     }
   //--- set interconnect
   object_first.Next(object_second);
   object_second.Prev(object_first);
   //--- use objects
   //--- ...
   //--- delete objects
   delete object_first;
   delete object_second;
  }
```

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht Daten eines Elements in der Liste mit den Daten eines anderen Listenelements.

```
virtual int  Compare(
   const CObject*  node,        // Element
   const int       mode=0       // Variante
   ) const
```

Parameter

node

[in] Zeiger auf ein Listenelement zum Vergleich

mode=0

[in] Vergleich-Variante

Rückgabewert

0 - im Falle die Elemente in der Liste gleich sind, -1 - wenn das Element in der Liste kleiner als der Knoten zum Vergleich (node) ist, 1 - wenn das Element in der Liste größer als ein Knoten für den Vergleich (node) ist.

Hinweis

Die Vergleichsmethode Compare() in der Klasse CObject gibt immer 0 zurück und führt keine Aktionen aus. Wenn Sie Daten der abgeleiteten Klasse vergleichen möchten, kann die Vergleichsmethode Compare(...) implementiert werden. Verwenden Sie den Parameter mode, um den multivariaten Vergleich zu implementieren.

Beispiel:

```
//--- example for CObject::Compare(...)
#include <Object.mqh>
//---
void OnStart()
  {
   CObject *object_first,*object_second;
   //---
   object_first=new CObject;
   if(object_first==NULL)
     {
      printf("Object create error");
      return;
     }
   object_second=new CObject;
   if(object_second==NULL)
     {
      printf("Object create error");
      delete object_first;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- set interconnect
   object_first.Next(object_second);
   object_second.Prev(object_first);
   //--- compare objects
   int result=object_first.Compare(object_second);
   //--- delete objects
   delete object_first;
   delete object_second;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Daten des Listenelements in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen() früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Methode Save(int) in der Klasse CObject gibt immer true zurück und führt keine Aktionen aus. Wenn Sie Daten der abgeleiteten Klasse in eine Datei speichern möchten, kann die Methode Save(int) implementiert werden.

Beispiel:

```
//--- example for CObject::Save(int)
#include <Object.mqh>
//---
void OnStart()
  {
   int    file_handle;
   CObject *object=new CObject;
   //---
   if(object!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- set objects data
   //--- . . .
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!object.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete object;
         FileClose(file_handle);
```

© 2000-2025, MetaQuotes Ltd.

```
         //---
         return;
        }
      FileClose(file_handle);
     }
   delete object;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Elementdaten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen() früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Methode Load(int) in der Klasse CObject gibt immer true zurück und führt keine Aktionen aus. Wenn Sie Daten der abgeleiteten Klasse aus einer Datei laden möchten, kann die Methode Load(int) implementiert werden.

Beispiel:

```
//--- example for CObject::Load(int)
#include <Object.mqh>
//---
void OnStart()
  {
   int    file_handle;
   CObject *object=new CObject;
   //---
   if(object!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!object.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete object;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
      }
   //--- use object
   //--- . . .
   delete object;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Typenidentifikator.

```
virtual int  Type() const
```

Rückgabewert

Typenidentifikator (für CObject - 0).

Beispiel:

```
//--- example for CObject::Type()
#include <Object.mqh>
//---
void OnStart()
  {
   CObject *object=new CObject;
   //---
   object=new CObject;
   if(object ==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get objects type
   int type=object.Type();
   //--- delete object
   delete object;
  }
```

© 2000-2025, MetaQuotes Ltd.

Datenstrukturen

Dieser Abschnitt enthält die technischen Details der Arbeit mit verschiedenen Datenstrukturen (Arrays, verkettete Listen, etc.) und die Beschreibungen der Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen von Datenstrukturen, sparen Sie Zeit bei der Entwicklung von verschiedenen benutzerdefinierten Data-Warehouses (einschließlich Verbundwerkdatenstrukturen).

Die Standardbibliothek MQL5 (in Bezug auf Datensätze) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Arrays.

Daten-Arrays

Mit den Klassen von dynamischen Daten-Arrays sparen Sie Zeit bei der Entwicklung von verschiedenen benutzerdefinierten Data-Warehouses (mehrdimensionale Arrays).

Die Standardbibliothek MQL5 (in Bezug auf Daten-Arrays) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Arrays.

Klasse	Beschreibung
CArray	Die Basisklasse des dynamischen Daten-Arrays
CArrayChar	Ein dynamisches Array von Variablen vom Typ char oder uchar
CArrayShort	Ein dynamisches Array von Variablen vom Typ short har oder ushort
CArrayInt	Ein dynamisches Array von Variablen vom Typ int oder uint
CArrayLong	Ein dynamisches Array von Variablen vom Typ long oder ulong
CArrayFloat	Ein dynamisches Array von Variablen vom Typ float
CArrayDouble	Ein dynamisches Array von Variablen vom Typ double
CArrayString	Ein dynamisches Array von Variablen vom Typ string
Die Basisklasse der Objekten CArrayObj	Ein dynamisches Array von Zeigern CObject
Die Basisklasse der Liste CList	Bietet die Möglichkeit, mit einer Liste der Instanzen der CObject-Klasse und ihrer Nachfolger zu arbeiten
CTreeNode	Bietet die Möglichkeit, mit den Knoten eines binären Baums CTree zu arbeiten

© 2000-2025, MetaQuotes Ltd.

Klasse	Beschreibung
CTree	Bietet die Möglichkeit, mit dem binären Baum der Instanzen der CTreeNode-Klasse und ihrer Nachfolger zu arbeiten

© 2000-2025, MetaQuotes Ltd.

CArray

CArray ist eine Basisklasse des dynamischen Arrays von Variablen.

Beschreibung

Klasse CArray wird für die Arbeit mit dem dynamischen Array von Variablen verwendet: Speicherzuweisung, Sortieren und die Arbeit mit Dateien.

Deklaration

```
class CArray : public CObject
```

Kopf

```
#include <Arrays\Array.mqh>
```

Vererbungshierarchie

CObject

CArray

Direkte Ableitungen

CArrayChar, CArrayDouble, CArrayFloat, CArrayInt, CArrayLong, CArrayObj, CArrayShort, CArrayString

Gruppen der Klassenmethode

Attribute	
Step	Erhält den Inkrement-Schritt der Arraygröße
Step	Setzt den Inkrement-Schritt des Arraygröße
Total	Erhält die Anzahl der Elementen im Array
Available	Erhält die Anzahl der freien Array-Elementen, die ohne zusätzliche Speicherzuweisung verfügbar sind
Max	Erhält die maximal mögliche Größe des Arrays ohne Neuzuweisung von Speicher
IsSorted	Erhält ein Zeichen der Sortierung des Arrays von bestimmten Variante
SortMode	Erhält den Sortiertyp eines Arrays
Reinigungsmethode	
Clear	Löscht alle Elemente des Arrays ohne Speicherfreigabe
Sortierung	
Sort	Sortiert ein Array nach der Variante

© 2000-2025, MetaQuotes Ltd.

Attribute	
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Step

Erhält den Inkrement-Schritt der Arraygröße.

```
int  Step() const
```

Rückgabewert

Inkrement-Schritt der Arraygröße.

Beispiel:

```
//--- example for CArray::Step()
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get resize step
   int step=array.Step();
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Step

Setzt den Inkrement-Schritt des Arraygröße.

```
bool  Step(
   int  step      // Schritt
   )
```

Parameter

step

[in] Der neue Wert des Schritts.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch einen Schritt weniger als oder gleich auf Null zu setzen.

Beispiel:

```
//--- example for CArray::Step(int)
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- set resize step
   bool result=array.Step(1024);
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Total

Erhält die Anzahl der Elementen im Array.

```
int  Total() const;
```

Rückgabewert

Die Anzahl der Elementen im Array.

Beispiel:

```
//--- example for CArray::Total()
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check total
   int total=array.Total();
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Available

Erhält die Anzahl der freien Array-Elementen, die ohne zusätzliche Speicherzuweisung verfügbar sind.

```
int  Available() const
```

Rückgabewert

Die Anzahl der freien Array-Elementen, die ohne zusätzliche Speicherzuweisung verfügbar sind.

Beispiel:

```
//--- example for CArray::Available()
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check available
   int available=array.Available();
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Max

Erhält die maximal mögliche Größe des Arrays ohne Neuzuweisung von Speicher.

```
int  Max() const
```

Rückgabewert

Die maximal mögliche Größe des Arrays ohne Neuzuweisung von Speicher.

Beispiel:

```
//--- example for CArray::Max()
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check maximum size
   int max=array.Max();
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

IsSorted

Erhält ein Zeichen der Sortierung des Arrays von bestimmten Variante.

```
bool  IsSorted(
   int  mode=0      // Sortieroption
   ) const
```

Parameter

mode=0

[in] Geprüfte Sortieroption.

Rückgabewert

Flag der Listesortierung. Wenn eine Liste nach der angegebenen Variante sortiert ist, wird true zurückgegeben, ansonsten false.

Hinweis

Das Zeichen der Sortierung des Arrays kann nicht direkt geändert werden. Das Zeichen der Sortierung wird durch die Methode Sort() gesetzt werden, und wird durch eine der Methoden zum Hinzufügen/Einfügen außer InsertSort(...) zurückgesetzt.

Beispiel:

```
//--- example for CArray::IsSorted()
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check sorted
   if(array.IsSorted())
     {
      //--- use methods for sorted array
      //--- ...
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SortMode

Erhält den Sortiertyp eines Arrays

```
int  SortMode() const;
```

Rückgabewert

Sortiertyp

Beispiel:

```
//--- example for CArray::SortMode()
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check sort mode
   int sort_mode=array.SortMode();
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Clear

Deletes all of the array elements without memory release.

```
void  Clear()
```

Return Value

None.

Example:

```
//--- example for CArray::Clear()
#include <Arrays\Array.mqh>
//---
void OnStart()
 {
 CArray *array=new CArray;
 //---
 if(array==NULL)
 {
 printf("Object create error");
 return;
 }
 //--- use array
 //--- ...
 //--- clear array
 array.Clear();
 //--- delete array
 delete array;
 }
```

© 2000-2025, MetaQuotes Ltd.

Sort

Sortiert ein Array nach der Variante.

```
void  Sort(
   int  mode=0      // Sortieroption
   )
```

Parameter

mode=0

[in] Array-Sortiertyp.

Rückgabewert

Nichts.

Hinweis

Arrays werden immer in aufsteigender Reihenfolge sortiert. Für die Arrays vom einfachen Datentypen (CArrayChar, CArrayShort etc.) wird der Parameter mode nicht verwendet. Für das Array CArrayObj müssen die multivarianten Sortierungen in der Sort(int)-Methode von abgeleiteten Klassen implementiert werden.

Beispiel:

```
//--- example for CArray::Sort(int)
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   CArray *array=new CArray;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- sorting by mode 0
   array.Sort(0);
   //--- use array
   //--- ...
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArray::Save(int)
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   int     file_handle;
   CArray *array=new CArray;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- delete array
   delete array;
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArray::Load(...)
#include <Arrays\Array.mqh>
//---
void OnStart()
  {
   int     file_handle;
   CArray *array=new CArray;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- delete array
   delete array;
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

CArrayChar

CArrayChar ist eine Klasse des dynamischen Arrays von Variablen vom Typ char oder uchar.

Beschreibung

Klasse CArrayChar bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ char oder uchar zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayChar : public CArray
```

Kopf

```
#include <Arrays\ArrayChar.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayChar

Gruppen der Klassenmethode

Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
	Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayChar::Reserve(int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayChar::Resize(int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayChar::Shutdown()
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   char  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayChar::Add(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(i))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const char&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayChar::AddArray(const char &[])
#include <Arrays\ArrayChar.mqh>
//---
char src[];
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayChar*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayChar-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayChar::AddArray(const CArrayChar*)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayChar *src=new CArrayChar;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   char  element,     // Element
   int   pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayChar::Insert(char,int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(i,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const char&  src[],     // Quelle-Array
   int           pos       // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayChar::InsertArray(const char &[],int)
#include <Arrays\ArrayChar.mqh>
//---
char src[];
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayChar*  src,      // Ein Zeiger auf die Quelle
   int           pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayChar-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayChar::InsertArray(const CArrayChar*,int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayChar *src=new CArrayChar;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const char&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayChar::AssignArray(const char &[])
#include <Arrays\ArrayChar.mqh>
//---
char src[];
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayChar*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayChar-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayChar::AssignArray(const CArrayChar*)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayChar *src  =new CArrayChar;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int   pos,          // Position
   char  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayChar::Update(int,char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,'A'))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayChar::Shift(int,int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayChar::Delete(int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayChar::DeleteRange(int,int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
char  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, CHAR_MAX - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann CHAR_MAX ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayChar::At(int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      char result=array.At(i);
      if(result==CHAR_MAX && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const char&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayChar::CompareArray(const char &[])
#include <Arrays\ArrayChar.mqh>
//---
char src[];
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const CArrayChar*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayChar-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayChar::CompareArray(const CArrayChar*)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayChar *src=new CArrayChar;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   char  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayChar::InsertSort(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort('A'))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::Search(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search('A')!=-1) printf("Element found");
   else                      printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::SearchGreat(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat('A')!=-1) printf("Element found");
   else                           printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::SearchLess(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess('A')!=-1) printf("Element found");
   else                          printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::SearchGreatOrEqual(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual('A')!=-1) printf("Element found");
   else                                  printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::SearchLessOrEqual(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual('A')!=-1) printf("Element found");
   else                                 printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::SearchFirst(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst('A')!=-1) printf("Element found");
   else                           printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayChar::SearchLast(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast('A')!=-1) printf("Element found");
   else                          printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   char  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayChar::SearchLinear(char)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear('A')!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayChar::Save(int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayChar *array=new CArrayChar;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- delete array
   delete array;
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayChar::Load(int)
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayChar *array=new CArrayChar;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       printf("Element[%d] = '%c'",i,array.At(i));
      }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayChar - 77).

Beispiel:

```
//--- example for CArrayChar::Type()
#include <Arrays\ArrayChar.mqh>
//---
void OnStart()
  {
   CArrayChar *array=new CArrayChar;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayShort

CArrayShort ist eine Klasse des dynamischen Arrays von Variablen vom Typ short oder ushort.

Beschreibung

Klasse CArrayShort bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ short oder ushort zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayShort : public CArray
```

Kopf

```
#include <Arrays\ArrayShort.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayShort

Gruppen der Klassenmethode

Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
	Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayShort::Reserve(int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayShort::Resize(int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayShort::Shutdown()
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   short  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayShort::Add(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(i))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const short&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayShort::AddArray(const short &[])
#include <Arrays\ArrayShort.mqh>
//---
short src[];
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayShort*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayShort-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayShort::AddArray(const CArrayShort*)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayShort *src=new CArrayShort;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   short  element,     // Element für Einfügen
   int    pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayShort::Insert(short,int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(i,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const short&  src[],     // Quelle-Array
   int            pos        // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayShort::InsertArray(const short &[],int)
#include <Arrays\ArrayShort.mqh>
//---
short src[];
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayShort*  src,      // Ein Zeiger auf die Quelle
   int            pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayShort-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayShort::InsertArray(const CArrayShort*,int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayShort *src=new CArrayShort;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const short&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayShort::AssignArray(const short &[])
#include <Arrays\ArrayShort.mqh>
//---
short src[];
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayShort*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayShort-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayShort::AssignArray(const CArrayShort*)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayShort *src  =new CArrayShort;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int    pos,          // Position
   short  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayShort::Update(int,short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,100))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayShort::Shift(int,int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayShort::Delete(int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayShort::DeleteRange(int,int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
short  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, SHORT_MAX - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann SHORT_MAX ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayShort::At(int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      short result=array.At(i);
      if(result==SHORT_MAX && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const short&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayShort::CompareArray(const short &[])
#include <Arrays\ArrayShort.mqh>
//---
short src[];
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const CArrayShort*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayShort-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayShort::CompareArray(const CArrayShort*)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayShort *src=new CArrayShort;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   short  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayShort::InsertSort(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort(100))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::Search(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search(100)!=-1) printf("Element found");
   else                      printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::SearchGreat(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat(100)!=-1) printf("Element found");
   else                           printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::SearchLess(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess(100)!=-1) printf("Element found");
   else                          printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::SearchGreatOrEqual(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual(100)!=-1) printf("Element found");
   else                                  printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::SearchLessOrEqual(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual(100)!=-1) printf("Element found");
   else                                 printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::SearchFirst(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst(100)!=-1) printf("Element found");
   else                           printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayShort::SearchLast(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast(100)!=-1) printf("Element found");
   else                          printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   short  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayShort::SearchLinear(short)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear(100)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayShort::Save(int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayShort *array=new CArrayShort;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      array.Add(i);
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
      }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayShort::Load(int)
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayShort *array=new CArrayShort;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      printf("Element[%d] = %d",i,array.At(i));
     }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayShort - 82).

Beispiel:

```
//--- example for CArrayShort::Type()
#include <Arrays\ArrayShort.mqh>
//---
void OnStart()
  {
   CArrayShort *array=new CArrayShort;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayInt

CArrayInt ist eine Klasse des dynamischen Arrays von Variablen vom Typ int oder uint.

Beschreibung

Klasse CArrayInt bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ int oder uint zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayInt : public CArray
```

Kopf

```
#include <Arrays\ArrayInt.mqh>
```

Vererbungshierarchie

CObject

CArray

CArrayInt

Direkte Ableitungen

CSpreadBuffer

Gruppen der Klassenmethode

Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
	Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayInt::Reserve(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayInt::Resize(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayInt::Shutdown()
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   int  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayInt::Add(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(i))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const int&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayInt::AddArray(const int &[])
#include <Arrays\ArrayInt.mqh>
//---
int src[];
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayInt*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayInt-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayInt::AddArray(const CArrayInt*)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayInt *src=new CArrayInt;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   int  element,     // Element für Einfügen
   int  pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayInt::Insert(int,int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(i,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const int&  src[],     // Quelle-Array
   int          pos        // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayInt::InsertArray(const int &[],int)
#include <Arrays\ArrayInt.mqh>
//---
int src[];
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayInt*  src,      // Ein Zeiger auf die Quelle
   int          pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayInt-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayInt::InsertArray(const CArrayInt*,int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayInt *src=new CArrayInt;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const int&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayInt::AssignArray(const int &[])
#include <Arrays\ArrayInt.mqh>
//---
int src[];
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayInt*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayInt-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayInt::AssignArray(const CArrayInt*)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayInt *src  =new CArrayInt;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int  pos,          // Position
   int  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayInt::Update(int,int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,10000))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayInt::Shift(int,int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayInt::Delete(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayInt::DeleteRange(int,int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
int  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, INT_MAX - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann INT_MAX ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayInt::At(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      int result=array.At(i);
      if(result==INT_MAX && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const int&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayInt::CompareArray(const int &[])
#include <Arrays\ArrayInt.mqh>
//---
int src[];
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const CArrayInt*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayInt-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayInt::CompareArray(const CArrayInt*)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayInt *src=new CArrayInt;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   int  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayInt::InsertSort(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort(10000))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt::Search(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search(10000)!=-1) printf("Element found");
   else                        printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt::SearchGreat(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat(10000)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt::SearchLess(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess(10000)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt::SearchGreatOrEqual(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual(10000)!=-1) printf("Element found");
   else                                    printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt::SearchLessOrEqual(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual(10000)!=-1) printf("Element found");
   else                                   printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt:: SearchFirst(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst(10000)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayInt::SearchLast(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast(10000)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   int  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayInt::SearchLinear(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear(10000)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayInt::Save(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   int        file_handle;
   CArrayInt *array=new CArrayInt;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      array.Add(i);
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
      }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayInt::Load(int)
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   int        file_handle;
   CArrayInt *array=new CArrayInt;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       printf("Element[%d] = %d",i,array.At(i));
      }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayInt - 82).

Beispiel:

```
//--- example for CArrayInt::Type()
#include <Arrays\ArrayInt.mqh>
//---
void OnStart()
  {
   CArrayInt *array=new CArrayInt;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayLong

CArrayLong ist eine Klasse des dynamischen Arrays von Variablen vom Typ long oder ulong.

Beschreibung

Klasse CArrayLong bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ long oder ulong zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayLong : public CArray
```

Kopf

```
#include <Arrays\ArrayLong.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayLong

Direkte Ableitungen

CRealVolumeBuffer, CTickVolumeBuffer, CTimeBuffer

Gruppen der Klassenmethode

Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
	Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayLong::Reserve(int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayLong::Resize(int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayLong::Shutdown()
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   long  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayLong::Add(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(i))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const long&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayLong::AddArray(const long &[])
#include <Arrays\ArrayLong.mqh>
//---
long src[];
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayLong*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayLong-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayLong::AddArray(const CArrayLong*)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayLong *src=new CArrayLong;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   long  element,     // Element für Einfügen
   int   pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayLong::Insert(long,int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(i,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const long&  src[],     // Quelle-Array
   int           pos        // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayLong::InsertArray(const long &[],int)
#include <Arrays\ArrayLong.mqh>
//---
long src[];
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayLong*  src,      // Ein Zeiger auf die Quelle
   int           pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayLong-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayLong::InsertArray(const CArrayLong*,int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayLong *src=new CArrayLong;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const long&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayLong::AssignArray(const long &[])
#include <Arrays\ArrayLong.mqh>
//---
long src[];
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayLong*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayLong-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayLong::AssignArray(const CArrayLong*)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayLong *src  =new CArrayLong;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int   pos,          // Position
   long  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayLong::Update(int,long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,1000000))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayLong::Shift(int,int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayLong::Delete(int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayLong::DeleteRange(int,int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
long  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, LONG_MAX - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann LONG_MAX ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayLong::At(int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      long result=array.At(i);
      if(result==LONG_MAX && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const long&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayLong::CompareArray(const long &[])
#include <Arrays\ArrayLong.mqh>
//---
long src[];
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArrayconst

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArrayconst(
   const CArrayLong*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayLong-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayLong::CompareArray(const CArrayLong*)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayLong *src=new CArrayLong;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   long  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayLong::InsertSort(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort(1000000))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::Search(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search(1000000)!=-1) printf("Element found");
   else                          printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::SearchGreat(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat(1000000)!=-1) printf("Element found");
   else                               printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::SearchLess(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess(1000000)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::SearchGreatOrEqual(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual(1000000)!=-1) printf("Element found");
   else                                      printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::SearchLessOrEqual(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual(1000000)!=-1) printf("Element found");
   else                                     printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::SearchFirst(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst(1000000)!=-1) printf("Element found");
   else                               printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayLong::SearchLast(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast(1000000)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   long  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayLong::SearchLinear(long)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear(1000000)!=-1) printf("Element found");
   else                                printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayLong::Save(int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayLong *array=new CArrayLong;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      array.Add(i);
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
     }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayLong::Load(int)
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayLong *array=new CArrayLong;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       printf("Element[%d] = %I64",i,array.At(i));
      }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayFloat - 84).

Beispiel:

```
//--- example for CArrayLong::Type()
#include <Arrays\ArrayLong.mqh>
//---
void OnStart()
  {
   CArrayLong *array=new CArrayLong;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayFloat

CArrayFloat ist eine Klasse des dynamischen Arrays von Variablen vom Typ float.

Beschreibung

Klasse CArrayFloat bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ float zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayFloat : public CArray
```

Kopf

```
#include <Arrays\ArrayFloat.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayFloat

Gruppen der Klassenmethode

Attribute	
Delta	Setzt die Vergleich-Toleranz
Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen

© 2000-2025, MetaQuotes Ltd.

Attribute	
	Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Attribute	
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Delta

Setzt die Vergleich-Toleranz.

```
void  Delta(
   float  delta      // Toleranz
   )
```

Parameter

delta

[in] Der neue Wert der Vergleich-Toleranz.

Rückgabewert

Kein

Hinweis

Die Vergleich-Toleranz wird bei der Suche verwendet. Die Werte sind gleich, wenn die Differenz kleiner oder gleich der Toleranz ist. Die Standardtoleranz ist 0,0.

Beispiel:

```
//--- example for CArrayFloat::Delta(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- set compare variation
   array.Delta(0.001);
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayFloat::Reserve(int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayFloat::Resize(int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayFloat::Shutdown()
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   float  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayFloat::Add(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(i))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const float&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayFloat::AddArray(const float &[])
#include <Arrays\ArrayFloat.mqh>
//---
float src[];
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayFloat*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayFloat-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayFloat::AddArray(const CArrayFloat*)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayFloat *src=new CArrayFloat;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   float  element,     // Element für Einfügen
   int    pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayFloat::Insert(float,int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(i,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const float&  src[],     // Quelle-Array
   int            pos        // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayFloat::InsertArray(const float &[],int)
#include <Arrays\ArrayFloat.mqh>
//---
float src[];
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayFloat*  src,      // Ein Zeiger auf die Quelle
   int            pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayFloat-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayFloat::InsertArray(const CArrayFloat*,int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayFloat *src=new CArrayFloat;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const float&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayFloat::AssignArray(const float &[])
#include <Arrays\ArrayFloat.mqh>
//---
float src[];
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayFloat*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayFloat-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayFloat::AssignArray(const CArrayFloat*)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayFloat *src  =new CArrayFloat;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int    pos,          // Position
   float  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayFloat::Update(int,float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,100.0))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayFloat::Shift(int,int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayFloat::Delete(int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayFloat::DeleteRange(int,int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
float  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, FLT_MAX - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann FLT_MAX ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayFloat::At(int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      float result=array.At(i);
      if(result==FLT_MAX && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const float&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayFloat::CompareArray(const float &[])
#include <Arrays\ArrayFloat.mqh>
//---
float src[];
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArrayconst

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArrayconst(
   const CArrayFloat*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayFloat-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayFloat::CompareArray(const CArrayFloat*)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayFloat *src=new CArrayFloat;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   float  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayFloat::InsertSort(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort(100.0))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat::Search(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search(100.0)!=-1) printf("Element found");
   else                        printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat::SearchGreat(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat(100.0)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat:: SearchLess(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess(100.0)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat::SearchGreatOrEqual(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual(100.0)!=-1) printf("Element found");
   else                                    printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat::SearchLessOrEqual(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual(100.0)!=-1) printf("Element found");
   else                                   printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat::SearchFirst(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst(100.0)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayFloat::SearchLast(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast(100.0)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   float  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayFloat::SearchLinear(float)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear(100.0)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayFloat::Save(int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      array.Add(i);
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
      }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayFloat::Load(int)
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
     {
      printf("Element[%d] = %f",i,array.At(i));
     }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayFloat - 86).

Beispiel:

```
//--- example for CArrayFloat::Type()
#include <Arrays\ArrayFloat.mqh>
//---
void OnStart()
  {
   CArrayFloat *array=new CArrayFloat;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayDouble

CArrayDouble ist eine Klasse des dynamischen Arrays von Variablen vom Typ double.

Beschreibung

Klasse CArrayDouble bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ double zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayDouble : public CArray
```

Kopf

```
#include <Arrays\ArrayDouble.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble

Direkte Ableitungen

CDoubleBuffer

Gruppen der Klassenmethode

Attribute	
Delta	Setzt die Vergleich-Toleranz
Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu

© 2000-2025, MetaQuotes Ltd.

Attribute	
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Suche nach Extrema	
Minimum	Erhält den Index des Minimalwerts im bestimmten Bereich
Maximum	Erhält den Index des Maximalwerts im bestimmten Bereich
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein

© 2000-2025, MetaQuotes Ltd.

Attribute	
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Delta

Setzt die Vergleich-Toleranz.

```
void  Delta(
   double  delta      // Toleranz
   )
```

Parameter

delta

[in] Der neue Wert der Vergleich-Toleranz.

Rückgabewert

Kein

Hinweis

Die Vergleich-Toleranz wird bei der Suche verwendet. Die Werte sind gleich, wenn die Differenz kleiner oder gleich der Toleranz ist. Die Standardtoleranz ist 0,0.

Beispiel:

```
//--- example for CArrayDouble::Delta(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- set compare variation
   array.Delta(0.001);
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayDouble::Reserve(int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayDouble::Resize(int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayDouble::Shutdown()
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   double  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayDouble::Add(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(i))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const double&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayDouble::AddArray(const double &[])
#include <Arrays\ArrayDouble.mqh>
//---
double src[];
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayDouble*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayDouble-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayDouble::AddArray(const CArrayDouble*)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayDouble *src=new CArrayDouble;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   double  element,     // Element für Einfügen
   int     pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayDouble::Insert(double,int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(i,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const double&  src[],     // Quelle Array
   int            pos        // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayDouble::InsertArray(const double &[],int)
#include <Arrays\ArrayDouble.mqh>
//---
double src[];
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayDouble*  src,      // Ein Zeiger auf die Quelle
   int            pos       // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayDouble-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayDouble::InsertArray(const CArrayDouble*,int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayDouble *src=new CArrayDouble;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const double&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayDouble::AssignArray(const double &[])
#include <Arrays\ArrayDouble.mqh>
//---
double src[];
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayDouble*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayDouble-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayDouble::AssignArray(const CArrayDouble*)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayDouble *src  =new CArrayDouble;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int     pos,          // Position
   double  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayDouble::Update(int,double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,100.0))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayDouble::Shift(int,int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayDouble::Delete(int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayDouble::DeleteRange(int,int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
double  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, DBL_MAX - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann DBL_MAX ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayDouble::At(int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      double result=array.At(i);
      if(result==DBL_MAX && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const double&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayDouble::CompareArray(const double &[])
#include <Arrays\ArrayDouble.mqh>
//---
double src[];
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const CArrayDouble*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayDouble-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayDouble::CompareArray(const CArrayDouble*)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayDouble *src=new CArrayDouble;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Minimum

Erhält den Index des minimalen Array-Elements im angegebenen Bereich.

```
int  Minimum(
   int  start,      // Startindex
   int  count       // Anzahl
   ) const
```

Parameter

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

Rückgabewert

Der Index des minimalen Elements im angegebenen Bereich.

© 2000-2025, MetaQuotes Ltd.

Maximum

Erhält den Index des maximalen Array-Elements im angegebenen Bereich.

```
int  Maximum(
   int  start,      // Startindex
   int  count       // Anzahl
   ) const
```

Parameter

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

Rückgabewert

Der Index des maximalen Elements im angegebenen Bereich.

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   double  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayDouble::InsertSort(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort(100.0))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble::Search(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search(100.0)!=-1) printf("Element found");
   else                        printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble::SearchGreat(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat(100.0)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble:: SearchLess(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess(100.0)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble::SearchGreatOrEqual(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual(100.0)!=-1) printf("Element found");
   else                                    printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble::SearchLessOrEqual(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual(100.0)!=-1) printf("Element found");
   else                                   printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble::SearchFirst(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst(100.0)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayDouble::SearchLast(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast(100.0)!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   double  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayDouble::SearchLinear(double)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear(100.0)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayDouble::Save(int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   int           file_handle;
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      array.Add(i);
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
      }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayDouble::Load(int)
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   int           file_handle;
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       printf("Element[%d] = %f",i,array.At(i));
      }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayDouble - 87).

Beispiel:

```
//--- example for CArrayDouble::Type()
#include <Arrays\ArrayDouble.mqh>
//---
void OnStart()
  {
   CArrayDouble *array=new CArrayDouble;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayString

CArrayString ist eine Klasse des dynamischen Arrays von Variablen vom Typ string.

Beschreibung

Klasse CArrayString bietet die Möglichkeit mit einem dynamischen Array von Variablen vom Typ string zu arbeiten. Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Deklaration

```
class CArrayString : public CArray
```

Kopf

```
#include <Arrays\ArrayString.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayString

Gruppen der Klassenmethode

Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Speicherfreigabe
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
	Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Speicherverwaltung	
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
SearchLinear	Sucht nach einem Element in einem Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayString::Reserve(int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reserve memory
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayString::Resize(int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Speicherfreigabe.

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayString::Shutdown()
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   string  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Beispiel:

```
//--- example for CArrayString::Add(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(IntegerToString(i)))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const string&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayString::AddArray(const string &[])
#include <Arrays\ArrayString.mqh>
//---
string src[];
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayString*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayString-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Beispiel:

```
//--- example for CArrayString::AddArray(const CArrayString*)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayString *src=new CArrayString;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   string  element,     // Element für Einfügen
   int     pos          // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayString::Insert(string,int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(IntegerToString(i),0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const string&  src[],     // Quelle-Array
   int             pos        // Position
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayString::InsertArray(const string &[],int)
#include <Arrays\ArrayString.mqh>
//---
string src[];
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   CArrayString*  src,      // Ein Zeiger auf die Quelle
   int             pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayString-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Beispiel:

```
//--- example for CArrayString::InsertArray(const CArrayString*,int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayString *src=new CArrayString;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const string&  src[]      // Quelle-Array
   )
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayString::AssignArray(const string &[])
#include <Arrays\ArrayString.mqh>
//---
string src[];
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayString*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayString-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Beispiel:

```
//--- example for CArrayString::AssignArray(const CArrayString*)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayString *src  =new CArrayString;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   //--- arrays is identical
   //--- delete source array
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int     pos,          // Position
   string  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Beispiel:

```
//--- example for CArrayString::Update(int, string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,"ABC"))
     {
      printf("Update error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayString::Shift(int,int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Beispiel:

```
//--- example for CArrayString::Delete(int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete element
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Beispiel:

```
//--- example for CArrayString::DeleteRange(int,int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
string  At(
   int  pos      // Position
   ) const
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, "" - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten (in diesem Fall ist der letzte Fehlercode ERR_OUT_OF_RANGE).

Hinweis

Natürlich kann "" ein gültiger Wert des Array-Elements sein, darum überprüfen Sie immer den letzten Fehlercode, wenn Sie solchen Wert erhalten.

Beispiel:

```
//--- example for CArrayString::At(int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      string result=array.At(i);
      if(result=="" && GetLastError()==ERR_OUT_OF_RANGE)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
```

© 2000-2025, MetaQuotes Ltd.

```
      //--- . . .
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const string&  src[]      // Quelle-Array
   ) const
```

Parameter

src[]

[in] Referenz auf das Quelle-Array.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayString::CompareArray(const string &[])
#include <Arrays\ArrayString.mqh>
//---
string src[];
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArrays(
   const CArrayString*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayString-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayString::CompareArray(const CArrayString*)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayString *src=new CArrayString;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- add source arrays elements
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   string  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Beispiel:

```
//--- example for CArrayString::InsertSort(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort("ABC"))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString::Search(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.Search("ABC")!=-1) printf("Element found");
   else                        printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString::SearchGreat(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreat("ABC")!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString:: SearchLess(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLess("ABC")!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString:: SearchGreatOrEqual(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchGreatOrEqual("ABC")!=-1) printf("Element found");
   else                                    printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString:: SearchLessOrEqual(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLessOrEqual("ABC")!=-1) printf("Element found");
   else                                   printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString:: SearchFirst(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchFirst("ABC")!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayString:: SearchLast(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- search element
   if(array.SearchLast("ABC")!=-1) printf("Element found");
   else                            printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLinear

Sucht nach einem Element in einem Array, das gleich dem Muster ist.

```
int  SearchLinear(
   string  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Hinweis

Die Suche wird als Reihenfolgezugriff auf Array-Elementen durchgeführt (Linearsuche für unsortierten Arrays).

Beispiel:

```
//--- example for CArrayString::SearchLinear(string)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- search element
   if(array.SearchLinear("ABC")!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayString::Save(int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   int           file_handle;
   CArrayString *array=new CArrayString;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      array.Add(IntegerToString(i));
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
```

© 2000-2025, MetaQuotes Ltd.

```
         }
       FileClose(file_handle);
     }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayString::Load(int)
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   int           file_handle;
   CArrayString *array=new CArrayString;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use arrays elements
   for(int i=0;i<array.Total();i++)
```

© 2000-2025, MetaQuotes Ltd.

```
    {
     printf("Element[%d] = '%s'",i,array.At(i));
    }
  delete array;
}
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayString - 89).

Beispiel:

```
//--- example for CArrayString::Type()
#include <Arrays\ArrayString.mqh>
//---
void OnStart()
  {
   CArrayString *array=new CArrayString;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CArrayObj

CArrayObj ist eine Klasse des dynamischen Arrays von Zeigern auf Instanzen von CObject-Klasse und ihre geerbten Klassen.

Beschreibung

CArrayObj bietet die Möglichkeit mit einem dynamischen Array von Zeigern auf Instanzen von CObject-Klasse und ihre geerbten Klassen. Dies erlaubt mit mehrdimensionalen dynamischen Arrays von primitiven Datentypen und mit komplexen Datenstrukturen zu arbeiten.

Die Klasse erlaubt Array-Elementen hinzuzufügen/einzufügen/zu löschen, ein Array zu sortieren, in einem Sortierten Array zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Es gibt bestimmte Arbeitseinzelheiten für die Klasse CArrayObj.

Deklaration

```
class CArrayObj : public CArray
```

Kopf

```
#include <Arrays\ArrayObj.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj

Direkte Ableitungen

CIndicators, CSeries

Gruppen der Klassenmethode

Attribute	
FreeMode	Erhält den Flag von Speicherverwaltung
FreeMode	Setzt den Flag von Speicherverwaltung
Speicherverwaltung	
Reserve	Reserviert Speicher, um die Größe des Arrays zu erhöhen
Resize	Setzt eine neue Größe des Arrays (kleinere)
Shutdown	Klärt ein Array mit vollständige Freigabe vom Arrary-Speicher (nicht seiner Elementen).
Das Erstellen eines neuen Elements	

© 2000-2025, MetaQuotes Ltd.

Attribute	
virtual CreateElement	Erstellt ein neues Element des Arrays an der angegebene Position.
Füllung	
Add	Fügt ein Element am Ende des Arrays hinzu
AddArray	Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu
Insert	Fügt ein Element in das Array an die angegebene Position ein
InsertArray	Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein
AssignArray	Kopiert Elementen aus einem anderen Array ins angegebenen Array
Änderung	
Update	Ändert ein Element an der angegebenen Position des Arrays
Shift	Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung
Löschen	
Detach	Erhält ein Element an der angegebene Position und löscht es vom Array
Delete	Löscht ein Element an der angegebene Position des Arrays
DeleteRange	Löscht eine Gruppe von Elementen an der angegebene Position des Arrays
Clear	Löscht alle Elemente des Arrays ohne Speicherfreigabe.
Zugriff	
At	Erhält ein Element an der angegebene Position des Arrays
Vergleich	
CompareArray	Vergleicht ein Array mit einem anderen Array
Operationen mit einem sortierten Array	
InsertSort	Fügt ein Element in ein sortiertes Array ein
Search	Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Attribute	
SearchGreat	Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist
SearchLess	Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist
SearchGreatOrEqual	Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist
SearchLessOrEqual	Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist
SearchFirst	Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist
SearchLast	Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist
Eingabe/Ausgabe	
virtual Save	Speichert Array-Daten in eine Datei
virtual Load	Lädt Array-Daten aus einer Datei
virtual Type	Erhält die Array-Typ-Identifikator.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Arrays von abgeleiteten Klassen von CObject (einschließlich aller Klassen der Standard-Bibliothek) haben praktische Anwendung.

Betrachten wir zum Beispiel die Implementierung einer zweidimensionalen Anordnung:

```
#include <Arrays\ArrayDouble.mqh>
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   int i,j;
   int first_size=10;
   int second_size=100;
//--- create array
   CArrayObj    *array=new CArrayObj;
   CArrayDouble *sub_array;
//---
   if(array==NULL)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Object create error");
      return;
     }
//--- create subarrays
   for(i=0;i<first_size;i++)
     {
      sub_array=new CArrayDouble;
      if(sub_array==NULL)
        {
         delete array;
         printf("Object create error");
         return;
        }
      //--- fill array
      for(j=0;j<second_size;j++)
        {
         sub_array.Add(i*j);
        }
      array.Add(sub_array);
     }
//--- create array OK
   for(i=0;i<first_size;i++)
     {
      sub_array=array.At(i);
      for(j=0;j<second_size;j++)
        {
         double element=sub_array.At(j);
         //--- use array element
        }
     }
   delete array;
  }
```

Arbeitseinzelheiten

Die Klasse hat einen Mechanismus zur dynamischen Speicherverwaltung, so müssen Sie bei der Arbeit mit den Array-Elementen sehr vorsichtig sein.

Der Speicherverwaltungsmechanismus kann durch die Methode FreeMode(bool) aktiviert/deaktiviert werden. Der Speicherverwaltungsmechanismus ist standardmäßig aktiviert.

Dementsprechend gibt es zwei Möglichkeiten für die Arbeit mit der CArrayObj-Klasse:

1. Der Speicherverwaltungsmechanismus ist aktiviert. (Standard)

In diesem Fall übernimmt CArrayObj die Verantwortung für die Speicherfreigabe von Elementen nach der Entfernung aus dem Array. Das Programm des Benutzers soll nicht die Array-Elemente freigeben.

Anwendungsbeispiel:

© 2000-2025, MetaQuotes Ltd.

```
   int i;
//--- ein Array erstellen
   CArrayObj *array=new CArrayObj;
//--- das Array mit Elementen einfüllen
   for(i=0;i<10;i++) array.Add(new CObject);
//--- etwas tun
   for(i=0;i<array.Total();i++)
     {
      CObject *object=array.At(i);
      //--- Operationen mit dem Element
      . . .
     }
//--- das Array mit Elementen löschen
   delete array;
```

2. Der Speicherverwaltungsmechanismus ist deaktiviert.

In diesem Fall übernimmt CArrayObj keine Verantwortung für die Speicherfreigabe von Elementen nach der Entfernung aus dem Array. Das Programm des Benutzers muss die Array-Elemente freigeben.

Anwendungsbeispiel:

```
   int i;
//--- ein Array erstellen
   CArrayObj *array=new CArrayObj;
//--- den Speicherverwaltungsmechanismus deaktivieren
   array.FreeMode(false);
//--- das Array mit Elementen einfüllen
   for(i=0;i<10;i++) array.Add(new CObject);
//--- etwas tun
   for(i=0;i<array.Total();i++)
     {
      CObject *object=array.At(i);
      //--- Operationen mit dem Element
      . . .
     }
//--- Array-Elemente löschen
   while(array.Total()) delete array.Detach();
//--- Das leere Array löschen
   delete array;
```

© 2000-2025, MetaQuotes Ltd.

FreeMode

Erhält ein Flag von Speicherverwaltung.

```
bool  FreeMode() const
```

Rückgabewert

Ein Flag von Speicherverwaltung.

Beispiel:

```
//--- example for CArrayObj::FreeMode()
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get free mode flag
   bool array_free_mode=array.FreeMode();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

FreeMode

Setzt das Flag von Speicherverwaltung.

```
void  FreeMode(
   bool  mode      // ein neues Flag
   )
```

Parameter

mode

[in] Der neue Wert des Flags von Speicherverwaltung.

Rückgabewert

Nichts.

Hinweis

Setzen vom Speicherverwaltung-Flags ist ein wichtiger Punkt bei der Verwendung von der Klasse CArrayObj. Da die Array-Elemente sind Zeiger auf dynamische Objekte, ist es wichtig zu bestimmen, was mit ihnen beim Löschen aus dem Array zu tun.

Wenn ein Flag gesetzt ist, wird das Element automatisch durch den Operator delete beim Löschen aus dem Array gelöscht werden. Wenn kein Flag gesetzt ist, wird angenommen, dass ein Zeiger auf ein gelöschtes Objekt noch irgendwo im Anwenderprogramm bleibt und wird bei der Programm danach freigegeben.

Wenn das Anwenderprogramm das Flag der Speicherverwaltung löscht, muss der Benutzer die Verantwortung für die Entfernung von Array-Elementen vor dem Ende des Programms verstehen, denn sonst bleibt der Speicher, der durch den Elementen während der Erstellung mit dem new-Operator belegt war, nicht freigegeben.

Wenn Datenmenge groß ist, kann dies zu Fehlfunktionen des Terminals führen. Wenn das Anwenderprogramm löscht nicht das Flag der Speicherverwaltung, gibt es eine andere "Fallgrube".

Verwendung von irgendwo in den lokalen Variablen gespeicherten Array-Elemente-Zeigern nach dem Entfernen des Array wird zu einem kritischen Fehler und Programmabbruch führen. Standardmäßig ist das Speicherverwaltung-Flag festgelegt, d.h. die Array-Klasse ist für die Freigabe des Speichers verantwortlich.

Beispiel:

```
//--- example for CArrayObj::FreeMode(bool)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
```

© 2000-2025, MetaQuotes Ltd.

```
      printf("Object create error");
      return;
     }
   //--- reset free mode flag
   array.FreeMode(false);
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Reserve

Reserviert Speicher, um die Größe des Arrays zu erhöhen.

```
bool  Reserve(
   int  size      // Größe
   )
```

Parameter

size

[in] Anzahl der zusätzlichen Array-Elemente.

Rückgabewert

Gibt bei Erfolg true zurück, false beim Versuch die Größer kleiner als oder gleich auf Null anzufordern oder wenn die Array-Größe nicht geändert werden konnte.

Hinweis

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) erhöht.

Beispiel:

```
//--- example for CArrayObj::Reserve(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   if(!array.Reserve(1024))
     {
      printf("Reserve error");
      delete array;
      return;
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Resize

Setzt eine neue Größe des Arrays (kleinere).

```
bool  Resize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Größe des Arrays.

Rückgabewert

Gibt bei Erfolg true zurück, oder false beim Versuch, die Größe kleiner als Null zu setzen.

Hinweis

Ändern der Größe des Arrays ermöglicht die optimale Nutzung des Speichers. Die Elemente auf der rechten Seite sind verloren. Speicher verlorener Elemente wird freigegeben oder nicht, abhängig von der Speicherverwaltungsmodus.

Um die Fragmentierung des Speichers zu reduzieren, wird die Array-Größe mit den Schritt, der zuvor durch die Methode Step(int) definierten wurde, oder mit dem Schritt 16 (Standard) geändert.

Beispiel:

```
//--- example for CArrayObj::Resize(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- resize array
   if(!array.Resize(10))
     {
      printf("Resize error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente des Arrays ohne Speicherfreigabe.

```
void  Clear()
```

Rückgabewert

Nichts.

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher von gelöschten Elementen freigegeben.

Beispiel:

```
//--- example for CArrayObj::Clear()
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- clear array
   array.Clear();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shutdown

Klärt ein Array mit vollständige Freigabe vom Arrary-Speicher (nicht seiner Elementen).

```
bool  Shutdown()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher von gelöschten Elementen freigegeben.

Beispiel:

```
//--- example for CArrayObj::Shutdown()
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shutdown array
   if(!array.Shutdown())
     {
      printf("Shutdown error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CreateElement

Erstellt ein neues Element des Arrays an der angegebene Position.

```
bool  CreateElement(
   int  index      // Position
   )
```

Parameter

index

[in] Die Position, in der Sie ein neues Element erstellen möchten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht erstellt werden kann.

Hinweis

Die Methode CreateElement(int) in der Klasse CArrayObject gibt immer false zurück und führt keine Aktionen aus. Falls erforderlich, sollte die Methode CreateElement(int) in der abgeleiteten Klasse implementiert werden.

Beispiel:

```
//--- example for CArrayObj::CreateElement(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   int         size=100;
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- fill array
   array.Reserve(size);
   for(int i=0;i<size;i++)
     {
      if(!array.CreateElement(i))
        {
         printf("Element create error");
         delete array;
         return;
        }
     }
   //--- use array
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende des Arrays hinzu.

```
bool  Add(
   CObject*  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements ins Array hinzuzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht hinzugefügt werden kann.

Hinweis

Das Element wird nicht ins Array hinzugefügt werden, wenn ein ungültiger Zeiger (zB NULL) als Parameter übergeben wird.

Beispiel:

```
//--- example for CArrayObj::Add(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 arrays elements
   for(int i=0;i<100;i++)
     {
      if(!array.Add(new CObject))
        {
         printf("Element addition error");
         delete array;
         return;
        }
     }
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

AddArray

Am Ende des Arrays fügt Elementen aus einem anderen Array hinzu.

```
bool  AddArray(
   const CArrayObj *  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayDouble-Klasse, die eine Quelle von Elementen für Hinzufügen ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht hinzugefügt werden können.

Hinweis

Hinzufügen von Elementen aus dem Array ins Array ist eigentlich das Kopieren von Zeigern. Deshalb, wenn Sie die Methode anrufen, gibt es eine "Fallgrube" - ein Zeiger auf das dynamischen Objekt kann in mehr als eine Variable existieren.

```
//--- example
extern bool       make_error;
extern int        error;
extern CArrayObj *src;
//--- Eine Instanz der Klasse CArrayObj erstellen
//--- Der Speicherverwaltungsmechanismus ist aktiviert
CArrayObj *array=new CArrayObj;
//--- Elemente aus dem Quelle-Array hinzufügen (kopieren)
if(array!=NULL)
   bool result=array.AddArray(src);
if(make_error)
  {
   //--- Fehler machen
   switch(error)
     {
      case 0:
         //--- das Quelle-Array löschen, ohne sein Flag von Speichersteuerung zu prüfe
         delete src;
         //--- das Ergebnis:
         //--- Anruf ist möglich
         //--- ein "ungültiger" Zeiger im Empfänger-Array
         break;
      case 1:
         //--- den Speicherverwaltungsmechanismus im Quelle-Array deaktivieren
         if(src.FreeMode()) src.FreeMode(false);
         //--- aber das Quelle-Array nicht löschen
         //--- das Ergebnis:
         //--- nach dem Löschen des Empfänger-Array ist Anruf möglich
```

© 2000-2025, MetaQuotes Ltd.

```
         //--- mit solchem Zeiger im Quelle-Array
         break;
      case 2:
         //--- den Speicherverwaltungsmechanismus im Quelle-Array deaktivieren
         src.FreeMode(false);
         //--- den Speicherverwaltungsmechanismus im Empfänger-Array deaktivieren
         array.FreeMode(false);
         //--- das Ergebnis:
         //--- Nach Abschluss des Programms, haben wir die "Speicherleck"
         break;
     }
  }
else
  {
   //--- den Speicherverwaltungsmechanismus im Quelle-Array deaktivieren
   if(src.FreeMode()) src.FreeMode(false);
   //--- das Quelle-Array löschen
   delete src;
   //--- das Ergebnis:
   //--- Anruf von Elementen des Empfänger-Arrays ist korrekt
   //--- Löschen vom Empfänger-Array löscht seinen Elemente
  }
```

Beispiel:

```
//--- example for CArrayObj::AddArray(const CArrayObj*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayObj *src=new CArrayObj;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- reset free mode flag
   src.FreeMode(false);
   //--- fill source array
   //--- . . .
   //--- add another array
   if(!array.AddArray(src))
     {
      printf("Array addition error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array without elements
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in das Array an die angegebene Position ein.

```
bool  Insert(
   CObject*  element,      // Element für Einfügen
   int       pos           // Position
   )
```

Parameter

element

[in] Wert des Elements ins Array einzufügen.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Hinweis

Das Element wird nicht ins Array hinzugefügt werden, wenn ein ungültiger Zeiger (zB NULL) als Parameter übergeben wird.

Beispiel:

```
//--- example for CArrayObj::Insert(CObject*,int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert elements
   for(int i=0;i<100;i++)
     {
      if(!array.Insert(new CObject,0))
        {
         printf("Insert error");
         delete array;
         return;
        }
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertArray

Fügt Elementen aus einem anderen Array ins angegebene Array beginnend mit der angegebenen Position ein.

```
bool  InsertArray(
   const CArrayObj*  src,     // Ein Zeiger auf die Quelle
   int               pos      // Position
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayObj-Klasse, die eine Quelle von Elementen für Einfügen ist.

pos

[in] Die Einfügeposition im Array

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht eingefügt werden können.

Hinweis

Siehe CArrayObj::AddArray(const CArrayObj*).

Beispiel:

```
//--- example for CArrayObj::InsertArray(const CArrayObj*,int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayObj *src=new CArrayObj;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- reset free mode flag
```

© 2000-2025, MetaQuotes Ltd.

```
   src.FreeMode(false);
   //--- fill source array
   //--- . . .
   //--- insert another array
   if(!array.InsertArray(src,0))
     {
      printf("Array inserting error");
      delete src;
      delete array;
      return;
     }
   //--- delete source array without elements
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

AssignArray

Kopiert Elementen aus einem anderen Array ins angegebenen Array.

```
bool  AssignArray(
   const CArrayObj*  src      // Ein Zeiger auf die Quelle
   )
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayObj-Klasse, die eine Quelle von Elementen für Kopieren ist.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht kopiert werden können.

Hinweis

Wenn das Empfänger-Array vor dem Aufruf von AssignArray nicht leer ist, werden alle seine Elemente aus dem Array gelöscht werden. Wenn ein Flag von Speicherverwaltung gesetzt ist, dann wird der Speicher von gelöschten Elementen befreit werden. Das Empfänger-Array wird eine exakte Kopie von Quelle-Array. Für weitere Informationen siehe CArrayObj::AddArray(const CArrayObj*).

Beispiel:

```
//--- example for CArrayObj::AssignArray(const CArrayObj*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayObj *src=new CArrayObj;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- reset free mode flag
   src.FreeMode(false);
   //--- fill source array
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- . . .
   //--- assign another array
   if(!array.AssignArray(src))
     {
      printf("Array assigned error");
      delete src;
      delete array;
      return;
     }
   //--- arrays is identical
   //--- delete source array without elements
   delete src;
   //--- use array
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Ändert ein Element an der angegebenen Position des Arrays.

```
bool  Update(
   int       pos,          // Position
   CObject*  element       // Wert
   )
```

Parameter

pos

[in] Die Position eines Elements im Array zu ändern

element

[in] Der neue Wert des Elements

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht geändert werden kann.

Hinweis

Das Element wird nicht geändert werden, wenn ein ungültiger Zeiger (zB NULL) als Parameter übergeben wird. Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher von ersetzten Elementen wieder freigegeben.

Beispiel:

```
//--- example for CArrayObj::Update(int,CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- update element
   if(!array.Update(0,new CObject))
     {
      printf("Update error");
      delete array;
      return;
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Shift

Verschiebt ein Element aus der angegebenen Array-Position an der angegebenen Verschiebung.

```
bool  Shift(
   int  pos,        // Position
   int  shift       // Verschiebung
   )
```

Parameter

pos

[in] Die Position des verschobenen Elements im Array.

shift

[in] Verschiebung (ein positiver oder negativer Wert).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CArrayObj::Shift(int,int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- shift element
   if(!array.Shift(10,-5))
     {
      printf("Shift error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Detach

Nimmt ein Element aus der angegebene Position des Arrays aus.

```
CObject*  Detach(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des ausgenommenen Elements im Array.

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das ausgenommenen Element zurück, NULL wenn das Element nicht ausgenommen werden kann.

Hinweis

Bei der Ausnahme aus einem Array wird das Element nicht gelöscht in jedem Zustand des Flags von Speicherverwaltung. Ein Zeiger auf ein ausgenommenen Element soll nach der Verwendung freigegeben werden.

Beispiel:

```
//--- example for CArrayObj::Detach(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   CObject *object=array.Detach(0);
   if(object==NULL)
     {
      printf("Detach error");
      delete array;
      return;
     }
   //--- use element
   //--- . . .
   //--- delete element
```

© 2000-2025, MetaQuotes Ltd.

```
   delete object;
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebene Position des Arrays.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher von gelöschten Elementen freigegeben.

Beispiel:

```
//--- example for CArrayObj::Delete(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   if(!array.Delete(0))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteRange

Löscht eine Gruppe von Elementen an der angegebene Position des Arrays.

```
bool  DeleteRange(
   int  from,     // Position des ersten Elements
   int  to        // Position des letzten Elements
   )
```

Parameter

from

[in] Position des ersten gelöschten Elements im Array.

to

[in] Position des letzten gelöschten Elements im Array.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht gelöscht werden können.

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher von gelöschten Elementen freigegeben.

Beispiel:

```
//--- example for CArrayObj::DeleteRange(int,int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- delete elements
   if(!array.DeleteRange(0,10))
     {
      printf("Delete error");
      delete array;
      return;
     }
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element an der angegebene Position des Arrays.

```
CObject*  At(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gewünschten Elements im Array.

Rückgabewert

Elementwert beim Erfolg, Null - beim Versuch ein Element aus nicht vorhandenen Position zu erhalten.

Beispiel:

```
//--- example for CArrayObj::At(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add elements
   //--- . . .
   for(int i=0;i<array.Total();i++)
     {
      CObject *result=array.At(i);
      if(result==NULL)
        {
         //--- Fehler beim Lesen aus einem Array
         printf("Get element error");
         delete array;
         return;
        }
      //--- use element
      //--- . . .
     }
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

CompareArray

Vergleicht ein Array mit einem anderen Array.

```
bool  CompareArray(
   const CArrayObj*  src      // Ein Zeiger auf die Quelle
   ) const
```

Parameter

src

[in] Ein Zeiger auf die Instanz der CArrayObj-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Arrays gleich sind, ansonsten false.

Beispiel:

```
//--- example for CArrayObj::CompareArray(const CArrayObj*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source array
   CArrayObj *src=new CArrayObj;
   if(src==NULL)
     {
      printf("Object create error");
      delete array;
      return;
     }
   //--- fill source array
   //--- . . .
   //--- compare with another array
   int result=array.CompareArray(src);
   //--- delete arrays
   delete src;
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

InsertSort

Fügt ein Element in ein sortiertes Array ein.

```
bool  InsertSort(
   CObject*  element      // Element für Einfügen
   )
```

Parameter

element

[in] Wert des Elements um ins sortierte Array einzufügen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht eingefügt werden kann.

Hinweis

Das Element wird nicht ins Array hinzugefügt werden, wenn ein ungültiger Zeiger (zB NULL) als Parameter übergeben wird.

Beispiel:

```
//--- example for CArrayObj::InsertSort(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- insert element
   if(!array.InsertSort(new CObject))
     {
      printf("Insert error");
      delete array;
      return;
     }
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einem sortierten Array, das gleich dem Muster ist.

```
int  Search(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj::Search(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.Search(sample)!=-1) printf("Element found");
   else                         printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreat

Sucht nach einem Element in einem sortierten Array, das größer als das Muster ist.

```
int  SearchGreat(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj::SearchGreat(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.SearchGreat(sample)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLess

Sucht nach einem Element in einem sortierten Array, das kleiner als das Muster ist.

```
int  SearchLess(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj:: SearchLess(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.SearchLess(sample)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchGreatOrEqual

Sucht nach einem Element in einem sortierten Array, das größer als oder gleich dem Muster ist.

```
int  SearchGreatOrEqual(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj::SearchGreatOrEqual(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.SearchGreatOrEqual(sample)!=-1) printf("Element found");
   else                                     printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLessOrEqual

Sucht nach einem Element in einem sortierten Array, das kleiner als oder gleich dem Muster ist.

```
int  SearchLessOrEqual(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj:: SearchLessOrEqual(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.SearchLessOrEqual(sample)!=-1) printf("Element found");
   else                                    printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchFirst

Sucht nach dem ersten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchFirst(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj::SearchFirst(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.SearchFirst(sample)!=-1) printf("Element found");
   else                              printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

SearchLast

Sucht nach dem letzten Element in einem sortierten Array, das gleich dem Muster ist.

```
int  SearchLast(
   CObject*  element      // Muster
   ) const
```

Parameter

element

[in] Element-Muster um im Array zu suchen.

Rückgabewert

Die Position des gefundenen Elements beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CArrayObj:: SearchLast(CObject*)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- sort array
   array.Sort();
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete array;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(array.SearchLast(sample)!=-1) printf("Element found");
   else                             printf("Element not found");
   //--- delete array
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Array-Daten in eine Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CArrayObj::Save(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   int        file_handle;
   CArrayObj *array=new CArrayObj;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add arrays elements
   //--- . . .
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Array-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Beim Lesen Array-Elementen aus einer Datei wird die Methode CArrayObj::CreateElement(int) für jedes Element aufgerufen.

Beispiel:

```
//--- example for CArrayObj::Load(int)
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   int        file_handle;
   CArrayObj *array=new CArrayObj;
   //---
   if(array!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!array.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete array;
         FileClose(file_handle);
         //---
         return;
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      FileClose(file_handle);
     }
   //--- use arrays elements
   //--- . . .
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Array-Typ-Identifikator.

```
virtual int  Type() const
```

Rückgabewert

Identifikator vom Arraytyp (für CArrayObj - 7778).

Beispiel:

```
//--- example for CArrayObj::Type()
#include <Arrays\ArrayObj.mqh>
//---
void OnStart()
  {
   CArrayObj *array=new CArrayObj;
   //---
   if(array==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get array type
   int type=array.Type();
   //--- delete array
   delete array;
  }
```

© 2000-2025, MetaQuotes Ltd.

CList

CList ist eine Klasse der dynamischen Liste von Instanzen von CObject-Klasse und ihre geerbten Klassen.

Beschreibung

Klasse CList bietet die Möglichkeit, mit einer Liste der Instanzen der CObject-Klasse und ihrer Nachfolger zu arbeiten. Die Klasse erlaubt Elemente der Liste hinzuzufügen/einzufügen/zu löschen, eine Liste zu sortieren, in einer sortierten Liste zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Es gibt bestimmte Arbeitseinzelheiten für die Klasse CList. Die Klasse hat einen Mechanismus zur dynamischen Speicherverwaltung, so müssen Sie bei der Arbeit mit den Elementen der Liste sehr vorsichtig sein.

Die Arbeitseinzelheiten vom Speicherverwaltungsmechanismus sind den in CArrayObj beschriebenen Einzelheiten ähnlich.

Deklaration

```
class CList : public CObject
```

Kopf

```
#include <Arrays\List.mqh>
```

Vererbungshierarchie

CObject
  CList

Gruppen der Klassenmethode

Attribute	
FreeMode	Erhält das Flag von Speicherverwaltung beim Löschen von Elementen aus der Liste.
FreeMode	Setzt das Flag von Speicherverwaltung beim Löschen von Elementen aus der Liste
Total	Erhält die Anzahl der Elementen in der Liste
IsSorted	Erhält das Flag der Listesortierung
SortMode	Erhält den Sortiertyp
Das Erstellen eines neuen Elements	
virtual CreateElement	Erstellt ein neues Element in der Liste
Füllung	

© 2000-2025, MetaQuotes Ltd.

Attribute	
Add	Fügt ein Element am Ende der Liste hinzu
Insert	Fügt ein Element in die Liste an die angegebene Position ein
Löschen	
DetachCurrent	Nimmt ein Element aus der aktuelle Position der Liste aus, ohne es "körperlich" zu löschen.
DeleteCurrent	Löscht ein Element an der aktuellen Position der Liste
Delete	Löscht ein Element an der angegebenen Position der Liste
Clear	Löscht alle Elemente der Liste
Navigation	
IndexOf	Erhält den Index des angegebenen Listenelements
GetNodeAtIndex	Erhält ein Element an der angegebene Position der Liste
GetFirstNode	Erhält das erste Elemente in der Liste
GetPrevNode	Erhält das vorhergehende Element in der Liste.
GetCurrentNode	Erhält das aktuelle Listenelement
GetNextNode	Erhält das nachfolgende Element in der Liste.
GetLastNode	Erhält das letzte Element in der Liste.
Verschieben von Elementen	
Sort	Sortiert eine Liste
MoveToIndex	Verschiebt das aktuelle Listenelement an die angegebene Position
Exchange	Tauscht die Elemente in der Liste aus
Vergleich	
CompareList	Vergleicht eine Liste mit einer anderen Liste
Suche	
Search	Sucht nach einem Element in einer sortierten Liste, das gleich dem Muster ist.
Eingabe/Ausgabe	
virtual Save	Speichert Daten der Liste in einer Datei
virtual Load	Lädt Daten der Liste aus einer Datei
virtual Type	Erhält die Identifikator des Listentyps.

Methoden geerbt von der Klasse CObject

© 2000-2025, MetaQuotes Ltd.

Prev, Prev, Next, Next, Compare

© 2000-2025, MetaQuotes Ltd.

FreeMode

Erhält das Flag von Speicherverwaltung beim Löschen von Elementen aus der Liste.

```
bool  FreeMode() const
```

Rückgabewert

Ein Flag von Speicherverwaltung.

Beispiel:

```
//--- example for CList::FreeMode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get free mode flag
   bool list_free_mode=list.FreeMode();
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

FreeMode

Setzt das Flag von Speicherverwaltung beim Löschen von Elementen aus der Liste.

```
void  FreeMode(
   bool  mode      // ein neuer Wert
   )
```

Parameter

mode

[in] Der neue Wert des Flags von Speicherverwaltung.

Hinweis

Setzen vom Speicherverwaltung-Flags ist ein wichtiger Punkt bei der Verwendung von der Klasse CList. Da die Elemente der Liste sind Zeiger auf dynamische Objekte, ist es wichtig zu bestimmen, was mit ihnen beim Löschen aus der Liste zu tun. Wenn ein Flag gesetzt ist, wird das Element automatisch durch den Operator delete beim Löschen aus der Liste gelöscht werden. Wenn kein Flag gesetzt ist, wird angenommen, dass ein Zeiger auf ein gelöschtes Objekt noch irgendwo im Anwenderprogramm bleibt und wird bei der Programm danach freigegeben.

Wenn das Anwenderprogramm das Flag der Speicherverwaltung löscht, muss der Benutzer die Verantwortung für die Entfernung von Elementen der Liste vor dem Ende des Programms verstehen, denn sonst bleibt der Speicher, der durch den Elementen während der Erstellung mit dem new-Operator belegt war, nicht freigegeben. Wenn Datenmenge groß ist, kann dies zu Fehlfunktionen des Terminals führen.

Wenn das Anwenderprogramm löscht nicht das Flag der Speicherverwaltung, gibt es eine andere "Fallgrube". Verwendung von irgendwo in den lokalen Variablen gespeicherten Listenelemente-Zeigern nach dem Entfernen der Liste wird zu einem kritischen Fehler und Programmabbruch führen. Standardmäßig ist das Speicherverwaltung-Flag festgelegt, d.h. die Liste-Klasse ist für die Freigabe des Speichers verantwortlich.

Beispiel:

```
//--- example for CList::FreeMode(bool)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- reset free mode flag
   list.FreeMode(false);
   //--- use list
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- . . .
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Total

Erhält die Anzahl der Elementen in der Liste.

```
int  Total() const
```

Rückgabewert

Die Anzahl der Elementen in der Liste.

Beispiel:

```
//--- example for CList::Total()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check total
   int total=list.Total();
   //--- use list
   //--- ...
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

IsSorted

Erhält das Flag der Listesortierung.

```
bool  IsSorted(
   int  mode=0      // Sortieroption
   ) const
```

Parameter

mode=0

[in] Geprüfte Sortieroption

Rückgabewert

Flag der Listesortierung. Wenn eine Liste nach der angegebenen Variante sortiert ist, wird true zurückgegeben, ansonsten false.

Hinweis

Ein Flag der Sortierung der Liste kann nicht direkt geändert werden. Ein Flag wird durch die Methode Sort(int) gesetzt werden, und wird durch eine der Methoden zum Hinzufügen/Einfügen gelöscht.

Beispiel:

```
//--- example for CList::IsSorted()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check sorted
   if(list.IsSorted(0))
     {
      //--- use methods for sorted list
      //--- ...
     }
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

SortMode

Erhält den Sortiertyp.

```
int  SortMode() const
```

Rückgabewert

Sortiertyp oder -1 wenn die Liste nicht sortiert ist.

Beispiel:

```
//--- example for CList::SortMode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- check sort mode
   int sort_mode=list.SortMode();
   //--- use list
   //--- ...
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

CreateElement

Erstellt ein neues Element in der Liste.

```
CObject*  CreateElement()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das neu erstellten Element zurück, NULL wenn das Element nicht erstellt werden kann.

Hinweis

Die Methode CreateElement() in der Klasse CList gibt immer NULL zurück und führt keine Aktionen aus. Falls erforderlich, sollte die Methode CreateElement() in der abgeleiteten Klasse implementiert werden.

Beispiel:

```
//--- example for CList::CreateElement(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   int    size=100;
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- fill list
   for(int i=0;i<size;i++)
     {
      CObject *object=list.CreateElement();
      if(object==NULL)
        {
         printf("Element create error");
         delete list;
         return;
        }
      list.Add(object);
     }
   //--- use list
   //--- . . .
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element am Ende der Liste hinzu.

```
int  Add(
   CObject*  element      // ein Element um hinzuzufügen
   )
```

Parameter

element

[in] Wert des Elements in die Liste hinzuzufügen.

Rückgabewert

Der Index des hinzugefügten Elements, oder -1, wenn ein Fehler auftritt.

Hinweis

Ein Element wird nicht in eine Liste hinzugefügt werden, wenn ein ungültiger Zeiger (zB NULL) als Parameter übergeben wird.

Beispiel:

```
//--- example for CList::Add(Cobject*)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add 100 elements
   for(int i=0;i<100;i++)
     {
      if(list.Add(new CObject)==-1)
        {
         printf("Element addition error");
         delete list;
         return;
        }
     }
   //--- use list
   //--- . . .
   //--- delete list
   delete list;
```

© 2000-2025, MetaQuotes Ltd.

```
}
```

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element in die Liste an die angegebene Position ein.

```
int  Insert(
   CObject*  element,      // Element für Einfügen
   int       pos           // Position
   )
```

Parameter

element

[in] Wert des Elements, das in die Liste eingefügt wird.

pos

[in] Die Einfügeposition in der Liste

Rückgabewert

Der Index des eingefügten Elements, oder -1, wenn ein Fehler auftritt.

Hinweis

Ein Element wird nicht in eine Liste hinzugefügt werden, wenn ein ungültiger Zeiger (zB NULL) als Parameter übergeben wird.

Beispiel:

```
//--- example for CList::Insert(CObject*,int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- insert 100 elements
   for(int i=0;i<100;i++)
     {
      if(list.Insert(new CObject,0)==-1)
        {
         printf("Element insert error");
         delete list;
         return;
        }
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- use list
   //--- . . .
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

DetachCurrent

Nimmt ein Element aus der aktuelle Position der Liste aus, ohne es "körperlich" zu löschen.

```
CObject*  DetachCurrent()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das ausgenommenen Element zurück, NULL wenn das Element nicht ausgenommen werden kann.

Hinweis

Bei der Ausnahme aus einer Liste wird das Element nicht gelöscht in jedem Zustand des Flags von Speicherverwaltung. Ein Zeiger auf ein ausgenommenen Element soll nach der Verwendung wieder freigegeben werden.

Beispiel:

```
//--- example for CList::DetachCurrent()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.DetachCurrent();
   if(object==NULL)
     {
      printf("Detach error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- delete element
   delete object;
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteCurrent

Löscht ein Element an der aktuellen Position der Liste.

```
bool  DeleteCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher vom gelöschten Element wieder freigegeben.

Beispiel:

```
//--- example for CList::DeleteCurrent()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   if(!list.DeleteCurrent())
     {
      printf("Delete error");
      delete list;
      return;
     }
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element an der angegebenen Position der Liste.

```
bool  Delete(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des gelöschten Elements in der Liste.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht gelöscht werden kann.

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher vom gelöschten Element wieder freigegeben.

Beispiel:

```
//--- example for CList::Delete(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   if(!list.Delete(0))
     {
      printf("Delete error");
      delete list;
      return;
     }
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente der Liste.

```
void  Clear()
```

Hinweis

Wenn der Speicherverwaltungsmechanismus aktiviert ist, wird der Speicher von gelöschten Elementen freigegeben.

Beispiel:

```
//--- example for CList::Clear()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   //--- clear list
   list.Clear();
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

IndexOf

Erhält den Index des angegebenen Listenelements.

```
int  IndexOf(
   CObject*  element      // ein Zeiger auf das Element
   )
```

Parameter

element

[in] Zeiger auf ein Listenelement.

Rückgabewert

Der Index des Listenelements oder -1.

Beispiel:

```
//--- example for CList::IndexOf(CObject*)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   CObject *object=new CObject;
   if(object==NULL)
     {
      printf("Element create error");
      delete list;
      return;
     }
   if(list.Add(object))
     {
      int pos=list.IndexOf(object);
     }
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

GetNodeAtIndex

Erhält ein Element an der angegebene Position der Liste.

```
CObject*  GetNodeAtIndex(
   int  pos      // Position
   )
```

Parameter

pos

[in] Position des Elements in der Liste.

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das Element zurück, NULL wenn der Zeiger nicht erhalten werden kann.

Beispiel:

```
//--- example for CList::GetNodeAtIndex(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.GetNodeAtIndex(10);
   if(object==NULL)
     {
      printf("Get node error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- do not delete element
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

GetFirstNode

Erhält das erste Elemente in der Liste.

```
CObject*  GetFirstNode()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das erste Element zurück, NULL wenn der Zeiger nicht erhalten werden kann.

Beispiel:

```
//--- example for CList::GetFirstNode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.GetFirstNode();
   if(object==NULL)
     {
      printf("Get node error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- do not delete element
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

GetPrevNode

Erhält das vorhergehende Element in der Liste.

```
CObject*  GetPrevNode()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das vorhergehende Element zurück, NULL wenn der Zeiger nicht erhalten werden kann.

Beispiel:

```
//--- example for CList::GetPrevNode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.GetPrevNode();
   if(object==NULL)
     {
      printf("Get node error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- do not delete element
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

GetCurrentNode

Erhält das aktuelle Listenelement.

```
CObject*  GetCurrentNode()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das aktuelle Element zurück, NULL wenn der Zeiger nicht erhalten werden kann.

Beispiel:

```
//--- example for CList::GetCurrentNode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.GetCurrentNode();
   if(object==NULL)
     {
      printf("Get node error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- do not delete element
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

GetNextNode

Erhält das nachfolgende Element in der Liste.

```
CObject*  GetNextNode()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das nachfolgende Element zurück, NULL wenn der Zeiger nicht erhalten werden kann.

Beispiel:

```
//--- example for CList::GetNextNode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.GetNextNode();
   if(object==NULL)
     {
      printf("Get node error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- do not delete element
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

GetLastNode

Erhält das letzte Element in der Liste.

```
CObject*  GetLastNode()
```

Rückgabewert

Gibt bei Erfolg einen Zeiger auf das letzte Element zurück, NULL wenn der Zeiger nicht erhalten werden kann.

Beispiel:

```
//--- example for CList::GetLastNode()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add list elements
   //--- . . .
   CObject *object=list.GetLastNode();
   if(object==NULL)
     {
      printf("Get node error");
      delete list;
      return;
     }
   //--- use element
   //--- . . .
   //--- do not delete element
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Sort

Sortiert eine Liste.

```
void  Sort(
   int  mode      // Sortieroption
   )
```

Parameter

mode

[in] Sortieroption.

Rückgabewert

Nichts.

Hinweis

Listen werden immer in aufsteigender Reihenfolge sortiert.

Beispiel:

```
//--- example for CList::Sort(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- sorting by mode 0
   list.Sort(0);
   //--- use list
   //--- ...
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

MoveToIndex

Verschiebt das aktuelle Listenelement an die angegebene Position.

```
bool  MoveToIndex(
   int  pos      // Position
   )
```

Parameter

pos

[in] Die Position in der Liste für Verschiebung.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Element nicht verschoben werden kann.

Beispiel:

```
//--- example for CList::MoveToIndex(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- move current node to begin
   list.MoveToIndex(0);
   //--- use list
   //--- . . .
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Exchange

Tauscht die Elemente in der Liste aus.

```
bool  Exchange(
   CObject*  node1,     // ein Element der Liste
   CObject*  node2      // ein Element der Liste
   )
```

Parameter

node1

[in] Ein Element der Liste

node2

[in] Ein Element der Liste

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Elementen nicht ausgetauscht werden können.

Beispiel:

```
//--- example for CList::Exchange(CObject*,CObject*)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- exchange
   list.Exchange(list.GetFirstNode(),list.GetLastNode());
   //--- use list
   //--- . . .
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

CompareList

Vergleicht eine Liste mit einer anderen Liste.

```
bool  CompareList(
   CList*  list      // die Liste mit der die angegebene Liste vergleicht wird
   )
```

Parameter

list

[in] Ein Zeiger auf die Instanz der CList-Klasse, die eine Quelle von Elementen für Vergleich ist.

Rückgabewert

Gibt true zurück, wenn die Listen gleich sind, ansonsten false.

Beispiel:

```
//--- example for CList::CompareList(const CList*)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- create source list
   CList *src=new CList;
   if(src==NULL)
     {
      printf("Object create error");
      delete list;
      return;
     }
   //--- fill source list
   //--- . . .
   //--- compare with another list
   bool result=list.CompareList(src);
   //--- delete lists
   delete src;
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Search

Sucht nach einem Element in einer sortierten Liste, das gleich dem Muster ist.

```
CObject*  Search(
   CObject*  element      // Muster
   )
```

Parameter

element

[in] Element-Muster um in der Liste zu suchen.

Rückgabewert

Ein Zeiger auf das gefundene Element beim Erfolg, -1 - wenn kein Element gefunden ist.

Beispiel:

```
//--- example for CList::Search(CObject*)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add lists elements
   //--- . . .
   //--- sort list
   list.Sort(0);
   //--- create sample
   CObject *sample=new CObject;
   if(sample==NULL)
     {
      printf("Sample create error");
      delete list;
      return;
     }
   //--- set sample attributes
   //--- . . .
   //--- search element
   if(list.Search(sample)!=NULL) printf("Element found");
   else                          printf("Element not found");
   //--- delete list
```

© 2000-2025, MetaQuotes Ltd.

```
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Daten der Liste in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CList::Save(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   int    file_handle;
   CList *list=new CList;
   //---
   if(list!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- add lists elements
   //--- . . .
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!list.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         delete list;
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Daten der Liste aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Beim Lesen Listenelementen aus einer Datei wird die Methode CList::CreateElement() für jedes Element aufgerufen.

Beispiel:

```
//--- example for CLoad::Load(int)
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   int    file_handle;
   CList *list=new CList;
   //---
   if(list!=NULL)
     {
      printf("Object create error");
      return;
     }
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!list.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         delete list;
         FileClose(file_handle);
         //---
         return;
        }
```

© 2000-2025, MetaQuotes Ltd.

```
      FileClose(file_handle);
     }
   //--- use list elements
   //--- . . .
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Listentyps.

```
virtual int  Type()
```

Rückgabewert

Identifikator vom Listentyp (für CList - 7779).

Beispiel:

```
//--- example for CList::Type()
#include <Arrays\List.mqh>
//---
void OnStart()
  {
   CList *list=new CList;
   //---
   if(list==NULL)
     {
      printf("Object create error");
      return;
     }
   //--- get list type
   int type=list.Type();
   //--- delete list
   delete list;
  }
```

© 2000-2025, MetaQuotes Ltd.

CTreeNode

Class CTreeNode is a class of node of the binary tree CTree.

Description

Class CTreeNode provides the ability to work with nodes of the binary tree CTree. Options of navigation through the tree is implemented in the class. Besides that methods of work with a file are implemented.

Declaration

```
class CTreeNode : public CObject
```

Title

```
#include <Arrays\TreeNode.mqh>
```

Vererbungshierarchie

CObject

CTreeNode

Direkte Ableitungen

CTree

Class Methods

Attributes	
Owner	Gets/sets the pointer of the owner node
Left	Gets/sets the pointer of the left node
Right	Gets/sets the pointer of the right node
Balance	Gets the node balance
BalanceL	Gets the balance of the left sub-branch of the node
BalanceR	Gets the balance of the right sub-branch of the node
Creation of a new element	
CreateSample	Creates a new node instance
Comparison	
RefreshBalance	Recalculates the node balance
Search	
GetNext	Gets the pointer of the next node
Input/Output	

© 2000-2025, MetaQuotes Ltd.

Attributes	
SaveNode	Saves the node data to a file
LoadNode	Downloads the node data from a file
virtual Type	Gets the identifier of the node type

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Compare

Trees of CTreeNode class descendants get practical application.

A descendant of class CTreeNode must have predefined methods: CreateSample that creates a new instance of the descendant class of CTreeNode, Compare that compares values of key fields of the descendant class of CTreeNode, Type (if it's necessary to identify a node), SaveNode and LoadNode (if it's necessary to work with a file).

Let's consider and example of a CTree descendant class.

```
//+------------------------------------------------------------------+
//|                                                   MyTreeNode.mq5 |
//|                        Copyright 2010, MetaQuotes Software Corp. |
//|                                             https://www.metaquotes.net/ |
//+------------------------------------------------------------------+
#property copyright "2010, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
//---
#include <Arrays\TreeNode.mqh>
//+------------------------------------------------------------------+
//| Describe classderived from CTreeNode.                            |
//+------------------------------------------------------------------+
//| Class CMyTreeNode.                                               |
//| Purpose: Class of element of a binary tree.                      |
//|              Descendant of class CTreeNode.                      |
//+------------------------------------------------------------------+
class CMyTreeNode : public CTreeNode
  {
protected:
   //--- user's data
   long              m_long;           // key field of type long
   double            m_double;         // custom variable of type double
   string            m_string;         // custom variable of type string
   datetime          m_datetime;       // custom variable of type datetime

public:
                     CMyTreeNode();
   //--- methods of accessing these user's data
   long              GetLong(void)                { return(m_long); }
   void              SetLong(long value)          { m_long=value;   }
```

© 2000-2025, MetaQuotes Ltd.

```
   double            GetDouble(void)               { return(m_double); }
   void              SetDouble(double value)       { m_double=value;  }
   string            GetString(void)               { return(m_string); }
   void              SetString(string value)       { m_string=value;  }
   datetime          GetDateTime(void)             { return(m_datetime); }
   void              SetDateTime(datetime value)   { m_datetime=value;  }
   //--- methods of working with files
   virtual bool      Save(int file_handle);
   virtual bool      Load(int file_handle);
protected:
   virtual int       Compare(const CObject *node,int mode);
   //--- methods of creating class instances
   virtual CTreeNode* CreateSample();
  };
//+------------------------------------------------------------------+
//| CMyTreeNode class constructor.                                   |
//| INPUT:  none.                                                    |
//| OUTPUT: none.                                                    |
//| REMARK: none.                                                    |
//+------------------------------------------------------------------+
void CMyTreeNode::CMyTreeNode()
  {
//--- initialization of user's data
   m_long        =0;
   m_double      =0.0;
   m_string      ="";
   m_datetime    =0;
  }
//+------------------------------------------------------------------+
//| Comparison with another three node by the specified algorithm.   |
//| INPUT:  node - array element to compare,                         |
//|         mode - identifier of comparison algorithm.               |
//| OUTPUT: result of comparison (>0,0,<0).                          |
//| REMARK: none.                                                    |
//+------------------------------------------------------------------+
int CMyTreeNode::Compare(const CObject *node,int mode)
  {
//--- parameter mode is ignored, because tree construction algorithm is the only one
   int res=0;
//--- explicit type casting
   CMyTreeNode *n=node;
   res=(int)(m_long-n.m_long);
//---
   return(res);
  }
//+------------------------------------------------------------------+
//| Creation of a new class instance.                                |
//| INPUT:  none.                                                    |
//| OUTPUT: pointer to a new instance of class CMyTreeNode.          |
```

© 2000-2025, MetaQuotes Ltd.

```
//| REMARK: none.                                                    |
//+------------------------------------------------------------------+
CTreeNode* CMyTreeNode::CreateSample()
  {
   CMyTreeNode *result=new CMyTreeNode;
//---
   return(result);
  }
//+------------------------------------------------------------------+
//| Write tree node data to a file.                                  |
//| INPUT:  file_handle -handle of a file pre-opened for writing.    |
//| OUTPUT: true if OK, otherwise false.                             |
//| REMARK: none.                                                    |
//+------------------------------------------------------------------+
bool CMyTreeNode::Save(int file_handle)
  {
   uint i=0,len;
//--- checks
   if(file_handle<0) return(false);
//--- writing user data
//--- writing custom variable of type long
   if(FileWriteLong(file_handle,m_long)!=sizeof(long))          return(false);
//--- writing custom variable of type double
   if(FileWriteDouble(file_handle,m_double)!=sizeof(double))    return(false);
//--- writing custom variable of type string
   len=StringLen(m_string);
//--- write string length
   if(FileWriteInteger(file_handle,len,INT_VALUE)!=INT_VALUE)   return(false);
//--- write the string
   if(len!=0 && FileWriteString(file_handle,m_string,len)!=len) return(false);
//--- writing custom variable of type datetime
   if(FileWriteLong(file_handle,m_datetime)!=sizeof(long))      return(false);
//---
   return(true);
  }
//+------------------------------------------------------------------+
//| Read tree node data from a file.                                 |
//| INPUT:  file_handle -handle of a file pre-opened for reading.    |
//| OUTPUT: true if OK, otherwise false.                             |
//| REMARK: none.                                                    |
//+------------------------------------------------------------------+
bool CMyTreeNode::Load(int file_handle)
  {
   uint i=0,len;
//--- checks
   if(file_handle<0) return(false);
//--- reading
   if(FileIsEnding(file_handle)) return(false);
//--- reading custom variable of type char
```

© 2000-2025, MetaQuotes Ltd.

```
//--- reading custom variable of type long
   m_long=FileReadLong(file_handle);
//--- reading custom variable of type double
   m_double=FileReadDouble(file_handle);
//--- reading custom variable of type string
//--- read the string length
   len=FileReadInteger(file_handle,INT_VALUE);
//--- read the string
   if(len!=0) m_string=FileReadString(file_handle,len);
   else       m_string="";
//--- reading custom variable of type datetime
   m_datetime=FileReadLong(file_handle);
//---
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

Owner

Erhält den Zeiger auf den Halter-Knoten.

```
CTreeNode*  Owner()
```

Rückgabewert

Ein Zeiger auf den Halter-Knoten.

Owner

Setzt den Zeiger auf den Halter-Knoten.

```
void  Owner(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Der neue Wert des Zeigers auf den Halter-Knoten.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Left

Erhält den Zeiger auf den linken Knoten.

```
CTreeNode*  Left()
```

Rückgabewert

Ein Zeiger auf den linken Knoten.

Left

Setzt den Zeiger auf den linken Knoten.

```
void  Left(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Der neue Wert des Zeigers auf den linken Knoten.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Right

Erhält den Zeiger auf den rechten Knoten.

```
CTreeNode*  Right()
```

Rückgabewert

Ein Zeiger auf den rechten Knoten.

Right

Setzt den Zeiger auf den rechten Knoten.

```
void  Right(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Der neue Wert des Zeigers auf den rechten Knoten.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Balance

Erhält das Gleichgewicht des Knotens.

```
int  Balance() const
```

Rückgabewert

Das Gleichgewicht des Knotens.

© 2000-2025, MetaQuotes Ltd.

BalanceL

Erhält das Gleichgewicht des linken Unterzweigs des Knotens.

```
int  BalanceL() const
```

Rückgabewert

Das Gleichgewicht des linken Unterzweigs des Knotens.

© 2000-2025, MetaQuotes Ltd.

BalanceR

Erhält das Gleichgewicht des rechten Unterzweigs des Knotens.

```
int  BalanceR() const
```

Rückgabewert

Das Gleichgewicht des rechten Unterzweigs des Knotens.

© 2000-2025, MetaQuotes Ltd.

CreateSample

Erstellt eine neue Instanz des Knotens.

```
virtual CTreeNode*  CreateSample()
```

Rückgabewert

Ein Zeiger auf die neue Knoten-Instanz oder NULL.

© 2000-2025, MetaQuotes Ltd.

RefreshBalance

Neu berechnet das Gleichgewicht des Knotens.

```
int  RefreshBalance()
```

Rückgabewert

Das Gleichgewicht des Knotens.

© 2000-2025, MetaQuotes Ltd.

GetNext

Erhält den Zeiger auf den nachfolgenden Knoten.

```
CTreeNode*  GetNext(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Knoten von Suchanfang.

Rückgabewert

Ein Zeiger auf den nachfolgenden Knoten.

© 2000-2025, MetaQuotes Ltd.

SaveNode

Schreibt Knotendaten in eine Datei.

```
bool  SaveNode(
   int  file_handle      // Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

LoadNode

Lädt Knotendaten aus einer Datei.

```
bool  LoadNode(
   int          file_handle,      // Handle
   CTreeNode*   main              // Knoten
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die früher geöffnet wurde.

main

[in] Knoten für Daten.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Knotentyps.

```
virtual int  Type() const
```

Rückgabewert

Die Identifikator des Knotentyps.

© 2000-2025, MetaQuotes Ltd.

CTree

CTree ist eine Klasse vom binären Baum der Instanzen von CTreeNode-Klasse und ihre geerbten Klassen.

Beschreibung

Klasse CTree bietet die Möglichkeit, mit dem binären Baum der Instanzen der CTreeNode-Klasse und ihrer Nachfolger zu arbeiten. Die Klasse erlaubt Baumelemente hinzuzufügen/einzufügen/zu löschen und im Baum zu suchen. Darüber hinaus sind Methoden für Arbeit mit einer Datei implementiert.

Es sei darauf hingewiesen, dass die Klasse CTree hat keinen dynamischen Speicherverwaltungsmechanismus (im Gegensatz zu Klassen CList und CArrayObj). Alle Baumknoten werden mit der Speicherfreigabe entfernt.

Deklaration

```
class CTree : public CTreeNode
```

Kopf

```
#include <Arrays\Tree.mqh>
```

Vererbungshierarchie

CObject
  CTreeNode
    CTree

Gruppen der Klassenmethode

Attribute	
Root	Erhält den Stammknoten des Baumes
Das Erstellen eines neuen Elements	
virtual CreateElement	Erstellt eine neue Instanz des Knotens
Füllung	
Insert	Fügt einen Knoten im Baum hinzu
Löschen	
Detach	Nimmt den angegebenen Knoten aus dem Baum aus
Delete	Löscht den angegebenen Knoten aus dem Baum
Clear	Löscht alle Baumknoten
Suche	
Find	Sucht nach einem Knoten im Baum das gleich dem Muster ist

© 2000-2025, MetaQuotes Ltd.

Attribute	
Eingabe/Ausgabe	
virtual Save	Speichert Daten des Baums in einer Datei
virtual Load	Lädt Baum-Daten aus einer Datei
virtual Type	Erhält die Identifikator des Baumtyps.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CTreeNode

Parent, Parent, Left, Left, Right, Right, Balance, BalanceL, BalanceR, RefreshBalance, GetNext, SaveNode, LoadNode

Bäume von abgeleiteten Klassen von CTreeNode - abgeleitete Klassen von CTree - haben praktische Anwendung.

Eine abgeleitete Klasse von CTree sollte die zugeordnete Methode Create haben, die eine neue Instanz der abgeleitete Klasse CTreeNode erstellt.

Betrachten Sie das Beispiel für eine abgeleitete Klasse CTree.

```
//+------------------------------------------------------------------+
//|                                                      MyTree.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//| www.metaquotes.net |
//+------------------------------------------------------------------+
#property copyright "2010, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
//---
#include <Arrays\Tree.mqh>
#include "MyTreeNode.mqh"
//---
input int extCountedNodes = 100;
//+------------------------------------------------------------------+
//| Beschreiben von CTree abgeleitete CMyTree-Klasse.          |
//+------------------------------------------------------------------+
//| Klasse CMyTree.                                                  |
//| Zweck: Bauen und Navigation des binären Suchbaums.      |
//+------------------------------------------------------------------+
class CMyTree : public CTree
  {
public:
   //--- Methoden zur Suche nach Benutzerdaten in der Baumstruktur
   CMyTreeNode*         FindByLong(long find_long);
   //--- Methode zur Erstellung eines Baumelements
   virtual CTreeNode *CreateElement();
  };
```

© 2000-2025, MetaQuotes Ltd.

```
//---
CMyTree MyTree;
//+------------------------------------------------------------------+
//| Erstellen ein neues Baumknotens.                                 |
//| INPUT:  keine.                                                   |
//| OUTPUT: Zeiger auf den neuen Baumknoten wenn OK, oder NULL.      |
//| REMARK: keine.                                                   |
//+------------------------------------------------------------------+
CTreeNode *CMyTree::CreateElement()
  {
   CMyTreeNode *node=new CMyTreeNode;
//---
   return(node);
  }
//+------------------------------------------------------------------+
//| Suche nach Element in der Liste mit Wert m_long.                 |
//| INPUT:  find_long - gewünschter Wert.                            |
//| OUTPUT: Zeiger auf gefundenes Listenelement oder NULL.           |
//| REMARK: keine.                                                   |
//+------------------------------------------------------------------+
CMyTreeNode* CMyTree::FindByLong(long find_long)
  {
   CMyTreeNode *res=NULL;
   CMyTreeNode *node;
//--- Baumknoten erstellen um Suchoptionen zu übertragen
   node=new CMyTreeNode;
   if(node==NULL) return(NULL);
   node.SetLong(find_long);
//---
   res=Find(node);
   delete node;
//---
   return(res);
  }
//+------------------------------------------------------------------+
//| Script "Testieren von Klasse CMyTree"                            |
//+------------------------------------------------------------------+
//---  Array für Initialisieren von Strings
string str_array[11]={"p","oo","iii","uuuu","yyyyy","ttttt","rrrr","eee","ww","q","999
//---
int OnStart() export
  {
   int          i;
   uint         pos;
   int          beg_time,end_time;
   CMyTreeNode *node; //--- Ein temporärer Zeiger auf eine Instanz der CMyTreeNode-Kla
//---
   printf("Start test %s.",__FILE__);
//--- Füllen MyTree mit den Instanzen der MyTreeNode-Klasse, ihre Anzahl ist extCounte
```

© 2000-2025, MetaQuotes Ltd.

```
   beg_time=GetTickCount();
   for(i=0;i<extCountedNodes;i++)
     {
      node=MyTree.CreateElement();
      if(node==NULL)
        {
         //--- Notausgang
         printf("%s (%4d): create error",__FILE__,__LINE__);
         return(__LINE__);
        }
      NodeSetData(node,i);
      node.SetLong(i);
      MyTree.Insert(node);
     }
   end_time=GetTickCount();
   printf("Zeit der Füllung von MyTree %d ms.",end_time-beg_time);
//--- Einen temporären Baum TmpMyTree erstellen.
   CMyTree TmpMyTree;
//--- 50% der Baumelementen ausnehmen (alle gerade Zahle)
//--- und sie in den temporären Baum TmpMyTree hinzufügen.
   beg_time=GetTickCount();
   for(i=0;i<extCountedNodes;i+=2)
     {
      node=MyTree.FindByLong(i);
      if(node!=NULL)
         if(MyTree.Detach(node)) TmpMyTree.Insert(node);
     }
   end_time=GetTickCount();
   printf("Zeit der Löschung von %d Elementen aus MyTree ist %d ms.",end_time-beg_time
//--- Die ausgenommene Elementen zurückgeben
   node=TmpMyTree.Root();
   while(node!=NULL)
     {
      if(TmpMyTree.Detach(node)) MyTree.Insert(node);
      node=TmpMyTree.Root();
     }
//--- Arbeit der Methode Save(int file_handle) überprüfen;
   int file_handle;
   file_handle=FileOpen("MyTree.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!MyTree.Save(file_handle))
        {
         //--- Fehler beim Schreiben in eine Datei
         //--- Notausgang
         printf("%s: Error %d in %d!",__FILE__,GetLastError(),__LINE__);
         //--- Vergessen SIe nicht die Datei zu schließen!!!
         FileClose(file_handle);
         return(__LINE__);
```

© 2000-2025, MetaQuotes Ltd.

```
        }
      FileClose(file_handle);
     }
//--- Arbeit der Methode Load(int file_handle) überprüfen;
   file_handle=FileOpen("MyTree.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!TmpMyTree.Load(file_handle))
        {
         //--- Fehler beim Lesen aus der Datei
         //--- Notausgang
         printf("%s: Error %d in %d!",__FILE__,GetLastError(),__LINE__);
         //--- Vergessen SIe nicht die Datei zu schließen!!!
         FileClose(file_handle);
         return(__LINE__);
        }
      FileClose(file_handle);
     }
//---
   MyTree.Clear();
   TmpMyTree.Clear();
//---
   printf("End test %s. OK!",__FILE__);
//---
   return(0);
  }
//+------------------------------------------------------------------+
//| Funktion druckt Inhalt von node in Journal                       |
//+------------------------------------------------------------------+
void NodeToLog(CMyTreeNode *node)
  {
   printf("   %I64d,%f,'%s','%s'",
              node.GetLong(),node.GetDouble(),
              node.GetString(),TimeToString(node.GetDateTime()));
  }
//+------------------------------------------------------------------+
//| Die Funktion "füllt" node mit Zufallswerte                       |
//+------------------------------------------------------------------+
void NodeSetData(CMyTreeNode *node,int mode)
  {
   if(mode%2==0)
     {
      node.SetLong(mode*MathRand());
      node.SetDouble(MathPow(2.02,mode)*MathRand());
     }
   else
     {
      node.SetLong(mode*(long)(-1)*MathRand());
      node.SetDouble(-MathPow(2.02,mode)*MathRand());
```

© 2000-2025, MetaQuotes Ltd.

```
      }
   node.SetString(str_array[mode%10]);
   node.SetDateTime(10000*mode);
  }
```

© 2000-2025, MetaQuotes Ltd.

Root

Erhält den Stammknoten des Baumes.

```
CTreeNode*  Root() const
```

Rückgabewert

Zeiger auf den Stammknoten des Baums.

© 2000-2025, MetaQuotes Ltd.

CreateElement

Erstellt eine neue Instanz des Knotens.

```
virtual CTreeNode*  CreateElement()
```

Rückgabewert

Ein Zeiger auf die neue Knoten-Instanz oder NULL.

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt einen Knoten im Baum ein.

```
CTreeNode*  Insert(
   CTreeNode*  new_node      // Knoten
   )
```

Parameter

new_node

[in] Zeiger auf den Knoten, den Sie einfügen möchten.

Rückgabewert

Ein Zeiger auf den Halter-Knoten oder NULL.

© 2000-2025, MetaQuotes Ltd.

Detach

Nimmt den angegebenen Knoten aus dem Baum aus.

```
bool  Detach(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Zeiger auf den Knoten, den Sie ausnehmen möchten.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Nach der Ausnahme aus dem Baum wird der Knoten-Zeiger freigegeben. Der Baum ist ausgewogen.

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht den angegebenen Knoten aus dem Baum.

```
bool  Delete(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Zeiger auf den Knoten, den Sie löschen möchten.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Nach der Entnahme aus dem Baum wird der Knoten-Zeiger freigegeben. Der Baum ist ausgewogen.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Baumknoten.

```
void  Clear()
```

Rückgabewert

Nichts.

Hinweis

Nach der Entnahme aus dem Baum werden die Knoten-Zeiger freigegeben.

© 2000-2025, MetaQuotes Ltd.

Find

Sucht nach einem Knoten im Baum das gleich dem Muster ist.

```
CTreeNode*  Find(
   CTreeNode*  node      // Knoten
   )
```

Parameter

node

[in] Der Knoten der die Musterdaten enthalten.

Rückgabewert

Ein Zeiger auf den gefundenen Knoten oder NULL.

© 2000-2025, MetaQuotes Ltd.

Save

Schreibt Daten des Baums in eine Datei

```
virtual bool  Save(
   int  file_handle      // Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt Baum-Daten aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Baumtyps.

```
virtual int  Type() const
```

Rückgabewert

Die Identifikator des Baumtyps.

© 2000-2025, MetaQuotes Ltd.

Template-Sammlungen von Daten

Die Bibliothek beinhaltet Klassen und Schnittstellen für die Definition von Template-Sammlungen, die es dem Nutzer erlauben, stark typisierte Sammlungen zu erstellen. Sie bieten mehr Bequemlichkeit und sichern eine höhere Leistung bei der Datenverarbeitung, als gewöhnliche typisierte Sammlungen.

Die Bibliothek befindet sich im Verzeichnis des Terminals im Ordner Include\Generic.

Objekte:

Objekt	Beschreibung	Typ
ICollection	Schnittstelle für die Implementierung von Template-Sammlungen	INTERFACE
IEqualityComparable	Schnittstelle für die Implementierung von Objekten, die man miteinander vergleichen kann	INTERFACE
IComparable	Schnittstelle für die Implementierung von Objekten, die man miteinander im Sinne von "größer als, kleiner als oder gleich" vergleichen kann	INTERFACE
IComparer	Schnittstelle für die Implementierung einer allgemeinen Klasse, die Objekte vom Typ T im Sinne von "größer als, kleiner als oder gleich" vergleicht	INTERFACE
IEqualityComparer	Schnittstelle für die Implementierung einer allgemeinen Klasse, die feststellt, ob zwei Objekte vom Typ T gleich sind	INTERFACE
IList	Schnittstelle für die Implementierung von Template-Datenlisten	INTERFACE
IMap	Schnittstelle für die Implementierung von Template-Sammlungen von Schlüssel-Wert-Paaren	INTERFACE
ISet	Schnittstelle für die Implementierung von Template-Datenmengen	INTERFACE
CDefaultComparer	Eine Hilfsklasse, die die Template-Schnittstelle IComparer<T> basierend auf den globalen Compare Methoden implementiert	CLASS
CDefaultEqualityComparer	Eine Hilfsklasse, die die Template-Schnittstelle IEqualityComparer<T> mithilfe der globalen Methoden Equals<T> und GetHashCode implementiert	CLASS

© 2000-2025, MetaQuotes Ltd.

Objekt	Beschreibung	Typ
CArrayList	Eine Template-Klasse, die die Schnittstelle IList<T> implementiert	CLASS
CKeyValuePair	Die Klasse implementiert ein Schlüssel-Wert-Paar	CLASS
CHashMap	Eine Template-Klasse, die die Schnittstelle IMap<TKey, TValue> implementiert	CLASS
CHashSet	Eine Template-Klasse, die die Schnittstelle ISet<T> implementiert	CLASS
CLinkedListNode	Eine Hilfsklasse, die für die Implementierung der Klasse CLinkedListNode<T> notwendig ist	CLASS
CLinkedList	Eine Hilfsklasse, die die Schnittstelle ICollection<T> implementiert	CLASS
CQueue	Eine Hilfsklasse, die die Schnittstelle ICollection<T> implementiert	CLASS
CRedBlackTreeNode	Eine Hilfsklasse, die für die Implementierung der Klasse CRedBlackTree<T> notwendig ist	CLASS
CRedBlackTree	Eine Hilfsklasse, die die Schnittstelle ICollection<T> implementiert	CLASS
CSortedMap	Eine Template-Klasse, die die Schnittstelle IMap<TKey, TValue> implementiert	CLASS
CSortedSet	Eine Template-Klasse, die die Schnittstelle ISet<T> implementiert	CLASS
CStack	Eine Hilfsklasse, die die Schnittstelle ICollection<T> implementiert	CLASS

Globale Methoden:

Methode	Beschreibung
ArrayBinarySearch	Sucht nach dem angegebenen Wert in einem aufsteigend sortierten eindimensionalen Array unter Verwendung der Schnittstelle IComparable<T> für den Vergleich von Elementen
ArrayIndexOf	Sucht nach dem ersten Auftreten eines Wertes in einem eindimensionalem Array
ArrayLastIndexOf	Sucht nach dem letzten Auftreten eines Wertes in einem eindimensionalem Array

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
ArrayReverse	Ändert die Sequenz von Elementen in einem eindimensionalen Array
Compare	Vergleicht zwei Elemente und stellt fest, ob eines "größer als, kleiner als oder gleich" dem anderen ist
Equals	Vergleicht, ob zwei Werte gleich sind
GetHashCode	Berechnet den Wert des Hashcodes

© 2000-2025, MetaQuotes Ltd.

ICollection<T>

ICollection<T> ist eine Schnittstelle für die Implementierung von Template-Sammlungen.

Beschreibung

Die Schnittstelle ICollection<T> definiert die grundlegenden Methoden für die Arbeit mit Sammlungen: Berechnung der Anzahl der Elemente, Bereinigung der Sammlung, Hinzufügen und Entfernen von Elementen und andere.

Deklaration

```
template<typename T>
interface ICollection
```

Überschrift

```
#include <Generic\Interfaces\ICollection.mqh>
```

Vererbungshierarchie

ICollection

Direkte Unterklassen

CLinkedList, CQueue, CRedBlackTree, CStack, IList, IMap, ISet

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einer Sammlung ein Element hinzu
Count	Gibt die Anzahl der Elemente in einer Sammlung zurück
Contains	Stellt fest, ob die Sammlung ein Element mit dem angegebenen Wert beinhaltet
CopyTo	Kopiert alle Elemente einer Sammlung in den angegebenen Array beginnend mit einem bestimmten Index.
Clear	Löscht alle Elemente einer Sammlung
Remove	Entfernt das erste Auftreten des angegebenen Elements aus einer Sammlung.

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einer Sammlung ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einer Sammlung zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob eine Sammlung ein Element mit dem angegebenen Wert beinhaltet.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn es in der Sammlung ein Element mit dem angegebenen Wert gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente einer Sammlung in den angegebenen Array beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],     // Array, in den die Elemente geschrieben werden
   const int   dst_start=0      // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in den die Elemente der Menge geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer Sammlung.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten des angegebenen Elements aus einer Sammlung.

```
bool Remove(
   T  item     // Wert des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IEqualityComparable<T>

Die Schnittstelle IEqualityComparable<T> ist eine Schnittstelle für die Implementierung von Objekten, die man miteinander vergleichen kann.

Beschreibung

Die Schnittstelle IEqualityComparable<T> definiert die Methoden für das Abrufen des Hashcodes des aktuellen Objekts und für den Vergleich hinsichtlich der Gleichheit mit einem anderen Objekt vom selben Typ.

Deklaration

```
template<typename T>
interface IEqualityComparable
```

Überschrift

```
#include <Generic\Interfaces\IEqualityComparable.mqh>
```

Vererbungshierarchie

IEqualityComparable

Direkte Unterklassen

IComparable

Methoden der Klasse

Methode	Beschreibung
Equals	Vergleicht das aktuelle Objekt mit dem angegebenen Wert
HashCode	Berechnet den Wert des Hashcodes für das aktuelle Objekt

© 2000-2025, MetaQuotes Ltd.

Equals

Vergleicht das aktuelle Objekt mit dem angegebenen Wert.

```
bool Equals(
   T  value     // Wert für den Vergleich
   );
```

Parameter

value

[in] Wert, mit welchem das aktuelle Objekt verglichen wird.

Rückgabewert

Gibt true zurück, wenn die Objekte gleich sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

HashCode

Berechnet den Wert des Hashcodes für das aktuelle Objekt.

```
int HashCode();
```

Rückgabewert

Gibt den Hashcode zurück.

© 2000-2025, MetaQuotes Ltd.

IComparable<T>

Die Schnittstelle IComparable<T> ist eine Schnittstelle für die Implementierung von Objekten, die man miteinander im Sinne von "größer als, kleiner als oder gleich" vergleichen kann.

Beschreibung

Die Schnittstelle IComparable<T> definiert die Methode für den Vergleich des aktuellen Objekts mit einem anderen Objekt vom selben Typ, auf dessen Basis die Sammlung dieser Objekte sortiert werden kann.

Deklaration

```
template<typename T>
interface IComparable : public IEqualityComparable<T>
```

Überschrift

```
#include <Generic\Interfaces\IComparable.mqh>
```

Vererbungshierarchie

IEqualityComparable

IComparable

Direkte Unterklassen

CKeyValuePair

Methoden der Klasse

Methode	Beschreibung
Compare	Vergleicht das aktuelle Objekt mit dem angegebenen Wert

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht das aktuelle Objekt mit dem angegebenen Wert.

```
int Compare(
   T  value     // Wert für den Vergleich
   );
```

Parameter

value

[in] Wert, mit welchem das aktuelle Objekt verglichen wird.

Rückgabewert

Gibt die Zahl zurück, die das Verhältnis zwischen dem aktuellen und dem übergebenen Objekten darstellt:

- wenn das Ergebnis kleiner als Null ist, ist das aktuelle Objekt kleiner als das übergebene Objekt
- wenn das Ergebnis gleich Null ist, ist das aktuelle Objekt gleich dem übergebenen Objekt
- wenn das Ergebnis größer als Null ist, ist das aktuelle Objekt größer als das übergebene Objekt

© 2000-2025, MetaQuotes Ltd.

IComparer<T>

Die Schnittstelle IComparer<T> ist eine Schnittstelle für die Implementierung einer universellen Klasse, die zwei Objekte vom Typ T im Sinne von "größer als, kleiner als oder gleich" miteinander vergleicht.

Beschreibung

Die Schnittstelle IComparable<T> definiert die Methode für den Vergleich zweier Objekte vom Typ T, auf dessen Basis die Sammlung dieser Objekte sortiert werden kann.

Deklaration

```
template<typename T>
interface IComparer
```

Überschrift

```
#include <Generic\Interfaces\IComparer.mqh>
```

Vererbungshierarchie

IComparer

Direkte Unterklassen

CDefaultComparer

Methoden der Klasse

Methode	Beschreibung
Compare	Vergleicht zwei Werte vom Typ T

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht zwei Werte vom Typ T.

```
int Compare(
   T  x,      // erster Wert
   T  y       // zweiter Wert
   );
```

Parameter

x

[in] Der erste Wert für den Vergleich.

y

[in] Der erste Wert für den Vergleich.

Rückgabewert

Gibt das Verhältnis zwischen den zwei zu vergleichenden Werten zurück:

- wenn das Ergebnis kleiner als Null ist, ist x kleiner als y (x<y)
- wenn das Ergebnis gleich Nul, ist x gleich y (x=y)
- wenn das Ergebnis größer als Null ist, ist x größer als y (x>y)

© 2000-2025, MetaQuotes Ltd.

IEqualityComparer<T>

Die Schnittstelle IEqualityComparer<T> ist eine Schnittstelle für die Implementierung einer universellen Klasse, die zwei Objekte vom Typ T miteinander vergleicht.

Beschreibung

Die Schnittstelle IEqualityComparer<T> definiert die Methoden für das Abrufen des Hashcodes eines Objekts vom Typ T und für den Vergleich hinsichtlich der Gleichheit von zwei Objekten vom Typ T.

Deklaration

```
template<typename T>
interface IEqualityComparer
```

Überschrift

```
#include <Generic\Interfaces\IEqualityComparer.mqh>
```

Vererbungshierarchie

IEqualityComparer

Direkte Unterklassen

CDefaultEqualityComparer

Methoden der Klasse

Methode	Beschreibung
Equals	Vergleicht zwei Werte vom Typ T
HashCode	Berechnet den Wert des Hash-Codes von einem Objekt vom Typ T

© 2000-2025, MetaQuotes Ltd.

Equals

Vergleicht zwei Werte vom Typ T.

```
bool Equals(
   T  x,      // erster Wert
   T  y       // zweiter Wert
   );
```

Parameter

x

[in] Der erste Wert für den Vergleich.

y

[in] Der zweite Wert für den Vergleich.

Rückgabewert

Gibt true zurück, wenn die Werte gleich sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

HashCode

Berechnet den Wert des Hashcodes von einem Objekt vom Typ T.

```
int HashCode(
   T  value     // Objekt für die Berechnung
   );
```

Parameter

value

[in] Objekt, für welchen der Hashcode erhalten werden muss.

Rückgabewert

Gibt den Hashcode zurück.

© 2000-2025, MetaQuotes Ltd.

IList<T>

Die Schnittstelle IList<T> ist eine Schnittstelle für die Implementierung von Template-Datenlisten.

Beschreibung

Die Schnittstelle IList<T> definiert die grundlegenden Methoden für die Arbeit mit Listen: Zugriff auf ein Element nach Index, Suche und Löschen eines Elements, Sortierung und andere.

Deklaration

```
template<typename T>
interface IList : public ICollection<T>
```

Überschrift

```
#include <Generic\Interfaces\IList.mqh>
```

Vererbungshierarchie

ICollection

IList

Direkte Unterklassen

CArrayList

Methoden der Klasse

Methode	Beschreibung
TryGetValue	Erhält ein Element einer Liste nach dem angegebenen Index
TrySetValue	Ändert den Wert einer Liste nach dem angegebenen Index
Insert	Fügt ein Element nach dem angegebenen Index in die Liste ein
IndexOf	Sucht nach dem ersten Auftreten eines Wertes in der Liste
LastIndexOf	Sucht nach dem letzten Auftreten eines Wertes in der Liste
RemoveAt	Löscht ein Element aus der Liste nach dem angegebenen Index

© 2000-2025, MetaQuotes Ltd.

TryGetValue

Erhält ein Element der Liste nach dem angegebenen Index

```
bool TryGetValue(
   const int  index,      // Index des Elements
   T&         value       // die Variable
   );
```

Parameter

index

[in] Index des Elements der Liste.

&value

[out] Variable, in welche der angegebene Wert eines Elements aus der Liste geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrySetValue

Ändert den Wert der Liste nach dem angegebenen Index.

```
bool TrySetValue(
   const int  index,      // Index des Elements
   T          value       // neuer Wert
   );
```

Parameter

index

[in] Index des Elements der Liste.

value

[in] Neuer Wert, der dem angegebenen Element der Liste zugewiesen werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element nach dem angegebenen Index in die Liste ein.

```
bool Insert(
   const int  index,      // Index, nach welchem ein Element eingefügt wird
   T          item        // Wert, der eingefügt wird
   );
```

Parameter

index

[in] Index, nach welchem ein Element eingefügt wird.

item

[in] Wert, der nach dem angegebenen Index eingefügt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IndexOf

Sucht nach dem ersten Auftreten eines Wertes in einer Liste.

```
int IndexOf(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt den Index des ersten gefundenen Elements zurück. Wenn kein Wert gefunden wurde, gibt -1 zurück.

© 2000-2025, MetaQuotes Ltd.

LastIndexOf

Sucht nach dem letzten Auftreten eines Wertes in einer Liste.

```
int LastIndexOf(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt den Index des letzten gefundenen Elements zurück. Wenn kein Wert gefunden wurde, gibt -1 zurück.

© 2000-2025, MetaQuotes Ltd.

RemoveAt

Entfernt ein Element aus der liste nach dem angegebenen Index.

```
bool RemoveAt(
   const int  index     // Index des Elements
   );
```

Parameter

index

[in] Index des Elements, das gelöscht werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IMap<TKey, TValue>

Die Schnittstelle IMap<TKey, TValue> ist eine Schnittstelle für die Implementierung von Template-Sammlungen von Schlüssel-Wert-Paaren.

Beschreibung

Die Schnittstelle IMap<TKey, TValue> definiert die grundlegende Methoden für das Arbeiten mit Sammlungen, deren Daten als Schlüssel-Wert-Paare gespeichert werden.

Deklaration

```
template<typename TKey, typename TValue>
interface IMap : public ICollection<TKey>
```

Überschrift

```
#include <Generic\Interfaces\IMap.mqh>
```

Vererbungshierarchie

ICollection

IMap

Direkte Unterklassen

CHashMap, CSortedMap

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einer Sammlung ein Schlüssel-Wert-Paar hinzu
Contains	Stellt fest, ob eine Sammlung ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel und Wert enthält
Remove	Entfernt das erste Auftreten eines Schlüssel-Wert-Paares mit dem angegebenen Schlüssel aus einer Sammlung
TryGetValue	Erhält ein Element nach dem angegebenen Schlüssel aus einer Sammlung
TrySetValue	Ändert den Wert eines Schlüssel-Wert-Paares aus einer Sammlung nach dem angegebenen Schlüssel
CopyTo	Kopiert alle Schlüssel-Wert-Paare aus einer Sammlung in die angegebenen Arrays beginnend mit einem bestimmten Index

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einer Sammlung ein Schlüssel-Wert-Paar hinzu.

```
bool Add(
   TKey    key,       // Schlüssel
   TValue  value      // Wert
   );
```

Parameter

key

[in] Schlüssel.

value

[in] Wert.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob eine Sammlung ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel und Wert enthält.

```
bool Contains(
   TKey    key,       // Schlüssel
   TValue  value      // Wert
   );
```

Parameter

key

[in] Schlüssel.

value

[in] Wert.

Rückgabewert

Gibt true zurück, wenn es in der Sammlung ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel und Wert gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten eines Schlüssel-Wert-Paares mit dem angegebenen Schlüssel einer Sammlung.

```
bool Remove(
   TKey  key      // Schlüssel
   );
```

Parameter

key

[in] Schlüssel.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TryGetValue

Erhält ein Element einer Sammlung nach dem angegebenen Schlüssel.

```
bool TryGetValue(
   TKey      key,       // Schlüssel
   TValue&   value      // Variable, in welche der Wert geschrieben wird
   );
```

Parameter

key

[in] Schlüssel.

&value

[out] Variable, in welche der angegebene Wert des Schlüssel-Wert-Paares geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrySetValue

Ändert den Wert eines Schlüssel-Wert-Paares aus einer Sammlung nach dem angegebenen Schlüssel.

```
bool TrySetValue(
   TKey    key,       // Schlüssel
   TValue  value      // neuer Wert
   );
```

Parameter

key

[in] Schlüssel.

value

[in] Neuer Wert, der dem angegebenen Schlüssel-Wert-Paar zugewiesen werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Schlüssel-Wert-Paare aus einer Sammlung in die angegebenen Arrays beginnend mit einem bestimmten Index.

```
int CopyTo(
   TKey&       dst_keys[],       // Array für Schlüssel
   TValue&     dst_values[],     // Array für Werte
   const int   dst_start=0       // Anfangsindex
   );
```

Parameter

&dst_keys[]

[out] Array, in welchen alle Schlüssel einer Sammlung geschrieben werden.

&dst_values[]

[out] Array, in welchen Werte der entsprechenden Schlüssel aus einer Sammlung geschrieben werden.

dst_start=0

[in] Index in den Arrays, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl kopierter Schlüssel-Wert-Paare zurück.

© 2000-2025, MetaQuotes Ltd.

ISet<T>

Die Schnittstelle ISet<T> ist eine Schnittstelle für die Implementierung von Template-Datenmengen.

Beschreibung

Die Schnittstelle ISet<T> definiert die grundlegenden Methoden für die Arbeit mit Mengen: Vereinigung, Schnitt, Definition strikter Teilmengen usw.

Deklaration

```
template<typename T>
interface ISet : public ICollection<T>
```

Überschrift

```
#include <Generic\Interfaces\ISet.mqh>
```

Vererbungshierarchie

ICollection

ISet

Direkte Unterklassen

CHashSet, CSortedSet

Methoden der Klasse

Methode	Beschreibung
ExceptWith	Führt die Operation des Komplements der aktuellen und der übergebenen Sammlung (Arrays) aus
IntersectWith	Führt die Operation des Schnitts der aktuellen und der übergebenen Sammlung (Arrays) aus
SymmetricExceptWith	Führt die Operation der symmetrischen Differenz der aktuellen und der übergebenen Sammlung (Arrays) aus
UnionWith	Führt die Operation der Vereinigung der aktuellen und der übergebenen Sammlung (Arrays) aus
IsProperSubsetOf	Stellt fest, ob die aktuelle Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist
IsProperSupersetOf	Stellt fest, ob die aktuelle Menge eine strikte Obermenge der angegebenen Sammlung oder Arrays ist
IsSubsetOf	Stellt fest, ob die aktuelle Menge eine Obermenge der angegebenen Sammlung oder Arrays ist
IsSupersetOf	Stellt fest, ob die aktuelle Menge eine Obermenge der angegebenen Sammlung oder Arrays ist

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
Overlaps	Stellt fest, ob sich die aktuelle Menge und die angegebene Sammlung oder Array überlappen
SetEquals	Stellt fest, ob die aktuelle Menge alle Elemente der angegebenen Sammlung oder Arrays beinhaltet

© 2000-2025, MetaQuotes Ltd.

ExceptWith

Führt die Operation des Komplements der aktuellen und der übergebenen Sammlung (Arrays) aus. D.h. entfernt aus der aktuellen Sammlung (Arrays) alle Elemente, die in der angegebenen Sammlung (Array) auch vorhanden sind.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void ExceptWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void ExceptWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, die von der aktuellen Menge ausgenommen wird.

&collection[]

[in] Array, der von der aktuellen Menge ausgenommen wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

IntersectWith

Führt die Operation des Schnitts der aktuellen und der übergebenen Sammlung (Arrays) aus D.h. in der aktuellen Sammlung bleiben nur die Elemente, die auch in der angegebenen Sammlung (Array) vorhanden sind.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void IntersectWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void IntersectWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, mit welcher das Produkt gebildet wird.

&collection[]

[in] Array, mit welchem das Produkt gebildet wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

SymmetricExceptWith

Führt die Operation der symmetrischen Differenz der aktuellen und der übergebenen Sammlung (Arrays) aus. D.H. die aktuelle Sammlung (Array) wird nur die Elemente enthalten, die entweder im Quellobjekt oder im übergebenen Objekt vorhanden waren, aber nicht gleichzeitig in beiden.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void SymmetricExceptWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void SymmetricExceptWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung für die symmetrische Differenz.

&collection[]

[in] Array für die symmetrische Differenz.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

UnionWith

Führt die Operation der Vereinigung der aktuellen und der übergebenen Sammlung (Arrays) aus. D.h. fügt der aktuellen Sammlung (Array) fehlende Elemente aus der angegebenen Sammlung (Array) hinzu.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void UnionWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void UnionWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, mit welcher die aktuelle Menge vereinigt wird.

&collection[]

[in] Array, mit welchem die aktuelle Menge vereinigt wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

IsProperSubsetOf

Stellt fest, ob die aktuelle Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsProperSubsetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsProperSubsetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle Menge eine strikte Untermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsProperSupersetOf

Stellt fest, ob die aktuelle Menge eine strikte Obermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsProperSupersetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsProperSupersetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle Menge eine strikte Obermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsSubsetOf

Stellt fest, ob die aktuelle Menge eine Untermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsSubsetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsSubsetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true in dem Fall zurück, wenn die aktuelle Menge eine Untermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsSupersetOf

Stellt fest, ob die aktuelle Menge eine Obermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsSupersetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsSupersetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true in dem Fall zurück, wenn die aktuelle Menge eine Obermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Overlaps

Stellt fest, ob sich die aktuelle Menge und die angegebene Sammlung oder Array überlappen.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool Overlaps(
   ICollection<T>*  collection     // Sammlung für den Vergleich
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool Overlaps(
   T&  array[]                     // Array für den Vergleich
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen der Überlappung.

&collection[]

[in] Array für das Feststellen der Überlappung.

Rückgabewert

Gibt true zurück, wenn sich die aktuelle Menge und Sammlung oder Array überlappen, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetEquals

Stellt fest, ob die aktuelle Menge alle Elemente der angegebenen Sammlung oder Arrays beinhaltet.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool SetEquals(
   ICollection<T>*  collection     // Sammlung für den Vergleich
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool SetEquals(
   T&  array[]                     // Array für den Vergleich
   );
```

Parameter

**collection*

[in] Sammlungen für den Vergleich von Elementen.

&collection[]

[in] Sammlungen für den Vergleich von Elementen.

Rückgabewert

Gibt true zurück, wenn alle Elemente der angegebenen Sammlung oder des Arrays in der aktuellen Menge vorhanden sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CDefaultComparer<T>

Die Klasse CDefaultComparer<T> ist eine Hilfsklasse, die die Template-Schnittstelle IComparer<T> basierend auf den globalen Methoden Compare implementiert.

Beschreibung

Die Klasse CDefaultComparer<T> wird in Template-Sammlungen von Daten standardmäßig verwendet, sofern der Nutzer nicht eine andere Klasse explizit verwendet, die die Schnittstelle IComparer<T> implementiert.

Deklaration

```
template<typename T>
class CDefaultComparer : public IComparer<T>
```

Überschrift

```
#include <Generic\Internal\DefaultComparer.mqh>
```

Vererbungshierarchie

IComparer

CDefaultComparer

Methoden der Klasse

Methode	Beschreibung
Compare	Vergleicht zwei Werte vom Typ T

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht zwei Werte vom Typ T.

```
int Compare(
   T  x,      // erster Wert
   T  y       // zweiter Wert
   );
```

Parameter

x

[in] Der erste Wert für den Vergleich.

y

[in] Der zweite Wert für den Vergleich.

Rückgabewert

Gibt das Verhältnis zwischen den zwei zu vergleichenden Werten zurück:

- wenn das Ergebnis kleiner als Null ist, ist x kleiner als y (x<y)
- wenn das Ergebnis gleich Nul, ist x gleich y (x=y)
- wenn das Ergebnis größer als Null ist, ist x größer als y (x>y)

Hinweis

Die Werte x und y werden basierend auf einer Überladung der globalen Methode Compare je nach dem T-Typ verglichen.

© 2000-2025, MetaQuotes Ltd.

CDefaultEqualityComparer<T>

Die Klasse CDefaultEqualityComparer<T> ist eine Hilfsklasse, die die Template-Schnittstelle IEqualityComparer<T> basierend auf den globalen Methoden Equals<T> und GetHashCode implementiert.

Beschreibung

Die Klasse CDefaultEqualityComparer<T> wird in Template-Sammlungen von Daten standardmäßig verwendet, sofern der Nutzer nicht eine andere Klasse explizit verwendet, die die Schnittstelle IEqualityComparer<T> implementiert.

Deklaration

```
template<typename T>
class CDefaultEqualityComparer : public IEqualityComparer<T>
```

Überschrift

```
#include <Generic\Internal\DefaultEqualityComparer.mqh>
```

Vererbungshierarchie

IEqualityComparer

CDefaultEqualityComparer

Methoden der Klasse

Methode	Beschreibung
Equals	Vergleicht zwei Werte vom Typ T
HashCode	Berechnet den Wert des Hash-Codes von einem Objekt vom Typ T

© 2000-2025, MetaQuotes Ltd.

Equals

Vergleicht zwei Werte vom Typ T.

```
bool Equals(
   T  x,      // erster Wert
   T  y       // zweiter Wert
   );
```

Parameter

x

[in] Der erste Wert für den Vergleich.

y

[in] Der zweite Wert für den Vergleich.

Rückgabewert

Gibt true zurück, wenn die Werte gleich sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

HashCode

Berechnet den Wert des Hashcodes von einem Objekt vom Typ T.

```
int HashCode(
   T  value     // Objekt für die Berechnung
   );
```

Parameter

value

[in] Objekt, für welchen der Hashcode erhalten werden muss.

Rückgabewert

Gibt den Hashcode zurück.

© 2000-2025, MetaQuotes Ltd.

CRedBlackTreeNode<T>

Die Klasse CRedBlackTreeNode<T> ist eine Hilfsklasse für die Implementierung der Klasse CRedBlackTree<T>.

Beschreibung

Die Klasse CRedBlackTreeNode<T> ist ein Knoten des Rot-Schwarz-Baumes CRedBlackTree<T>. In dieser Klasse sind Methoden der Navigation im Baum implementiert.

Deklaration

```
template<typename T>
class CRedBlackTreeNode
```

Überschrift

```
#include <Generic\RedBlackTree.mqh>
```

Methoden der Klasse

Methode	Beschreibung
Value	Gibt zurück und setzt den Wert des Knotens
Parent	Gibt zurück und setzt den Wert des Knotens
Left	Gibt zurück und setzt einen Pointer auf den linken Knoten
Right	Gibt zurück und setzt einen Pointer auf den rechten Knoten
Color	Gibt zurück und setzt die Farbe des Knotens
IsLeaf	Stellt fest, ob dieser Knoten ein Blatt ist
CreateEmptyNode	Erstellt einen neuen schwarzen Knoten ohne Elter und Kinder und gibt einen Pointer zum Knoten zurück

© 2000-2025, MetaQuotes Ltd.

Value (Get-Methode)

Gibt den Wert des Knotens zurück.

```
T Value();
```

Rückgabewert

Gibt den Wert des Knotens zurück.

Value (Set-Methode)

Setzt den Wert des Knotens.

```
void Value(
   T  value     // Wert des Knotens
   );
```

Parameter

value

[in] Wert des Knotens.

© 2000-2025, MetaQuotes Ltd.

Parent (Get-Methode)

Gibt einen Pointer zum Elternknoten zurück.

```
CRedBlackTreeNode<T>* Parent();
```

Rückgabewert

Gibt einen Pointer zum Elternknoten zurück.

Parent (Set-Methode)

Setzt einen Pointer zum Elterknoten.

```
void Parent(
   CRedBlackTreeNode<T>*  node     // Pointer zum Elternknoten
   );
```

Parameter

node

[in] Pointer zum Elternknoten.

© 2000-2025, MetaQuotes Ltd.

Left (Get-Methode)

Gibt einen Pointer zum linken Knoten zurück.

```
CRedBlackTreeNode<T>* Left();
```

Rückgabewert

Gibt einen Pointer zum linken Knoten zurück.

Left (Set-Methode)

Setzt einen Pointer zum linken Knoten.

```
void Left(
   CRedBlackTreeNode<T>*  node     // Pointer zum linken Knoten
   );
```

Parameter

node

[in] Pointer zum linken Knoten.

© 2000-2025, MetaQuotes Ltd.

Right (Get-Methode)

Gibt einen Pointer zum rechten Knoten zurück.

```
CRedBlackTreeNode<T>* Right();
```

Rückgabewert

Gibt einen Pointer zum rechten Knoten zurück.

Right (Set-Methode)

Setzt einen Pointer zum rechten Knoten.

```
void Right(
   CRedBlackTreeNode<T>*  node     // Pointer zum rechten Knoten
   );
```

Parameter

node

[in] Pointer zum rechten Knoten.

© 2000-2025, MetaQuotes Ltd.

Color (Get-Methode)

Gibt die Farbe des Knotens zurück.

```
ENUM_RED_BLACK_TREE_NODE_TYPE Color();
```

Rückgabewert

Gibt die Farbe des Knotens zurück.

Color (Set-Methode)

Setzt die Farbe des Knotens.

```
void Color(
   ENUM_RED_BLACK_TREE_NODE_TYPE  clr     // Farbe des Knotens
   );
```

Parameter

clr

[in] Farbe des Knotens.

Hinweis

Die Farbe des Knotens wird von der Aufzählung ENUM_RED_BLACK_TREE_NODE_TYPE gesetzt und kann von zwei Typen sein:

- RED_BLACK_TREE_NODE_RED — die rote Farbe des Knotens;
- RED_BLACK_TREE_NODE_BLACK — die schwarze Farbe des Knotens.

© 2000-2025, MetaQuotes Ltd.

IsLeaf

Stellt fest, ob dieser Knoten ein Blatt ist.

```
bool IsLeaf();
```

Rückgabewert

Gibt true zurück, wenn der Knoten ein Blatt ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CreateEmptyNode

Erstellt einen neuen schwarzen Knoten ohne Elter und Kinder und gibt einen Pointer zum Knoten zurück.

```
static CRedBlackTreeNode<T>* CreateEmptyNode();
```

Rückgabewert

Gibt einen Pointer zum neuen Knoten zurück.

© 2000-2025, MetaQuotes Ltd.

CLinkedListNode<T>

Die Klasse CLinkedListNode<T> ist eine Hilfsklasse, die die Klasse CLinkedListNode<T> implementiert.

Beschreibung

Die Klasse CLinkedListNode<T> ist ein Knoten der doppelt verketteten Liste CLinkedListNode<T>. In dieser Klasse sind Methoden der Navigation in der Liste implementiert.

Deklaration

```
template<typename T>
class CLinkedListNode
```

Überschrift

```
#include <Generic\LinkedList.mqh>
```

Methoden der Klasse

Methode	Beschreibung
List	Gibt zurück und setzt einen Pointer zur doppelt verketteten Liste CLinkedList<T>
Next	Gibt zurück und setzt einen Pointer zum nachfolgenden Knoten
Previous	Gibt zurück und setzt einen Pointer zum vorherigen Knoten
Value	Gibt zurück und setzt den Wert des Knotens

© 2000-2025, MetaQuotes Ltd.

List (Get-Methode)

Gibt einen Pointer zur doppelt verketteten Liste CLinkedList<T> zurück.

```
CLinkedList<T>* List();
```

Rückgabewert

Gibt einen Pointer zur doppelt verketteten Liste CLinkedList<T> zurück.

List (Set-Methode)

Setzt einen Pointer zur doppelt verketteten Liste CLinkedList<T>.

```
void List(
   CLinkedList<T>*  value     // Pointer zur Liste
   );
```

Parameter

value

[in] Pointer zur doppelt verketteten Liste CLinkedList<T>.

© 2000-2025, MetaQuotes Ltd.

Next (Get-Methode)

Gibt einen Pointer zum nachfolgenden Knoten zurück.

```
CLinkedListNode<T>* Next();
```

Rückgabewert

Gibt einen Pointer zum nachfolgenden Knoten zurück.

Next (Set-Methode)

Setzt einen Pointer zum nachfolgenden Knoten.

```
void Next(
   CLinkedListNode<T>*  value     // Pointer zum nachfolgenden Knoten
   );
```

Parameter

value

[in] Pointer zum nachfolgenden Knoten.

© 2000-2025, MetaQuotes Ltd.

Previous (Get-Methode)

Gibt einen Pointer zum vorherigen Knoten zurück.

```
CLinkedListNode<T>* Previous();
```

Rückgabewert

Gibt einen Pointer zum vorherigen Knoten zurück.

Previous (Set-Methode)

Setzt einen Pointer zum vorherigen Knoten.

```
void Previous(
   CLinkedListNode<T>*  value     // Pointer zum vorherigen Knoten
   );
```

Parameter

value

[in] Pointer zum vorherigen Knoten.

© 2000-2025, MetaQuotes Ltd.

Value (Get-Methode)

Gibt den Wert des Knotens zurück.

```
T Value();
```

Rückgabewert

Gibt den Wert des Knotens zurück.

Value (Set-Methode)

Setzt den Wert des Knotens.

```
void Value(
   T  value     // Wert des Knotens
   );
```

Parameter

value

[in] Wert des Knotens.

© 2000-2025, MetaQuotes Ltd.

CKeyValuePair<TKey,TValue>

Die Klasse CKeyValuePair<TKey,TValue> implementiert ein Schlüssel-Wert-Paar.

Beschreibung

Die Klasse CKeyValuePair<TKey,TValue> implementiert Methoden für die Arbeit mit dem Schlüssel und dem Wert eines Schlüssel-Wert-Paares.

Deklaration

```
template<typename TKey, typename TValue>
class CKeyValuePair : public IComparable<CKeyValuePair<TKey,TValue>*>
```

Überschrift

```
#include <Generic\HashMap.mqh>
```

Vererbungshierarchie

IEqualityComparable
  IComparable
    CKeyValuePair

Methoden der Klasse

Methode	Beschreibung
Key	Gibt zurück und setzt den Schlüssel des Schlüssel-Wert-Paares
Value	Gibt zurück und setzt den Wert eines Schlüssel-Wert-Paares
Clone	Erstellt ein neues Schlüssel-Wert-Paar, dessen Schlüssel und Wert den aktuellen gleich sind
Compare	Vergleicht das aktuelle Schlüssel-Wert-Paar mit dem angegebenen Paar
Equals	Stellt fest, ob das aktuelle Schlüssel-Wert-Paar dem angegebenen Paar gleich ist
HashCode	Berechnet den Wert des Hash-Codes basierend auf dem Schlüssel-Wert-Paar

© 2000-2025, MetaQuotes Ltd.

Key (Get-Methode)

Gibt den Schlüssel eines Schlüssel-Wert-Paares zurück.

```
TKey Key();
```

Rückgabewert

Gibt den Schlüssel zurück.

Key (Set-Methode)

Setzt den Wert des Schlüssel-Wert-Paares zurück.

```
void Key(
   TKey  key     // Schlüssel
   );
```

Parameter

key

[in] Schlüssel.

© 2000-2025, MetaQuotes Ltd.

Value (Get-Methode)

Gibt den Wert des Schlüssel-Wert-Paares zurück.

```
TValue Value();
```

Rückgabewert

Gibt den Wert zurück.

Value (Set-Methode)

Setzt den Wert des Schlüssel-Wert-Paares.

```
void Value(
   TValue  value     // Wert
   );
```

Parameter

value

[in] Wert

© 2000-2025, MetaQuotes Ltd.

Clone

Erstellt ein neues Schlüssel-Wert-Paar, dessen Schlüssel und Wert den aktuellen gleich sind.

```
TValue>* Clone();
```

Rückgabewert

Gibt das neue Schlüssel-Wert-Paar zurück

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht das aktuelle Schlüssel-Wert-Paar mit dem angegebenen Paar.

```
int Compare(
   CKeyValuePair<TKeyTValue>*  pair     // das Paar für den Vergleich
   );
```

Parameter

pair

[in] Paar für den Vergleich.

Rückgabewert

Gibt die Zahl zurück, die das Verhältnis zwischen dem aktuellen und dem übergebenen Schlüssel-Wert-Paar darstellt:

- das Ergebnis ist kleiner als Null — das aktuelle Schlüssel-Wert-Paar ist kleiner als das übergebene Paar
- das Ergebnis ist gleich Null — das aktuelle Schlüssel-Wert-Paar ist gleich dem übergebenen Paar
- das Ergebnis ist größer als Null — das aktuelle Schlüssel-Wert-Paar ist größer

Hinweis

Zwei Schlüssel-Wert-Paare werden nach dem Schlüssel verglichen.

© 2000-2025, MetaQuotes Ltd.

Equals

Stellt fest, ob das aktuelle Schlüssel-Wert-Paar dem angegebenen Paar gleich ist.

```
bool Equals(
   CKeyValuePair<TKeyTValue>*  pair     // Paar für den Vergleich
   );
```

Parameter

pair

[in] Paar für den Vergleich

Rückgabewert

Gibt true zurück, wenn die Schlüssel-Wert-Paare gleich sind, andernfalls false.

Hinweis

Zwei Schlüssel-Wert-Paare werden nach dem Schlüssel verglichen.

© 2000-2025, MetaQuotes Ltd.

HashCode

Berechnet den Wert des Hash-Codes basierend auf einem Schlüssel-Wert-Paar.

```
int HashCode();
```

Rückgabewert

Gibt den Hashcode zurück.

Hinweis

Der Hachcode des eines Schlüssel-Wert-Paares ist gleich dem Hashcode des Schlüssels.

© 2000-2025, MetaQuotes Ltd.

CArrayList<T>

Die Klasse CArrayList<T> ist ein Template-Klasse, die die Schnittstelle IList<T> implementiert.

Beschreibung

Die Klasse CArrayList<T> implementiert eine dynamische Liste der Daten vom Typ T. Diese Klasse stellt die grundlegenden Methoden für das Arbeiten mit einer Liste bereit: Zugriff auf ein Element nach seinem Index, Suchen und Löschen des Elements, Sortieren und andere.

Deklaration

```
template<typename T>
class CArrayList : public IList<T>
```

Überschrift

```
#include <Generic\ArrayList.mqh>
```

Vererbungshierarchie

ICollection
  IList
    CArrayList

Methoden der Klasse

Methode	Beschreibung
Capacity	Gibt zurück und setzt die aktuelle Kapazität einer Liste
Count	Gibt die Anzahl der Elemente in einer Liste zurück
Contains	Legt fest, ob eine Liste das Element mit dem angegebenen Wert beinhaltet
TrimExcess	Reduziert die Kapazität einer Liste auf die tatsächliche Zahl der Elemente
TryGetValue	Erhält ein Element der Liste nach dem angegebene Index
TrySetValue	Setzt den Wert eines Elements der Liste nach dem angegebenen Index
Add	Fügt der Liste ein Element hinzu
AddRange	Fügt der Liste eine Sammlung oder ein Array von Elementen hinzu
Insert	Fügt ein Element nach dem angegebenen Index in die Liste ein
InsertRange	Fügt eine Sammlung oder ein Array von Elementen nach dem angegebenen Index in die Liste ein
CopyTo	Kopiert alle Elemente einer Liste in den angegebenen Array, beginnend mit einem bestimmten Index

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
BinarySearch	Sucht nach dem angegebenen Wert in einer aufsteigend sortierten Liste
IndexOf	Sucht nach dem ersten Auftreten eines Wertes in der Liste
LastIndexOf	Sucht nach dem letzten Auftreten eines Wertes in der Liste
Clear	Löscht alle Elemente einer Sammlung
Remove	Entfernt das erste Auftreten des angegebenen Elements aus der Liste
RemoveAt	Löscht das angegebene Element nach dem angegebenen Index aus der Liste
RemoveRange	Entfernt einen Bereich von Elementen aus der Liste
Reverse	Ändert die Reihenfolge von Elementen in der Liste
Sort	Sortiert Elemente der Liste

© 2000-2025, MetaQuotes Ltd.

Capacity (Get-Methode)

Gibt die aktuelle Kapazität der Liste zurück.

```
int Capacity();
```

Rückgabewert

Gibt die aktuelle Kapazität der Liste zurück.

Capacity (Set-Methode)

Setzt die aktuelle Kapazität der Liste.

```
void Capacity(
   const int  capacity     // Wert der Kapazität
   );
```

Parameter

capacity

[in] Neuer Wert der Kapazität.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in der Liste zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Legt fest, ob die Liste das Element mit dem angegebenen Wert beinhaltet.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn das Element mit dem angegebenen Wert in der Liste vorhanden ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrimExcess

Reduziert die Kapazität einer Liste auf die tatsächliche Anzahl der Elemente und setzt dadurch den ungenutzten Speicherplatz frei.

```
void TrimExcess();
```

© 2000-2025, MetaQuotes Ltd.

TryGetValue

Erhält ein Element der Liste nach dem angegebenen Index.

```
bool TryGetValue(
   const int  index,      // Index
   T&         value       // Variable, in welcher der Wert des Elements geschrieben wi
   );
```

Parameter

index

[in] Index des Elements der Liste, dessen Wert erhalten werden muss.

&value

[out] Variable, in welche der Wert des Elements geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrySetValue

Setzt den Wert des Elements einer Liste nach dem angegebenen Index.

```
bool TrySetValue(
   const int  index,      // Index
   T          value       // Wert des Elements
   );
```

Parameter

index

[in] Index des Elements der Liste, dessen Wert gesetzt werden muss.

value

[in] Wert des Elements der Liste, der gesetzt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Add

Fügt der Liste ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

AddRange

Fügt der Liste eine Sammlung oder ein Array von Elementen hinzu.

Die Version für das Hinzufügen eines Arrays.

```
bool AddRange(
   const T&  array[]               // Array, der hinzugefügt wird
   );
```

Die Version für das Hinzufügen einer Sammlung.

```
bool AddRange(
   ICollection<T>*  collection     // Sammlung, die hinzugefügt wird
   );
```

Parameter

&array[]

[in] Array, der hinzugefügt wird.

**collection*

[in] Sammlung, die hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt ein Element nach dem angegebenen Index in die Liste ein.

```
bool Insert(
   const int  index,      // Index, der eingefügt werden muss
   T          item        // Wert, der eingefügt werden muss
   );
```

Parameter

index

[in] Index, nach welchem ein Element eingefügt wird.

item

[in] Wert, der nach dem angegebenen Index eingefügt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

InsertRange

Fügt eine Sammlung oder einen Array nach dem angegebenen Index in die Liste ein.

Eine Version für das Einfügen eines Arrays.

```
bool InsertRange(
   const int  index,               // Index, der eingefügt werden muss
   const T&   array[]              // Array, der eingefügt werden muss
   );
```

Eine Version für das Einfügen einer Sammlung.

```
bool InsertRange(
   const int        index,         // Index, der eingefügt werden muss
   ICollection<T>*  collection     // Sammlung, die eingefügt werden muss
   );
```

Parameter

index

[in] Index, nach welchem ein Element eingefügt wird.

&array[]

[in] Array, der nach dem angegebenen Index eingefügt werden muss.

**collection*

[in] Sammlung, die nach dem angegebenen Index eingefügt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente einer Liste in den angegebenen Array, beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],      // Array
   const int   dst_start=0       // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in den die Elemente der Liste geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

BinarySearch

Sucht nach dem angegebenen Wert in einer aufsteigend sortierten Liste.

Eine Version für die Suche nach dem angegebenen Wertebereich unter Verwendung der Klasse, die die Schnittstelle IComparable<T> für den Vergleich von Elementen implementiert.

```
int BinarySearch(
   const int       index,        // Anfangsindex
   const int       count,        // Suchbereich
   T               item,         // der gesuchte Wert
   IComparer<T>*   comparer      // Schnittstelle für den Vergleich
   );
```

Eine Version für die Suche unter Verwendung der Klasse, die die Schnittstelle IComparable<T> für den Vergleich von Elementen implementiert.

```
int BinarySearch(
   T               item,         // der gesuchte Wert
   IComparer<T>*   comparer      // Interface für den Vergleich
   );
```

Eine Version für die Suche unter Verwendung der globalen Methode ::Compare für den Vergleich von Elementen.

```
int BinarySearch(
   T  item                       // der gesuchte Wert
   );
```

Parameter

index

[in] Anfangsindex, mit welchem die Suche beginnt.

count

[in] Länge des Suchbereichs.

item

[in] Gesuchter Wert.

**comparer*

[in] Schnittstelle für den Vergleich der Elemente.

Rückgabewert

Gibt den Index des ermittelten Elements zurück. Wenn der Wert nicht ermittelt ist, gibt den Index des kleineren Elements zurück, das dem gesuchten Wert am nächsten kommt.

© 2000-2025, MetaQuotes Ltd.

IndexOf

Sucht nach dem ersten Auftreten eines Wertes in einer Liste.

Eine Version für die Suche in der ganzen Liste.

```
int IndexOf(
   T  item                    // der gesuchte Wert
   );
```

Eine Version für die Suche vom angegebenen Element bis zum Ende der Liste.

```
int IndexOf(
   T           item,          // der gesuchte Wert
   const int   start_index    // Anfangsindex
   );
```

Eine Version für die Suche beginnend mit dem angegebenen Element im angegebenen Bereich.

```
int IndexOf(
   T           item,          // der gesuchte Wert
   const int   start_index,   // Anfangsindex
   const int   count          // Suchbereich
   );
```

Parameter

item

[in] Gesuchter Wert.

start_index

[in] Anfangsindex, mit welchem die Suche beginnt.

count

[in] Länge des Suchbereichs.

Rückgabewert

Gibt den Index des ersten gefundenen Elements zurück. Wenn kein Wert gefunden wurde, gibt -1 zurück.

© 2000-2025, MetaQuotes Ltd.

LastIndexOf

Sucht nach dem letzten Auftreten eines Wertes in einer Liste.

Eine Version für die Suche in der ganzen Liste.

```
int LastIndexOf(
   T  item                    // der gesuchte Wert
   );
```

Eine Version für die Suche vom angegebenen Element bis zum Ende der Liste.

```
int LastIndexOf(
   T           item,          // der gesuchte Wert
   const int   start_index    // Anfangsindex
   );
```

Eine Version für die Suche beginnend mit dem angegebenen Element im angegebenen Bereich.

```
int LastIndexOf(
   T           item,          // der gesuchte Wert
   const int   start_index,   // Anfangsindex
   const int   count          // Suchbereich
   );
```

Parameter

item

[in] Gesuchter Wert.

start_index

[in] Anfangsindex, mit welchem die Suche beginnt.

count

[in] Länge des Suchbereichs.

Rückgabewert

Gibt den Index des letzten gefundenen Elements zurück. Wenn kein Wert gefunden wurde, gibt -1 zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer Sammlung.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten des angegebenen Elements in einer Liste.

```
bool Remove(
   T  item     // Wert des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

RemoveAt

Entfernt das angegebene Element nach dem angegebenen Index aus der Liste.

```
bool RemoveAt(
   const int  index     // Index
   );
```

Parameter

index

[in] Index des Elements, das entfernt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

RemoveRange

Entfernt einen Bereich der Elemente aus der Liste.

```
bool RemoveRange(
   const int  start_index,     // Anfangsindex
   const int  count            // Anzahl der Elemente
   );
```

Parameter

start_index

[in] Anfangsindex, mit welchem das Entfernen beginnt.

count

[in] Anzahl der zu löschenden Elemente.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Reverse

Ändert die Reihen der Elemente in der Liste.

Eine Version für das Arbeiten mit der ganzen Liste.

```
bool Reverse();
```

Eine Version für das Arbeiten mit einem vorgegebenen Bereich der Elemente einer Liste.

```
bool Reverse(
   const int  start_index,     // Anfangsindex
   const int  count            // Anzahl der Elemente
   );
```

Parameter

start_index

[in] Anfangsindex.

count

[in] Anzahl der Elemente der Liste, die in die Operation einbezogen sind.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Sort

Sortiert Elemente in der Liste.

Eine Version für das Sortieren aller Elemente der Liste.

```
bool Sort();
```

Eine Version für das Sortieren aller Elemente der Liste unter Verwendung der Klasse, die die Schnittstelle IComparable<T> für den Vergleich von Elementen implementiert.

```
bool Sort(
   IComparer<T>*  comparer         // Schnittstelle für den Vergleich
   );
```

Eine Version für das Sortieren des angegebenen Bereichs der Elemente der Liste unter Verwendung der Klasse, die die Schnittstelle IComparable<T> für den Vergleich von Elementen implementiert.

```
bool Sort(
   const int      start_index,     // Anfangsindex
   const int      count,           // Anzahl der Elemente
   IComparer<T>*  comparer         // Schnittstelle für den Vergleich
   );
```

Parameter

**comparer*

[in] Schnittstelle für den Vergleich der Elemente.

start_index

[in] Anfangsindex, mit welchem das Sortieren beginnt.

count

[in] Länge des Sortiersbereichs.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CHashMap<TKey,TValue>

Die Klasse CHashMap<TKey,TValue> ist eine Template-Klasse, die die Schnittstelle IMap<TKey,TValue> implementiert.

Beschreibung

Die Klasse CHashMap<TKey,TValue> ist eine Implementierung der Hashtabelle, deren Daten als nicht geordnete Schlüssel-Wert-Paare unter Einhaltung der Bedingung der Einmaligkeit des Schlüssels gespeichert werden. Diese Klasse stellt die grundlegenden Methoden für die Arbeit mit der Hashtabelle bereit: Zugriff auf den Wert nach Schlüssel, Suche und Entfernen eines Schlüssel-Wert-Paares und andere.

Deklaration

```
template<typename TKey, typename TValue>
class CHashMap : public IMap<TKey,TValue>
```

Überschrift

```
#include <Generic\HashMap.mqh>
```

Vererbungshierarchie

ICollection
  IMap
    CHashMap

Methoden der Klasse

Methode	Beschreibung
Add	Fügt der Hashtabelle ein Schlüssel-Wert-Paar hinzu
Count	Gibt die Anzahl der Elemente in der Hashtabelle hinzu
Comparer	Gibt den Pointer auf die Schnittstelle IEqualityComparer<T> zurück, der für eine Hashtabelle verwendet wird
Contains	Stellt fest, ob die Hashtabelle das angegebene Schlüssel-Wert-Paar beinhaltet
ContainsKey	Stellt fest, ob die Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel beinhaltet
ContainsValue	Die Klasse CHashMap<TKey, TValue> ist eine Template-Klasse, die die Schnittstelle IMap<TKey,TValue> implementiert
CopyTo	Kopiert alle Schlüssel-Wert-Paare aus der Hashtabelle in die angegebenen Arrays, beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente der Hashtabelle
Remove	Entfernt das erste Auftreten des Schlüssel-Wert-Paares der Hashtabelle

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
TryGetValue	Erhält ein Element nach dem angegebenen Schlüssel aus der Hashtabelle
TrySetValue	Ändert den Wert eines Schlüssel-Wert-Paares der Hashtabelle nach dem angegebenen Schlüssel

© 2000-2025, MetaQuotes Ltd.

Add

Fügt der Hashtabelle ein Schlüssel-Wert-Paar hinzu.

Eine Version für das Hinzufügen eines Schlüssel-Wert-Paares.

```
bool Add(
   CKeyValuePair<TKeyTValue>*  pair     // Schlüssel-Wert-Paar
   );
```

Eine Version für das Hinzufügen eines neuen Schlüssel-Wert-Paares mit dem angegebenen Schlüssel und Wert.

```
bool Add(
   TKey    key,                         // Schlüssel
   TValue  value                        // Wert
   );
```

Parameter

**pair*

[in] Schlüssel-Wert-Paar.

key

[in] Schlüssel.

value

[in] Wert.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in der Hashtabelle zurück.

```
int Count();
```

© 2000-2025, MetaQuotes Ltd.

Comparer

Gibt den Pointer auf die Schnittstelle IEqualityComparer<T> zurück, die für die Gestaltung der Hashtabelle verwendet wird.

```
IEqualityComparer<TKey>* Comparer() const;
```

Rückgabewert

Gibt den Pointer auf die Schnittstelle IEqualityComparer<T> zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob die Hashtabelle das angegebene Schlüssel-Wert-Paar beinhaltet.

Eine Version für das Arbeiten mit einem gebildeten Schlüssel-Wert-Paar.

```
bool Contains(
   CKeyValuePair<TKeyTValue>*  item     // Schlüssel-Wert-Paar
   );
```

Eine Version für das Arbeiten mit einem Schlüssel-Wert-Paar in Form eines separat angegebenen Schlüssels und Wertes.

```
bool Contains(
   TKey   key,                          // Schlüssel
   TValue value                         // Wert
   );
```

Parameter

**item*

[in] Schlüssel-Wert-Paar.

key

[in] Schlüssel.

value

[in] Wert.

Rückgabewert

Gibt true zurück, wenn es in der Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel und Wert gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

ContainsKey

Stellt fest, ob die Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel beinhaltet.

```
bool ContainsKey(
   TKey  key      // ключ
   );
```

Parameter

key

[in] Schlüssel.

Rückgabewert

Gibt true zurück, wenn ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel in der Hashtabelle vorhanden ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

ContainsValue

Stellt fest, ob die Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Wert beinhaltet.

```
bool ContainsValue(
   TValue  value     // Wert
   );
```

Parameter

value

[in] Wert.

Rückgabewert

Gibt true zurück, wenn die Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Wert beinhaltet, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Schlüssel-Wert-Paare aus der Hashtabelle in die angegebenen Arrays, beginnend mit einem bestimmten Index.

Eine Version für das Kopieren der Hashtabelle in den Array von Schlüssel-Wert-Paaren.

```
int CopyTo(
   CKeyValuePair<TKeyTValue>*&  dst_array[],     // Array, in den Schlüssel-Wert-Paare
   const int                    dst_start=0      // Anfangsindex
   );
```

Eine Version für das Kopieren der Hashtabelle in separate Arrays für Schlüssel und Werte.

```
int CopyTo(
   TKey&       dst_keys[],                       // Array, in den Schlüssel geschriebe
   TValue&     dst_values[],                     // Array, in den Werte geschrieben we
   const int   dst_start=0                       // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in den alle Paare aus der Hashtabelle geschrieben werden.

&dst_keys[]

[out] Array, in den alle Schlüssel aus der Hashtabelle geschrieben werden.

&dst_values[]

[out] Array, in den alle Werte der Hashtabelle geschrieben werden.

dst_start=0

[in] Index des Arrays, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl kopierter Schlüssel-Wert-Paare zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente der Hashtabelle.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten des Schlüssel-Wert-Paares der Hashtabelle.

Eine Version für das Entfernen des Schlüssel-Wert-Paares nach dem gebildeten Schlüssel-Wert-Paar.

```
bool Remove(
   CKeyValuePair<TKeyTValue>*  item     // Schlüssel-Wert-Paar
   );
```

Eine Version für das Entfernen des Schlüssel-Wert-Paares nach dem Schlüssel.

```
bool Remove(
   TKey  key                           // Schlüssel
   );
```

Parameter

**item*

[in] Schlüssel-Wert-Paar.

key

[in] Schlüssel.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TryGetValue

Erhält ein Element nach dem angegebenen Schlüssel aus der Hashtabelle.

```
bool TryGetValue(
   TKey      key,       // Schlüssel
   TValue&   value      // Variable, in welche die Werte geschrieben werden
   );
```

Parameter

key

[in] Schlüssel.

&value

[out] Variable, in welche der angegebene Wert des Schlüssel-Wert-Paares geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrySetValue

Ändert den Wert eines Schlüssel-Wert-Paares aus der Hashtabelle nach dem angegebenen Schlüssel.

```
bool TrySetValue(
   TKey    key,       // Schlüssel
   TValue  value      // neuer Wert
   );
```

Parameter

key

[in] Schlüssel.

value

[in] Neuer Wert, der dem angegebenen Schlüssel-Wert-Paar zugewiesen werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CHashSet<T>

Die Klasse CHashSet<T> ist ein Template-Klasse, die die Schnittstelle ISet<T> implementiert.

Beschreibung

Die Klasse CHashSet<T> implementiert eine nicht geordnete dynamische Datenmenge vom Typ T unter Einhaltung der Bedingung der Einmaligkeit jedes Wertes. Diese Klasse bietet die grundlegenden Methoden für die Arbeit und Operationen mit Datenmengen, Definieren strikter Untermengen und andere.

Deklaration

```
template<typename T>
class CHashSet : public ISet<T>
```

Überschrift

```
#include <Generic\HashSet.mqh>
```

Vererbungshierarchie

ICollection
  ISet
    CHashSet

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einer Menge ein Element hinzu
Count	Gibt die Anzahl der Elemente in einer Menge zurück
Comparer	Stellt fest, ob die Menge das Element mit dem angegebenen Wert beinhaltet
Contains	Gibt den Pointer auf die Schnittstelle IEqualityComparer<T> zurück, der für das Organisieren einer Menge verwendet wird
TrimExcess	Reduziert die Kapazität einer Menge auf die tatsächliche Anzahl der Elemente und setzt dadurch den ungenutzten Speicherplatz frei
CopyTo	Kopiert alle Elemente einer Menge in den angegebenen Array, beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente einer Menge
Remove	Entfernt das angegebene Element aus einer Menge
ExceptWith	Führt die Operation des Komplements der aktuellen und der übergebenen Sammlung (Arrays) aus

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
IntersectWith	Führt die Operation des Schnitts der aktuellen und der übergebenen Sammlung (Arrays) aus
SymmetricExceptWith	Führt die Operation der symmetrischen Differenz der aktuellen und der übergebenen Sammlung (Arrays) aus
UnionWith	Führt die Operation der Vereinigung der aktuellen und der übergebenen Sammlung (Arrays) aus
IsProperSubsetOf	Stellt fest, ob die aktuelle Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist
IsProperSupersetOf	Stellt fest, ob die aktuelle Menge eine strikte Obermenge der angegebenen Sammlung oder Arrays ist
IsSubsetOf	Stellt fest, ob die aktuelle Menge eine Obermenge der angegebenen Sammlung oder Arrays ist
IsSupersetOf	Stellt fest, ob die aktuelle Menge eine Obermenge der angegebenen Sammlung oder Arrays ist
Overlaps	Stellt fest, ob sich die aktuelle Menge und die angegebene Sammlung oder Array überlappen
SetEquals	Stellt fest, ob die aktuelle Menge alle Elemente der angegebenen Sammlung oder Arrays beinhaltet

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einer Menge ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einer Menge zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob eine Menge das Element mit dem angegebenen Wert beinhaltet.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn das Element mit dem angegebenen Wert in der Menge vorhanden ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Comparer

Gibt den Pointer auf die Schnittstelle IEqualityComparer<T> zurück, die für das Organisieren einer Menge verwendet wird.

```
IEqualityComparer<T>* Comparer() const;
```

Rückgabewert

Gibt den Pointer auf die Schnittstelle IEqualityComparer<T> zurück.

© 2000-2025, MetaQuotes Ltd.

TrimExcess

Reduziert die Kapazität einer Menge auf die tatsächliche Anzahl der Elemente und setzt dadurch den ungenutzten Speicherplatz frei.

```
void TrimExcess();
```

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente einer Menge in den angegebenen Array, beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],     // Array, in welchen die Elemente geschrieben werden
   const int   dst_start=0      // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in den die Elemente einer Menge geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer Menge.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das angegebene Element aus einer Menge.

```
bool Remove(
   T  item     // Wert des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

ExceptWith

Führt die Operation des Komplements der aktuellen und der übergebenen Sammlung (Arrays) aus. D.h. entfernt aus der aktuellen Sammlung (Arrays) alle Elemente, die in der angegebenen Sammlung (Array) auch vorhanden sind.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void ExceptWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void ExceptWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, die von der aktuellen Menge ausgenommen wird.

&collection[]

[in] Array, der von der aktuellen Menge ausgenommen wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

IntersectWith

Führt die Operation des Schnitts der aktuellen und der übergebenen Sammlung (Arrays) aus D.h. in der aktuellen Sammlung bleiben nur die Elemente, die auch in der angegebenen Sammlung (Array) vorhanden sind.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void IntersectWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void IntersectWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, mit welcher das Produkt gebildet wird.

&collection[]

[in] Array, mit welchem das Produkt gebildet wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

SymmetricExceptWith

Führt die Operation der symmetrischen Differenz der aktuellen und der übergebenen Sammlung (Arrays) aus. D.H. die aktuelle Sammlung (Array) wird nur die Elemente enthalten, die entweder im Quellobjekt oder im übergebenen Objekt vorhanden waren, aber nicht gleichzeitig in beiden.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void SymmetricExceptWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void SymmetricExceptWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung für die symmetrische Differenz.

&collection[]

[in] Array für die symmetrische Differenz.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

UnionWith

Führt die Operation der Vereinigung der aktuellen und der übergebenen Sammlung (Arrays) aus. D.h. fügt der aktuellen Sammlung (Array) fehlende Elemente aus der angegebenen Sammlung (Array) hinzu.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void UnionWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void UnionWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, mit welcher die aktuelle Menge vereinigt wird.

&collection[]

[in] Array, mit welchem die aktuelle Menge vereinigt wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

IsProperSubsetOf

Stellt fest, ob die aktuelle Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsProperSubsetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsProperSubsetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle Menge eine strikte Untermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsProperSupersetOf

Stellt fest, ob die aktuelle Menge eine strikte Obermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsProperSupersetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsProperSupersetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle Menge eine strikte Obermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsSubsetOf

Stellt fest, ob die aktuelle Menge eine Untermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsSubsetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsSubsetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true in dem Fall zurück, wenn die aktuelle Menge eine Untermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsSupersetOf

Stellt fest, ob die aktuelle Menge eine Obermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsSupersetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsSupersetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true in dem Fall zurück, wenn die aktuelle Menge eine Obermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Overlaps

Stellt fest, ob sich die aktuelle Menge und die angegebene Sammlung oder Array überlappen.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool Overlaps(
   ICollection<T>*  collection     // Sammlung für den Vergleich
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool Overlaps(
   T&  array[]                     // Array für den Vergleich
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen der Überlappung.

&collection[]

[in] Array für das Feststellen der Überlappung.

Rückgabewert

Gibt true zurück, wenn sich die aktuelle Menge und Sammlung oder Array überlappen, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetEquals

Stellt fest, ob die aktuelle Menge alle Elemente der angegebenen Sammlung oder Arrays beinhaltet.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool SetEquals(
   ICollection<T>*  collection     // Sammlung für den Vergleich
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool SetEquals(
   T&  array[]                     // Array für den Vergleich
   );
```

Parameter

**collection*

[in] Sammlungen für den Vergleich von Elementen.

&collection[]

[in] Array für den Vergleich von Elementen.

Rückgabewert

Gibt true zurück, wenn alle Elemente der angegebenen Sammlung oder des Arrays in der aktuellen Menge vorhanden sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CLinkedList<T>

Die Klasse CLinkedList<T> ist eine Template-Klasse, die die Schnittstelle ICollection<T> implementiert.

Beschreibung

Die Klasse CLinkedList<T> implementiert eine dynamische doppelt verkettete Liste der Daten vom Typ T. Diese Klasse stellt die wichtigsten Methoden für das Arbeiten mit doppelt verketteten Listen bereit: Hinzufügen, Entfernen, Suche nach dem Element und andere.

Deklaration

```
template<typename T>
class CLinkedList : public ICollection<T>
```

Überschrift

```
#include <Generic\LinkedList.mqh>
```

Vererbungshierarchie

ICollection

CLinkedList

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einer doppelt verketteten Liste ein Element hinzu
AddAfter	Fügt einer doppelt verketteten Liste ein Element nach dem angegebenen Knoten hinzu
AddBefore	Fügt einer doppelt verketteten Liste ein Element vor dem angegebenen Knoten hinzu
AddFirst	Fügt ein Element am Anfang einer doppelt verketteten Liste hinzu
AddLast	Fügt ein Element am Ende einer doppelt verketteten Liste hinzu
Count	Gibt die Anzahl der Elemente in einer doppelt verketteten Liste zurück
Head	Gibt einen Pointer zum ersten Knoten einer doppelt verketteten Liste zurück
First	Gibt einen Pointer zum ersten Knoten einer doppelt verketteten Liste zurück
Last	Gibt einen Pointer zum letzten Knoten einer doppelt verketteten Liste zurück
Contains	Stellt fest, ob die doppelt verkettete Liste das Element mit dem angegebenen Wert enthält

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
CopyTo	Kopiert alle Elemente einer Liste in den angegebenen Array, beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente einer Sammlung
Remove	Entfernt das erste Auftreten des angegebenen Elements aus der doppelt Liste
RemoveFirst	Entfernt das erste Element aus einer doppelt verketteten Liste
RemoveLast	Entfernt das letzte Element aus einer doppelt verketteten Liste
Find	Sucht nach dem ersten Auftreten des angegebenen Wertes in einer doppelt verketteten Liste
FindLast	Sucht nach dem # Auftreten des angegebenen Wertes in einer doppelt verketteten Liste

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einer doppelt verketteten Liste ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

AddAfter

Fügt einer doppelt verketteten Liste ein Element nach dem angegebenen Knoten hinzu.

Eine Version zum Hinzufügen eines Elements nach dem Wert.

```
CLinkedListNode<T>* AddAfter(
   CLinkedListNode<T>*  node,         // Knoten, nach welchem ein Element hinzugefügt w
   T                    value         // Element, das hinzugefügt wird
   );
```

Rückgabewert

Gibt einen Pointer zum hinzugefügten Knoten zurück.

Eine Version für das Hinzufügen eines Elements als einen nach dem Wert gebildeten Knoten.

```
bool AddAfter(
   CLinkedListNode<T>*  node,         // Knoten, nach welchen ein Element hinzugefügt w
   CLinkedListNode<T>*  new_node      // Knoten
   );
```

Parameter

node

[in] Knoten einer doppelt verketteten Liste, nach welchem ein neues Element hinzugefügt wird.

value

[in] Element, das hinzugefügt wird.

new_node

[in] Knoten zum Hinzufügen

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

AddBefore

Fügt einer doppelt verketteten Liste ein Element vor dem angegebenen Knoten hinzu.

Eine Version zum Hinzufügen eines Elements nach dem Wert.

```
CLinkedListNode<T>* AddBefore(
   CLinkedListNode<T>*  node,         // Knoten, vor welchem ein Element hinzugefügt wi
   T                    value         // das Element, das hinzugefügt wird
   );
```

Rückgabewert

Gibt einen Pointer zum hinzugefügten Knoten zurück.

Eine Version für das Hinzufügen eines Elements als einen nach dem Wert gebildeten Knoten.

```
bool AddBefore(
   CLinkedListNode<T>*  node,         // Knoten, vor welchem ein Element hinzugefügt wi
   CLinkedListNode<T>*  new_node      // der Knoten
   );
```

Parameter

node

[in] Der Knoten der doppelt verketteten Liste, bis welchem das neue Element hinzugefügt wird.

value

[in] Element, das hinzugefügt wird.

new_node

[in] Knoten zum Hinzufügen

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

AddFirst

Fügt ein Element am Anfang einer doppelt verketteten Liste hinzu.

Eine Version zum Hinzufügen eines Elements nach dem Wert.

```
CLinkedListNode<T>* AddFirst(
   T    value                     // Element zum Hinzufügen
   );
```

Rückgabewert

Gibt einen Pointer zum hinzugefügten Knoten zurück.

Eine Version für das Hinzufügen eines Elements als einen nach dem Wert gebildeten Knoten.

```
bool AddFirst(
   CLinkedListNode<T>*  node      // der Knoten
   );
```

Parameter

value

[in] Element, das hinzugefügt wird.

**node*

[in] Knoten zum Hinzufügen

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

AddLast

Fügt ein Element am Ende einer doppelt verketteten Liste hinzu.

Eine Version zum Hinzufügen eines Elements nach dem Wert.

```
CLinkedListNode<T>* AddLast(
   T  value                        // Element zum Hinzufügen
   );
```

Rückgabewert

Gibt einen Pointer zum hinzugefügten Knoten zurück.

Eine Version für das Hinzufügen eines Elements als einen nach dem Wert gebildeten Knoten

```
bool AddLast(
   CLinkedListNode<T>*  node      // Knoten, der hinzugefügt wird
   );
```

Parameter

value

[in] Element, das hinzugefügt wird.

**node*

[in] Knoten zum Hinzufügen

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einer doppelt verketteten Liste zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Head

Gibt einen Pointer zum ersten Knoten einer doppelt verketteten Liste zurück.

```
CLinkedListNode<T>* Head();
```

Rückgabewert

Gibt einen Pointer zum ersten Knoten zurück.

© 2000-2025, MetaQuotes Ltd.

First

Gibt einen Pointer zum ersten Knoten einer doppelt verketteten Liste zurück.

```
CLinkedListNode<T>* First();
```

Rückgabewert

Gibt einen Pointer zum ersten Knoten zurück.

© 2000-2025, MetaQuotes Ltd.

Last

Gibt einen Pointer zum letzten Knoten einer doppelt verketteten Liste zurück.

```
CLinkedListNode<T>* Last();
```

Rückgabewert

Gibt einen Pointer zum letzten Knoten zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob die doppelt verkettete Liste ein Element mit dem angegebenen Wert.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn es in der doppelt verketteten Liste ein Element mit dem angegeben Wert gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente einer doppelt verketteten Liste in den angegebenen Array beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],      // Array
   const int   dst_start=0       // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in welchen die Elemente der doppelt verketteten Liste geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer Sammlung.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten des angegebenen Elements aus einer doppelt verketteten Liste.

Eine Version für das Löschen eines Elements nach dem Wert.

```
bool Remove(
   T  item                        // Wert des Elements
   );
```

Eine Version für das Löschen eines Elements nach einem Pointer zum Knoten.

```
bool Remove(
   CLinkedListNode<T>*  node      // Knoten des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

**node*

[in] Knoten des Elements, das gelöscht werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

RemoveFirst

Löscht das erste Element aus einer doppelt verketteten Liste.

```
bool RemoveFirst();
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

RemoveLast

Löscht das letzte Element aus einer doppelt verketteten Liste.

```
bool RemoveLast();
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Find

Sucht nach dem ersten Auftreten des angegebenen Wertes in eine doppelt verkettete Liste.

```
CLinkedListNode<T>* Find(
   T  value     // der gesuchte Wert
   );
```

Parameter

value

[in] Gesuchter Wert.

Rückgabewert

Gibt einen Pointer zum ersten Knoten zurück, der den gesuchten Wert beinhaltet, wenn erfolgreich zurück, andernfalls NULL.

© 2000-2025, MetaQuotes Ltd.

FindLast

Sucht nach dem letzten Auftreten des angegebenen Wertes in einer doppelt verketteten Liste.

```
CLinkedListNode<T>* FindLast(
   T  value     // der gesuchte Wert
   );
```

Parameter

value

[in] Gesuchter Wert.

Rückgabewert

Gibt den Zeige auf den letzten Knoten zurück, der den gesuchten Wert beinhaltet, wenn erfolgreich zurück, andernfalls NULL.

© 2000-2025, MetaQuotes Ltd.

CQueue<T>

Die Klasse CQueue<T> ist eine Template-Klasse, die die Schnittstelle ICollection<T> implementiert.

Beschreibung

Die Klasse CQueue<T> ist eine dynamische Sammlung der Daten vom Typ T, realisiert als eine Warteschlange basierend auf dem Prinzip "First In — First Out" (FIFO).

Deklaration

```
template<typename T>
class CQueue : public ICollection<T>
```

Überschrift

```
#include <Generic\Queue.mqh>
```

Vererbungshierarchie

ICollection

CQueue

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einer Warteschlange ein Element hinzu
Enqueue	Fügt einer Warteschlange ein Element hinzu
Count	Gibt die Anzahl der Elemente in einer Warteschlange zurück
Contains	Bestimmt, ob die Warteschlange das Element mit dem angegebenen Wert beinhaltet
TrimExcess	Reduziert die Kapazität einer Warteschlange auf die tatsächliche Anzahl der Elemente und setzt dadurch den ungenutzten Speicherplatz frei
CopyTo	Kopiert alle Elemente einer Warteschlange in den angegebenen Array, beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente einer Warteschlange
Remove	Entfernt das erste Auftreten des angegebenen Elements aus der Warteschlange
Dequeue	Gibt das Anfangselement zurück und löscht es aus der Warteschlange
Peek	Gibt das Anfangselement zurück, ohne es aus der Warteschlange zu löschen

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einer Warteschlange ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Enqueue

Fügt einer Warteschlange ein Element hinzu.

```
bool Enqueue(
   T  value     // Element, das hinzugefügt wird
   );
```

Parameter

value

[in] Element, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einer Warteschlange zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Bestimmt, ob eine Warteschlange das Element mit dem angegebenen Wert beinhaltet.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn es ein Element mit dem angegebenen Wert in der Warteschlange gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrimExcess

Reduziert die Kapazität einer Warteschlange auf die tatsächliche Anzahl der Elemente und setzt dadurch den ungenutzten Speicherplatz frei.

```
void TrimExcess();
```

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente einer Warteschlange in den angegebenen Array, beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],      // Array
   const int   dst_start=0       // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in welchen die Elemente der Warteschlange geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer Warteschlange.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten des angegebenen Elements aus einer Warteschlange.

```
bool Remove(
   T  item     // Wert des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Dequeue

Gibt das Anfangselement zurück und löscht es aus der Warteschlange.

```
T Dequeue();
```

Rückgabewert

Gibt das Anfangselement zurück.

© 2000-2025, MetaQuotes Ltd.

Peek

Gibt das Anfangselement zurück, ohne es aus der Warteschlange zu löschen.

```
T Peek();
```

Rückgabewert

Gibt das Anfangselement zurück.

© 2000-2025, MetaQuotes Ltd.

CRedBlackTree<T>

Die Klasse CRedBlackTree<T> ist eine Template-Klasse, die die Schnittstelle ICollection<T> implementiert.

Beschreibung

Die Klasse CRedBlackTree<T> implementiert einen dynamischen Rot-Schwarz-Baum, in dessen Knoten die Daten vom Typ T gespeichert werden. Die Klasse bietet die grundlegenden Methoden für das Arbeiten mit Rot-Schwarz-Bäumen: Hinzufügen, Löschen, Suchen nach dem Höchst- bzw. Mindestwert usw.

Deklaration

```
template<typename T>
class CRedBlackTree : public ICollection<T>
```

Überschrift

```
#include <Generic\RedBlackTree.mqh>
```

Vererbungshierarchie

ICollection

CRedBlackTree

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einem Rot-Schwarz-Baum ein Element hinzu
Root	Gibt einen Pointer zur Wurzel eines Rot-Schwarz-Baumes zurück
Count	Gibt die Anzahl der Elemente in einem Rot-Schwarz-Baum zurück
Contains	Stellt fest, ob der Rot-Schwarz-Baum das Element mit dem angegebenen Wert beinhaltet
Comparer	Gibt einen Pointer zur Schnittstelle IComparer<T> zurück, die für das Organisieren eines Rot-Schwarz-Baumes verwendet wird
TryGetMin	Erhält das kleinste Element eines Rot-Schwarz-Baumes
TryGetMax	Erhält das höchste Element eines Rot-Schwarz-Baumes
CopyTo	Kopiert alle Elemente eines Rot-Schwarz-Baumes in den angegebenen Array beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente eines Rot-Schwarz-Baumes
Remove	Entfernt das Auftreten des angegebenen Elements aus einemRot-Schwarz-Baum

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
RemoveMin	Löscht das Element mit dem kleinsten Wert aus einem Rot-Schwarz-Baum
RemoveMax	Löscht das Element mit dem größten Wert aus einem Rot-Schwarz-Baum
Find	Sucht nach dem Auftreten des angegebenen Wertes in einem Rot-Schwarz-Baum
FindMax	Sucht nach dem Element mit dem höchsten Wert in einem Rot-Schwarz-Baum
FindMin	Sucht das Element mit dem kleinsten Wert in einem Rot-Schwarz-Baum

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einem Rot-Schwarz-Baum ein Element hinzu.

```
bool Add(
   T  value     // Element, das hinzugefügt wird
   );
```

Parameter

value

[in] Element, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einem Rot-Schwarz-Baum zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Root

Gibt einen Pointer zur Wurzel eines Rot-Schwarz-Baumes zurück.

```
CRedBlackTreeNode<T>* Root();
```

Rückgabewert

Gibt einen Pointer zur Wurzel zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob der Rot-Schwarz-Baum das Element mit dem angegebenen Wert beinhaltet.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn es ein Element mit dem angegebenen Wert im Rot-Schwarz-Baum gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Comparer

Gibt einen Pointer zur Schnittstelle IComparer<T> zurück, die für die Organisation eines Rot-Schwarz-Baumes verwendet wird.

```
IComparer<T>* Comparer() const;
```

Rückgabewert

Gibt einen Pointer zur Schnittstelle IComparer<T> zurück.

© 2000-2025, MetaQuotes Ltd.

TryGetMin

Erhält das kleinste Element aus einem Rot-Schwarz-Baum.

```
bool TryGetMin(
   T&  max     // Variable, in welche der Wert geschrieben wird
   );
```

Parameter

&min

[out] Variable, in welche der kleinste Wert geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TryGetMax

Erhält das höchste Element aus einem Rot-Schwarz-Baum.

```
bool TryGetMax(
   T&  max     // Variable, in welche der Wert geschrieben wird
   );
```

Parameter

&max

[out] Variable, in welche der höchste Wert geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente eines Rot-Schwarz-Baumes in den angegebenen Array beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],     // Array, in welchen die Elemente geschrieben werden
   const int   dst_start=0      // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in welchen die Elemente des Rot-Schwarz-Baumes geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente eines Rot-Schwarz-Baumes.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das Auftreten des angegebenen Elements aus einemRot-Schwarz-Baum.

Version für das Löschen eines Elements nach dem angegebenen Wert.

```
bool Remove(
   T  value                         // Wert des Elements
   );
```

Eine Version für das Löschen eines Elements nach einem Pointer zum Knoten.

```
bool Remove(
   CRedBlackTreeNode<T>*  node      // Knoten des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

**node*

[in] Knoten des Elements, das gelöscht werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

RemoveMin

Löscht das Element mit dem kleinsten Wert aus einem Rot-Schwarz-Baum.

```
bool RemoveMin();
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

RemoveMax

Löscht das Element mit dem größten Wert aus einem Rot-Schwarz-Baum.

```
bool RemoveMax();
```

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Find

Sucht nach dem Auftreten des angegebenen Wertes in einem Rot-Schwarz-Baum.

```
CRedBlackTreeNode<T>* Find(
   T  value     // gesuchter Wert
   );
```

Parameter

value

[in] Gesuchter Wert.

Rückgabewert

Gibt einen Pointer zum Knoten, der den gesuchten Wert enthält, zurück wenn erfolgreich, andernfalls NULL.

© 2000-2025, MetaQuotes Ltd.

FindMin

Sucht das Element mit dem kleinsten Wert im Rot-Schwarz-Baum.

```
CRedBlackTreeNode<T>* FindMin();
```

Rückgabewert

Gibt einen Pointer zum Knoten, der den kleinsten Wert beinhaltet, zurück wenn erfolgreich, andernfalls NULL.

© 2000-2025, MetaQuotes Ltd.

FindMax

Sucht das Element mit dem höchsten Wert im Rot-Schwarz-Baum.

```
CRedBlackTreeNode<T>* FindMax();
```

Rückgabewert

Gibt einen Pointer zum Knoten zurück, der den höchsten Wert beinhaltet, zurück wenn erfolgreich, andernfalls NULL.

© 2000-2025, MetaQuotes Ltd.

CSortedMap<TKey,TValue>

Die Klasse CSortedMap<TKey,TValue> ist eine Template-Klasse, die die Schnittstelle IMap<TKey,TValue> implementiert.

Beschreibung

Die Klasse CSortedMap<TKey,TValue> ist eine Implementierung einer dynamischen Hashtabelle, deren Daten in Form von nach Schlüssel und unter Einhaltung der Bedingung der Einmaligkeit des Schlüssels sortierten Schlüssel-Wert-Paaren gespeichert werden. Diese Klasse stellt die grundlegenden Methoden für die Arbeit mit der Hashtabelle bereit: Zugriff auf den Wert nach Schlüssel, Suche und Entfernen eines Schlüssel-Wert-Paares und andere.

Deklaration

```
template<typename TKey, typename TValue>
class CSortedMap : public IMap<TKey, TValue>
```

Überschrift

```
#include <Generic\SortedMap.mqh>
```

Vererbungshierarchie

ICollection
  IMap
    CSortedMap

Methoden der Klasse

Methode	Beschreibung
Add	Fügt der Hashtabelle ein Schlüssel-Wert-Paar hinzu
Count	Gibt die Anzahl der Elemente in der sortierten Hashtabelle zurück
Contains	Stellt fest, ob die sortierte Hashtabelle das angegebene Schlüssel-Wert-Paar enthält
ContainsKey	Stellt fest, ob die sortierte Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel enthält
ContainsValue	Stellt fest, ob die sortierte Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Wert beinhaltet
Comparer	Gibt einen Pointer zur Schnittstelle IComparer<T> zurück, der für die Gestaltung einer sortierten Hashtabelle verwendet wird
CopyTo	Kopiert alle Schlüssel-Wert-Paare aus der sortierten Hashtabelle in die angegebenen Arrays beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente der sortierten Hashtabelle

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
Remove	Entfernt das erste Auftreten des Schlüssel-Wert-Paares der sortierten Hashtabelle
TryGetValue	Erhält ein Element der sortierten Hashtabelle nach dem angegebenen Schlüssel
TrySetValue	Ändert den Wert eines Schlüssel-Wert-Paares der sortierten Hashtabelle nach dem angegebenen Schlüssel

© 2000-2025, MetaQuotes Ltd.

Add

Fügt der Hashtabelle ein Schlüssel-Wert-Paar hinzu.

Eine Version für das Hinzufügen eines Schlüssel-Wert-Paares.

```
bool Add(
   CKeyValuePair<TKeyTValue>*  pair     // Schlüssel-Wert-Paar
   );
```

Eine Version für das Hinzufügen eines neuen Schlüssel-Wert-Paares mit dem angegebenen Schlüssel und Wert.

```
bool Add(
   TKey    key,                         // Schlüssel
   TValue  value                        // Wert
   );
```

Parameter

**pair*

[in] Schlüssel-Wert-Paar.

key

[in] Schlüssel.

value

[in] Wert.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in der sortierten Hashtabelle zurück.

```
int Count();
```

© 2000-2025, MetaQuotes Ltd.

Comparer

Gibt einen Pointer zur Schnittstelle IComparer<T> zurück, die für die Gestaltung einer sortierten Hashtabelle verwendet wird.

```
IComparer<TKey>* Comparer() const;
```

Rückgabewert

Gibt einen Pointer zur Schnittstelle IComparer<T> zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob die sortierte Hashtabelle das angegebene Schlüssel-Wert-Paar enthält

Eine Version für das Arbeiten mit einem gebildeten Schlüssel-Wert-Paar.

```
bool Contains(
   CKeyValuePair<TKeyTValue>*  item     // Schlüssel-Wert-Paar
   );
```

Eine Version für das Arbeiten mit einem Schlüssel-Wert-Paar in Form eines separat angegebenen Schlüssels und Wertes.

```
bool Contains(
   TKey    key,                         // Schlüssel
   TValue  value                        // Wert
   );
```

Parameter

**item*

[in] Schlüssel-Wert-Paar.

key

[in] Schlüssel.

value

[in] Wert.

Rückgabewert

Gibt true zurück, wenn es in der sortierten Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel und Wert gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

ContainsKey

Stellt fest, ob die sortierte Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel enthält.

```
bool ContainsKey(
   TKey  key     // Schlüssel
   );
```

Parameter

key

[in] Schlüssel.

Rückgabewert

Gibt true zurück, wenn es in der sortierten Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Schlüssel gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

ContainsValue

Stellt fest, ob die sortierte Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Wert enthält.

```
bool ContainsValue(
   TValue  value     // Wert
   );
```

Parameter

value

[in] Wert.

Rückgabewert

Gibt true zurück, wenn es in der sortierten Hashtabelle ein Schlüssel-Wert-Paar mit dem angegebenen Wert gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Schlüssel-Wert-Paare aus einer sortierten Hashtabelle in die angegebenen Arrays beginnend mit einem bestimmten Index.

Version für das Kopieren der Hashtabelle in den Array von Schlüssel-Wert-Paaren.

```
int CopyTo(
   CKeyValuePair<TKeyTValue>*&  dst_array[],     // Array, in welchen Schlüssel-Wert-P
   const int                    dst_start=0      // Anfangsindex
   );
```

Eine Version für das Kopieren einer Hashtabelle in separate Arrays für Schlüssel und Werte.

```
int CopyTo(
   TKey&       dst_keys[],                       // Array für Schlüssel
   TValue&     dst_values[],                     // Array für Werte
   const int   dst_start=0                       // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in den alle Paare aus der Hashtabelle geschrieben werden.

&dst_keys[]

[out] Array, in den alle Schlüssel aus der Hashtabelle geschrieben werden.

&dst_values[]

[out] Array, in den alle Werte der Hashtabelle geschrieben werden.

dst_start=0

[in] Index in den Arrays, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Schlüssel-Wert-Paare zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer sortierten Hashtabelle.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das erste Auftreten des Schlüssel-Wert-Paares der sortierten Hashtabelle.

Eine Version für das Entfernen des Schlüssel-Wert-Paares nach dem gebildeten Schlüssel-Wert-Paar.

```
bool Remove(
   CKeyValuePair<TKeyTValue>*  item     // Schlüssel-Wert-Paar
   );
```

Eine Version für das Entfernen des Schlüssel-Wert-Paares nach dem Schlüssel.

```
bool Remove(
   TKey  key                           // Schlüssel
   );
```

Parameter

**item*

[in] Schlüssel-Wert-Paar.

key

[in] Schlüssel.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TryGetValue

Erhält ein Element aus der sortierten Hashtabelle nach dem angegebenen Schlüssel.

```
bool TryGetValue(
   TKey     key,       // Schlüssel
   TValue&  value      // Variable, in welche der Wert geschrieben wird
   );
```

Parameter

key

[in] Schlüssel.

&value

[out] Variable, in welche der angegebene Wert des Schlüssel-Wert-Paares geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrySetValue

Ändert den Wert des Schlüssel-Wert-Paares der sortierten Hashtabelle nach dem angegebenen Schlüssel.

```
bool TrySetValue(
   TKey    key,       // Schlüssel
   TValue  value      // neuer Wert
   );
```

Parameter

key

[in] Schlüssel.

value

[in] Neuer Wert, der dem angegebenen Schlüssel-Wert-Paar zugewiesen werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CSortedSet<T>

Die Klasse CSortedSet<T> ist eine Template-Klasse, die die Schnittstelle ISet<T> implementiert.

Beschreibung

Die Klasse CSortedSet<T> ist eine Implementierung einer sortierten dynamischen Menge von Daten des Typs T unter Einhaltung der Bedingung der Einmaligkeit jedes Wertes. Diese Klasse stellt die grundlegenden Methoden für die Arbeit und Operationen mit Datenmengen bereit: Vereinigung, Schnitt, Definieren strikter Teilmengen und andere.

Deklaration

```
template<typename T>
class CSortedSet : public ISet<T>
```

Überschrift

```
#include <Generic\SortedSet.mqh>
```

Vererbungshierarchie

ICollection
  ISet
    CSortedSet

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einer sortierten Menge ein Element hinzu
Count	Gibt die Anzahl der Elemente in einer sortierten Menge zurück
Contains	Stellt fest, ob die sortierte Menge das Element mit dem angegebenen Wert enthält.
Comparer	Gibt einen Pointer zur Schnittstelle IComparer<T> zurück, die für das Organisieren einer sortierten Menge verwendet wird
TryGetMin	Erhält das kleinste Element aus der sortierten Menge
TryGetMax	Erhält das höchste Element aus einer sortierten Menge
CopyTo	Kopiert alle Elemente der sortierten Menge in den angegebenen Array, beginnend mit einen bestimmten Index
Clear	Löscht alle Elemente einer sortierten Menge
Remove	Entfernt das Auftreten des angegebenen Elements aus einer sortierten Menge
ExceptWith	Führt die Operation des Komplements der aktuellen und der übergebenen Sammlung (Arrays) aus

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
IntersectWith	Führt die Operation des Schnitts der aktuellen und der übergebenen Sammlung (Arrays) aus
SymmetricExceptWith	Führt die Operation der symmetrischen Differenz der aktuellen und der übergebenen Sammlung (Arrays) aus
UnionWith	Führt die Operation der Vereinigung der aktuellen und der übergebenen Sammlung (Arrays) aus
IsProperSubsetOf	Stellt fest, ob die aktuelle sortierte Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist
IsProperSupersetOf	Stellt fest, ob die aktuelle sortierte Menge eine strikte Obermenge der angegebenen Sammlung oder Arrays ist
IsSubsetOf	Stellt fest, ob die aktuelle sortierte Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist
IsSupersetOf	Stellt fest, ob die aktuelle sortierte Menge eine Obermenge der angegebenen Sammlung oder Arrays ist
Overlaps	Stellt fest, ob sich die aktuelle sortierte Menge und die angegebene Sammlung oder Array überlappen.
SetEquals	Stellt fest, ob die aktuelle sortierte Menge alle Elemente der angegebenen Sammlung oder Arrays enthält
GetViewBetween	Erhält eine Untermenge von der aktuellen sortierten Menge, die durch den minimalen und maximalen Wert gesetzt wurde
GetReverse	Erhält eine Kopie der aktuellen sortierten Menge, in welcher sich alle Elemente in der umgekehrten Reihenfolge befinden

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einer sortierten Menge ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einer sortierten Menge zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob die Menge ein Element mit dem angegebenen Wert enthält.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn das Element mit dem angegebenen Wert in der Menge vorhanden ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Comparer

Gibt einen Pointer zur Schnittstelle IComparer<T> zurück, die für das Organisieren einer sortierten Menge verwendet wird.

```
IComparer<T>* Comparer() const;
```

Rückgabewert

Gibt einen Pointer zur Schnittstelle IComparer<T> zurück.

© 2000-2025, MetaQuotes Ltd.

TryGetMin

Erhält das kleinste Element aus einer sortierten Menge.

```
bool TryGetMin(
   T&  max     // Variable, in welche der Wert geschrieben wird
   );
```

Parameter

&min

[out] Variable, in welche der kleinste Wert geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TryGetMax

Erhält das höchste Element aus einer sortierten Menge.

```
bool TryGetMax(
   T&  max     // Variable, in welche der Wert geschrieben wird
   );
```

Parameter

&max

[out] Variable, in welche der höchste Wert geschrieben wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente einer sortierten Menge in den angegebenen Array beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],      // Array
   const int   dst_start=0       // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in den die Elemente einer Menge geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente einer sortierten Menge.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Entfernt das Auftreten des angegebenen Elements aus einer sortierten Menge.

```
bool Remove(
   T  item     // Wert des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

ExceptWith

Führt die Operation des Komplements der aktuellen und der übergebenen Sammlung (Arrays) aus. D.h. entfernt aus der aktuellen Sammlung (Arrays) alle Elemente, die in der angegebenen Sammlung (Array) auch vorhanden sind.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void ExceptWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void ExceptWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, die von der aktuellen sortierten Menge ausgenommen wird.

&collection[]

[in] Array, der von der aktuellen sortierten Menge ausgenommen wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

IntersectWith

Führt die Operation des Schnitts der aktuellen und der übergebenen Sammlung (Arrays) aus D.h. in der aktuellen Sammlung bleiben nur die Elemente, die auch in der angegebenen Sammlung (Array) vorhanden sind.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void IntersectWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void IntersectWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, mit welcher das Produkt gebildet wird.

&collection[]

[in] Array, mit welchem das Produkt gebildet wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

SymmetricExceptWith

Führt die Operation der symmetrischen Differenz der aktuellen und der übergebenen Sammlung (Arrays) aus. D.H. die aktuelle Sammlung (Array) wird nur die Elemente enthalten, die entweder im Quellobjekt oder im übergebenen Objekt vorhanden waren, aber nicht gleichzeitig in beiden.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void SymmetricExceptWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void SymmetricExceptWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung für die symmetrische Differenz.

&collection[]

[in] Array für die symmetrische Differenz.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

UnionWith

Führt die Operation der Vereinigung der aktuellen und der übergebenen Sammlung (Arrays) aus. D.h. fügt der aktuellen Sammlung (Array) fehlende Elemente aus der angegebenen Sammlung (Array) hinzu.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
void UnionWith(
   ICollection<T>*  collection     // Sammlung
   );
```

Eine Version für die Arbeit mit einem Array.

```
void UnionWith(
   T&  array[]                     // Array
   );
```

Parameter

**collection*

[in] Sammlung, mit welcher die aktuelle Menge vereinigt wird.

&collection[]

[in] Array, mit welchem die aktuelle Menge vereinigt wird.

Hinweis

Das Ergebnis wird in die aktuelle Sammlung (Array) geschrieben.

© 2000-2025, MetaQuotes Ltd.

IsProperSubsetOf

Stellt fest, ob die aktuelle Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsProperSubsetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsProperSubsetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle sortierte Menge eine strikte Untermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsProperSupersetOf

Stellt fest, ob die aktuelle sortierte Menge eine strikte Obermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsProperSupersetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsProperSupersetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle sortierte Menge eine strikte Obermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsSubsetOf

Stellt fest, ob die aktuelle sortierte Menge eine strikte Untermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsSubsetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsSubsetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle sortierte Menge eine Untermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsSupersetOf

Stellt fest, ob die aktuelle sortierte Menge eine Obermenge der angegebenen Sammlung oder Arrays ist.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool IsSupersetOf(
   ICollection<T>*  collection     // Sammlung für das Feststellen des Verhältnisses
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool IsSupersetOf(
   T&  array[]                     // Array für das Feststellen des Verhältnisses
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen des Verhältnisses.

&collection[]

[in] Array für das Feststellen des Verhältnisses.

Rückgabewert

Gibt true zurück, wenn die aktuelle sortierte Menge eine Obermenge ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Overlaps

Stellt fest, ob sich die aktuelle sortierte Menge und die angegebenen Sammlung oder Array überlappen.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool Overlaps(
   ICollection<T>*  collection     // Sammlung für den Vergleich
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool Overlaps(
   T&  array[]                     // Array für den Vergleich
   );
```

Parameter

**collection*

[in] Sammlung für das Feststellen der Überlappung.

&collection[]

[in] Array für das Feststellen der Überlappung.

Rückgabewert

Gibt true zurück, wenn sich die aktuelle sortierte Menge und die Sammlung oder der Array überlappen, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

SetEquals

Stellt fest, ob die aktuelle sortierte Menge alle Elemente der angegebenen Sammlung oder Arrays enthält.

Eine Version für die Arbeit mit der Sammlung, die die Schnittstelle ICollection<T> implementiert.

```
bool SetEquals(
   ICollection<T>*  collection     // Sammlung für den Vergleich
   );
```

Eine Version für die Arbeit mit einem Array.

```
bool SetEquals(
   T&  array[]                     // Array für den Vergleich
   );
```

Parameter

**collection*

[in] Sammlungen für den Vergleich von Elementen.

&collection[]

[in] Array für den Vergleich von Elementen.

Rückgabewert

Gibt true zurück, wenn es alle Elemente der angegebenen Sammlung oder des Arrays in der aktuellen sortierten Menge gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

GetViewBetween

Erhält eine Untermenge von der aktuellen sortierten Menge, die durch den minimalen und maximalen Wert gesetzt wurde.

```
bool GetViewBetween(
   T&  array[],          // Array
   T   lower_value,      // der kleinste Wert
   T   upper_value       // der größte Wert
   );
```

Parameter

&array[]

[out] Array, in welchen die Untermenge geschrieben wird.

lower_value

[in] Mindestwert des Bereichs.

upper_value

[in] Höchstwert des Bereichs.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

GetReverse

Erhält eine Kopie der aktuellen sortierten Menge, in welcher sich alle Elemente in einer umgekehrten Reihenfolge befinden.

```
bool GetReverse(
   T&  array[]     // Array
   );
```

Parameter

&array[]

[out] Array.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

CStack<T>

Die Klasse CStack<T> ist eine Template-Klasse, die die Schnittstelle ICollection<T> implementiert.

Beschreibung

Die Klasse CStack<T> ist eine dynamische Kollektion der Daten vom Typ T, realisiert als ein Stack basierend auf dem Prinzip "Last In — First Out" (LIFO).

Deklaration

```
template<typename T>
class CStack : public ICollection<T>
```

Überschrift

```
#include <Generic\Stack.mqh>
```

Vererbungshierarchie

ICollection

CStack

Methoden der Klasse

Methode	Beschreibung
Add	Fügt einem Stack ein Element hinzu
Count	Gibt die Anzahl der Elemente in einem Stack zurück
Contains	Legt fest, ob ein Stack ein Element mit dem angegebenen Wert beinhaltet
TrimExcess	Reduziert die Kapazität eines Stacks auf die tatsächliche Anzahl der Elemente
CopyTo	Kopiert alle Elemente eines Stacks in den angegebenen Array beginnend mit einem bestimmten Index
Clear	Löscht alle Elemente eines Stacks
Remove	Löscht das erste Auftreten des angegebenen Elements aus einem Stack
Push	Fügt einem Stack ein Element hinzu
Peek	Gibt das Anfangselement zurück, ohne es aus dem Stack zu löschen
Pop	Gibt das Anfangselement zurück und löscht es aus dem Stack

© 2000-2025, MetaQuotes Ltd.

Add

Fügt einem Stack ein Element hinzu.

```
bool Add(
   T  value     // Wert des Elements
   );
```

Parameter

value

[in] Wert des Elements, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Count

Gibt die Anzahl der Elemente in einem Stack zurück.

```
int Count();
```

Rückgabewert

Gibt die Anzahl der Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Stellt fest, ob ein Stack ein Element mit dem angegebenen Wert beinhaltet.

```
bool Contains(
   T  item     // der gesuchte Wert
   );
```

Parameter

item

[in] Gesuchter Wert.

Rückgabewert

Gibt true zurück, wenn es ein Element mit dem angegebenen Wert in einem Stack gibt, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

TrimExcess

Reduziert die Kapazität eines Stacks auf die tatsächliche Anzahl der Elemente und setzt dadurch den ungenutzten Speicherplatz frei.

```
void TrimExcess();
```

© 2000-2025, MetaQuotes Ltd.

CopyTo

Kopiert alle Elemente eines Stacks in den angegebenen Array beginnend mit einem bestimmten Index.

```
int CopyTo(
   T&          dst_array[],     // Array, in welchen die Elemente geschrieben werden
   const int   dst_start=0      // Anfangsindex
   );
```

Parameter

&dst_array[]

[out] Array, in welchen alle Elemente des Stacks geschrieben werden.

dst_start=0

[in] Index im Array, mit welchem das Kopieren beginnt.

Rückgabewert

Gibt die Anzahl der kopierten Elemente zurück.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Elemente eines Stacks.

```
void Clear();
```

© 2000-2025, MetaQuotes Ltd.

Remove

Löscht das erste Auftreten des angegebenen Elements aus dem Stack.

```
bool Remove(
   T  item     // Wert des Elements
   );
```

Parameter

item

[in] Wert des Elements, der entfernt werden muss.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Push

Fügt einem Stack ein Element hinzu.

```
bool Push(
   T  value     // Element, das hinzugefügt wird
   );
```

Parameter

value

[in] Element, das hinzugefügt wird.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Peek

Gibt das Kopfelement eines Stacks zurück, ohne es aus dem Stack zu löschen.

```
T Peek();
```

Rückgabewert

Gibt das Kopfelement zurück.

© 2000-2025, MetaQuotes Ltd.

Pop

Gibt das Kopfelement zurück und löscht es aus dem Stack.

```
T Pop();
```

Rückgabewert

Gibt das Kopfelement zurück.

© 2000-2025, MetaQuotes Ltd.

ArrayBinarySearch

Sucht den angegebenen Wert in einem aufsteigend sortierten eindimensionalen Array unter Verwendung der Schnittstelle IComparable<T> für den Vergleich von Elementen.

```
template<typename T>
int ArrayBinarySearch(
   T&              array[],          // Array, in dem gesucht wird
   const int       start_index,      // Anfangsindex
   const int       count,            // Suchbereich
   T               value,            // der gesuchte Wert
   IComparer<T>*   comparer          // Schnittstelle für den Vergleich
   );
```

Parameter

&array[]

[out] Array, in dem gesucht wird.

value

[in] Gesuchter Wert.

**comparer*

[in] Schnittstelle für den Vergleich der Elemente.

start_index

[in] Anfangsindex, mit welchem die Suche beginnt.

count

[in] Länge des Suchbereichs.

Rückgabewert

Gibt den Index des ermittelten Elements zurück. Wenn der Wert nicht ermittelt ist, gibt den Index des kleineren Elements zurück, das dem gesuchten Wert am nächsten kommt.

© 2000-2025, MetaQuotes Ltd.

ArrayIndexOf

Sucht nach dem ersten Auftreten eines Wertes in einem eindimensionalem Array.

```
template<typename T>
int ArrayIndexOf(
   T&          array[],          // Array, in dem gesucht wird
   T           value,            // der gesuchte Wert
   const int   start_index,      // Anfangsindex
   const int   count             // Suchbereich
   );
```

Parameter

&array[]

[out] Array, in dem gesucht wird.

value

[in] Gesuchter Wert.

start_index

[in] Anfangsindex, mit welchem die Suche beginnt.

count

[in] Länge des Suchbereichs.

Rückgabewert

Gibt den Index des ersten gefundenen Elements zurück. Wenn kein Wert gefunden wurde, gibt -1 zurück.

© 2000-2025, MetaQuotes Ltd.

ArrayLastIndexOf

Sucht nach dem letzten Auftreten eines Wertes in einem eindimensionalem Array.

```
template<typename T>
int ArrayLastIndexOf(
   T&             array[],          // Array, in dem gesucht wird
   T          value,            // der gesuchte Wert
   const int  start_index,      // Anfangswert
   const int  count             // Suchbereich
   );
```

Parameter

&array[]

[out] Array, in dem gesucht wird.

value

[in] Gesuchter Wert.

start_index

[in] Anfangsindex, mit welchem die Suche beginnt.

count

[in] Länge des Suchbereichs.

Rückgabewert

Gibt den Index des letzten gefundenen Elements zurück. Wenn kein Wert gefunden wurde, gibt -1 zurück.

© 2000-2025, MetaQuotes Ltd.

ArrayReverse

Ändert die Sequenz der Elemente in einem eindimensionalen Array.

```
template<typename T>
bool ArrayReverse(
   T&          array[],          // Ausgangsarray
   const int   start_index,      // Anfangsindex
   const int   count             // Anzahl der Elemente
   );
```

Parameter

&array[]

[out] Ausgangsarray.

start_index

[in] Anfangsindex.

count

[in] Anzahl der Elemente des Arrays, die in die Operation einbezogen sind.

Rückgabewert

Wenn erfolgreich true, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht zwei Werte und stellt fest, ob einer "größer als, kleiner als oder gleich" dem anderen ist.

Eine Version für den Vergleich von zwei Werten vom Typ bool.

```
int Compare(
   const bool  x,          // erster Wert
   const bool  y           // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ char.

```
int Compare(
   const char  x,          // erster Wert
   const char  y           // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ uchar.

```
int Compare(
   const uchar  x,         // erster Wert
   const uchar  y          // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ short.

```
int Compare(
   const short  x,         // erster Wert
   const short  y          // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ ushort.

```
int Compare(
   const ushort  x,        // erster Wert
   const ushort  y         // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ color.

```
int Compare(
   const color  x,         // erster Wert
   const color  y          // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ int.

```
int Compare(
   const int  x,           // erster Wert
   const int  y            // zweiter Wert
   );
```

© 2000-2025, MetaQuotes Ltd.

Eine Version für den Vergleich von zwei Werten vom Typ uint.

```
int Compare(
   const uint  x,          // erster Wert
   const uint  y           // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ datetime.

```
int Compare(
   const datetime  x,      // erster Wert
   const datetime  y       // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ long.

```
int Compare(
   const long  x,          // erster Wert
   const long  y           // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ ulong.

```
int Compare(
   const ulong  x,         // erster Wert
   const ulong  y          // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ float.

```
int Compare(
   const float  x,         // erster Wert
   const float  y          // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ double.

```
int Compare(
   const double  x,        // erster Wert
   const double  y         // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ string.

```
int Compare(
   const string  x,        // erster Wert
   const string  y         // zweiter Wert
   );
```

Eine Version für den Vergleich von zwei Werten vom Typ.

```
template<typename T>
int Compare(
```

© 2000-2025, MetaQuotes Ltd.

```
   T  x,                  // erster Wert
   T  y                   // zweiter Wert
   );
```

Parameter

x

[in] Erster Wert.

y

[in] Zweiter Wert

Rückgabewert

Gibt das Verhältnis zwischen den zwei zu vergleichenden Werten zurück:

- wenn das Ergebnis kleiner als Null ist, ist x kleiner als y (x<y)
- wenn das Ergebnis gleich Nul, ist x gleich y (x=y)
- wenn das Ergebnis größer als Null ist, ist x größer als y (x>y)

Hinweis

Wenn der T-Typ ein Objekt ist, das die Schnittstelle IComparable<T> implementiert, werden die Objekte basierend auf seiner Vergleichsmethode Compare miteinander verglichen. In allen anderen Fällen wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Equals

Stellt durch einen Vergleich fest, ob zwei Werte gleich sind.

```
template<typename T>
bool Equals(
   T  x,      // erster Wert
   T  y       // zweiter Wert
   );
```

Parameter

x

[in] Erster Wert.

y

[in] Zweiter Wert.

Rückgabewert

Gibt true zurück, wenn die Objekte gleich sind, andernfalls false.

Hinweis

Wenn der T-Typ ein Objekt ist, das die Schnittstelle IEqualityComparable<T> implementiert, werden die Objekte basierend auf seiner Vergleichsmethode Equals verglichen. In allen anderen Fällen wird der Standardvergleich hinsichtlich der Gleichheit verwendet.

© 2000-2025, MetaQuotes Ltd.

GetHashCode

Berechnet den Wert des Hashcodes.

Eine Version für das Arbeiten mit dem Typ bool.

```
int GetHashCode(
   const bool   value          // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ char.

```
int GetHashCode(
   const char   value          // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ uchar.

```
int GetHashCode(
   const uchar   value         // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ short.

```
int GetHashCode(
   const short   value         // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ ushort.

```
int GetHashCode(
   const ushort   value        // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ color.

```
int GetHashCode(
   const color   value         // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ int.

```
int GetHashCode(
   const int   value           // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ uint.

```
int GetHashCode(
   const uint   value          // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ datetime.

```
int GetHashCode(
```

© 2000-2025, MetaQuotes Ltd.

```
   const datetime   value      // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ long.

```
int GetHashCode(
   const long   value          // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ ulong.

```
int GetHashCode(
   const ulong   value         // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ float.

```
int GetHashCode(
   const float   value         // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ double.

```
int GetHashCode(
   const double   value        // Wert
   );
```

Eine Version für das Arbeiten mit dem Typ string.

```
int GetHashCode(
   const string   value        // Wert
   );
```

Eine Version für das Arbeiten mit den anderen Typen.

```
template<typename T>
int GetHashCode(
   T   value                   // Wert
   );
```

Parameter

value

[in] Wert, für welchen der Hashcode erhalten werden muss.

Rückgabewert

Gibt den Hashcode zurück.

Hinweis

Wenn der T-Typ ein Objekt ist, das die Schnittstelle IEqualityComparable<T> implementiert, wird der Hashcode basierend auf seiner Methode HashCode erhalten. In allen anderen Fällen wird der Hashcode als Hashwert vom Namen des Typs dieses Wertes berechnet.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Dateioperationen

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen von Dateioperationen und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Dateioperationen-Klassen sparen Sie Zeit bei der Entwicklung von Anwendungen die Datei-E/A-Vorgänge verwenden.

Die Standardbibliothek MQL5 (in Bezug auf Dateioperationen) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Files.

Klassenname	Klassenbeschreibung
CFile	Die Basisklasse von Dateioperationen
CFileBin	Die Klasse für Operationen mit Binär-Dateien
CFileTxt	Die Klasse für Operationen Textdateien

© 2000-2025, MetaQuotes Ltd.

Klasse CFile

CFile ist eine Basisklasse für die Klassen CFileBin und CFileTxt.

Beschreibung

Klasse CFile bietet ihren abgeleiteten Klassen den Zugriff auf die allgemeine Funktionen von API MQL5 für die Arbeit mit Dateien und Verzeichnissen.

Deklaration

```
class CFile: public CObject
```

Kopf

```
#include <Files\File.mqh>
```

Vererbungshierarchie

CObject

CFile

Direkte Ableitungen

CFileBin, CFilePipe, CFileTxt

Gruppen der Klassenmethode

Attribute	
Handle	Erhält Handle der Datei
Filename	Erhält den Dateinamen
Flags	Erhält die Dateiflags
SetUnicode	Setzt/löscht das Flag FILE_UNICODE
SetCommon	Setzt/löscht das Flag FILE_COMMON
Allgemeine Methoden für die Arbeit mit Dateien	
Open	Öffnet eine Datei
Close	Schließt eine Datei
Delete	Löscht eine Datei
IsExist	Überprüft die Existenz der Datei
Copy	Kopiert eine Datei
Move	Bewegt/neu benennt eine Datei
Size	Erhält die Dateigröße

© 2000-2025, MetaQuotes Ltd.

Attribute	
Tell	Erhält die aktuelle Position des Dateizeigers
Seek	Setzt die Position des Dateizeigers
Flush	Schreibt alle Daten in eine Datei weg
IsEnding	Definiert das Ende der Datei
IsLineEnding	Definiert das Ende der Zeile
Allgemeine Methoden für die Arbeit mit Verzeichnissen	
FolderCreate	Erstellt ein Verzeichnis
FolderDelete	Löscht ein Verzeichnis
FolderClean	Löscht alles aus dem Verzeichnis
Allgemeine Methoden für die Suche nach Dateien und Verzeichnissen	
FileFindFirst	Startet die Suche nach Dateien
FileFindNext	Weiter sucht nach Dateien
FileFindClose	Schließt den Such-Handle

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Handle

Erhält den Handle der geöffneten Datei.

```
int  Handle()
```

Rückgabewert

Handle der offenen Datei, die an die Klasseninstanz gebunden ist. Wenn es keine gebundene Datei gibt, wird -1 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

FileName

Erhält den Namen der geöffneten Datei.

```
string  FileName()
```

Rückgabewert

Name der offenen Datei, die an die Klasseninstanz gebunden ist. Wenn es keine gebundene Datei gibt, wird "" zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Flags

Erhält die Dateiöffnungsflags

```
int  Flags()
```

Rückgabewert

Flags der Öffnung der an eine Klasseninstanz gebundenen Datei.

© 2000-2025, MetaQuotes Ltd.

SetUnicode

Setzt/löscht das Flag FILE_UNICODE.

```
void  SetUnicode(
   bool  unicode      // Flagwert
   )
```

Parameter

unicode

[in] Der neue Wert des Flags FILE_UNICODE.

Hinweis

Arbeit mit Zeichenfolgen abhängt vom Zustand des Flags FILE_UNICODE. Wenn das Flag gelöscht wird, dann wird der die Codierung ANSI (Single-Byte-Zeichen) verwendet. Wenn das Flag gesetzt ist, dann wird der die Codierung UNICODE (Double-Byte-Zeichen) verwendet. Wenn die Datei bereits geöffnet ist, kann das Flag nicht geändert werden.

© 2000-2025, MetaQuotes Ltd.

SetCommon

Setzt/löscht das Flag FILE_COMMON.

```
void  SetCommon(
   bool  common      // Flagwert
   )
```

Parameter

common

[in] Der neue Wert des Flags FILE_COMMON.

Hinweis

Vom Zustand des Flags FILE_COMMON abhängt, welches Verzeichnis für die Arbeit verwendet wird. Wenn das Flag gelöscht wird, dann wird der lokale Ordner des Terminals als ein Arbeitsverzeichnis verwendet. Wenn das Flag gesetzt ist, dann wird der gemeinsame Ordner als ein Arbeitsverzeichnis verwendet. Wenn die Datei bereits geöffnet ist, kann das Flag nicht geändert werden.

© 2000-2025, MetaQuotes Ltd.

Open

Öffnet die angegebene Datei und, wenn erfolgreich geöffnet, bindet sie an eine Klasseninstanz.

```
int  Open(
   const string  file_name,        // Dateiname
   int           flags,            // Flags
   short         delimiter=9       // Trennzeichen
   )
```

Parameter

file_name

[in] Name der Datei, die Sie öffnen.

flags

[in] Dateiöffnungsflags

delimiter=9

[in] Trennzeichen für die CSV-Datei.

Rückgabewert

Handle der geöffneten Datei.

Hinweis

Der Arbeitsordner wird beim zuvor durch Methode SetCommon() gesetzten/gelöschten Flag definiert .

© 2000-2025, MetaQuotes Ltd.

Close

Schließt die auf die Klasseninstanz gebundene Datei.

```
void  Close()
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht die auf die Klasseninstanz gebundene Datei.

```
void  Delete()
```

Delete

Löscht die angegebene Datei.

```
void  Delete(
   const string  file_name      // Dateiname
   )
```

Parameter

file_name

[in] Name der Datei, die gelöscht werden soll.

Hinweis

Der Arbeitsordner wird beim zuvor durch Methode SetCommon() gesetzten/gelöschten Flag definiert .

© 2000-2025, MetaQuotes Ltd.

IsExist

Überprüft die Existenz der Datei.

```
bool  IsExist(
   const string  file_name      // Dateiname
   )
```

Parameter

file_name

[in] Name der geprüften Datei.

Rückgabewert

true, wenn die angegebene Datei existiert.

© 2000-2025, MetaQuotes Ltd.

Copy

Kopiert eine Datei.

```
bool  Copy(
   const string  src_name,      // Dateiname
   int           src_flag,      // Flag
   const string  dst_name,      // Dateiname
   int           dst_flags      // Flags
   )
```

Parameter

src_name

[in] Name der Quelldatei.

src_flag

[in] Flags der Quelldatei (nur FILE_COMMON ist verwendet).

dst_name

[in] Name der Empfänger-Datei.

dst_flags

[in] Flags der der Empfänger-Datei (nur FILE_REWRITE und FILE_COMMON sind verwendet).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn eine Datei nicht kopiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Move

Bewegt/neu benennt eine Datei.

```
bool  Move(
   const string  src_name,      // Dateiname
   int           src_flag,      // Flag
   const string  dst_name,      // Dateiname
   int           dst_flags      // Flags
   )
```

Parameter

src_name

[in] Name der Quelldatei.

src_flag

[in] Flags der Quelldatei (nur FILE_COMMON ist verwendet).

dst_name

[in] Name der Empfänger-Datei.

dst_flags

[in] Flags der der Empfänger-Datei (nur FILE_REWRITE und FILE_COMMON sind verwendet).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn eine Datei nicht bewegt/neu benannt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Size

Erhält die Dateigröße in Bytes.

```
ulong  Size()
```

Rückgabewert

Die Dateigröße in Bytes. Wenn es keine gebundene Datei gibt, wird ULONG_MAX zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Tell

Erhält die aktuelle Position des Dateizeigers.

```
ulong  Tell()
```

Rückgabewert

Die aktuelle Position des Dateizeigers. Wenn es keine gebundene Datei gibt, wird ULONG_MAX zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Seek

Setzt die Position des Dateizeigers.

```
void  Seek(
   long                offset,     // Offset
   ENUM_FILE_POSITION  origin      // Startpunkt
   )
```

Parameter

offset

[in] Der Offset in Bytes (kann auch ein negativer Wert sein).

origin

[in] Startpunkt für Offset.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Dateizeiger nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Flush

Schreibt alle Daten aus der Datei-Eingabe-Ausgabe-Puffer in die Datei

```
void  Flush()
```

© 2000-2025, MetaQuotes Ltd.

IsEnding

Definiert das Ende der Datei beim Lesen.

```
bool  IsEnding()
```

Rückgabewert

Gibt true zurück, wenn das Dateiende beim Lesen oder Bewegen des Dateizeigers erreicht wurde.

© 2000-2025, MetaQuotes Ltd.

IsLineEnding

Definiert das Ende der Zeile in einer Textdatei beim Lesen.

```
bool  IsLineEnding()
```

Rückgabewert

Gibt true zurück, wenn das Zeilenende beim Lesen der txt- oder csv-Datei erreicht wurde.

© 2000-2025, MetaQuotes Ltd.

FolderCreate

Erstellt ein neues Verzeichnis.

```
bool  FolderCreate(
   const string  folder_name      // Ordnername
   )
```

Parameter

folder_name

[in] Name des Verzeichnisses, das Sie erstellen möchten. Es enthält den Pfad zum Ordner (in Bezug auf das mit dem Flag FILE_COMMON angegebenen Arbeitsverzeichnis).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn das Verzeichnis nicht erstellt werden konnte.

Hinweis

Der Arbeitsordner wird beim zuvor durch Methode SetCommon() gesetzten/gelöschten Flag definiert .

© 2000-2025, MetaQuotes Ltd.

FolderDelete

Löscht das angegebene Verzeichnis.

```
bool  FolderDelete(
   const string  folder_name      // Ordnername
   )
```

Parameter

folder_name

[in] Name des Verzeichnisses, das Sie löschen möchten. Es enthält den Pfad zum Ordner (in Bezug auf das mit dem Flag FILE_COMMON angegebenen Arbeitsverzeichnis).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn das Verzeichnis nicht gelöscht werden konnte.

Hinweis

Der Arbeitsordner wird beim zuvor durch Methode SetCommon() gesetzten/gelöschten Flag definiert .

© 2000-2025, MetaQuotes Ltd.

FolderClean

Löscht alles aus dem angegebenen Verzeichnis.

```
bool  FolderClean(
   const string  folder_name      // Ordnername
   )
```

Parameter

folder_name

[in] Name des Verzeichnisses, in dem Sie alles löschen möchten. Es enthält den Pfad zum Ordner (in Bezug auf das mit dem Flag FILE_COMMON angegebenen Arbeitsverzeichnis).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Operation fehlgeschlagen ist.

Hinweis

Der Arbeitsordner wird beim zuvor durch Methode SetCommon() gesetzten/gelöschten Flag definiert .

© 2000-2025, MetaQuotes Ltd.

FileFindFirst

Startet die Suche nach Dateien nach dem angegebenen Filter

```
int  FileFindFirst(
   const string  filter,        // Suchfilter
   string&       file_name      // Referenz
   )
```

Parameter

filter

[in] Suchfilter.

file_name

[out] Eine Referenz auf die Zeichenfolge, in der der Name der ersten gefundenen Datei im Erfolgsfall platziert wird.

Rückgabewert

Der Handle, der verwendet werden soll, um weiter nach Dateien mit Methoden FileFindNext() zu suchen, oder INVALID_HANDLE wenn es keine entsprechende Datei gibt.

Hinweis

Der Arbeitsordner wird beim zuvor durch Methode SetCommon() gesetzten/gelöschten Flag definiert .

© 2000-2025, MetaQuotes Ltd.

FileFindNext

Weiter sucht nach Dateien nach der Methode FileFindFirst()

```
bool  FileFindNext(
   int      search_handle,     // Such-Handle
   string&  file_name          // Referenz
   )
```

Parameter

search_handle

[in] Such-Handle, der durch Methode FileFindFirst() erhalten war.

file_name

[in] Eine Referenz auf die Zeichenfolge, in der der Name der gefundenen Datei im Erfolgsfall platziert wird.

Rückgabewert

Gibt bei Erfolg true zurück, oder false wenn es keine mehr entsprechende Dateien gibt.

© 2000-2025, MetaQuotes Ltd.

FileFindClose

Schließt den Such-Handle.

```
void  FileFindClose(
   int  search_handle      // Such-Handle
   )
```

Parameter

search_handle

[in] Such-Handle, der durch Methode FileFindFirst() erhalten war.

© 2000-2025, MetaQuotes Ltd.

Klasse CFileBin

CFileBin ist eine Klasse für den vereinfachten Zugriff auf die Binärdateien.

Beschreibung

Klasse CFileBin bietet den Zugriff auf die Binärdateien.

Deklaration

```
class CFileBin: public CFile
```

Kopf

```
#include <Files\FileBin.mqh>
```

Vererbungshierarchie

CObject
  CFile
    CFileBin

Gruppen der Klassenmethode

Öffnung	
Open	Öffnet eine Binärdatei
Schreibmethoden	
WriteChar	Schreibt eine Variable vom Typ char oder uchar
WriteShort	Schreibt eine Variable vom Typ short oder ushort
WriteInteger	Schreibt eine Variable vom Typ int oder uint
WriteLong	Schreibt eine Variable vom Typ long oder ulong
WriteFloat	Schreibt eine Variable vom Typ float
WriteDouble	Schreibt eine Variable vom Typ double
WriteString	Schreibt eine Variable vom Typ string
WriteCharArray	Schreibt ein Array von Variablen vom Typ char oder uchar
WriteShortArray	Schreibt ein Array von Variablen vom Typ short oder ushort
WriteIntegerArray	Schreibt ein Array von Variablen vom Typ int oder uint
WriteLongArray	Schreibt ein Array von Variablen vom Typ long oder ulong
WriteFloatArray	Schreibt ein Array von Variablen vom Typ float
WriteDoubleArray	Schreibt ein Array von Variablen vom Typ double
WriteObject	Schreibt Daten einer Instanz der CObject geerbten Klasse

© 2000-2025, MetaQuotes Ltd.

Öffnung	
Lesemethoden	
ReadChar	Liest eine Variable vom Typ char oder uchar
ReadShort	Liest eine Variable vom Typ short oder ushort
ReadInteger	Liest eine Variable vom Typ int oder uint
ReadLong	Liest eine Variable vom Typ long oder ulong
ReadFloat	Liest eine Variable vom Typ float
ReadDouble	Liest eine Variable vom Typ double
ReadString	Liest eine Variable vom Typ string
ReadCharArray	Liest ein Array von Variablen vom Typ char oder uchar
ReadShortArray	Liest ein Array von Variablen vom Typ short oder ushort
ReadIntegerArray	Liest ein Array von Variablen vom Typ int oder uint
ReadLongArray	Liest ein Array von Variablen vom Typ long oder ulong
ReadFloatArray	Liest ein Array von Variablen vom Typ float
ReadDoubleArray	Liest ein Array von Variablen vom Typ double
ReadObject	Liest Daten einer Instanz der CObject geerbten Klasse

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CFile

Handle, FileName, Flags, SetUnicode, SetCommon, Open, Close, Delete, Size, Tell, Seek, Flush, IsEnding, IsLineEnding, Delete, IsExist, Copy, Move, FolderCreate, FolderDelete, FolderClean, FileFindFirst, FileFindNext, FileFindClose

© 2000-2025, MetaQuotes Ltd.

Open

Öffnet die angegebene Binärdatei und, wenn erfolgreich geöffnet, bindet sie an eine Klasseninstanz.

```
int  Open(
   const string  file_name,     // Dateiname
   int           flags          // Flags
   )
```

Parameter

file_name

[in] Name der Datei, die Sie öffnen.

flags

[in] Dateiöffnungsflags (Flag FILE_BIN wird zwangsweise gesetzt).

Rückgabewert

Handle der geöffneten Datei.

© 2000-2025, MetaQuotes Ltd.

WriteChar

Schreibt eine Variable vom Typ char oder uchar in eine Datei.

```
uint  WriteChar(
   char  value      // Wert
   )
```

Parameter

value

[in] Variable für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteShort

Schreibt eine Variable vom Typ short oder ushort in eine Datei.

```
uint  WriteShort(
   short  value      // Wert
   )
```

Parameter

value

[in] Variable für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteInteger

Schreibt eine Variable vom Typ int oder uint in eine Datei.

```
uint  WriteInteger(
   int  value      // Wert
   )
```

Parameter

value

[in] Variable für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteLong

Schreibt eine Variable vom Typ long oder ulong in eine Datei.

```
uint  WriteLong(
   long  value      // Wert
   )
```

Parameter

value

[in] Variable für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteFloat

Schreibt eine Variable vom Typ float in eine Datei.

```
uint  WriteFloat(
   float  value      // Wert
   )
```

Parameter

value

[in] Variable für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteDouble

Schreibt eine Variable vom Typ double in eine Datei.

```
uint  WriteDouble(
   double  value      // Wert
   )
```

Parameter

value

[in] Variable für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteString

Schreibt eine Variable vom Typ string in eine Datei.

```
uint  WriteString(
   const string  value      // Wert
   )
```

Parameter

value

[in] String für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

WriteString

Schreibt eine Variable vom Typ string in eine Datei.

```
uint  WriteString(
   const string  value,     // Wert
   int           size       // Größe
   )
```

Parameter

value

[in] String für Schreiben.

size

[in] Anzahl der Zeichen zu schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteCharArray

Schreibt ein Array von Variablen vom Typ char oder uchar in eine Datei.

```
uint  WriteCharArray(
   char&  array[],            // Array
   int    start_item=0,       // Anfangselement
   int    items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Ein Array für Schreiben.

start_item=0

[in] Startelement für Schreiben.

items_count=-1

[in] Anzahl der Elementen für Schreiben (-1 - das ganze Array).

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteShortArray

Schreibt ein Array von Variablen vom Typ short oder ushort in eine Datei.

```
uint  WriteShortArray(
   short&  array[],            // Array
   int     start_item=0,       // Anfangselement
   int     items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Ein Array für Schreiben.

start_item=0

[in] Startelement für Schreiben.

items_count=-1

[in] Anzahl der Elementen für Schreiben (-1 - das ganze Array).

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteIntegerArray

Schreibt ein Array von Variablen vom Typ int oder uint in eine Datei.

```
uint  WriteIntegerArray(
   int&  array[],             // Array
   int   start_item=0,        // Anfangselement
   int   items_count=-1       // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Ein Array für Schreiben.

start_item=0

[in] Startelement für Schreiben.

items_count=-1

[in] Anzahl der Elementen für Schreiben (-1 - das ganze Array).

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteLongArray

Schreibt ein Array von Variablen vom Typ long oder ulong in eine Datei.

```
uint  WriteLongArray(
   long&  array[],            // Array
   int    start_item=0,       // Anfangselement
   int    items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Ein Array für Schreiben.

start_item=0

[in] Startelement für Schreiben.

items_count=-1

[in] Anzahl der Elementen für Schreiben (-1 - das ganze Array).

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteFloatArray

Schreibt ein Array von Variablen vom Typ float in eine Datei.

```
uint  WriteFloatArray(
   float&  array[],            // Array
   int     start_item=0,       // Anfangselement
   int     items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Ein Array für Schreiben.

start_item=0

[in] Startelement für Schreiben.

items_count=-1

[in] Anzahl der Elementen für Schreiben (-1 - das ganze Array).

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteDoubleArray

Schreibt ein Array von Variablen vom Typ double in eine Datei.

```
uint  WriteDoubleArray(
   double&  array[],             // Array
   int      start_item=0,        // Anfangselement
   int      items_count=-1       // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Ein Array für Schreiben.

start_item=0

[in] Startelement für Schreiben.

items_count=-1

[in] Anzahl der Elementen für Schreiben (-1 - das ganze Array).

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

WriteObject

Schreibt Daten einer Instanz der CObject geerbten Klasse in eine Datei.

```
bool  WriteObject(
   CObject*  object      // Zeiger
   )
```

Parameter

object

[in] Ein Zeiger auf die Instanz der CObject geerbten Klasse für Schreiben.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht geschrieben werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadChar

Liest eine Variable vom Typ char oder uchar aus einer Datei.

```
bool  ReadChar(
   char&  value      // Flagwert
   )
```

Parameter

value

[in] Die Referenz an die Variable um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadShort

Liest eine Variable vom Typ short oder ushort aus einer Datei.

```
bool  ReadShort(
   short&  value
   )
```

Parameter

value

[in] Die Referenz an die Variable um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadInteger

Liest eine Variable vom Typ int oder uint aus einer Datei.

```
bool  ReadInteger(
   int&  value      // Variable
   )
```

Parameter

value

[in] Die Referenz an die Variable um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadLong

Liest eine Variable vom Typ long oder ulong aus einer Datei.

```
bool  ReadLong(
   long&  value
   )
```

Parameter

value

[in] Die Referenz an die Variable um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadFloat

Liest eine Variable vom Typ float aus einer Datei.

```
bool  ReadFloat(
   float&  value      // Variable
   )
```

Parameter

value

[in] Die Referenz an die Variable um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadDouble

Liest eine Variable vom Typ double aus einer Datei.

```
bool  ReadDouble(
   double&  value
   )
```

Parameter

value

[in] Die Referenz an die Variable um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadString

Liest eine Variable vom Typ string aus einer Datei.

```
bool  ReadString(
   string&  value      // String
   )
```

Parameter

value

[in] Die Referenz an String um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

ReadString

Liest eine Variable vom Typ string aus einer Datei.

```
bool  ReadString(
   string&  value
   )
```

Parameter

value

[in] Die Referenz an String um die gelesenen Daten zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadCharArray

Liest ein Array von Variablen vom Typ char oder uchar aus einer Datei.

```
bool  ReadCharArray(
   char&  array[],            // Array
   int    start_item=0,       // Anfangselement
   int    items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Die Referenz an ein Array um die gelesenen Daten zu platzieren.

start_item=0

[in] Startelement für Lesen.

items_count=-1

[in] Anzahl der Elementen für Lesen (-1 -bis zum Ende der Datei).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadShortArray

Liest ein Array von Variablen vom Typ short oder ushort aus einer Datei.

```
bool  ReadShortArray(
   short&  array[],            // Array
   int     start_item=0,       // Anfangselement
   int     items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Die Referenz an ein Array um die gelesenen Daten zu platzieren.

start_item=0

[in] Startelement für Lesen.

items_count=-1

[in] Anzahl der Elementen für Lesen (-1 -bis zum Ende der Datei).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadIntegerArray

Liest ein Array von Variablen vom Typ int oder uint aus einer Datei.

```
bool  ReadIntegerArray(
   int&  array[],             // Array
   int   start_item=0,        // Anfangselement
   int   items_count=-1       // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Die Referenz an ein Array um die gelesenen Daten zu platzieren.

start_item=0

[in] Startelement für Lesen.

items_count=-1

[in] Anzahl der Elementen für Lesen (-1 -bis zum Ende der Datei).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadLongArray

Liest ein Array von Variablen vom Typ long oder ulong aus einer Datei.

```
bool  ReadLongArray(
   long&  array[],            // Array
   int    start_item=0,       // Anfangselement
   int    items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Die Referenz an ein Array um die gelesenen Daten zu platzieren.

start_item=0

[in] Startelement für Lesen.

items_count=-1

[in] Anzahl der Elementen für Lesen (-1 -bis zum Ende der Datei).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadFloatArray

Liest ein Array von Variablen vom Typ float aus einer Datei.

```
bool  ReadFloatArray(
   float&  array[],            // Array
   int     start_item=0,       // Anfangselement
   int     items_count=-1      // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Die Referenz an ein Array um die gelesenen Daten zu platzieren.

start_item=0

[in] Startelement für Lesen.

items_count=-1

[in] Anzahl der Elementen für Lesen (-1 -bis zum Ende der Datei).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadDoubleArray

Liest ein Array von Variablen vom Typ double aus einer Datei.

```
bool  ReadDoubleArray(
   double&  array[],             // Array
   int      start_item=0,        // Anfangselement
   int      items_count=-1       // Anzahl der Elemente
   )
```

Parameter

array[]

[in] Die Referenz an ein Array um die gelesenen Daten zu platzieren.

start_item=0

[in] Startelement für Lesen.

items_count=-1

[in] Anzahl der Elementen für Lesen (-1 -bis zum Ende der Datei).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

ReadObject

Liest Daten einer Instanz der CObject geerbten Klasse aus einer Datei.

```
bool  ReadObject(
   CObject*  object      // Zeiger
   )
```

Parameter

object

[in] Ein Zeiger auf die Instanz der CObject geerbten Klasse für Lesen.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Daten nicht gelesen werden konnten.

© 2000-2025, MetaQuotes Ltd.

Klasse CFileTxt

CFileTxt ist eine Klasse für den vereinfachten Zugriff auf die Textdateien.

Beschreibung

Klasse CFileTxt bietet den Zugriff auf die Textdateien.

Deklaration

```
class CFileTxt: public CFile
```

Kopf

```
#include <Files\FileTxt.mqh>
```

Vererbungshierarchie

CObject
  CFile
    CFileTxt

Gruppen der Klassenmethode

Öffnung	
Open	Öffnet eine Textdatei
Schreibmethoden	
WriteString	Schreibt eine Variable vom Typ string
Lesemethoden	
ReadString	Liest eine Variable vom Typ string

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CFile

Handle, FileName, Flags, SetUnicode, SetCommon, Open, Close, Delete, Size, Tell, Seek, Flush, IsEnding, IsLineEnding, Delete, IsExist, Copy, Move, FolderCreate, FolderDelete, FolderClean, FileFindFirst, FileFindNext, FileFindClose

© 2000-2025, MetaQuotes Ltd.

Open

Öffnet die angegebene Textdatei und, wenn erfolgreich geöffnet, bindet sie an eine Klasseninstanz.

```
int  Open(
   const string  file_name,      // Dateiname
   int           flags           // Flags
   )
```

Parameter

file_name

[in] Name der Datei, die Sie öffnen.

flags

[in] Dateiöffnungsflags (Flag FILE_TXT wird zwangsweise gesetzt).

Rückgabewert

Handle der geöffneten Datei.

© 2000-2025, MetaQuotes Ltd.

WriteString

Schreibt eine Variable vom Typ string in eine Datei.

```
uint  WriteString(
   const string  value      // String
   )
```

Parameter

value

[in] String für Schreiben.

Rückgabewert

Die Anzahl der geschriebenen Bytes.

© 2000-2025, MetaQuotes Ltd.

ReadString

Liest eine Variable vom Typ string aus einer Datei.

```
string  ReadString()
```

Rückgabewert

Gelesener String.

© 2000-2025, MetaQuotes Ltd.

String-Operationen

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen von String-Operationen und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen von String-Operationen sparen Sie Zeit bei der Entwicklung von benutzerdefinierten Anwendungen, die Textinformationen verarbeiten.

Die Standardbibliothek MQL5 (in Bezug auf Arbeit mit Strings) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Strings.

Klassenname	Klassenbeschreibung
CString	Klasse von String-Operationen

© 2000-2025, MetaQuotes Ltd.

Klasse CString

CString ist eine Klasse für den vereinfachten Zugriff auf die Variablen vom Typ string.

Beschreibung

Klasse CString bietet vereinfachten Zugriff auf die Funktionen von API MQL5 für die Arbeit mit Variablen vom Typ srting.

Deklaration

```
class CString: public CObject
```

Kopf

```
#include <Strings\String.mqh>
```

Vererbungshierarchie

CObject

CString

Gruppen der Klassenmethode

Datenzugriffsmethoden	
Str	Erhält eine Zeichenfolge
Len	Erhält die Länge des Strings
Copy	Erhält eine Kopie des Strings
Methoden zur Abfüllung	
Fill	Füllt einen String
Assign	Weist einen String zu
Append	Fügt einen String hinzu
Insert	Fügt einen String ein
Vergleichsmethoden	
Compare	Vergleicht Strings
CompareNoCase	Vergleicht Strings egal in welcher Groß-/Kleinschreibung
Methoden der Arbeit mit Teilstrings	
Left	Erhält einen Teilstring links
Right	Erhält einen Teilstring rechts
Mid	Ruft einen Teilstring aus der Mitte

© 2000-2025, MetaQuotes Ltd.

Datenzugriffsmethoden	
Methoden der Trimmung/Löschen	
Trim	Kürzt die linke und rechte Seite eines Strings
TrimLeft	Kürzt die linke Seite eines Strings
TrimRight	Kürzt die rechte Seite eines Strings
Clear	Löscht alle Zeichen aus dem String
Verwandlungsmethoden	
ToUpper	Einen String in Großbuchstaben verwandelt
ToLower	Einen String in Kleinbuchstaben verwandelt
Reverse	Kehrt einen String um
Suchmethoden	
Find	Sucht nach einem Teilstring von links nach rechts
FindRev	Sucht nach einem Teilstring von rechts nach links
Remove	Löscht einen Teilstring
Replace	Ersetzt einen Teilstring

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type

© 2000-2025, MetaQuotes Ltd.

Str

Erhält einen String.

```
string  Str() const;
```

Rückgabewert

Eine Kopie des Strings.

© 2000-2025, MetaQuotes Ltd.

Len

Erhält die Länge des Strings.

```
uint  Len() const;
```

Rückgabewert

Die Länge des Strings.

© 2000-2025, MetaQuotes Ltd.

Copy

Kopiert den String auf die Referenz.

```
void  Copy(
   string&  copy      // Referenz
   ) const;
```

Parameter

copy

[in] Referenz an den String.

Copy

Kopiert den String in eine Instanz der CString-Klasse.

```
void  Copy(
   CString*  copy      // Zeiger
   ) const;
```

Parameter

copy

[in] Ein Zeiger auf die Instanz der CString-Klasse für Daten.

© 2000-2025, MetaQuotes Ltd.

Fill

Füllt einen String mit dem angegebenen Zeichen.

```
bool  Fill(
   short  character      // Zeichen
   )
```

Parameter

character

[in] Zeichen um den String zu füllen.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der String nicht gefüllt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Assign

Weist einen String zu.

```
void  Assign(
   const string  str      // String
   )
```

Parameter

str

[in] Der String, der zugewiesen wird.

Assign

Weist einen String aus einer Instanz von CString zu.

```
void  Assign(
   CString*  str      // Zeiger
   )
```

Parameter

str

[in] Ein Zeiger auf die Instanz der CString-Klasse für Zuweisung.

© 2000-2025, MetaQuotes Ltd.

Append

Fügt einen String hinzu.

```
void  Append(
   const string  str      // String
   )
```

Parameter

str

[in] Der String, der hinzugefügt wird.

Append

Fügt einen String aus einer Instanz von CString hinzu.

```
void  Append(
   CString*  string      // Zeiger
   )
```

Parameter

string

[in] Ein Zeiger auf die Instanz der CString-Klasse für Hinzufügen.

© 2000-2025, MetaQuotes Ltd.

Insert

Fügt einen String and die angegebene Position ein.

```
uint  Insert(
   uint          pos,      // Position
   const string  str       // String
   )
```

Parameter

pos

[in] Die Einfügeposition.

str

[in] Der String, der eingefügt wird.

Rückgabewert

Die Länge des resultierenden Strings.

Insert

Fügt einen String aus einer Instanz von CString ad die angegebene Position ein.

```
uint  Insert(
   uint      pos,      // Position
   CString*  str       // Zeiger
   )
```

Parameter

pos

[in] Die Einfügeposition.

str

[in] Ein Zeiger auf die Instanz der CString-Klasse für Einfügen.

Rückgabewert

Die Länge des resultierenden Strings.

© 2000-2025, MetaQuotes Ltd.

Compare

Vergleicht mit einem String.

```
int  Compare(
   const string  str      // String
   ) const;
```

Parameter

str

[in] Der String zum Vergleich.

Rückgabewert

0 - wenn die Strings gleich sind, -1 - wenn der Klasse-Mitglied-String kleiner als der Parameter-String ist, 1 - wenn der Klasse-Mitglied-String größer als der Parameter-String ist.

Compare

Vergleicht mit dem String der eine Instanz von CString ist.

```
int  Compare(
   CString*  str      // Zeiger
   ) const;
```

Parameter

str

[in] Ein Zeiger auf die Instanz der CString-Klasse für Vergleich.

Rückgabewert

0 - wenn die Strings gleich sind, -1 - wenn der Klasse-Mitglied-String kleiner als der Parameter-String ist, 1 - wenn der Klasse-Mitglied-String größer als der Parameter-String ist.

© 2000-2025, MetaQuotes Ltd.

CompareNoCase

Vergleicht Strings egal in welcher Groß-/Kleinschreibung.

```
int  CompareNoCase(
   const string  str      // String
   ) const;
```

Parameter

str

[in] Der String zum Vergleich.

Rückgabewert

0 - wenn die Strings gleich sind, -1 - wenn der Klasse-Mitglied-String kleiner als der Parameter-String ist, 1 - wenn der Klasse-Mitglied-String größer als der Parameter-String ist.

CompareNoCase

Vergleicht mit dem String, der eine Instanz von CString ist, egal in welcher Groß-/Kleinschreibung.

```
int  CompareNoCase(
   CString*  str      // Zeiger
   ) const;
```

Parameter

str

[in] Ein Zeiger auf die Instanz der CString-Klasse für Vergleich.

Rückgabewert

0 - wenn die Strings gleich sind, -1 - wenn der Klasse-Mitglied-String kleiner als der Parameter-String ist, 1 - wenn der Klasse-Mitglied-String größer als der Parameter-String ist.

© 2000-2025, MetaQuotes Ltd.

Left

Erhält den Teilstring mit der angegebenen Länge beginnend mit dem Anfang des Strings

```
string  Left(
   uint  count      // Länge
   )
```

Parameter

count

[in] Länge der Teilstring.

Rückgabewert

Der resultierende String.

© 2000-2025, MetaQuotes Ltd.

Right

Erhält den Teilstring mit der angegebenen Länge beginnend mit dem Ende des Strings

```
string  Right(
   uint  count       // Länge
   )
```

Parameter

count

[in] Länge der Teilstring.

Rückgabewert

Der resultierende String.

© 2000-2025, MetaQuotes Ltd.

Mid

Erhält den Teilstring mit der angegebenen Länge beginnend mit der angegebene Position des Strings

```
string  Mid(
   uint  pos,        // Position
   uint  count       // Länge
   )
```

Parameter

pos

[in] Position des Teilstrings.

count

[in] Länge der Teilstring.

Rückgabewert

Der resultierende String.

© 2000-2025, MetaQuotes Ltd.

Trim

Löscht alle Zeichen aus dem Satz (zusätzlich ' ','\t','\r','\n') am Anfang und Ende des Strings.

```
int  Trim(
   const string  targets      // Satz
   )
```

Parameter

targets

[in] Eine Reihe von Zeichen zu löschen.

Rückgabewert

Die Anzahl der gelöschten Zeichen.

Beispiel:

```
//--- example for CString::Trim
#include <Strings\String.mqh>
//---
void OnStart()
  {
   CString str;
   //---
   str.Assign("   \t\tABCD\r\n");
   printf("Source string '%s'",str.Str());
   //---
   str.Trim("DA-DA-DA");
   printf("Result string '%s'",str.Str());
  }
```

© 2000-2025, MetaQuotes Ltd.

TrimLeft

Löscht alle Zeichen aus dem Satz (zusätzlich ' ','\t','\r','\n') am Anfang des Strings.

```
int  TrimLeft(
   const string  targets      // Satz
   )
```

Parameter

targets

[in] Eine Reihe von Zeichen zu löschen.

Rückgabewert

Die Anzahl der gelöschten Zeichen.

© 2000-2025, MetaQuotes Ltd.

TrimRight

Löscht alle Zeichen aus dem Satz (zusätzlich ' ','\t','\r','\n') am Ende des Strings.

```
int  TrimRight(
   const string  targets      // Satz
   )
```

Parameter

targets

[in] Eine Reihe von Zeichen zu löschen.

Rückgabewert

Die Anzahl der gelöschten Zeichen.

© 2000-2025, MetaQuotes Ltd.

Clear

Löscht alle Zeichen aus dem String

```
bool  Clear()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Operation fehlgeschlagen ist.

© 2000-2025, MetaQuotes Ltd.

ToUpper

Verwandelt alle Zeichen eines Strings in Großbuchstaben.

```
bool  ToUpper()
```

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Register nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ToLower

Verwandelt alle Zeichen eines Strings in Kleinbuchstaben.

```
bool  ToLower()
```

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Register nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Reverse

Kehrt einen String um (wechselt die Paaren aus Anfangs- und Endzeichen).

```
void  Reverse()
```

© 2000-2025, MetaQuotes Ltd.

Find

Sucht nach der ersten Aufnahme des Teilstrings beginnend mit der angegebenen Position.

```
int  Find(
   uint          start,          // Position
   const string  substring       // Teilstring
   ) const;
```

Parameter

start

[in] Startposition für Suche nach dem Teilstring.

substring

[in] Muster des Teilstrings für Suche.

Rückgabewert

Der Index der ersten Aufnahme des Teilstrings (-1, wenn der Teilstring nicht gefunden ist).

© 2000-2025, MetaQuotes Ltd.

FindRev

Sucht nach der letzten Aufnahme des Teilstrings.

```
int  FindRev(
   const string  substring      // Teilstring
   ) const;
```

Parameter

substring

[in] Muster des Teilstrings für Suche.

Rückgabewert

Der Index der letzten Aufnahme des Teilstrings (-1, wenn der Teilstring nicht gefunden ist).

© 2000-2025, MetaQuotes Ltd.

Remove

Löscht alle Aufnahmen der Teilstrings.

```
uint  Remove(
   const string  substring      // Teilstring
   )
```

Parameter

substring

[in] Muster des Teilstrings für Suche.

Rückgabewert

Die Anzahl der Löschungen des Teilstrings.

© 2000-2025, MetaQuotes Ltd.

Replace

Ersetzt alle Aufnahmen der Teilstrings.

```
uint  Replace(
   const string  substring,     // Teilstring
   const string  newstring      // Teilstring
   )
```

Parameter

substring

[in] Muster des Teilstrings für Suche.

newstring

[in] Muster des Teilstrings für Ersetzung.

Rückgabewert

Die Anzahl der Ersetzungen des Teilstrings.

© 2000-2025, MetaQuotes Ltd.

Grafische Objekte

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen von Grafischen Objekten und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen von Grafischen Objekten, sparen Sie Zeit bei der Entwicklung von benutzerdefinierten Anwendungen (Skripte, Expert Advisors).

Die Standardbibliothek MQL5 (in Bezug auf grafische Objekte) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\ChartObjects.

Klasse/Gruppe	Beschreibung
CChartObject	Die Basisklasse der grafischen Objekt
Linien	Klassengruppe "Linien"
Kanäle	Klassengruppe "Kanäle"
Gann-Werkzeuge	Klassengruppe "Gann"
Fibonacci-Werkzeuge	Klassengruppe "Fibonacci"
Elliott Werkzeuge	Klassengruppe "Elliott"
Figuren	Klassengruppe "Figuren"
Pfeile	Gruppenklasse "Pfeile"
Steuerelemente	Klassengruppe "Steuerelemente"

© 2000-2025, MetaQuotes Ltd.

CChartObject

CChartObject ist eine Basisklasse für die Klassen von grafischen Objekten in der Standardbibliothek MQL5.

Beschreibung

Klasse CChartObject bietet allen ihren abgeleiteten Klassen den vereinfachten Zugriff auf die Funktionen API MQL5.

Deklaration

```
class CChartObject : public CObject
```

Kopf

```
#include <ChartObjects\ChartObject.mqh>
```

Vererbungshierarchie

CObject

CChartObject

Direkte Ableitungen

CChartObjectArrow, CChartObjectBitmap, CChartObjectBmpLabel, CChartObjectCycles, CChartObjectElliottWave3, CChartObjectEllipse, CChartObjectFiboArc, CChartObjectFiboFan, CChartObjectFiboTimes, CChartObjectHLine, CChartObjectRectangle, CChartObjectSubChart, CChartObjectText, CChartObjectTrend, CChartObjectTriangle, CChartObjectVLine

Gruppen der Klassenmethode

Attribute	
ChartId	Erhält die ID des Charts, zu dem ein grafisches Objekt gehört
Window	Erhält die Nummer des Chartfensters, in dem ein grafisches Objekt sich befindet
Name	Erhält/setzt den Namen eines grafischen Objektes
NumPoints	Erhält die Anzahl der Ankerpunkte
Füllung	
Attach	Bindet ein grafisches Objekt des Charts
SetPoint	Setzt die Ankerpunktparameter
Löschen	
Delete	Löscht ein grafisches Objekt des Charts
Detach	Löst ein grafisches Objekt des Charts
Verschiebung	

© 2000-2025, MetaQuotes Ltd.

Attribute	
ShiftObject	Die relative Verschiebung des Objekts
ShiftPoint	Die relative Verschiebung des Objektpunkts
Objekteigenschaften	
Time	Erhält/setzt die Zeitkoordinate eines Objektpunkts
Price	Erhält/setzt die Preiskoordinate eines Objektpunkts
Color	Erhält/setzt die Farbe eines Objekts
Style	Erhält/setzt den Linienstil eines Objekts
Width	Erhält/setzt die Linienbreite eines Objekts
Background	Erhält/setzt ein Flag der Objektzeichnung im Hintergrund
Selected	Erhält/setzt ein Flag der Objektwahl
Selectable	Erhält/setzt ein Flag der "Wählbarkeit" eines Objektes
Description	Erhält/setzt den Text eines Objekts
Tooltip	Erhält/setzt den Text von Tooltip eines Objekts
Timeframes	Erhält/setzt die Maske von Flags der Objektanzeige
Z_Order	Erhält/setzt die Priorität eines grafischen Objektes für Erhaltung eines Ereignisses von Mausklick auf den Chart
CreateTime	Erhält die Objekterstellungszeit
Die Eigenschaften der Objektebenen	
LevelsCount	Erhält/setzt die Anzahl der Objektebenen
LevelColor	Erhält/setzt die Farbe der Ebenelinie
LevelStyle	Erhält/setzt den Stil der Ebenelinie
LevelWidth	Erhält/setzt die Breite der Ebenelinie
LevelValue	Erhält/setzt den Wert der Ebene
LevelDescription	Erhält/setzt den Text einer Ebene
Zugriff auf Funktionen API MQL5	
GetInteger	Erhält den Wert der Objekteigenschaft
SetInteger	Setzt den Wert der Objekteigenschaft
GetDouble	Erhält den Wert der Objekteigenschaft
SetDouble	Setzt den Wert der Objekteigenschaft

© 2000-2025, MetaQuotes Ltd.

Attribute	
GetString	Erhält den Wert der Objekteigenschaft
SetString	Setzt den Wert der Objekteigenschaft
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

© 2000-2025, MetaQuotes Ltd.

ChartId

Erhält die ID des Charts, zu dem ein grafisches Objekt gehört.

```
long  ChartId() const
```

Rückgabewert

Die ID des Charts auf dem ein grafisches Objekt sich befindet. Wenn es kein gebundenes Objekt gibt, wird -1 zurückgegeben.

Beispiel:

```
//--- example for CChartObject::ChartId
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get chart idintifier of chart object
   long chart_id=object.ChartId();
  }
```

© 2000-2025, MetaQuotes Ltd.

Window

Erhält die Nummer des Chartfensters, in dem ein grafisches Objekt sich befindet.

```
int  Window() const
```

Rückgabewert

Die Nummer des Chartfensters, in dem ein grafisches Objekt sich befindet (0 - das Hauptfenster). Wenn es kein gebundenes Objekt gibt, wird -1 zurückgegeben.

Beispiel:

```
//--- example for CChartObject::Window
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get window of chart object
   int window=object.Window();
  }
```

© 2000-2025, MetaQuotes Ltd.

Name (Get-Methode)

Erhält den Namen eines grafischen Objektes.

```
string  Name() const
```

Rückgabewert

Der Name eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird NULL zurückgegeben.

Name (Set-Methode)

Setzt den Namen eines grafischen Objektes

```
bool  Name(
   string  name      // der neue Namen
   )
```

Parameter

name

[in] Der neue Name eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Name nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Name
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get name of chart object
   string object_name=object.Name();
   if(object_name!="MyChartObject")
     {
     //--- set name of chart object
     object.Name("MyChartObject");
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

NumPoints

Erhält die Anzahl der Ankerpunkte des graphischen Objektes.

```
int  NumPoints() const
```

Rückgabewert

Die Anzahl der Ankerpunkte eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Beispiel:

```
//--- example for CChartObject::NumPoints
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get points count of chart object
   int points=object.NumPoints();
  }
```

© 2000-2025, MetaQuotes Ltd.

Attach

Bindet ein grafisches Objekt an eine Klasseninstanz.

```
bool  Attach(
   long     chart_id,      // Chart ID
   string   name,          // Objektname
   int      window,        // Chartfenster
   int      points         // die Anzahl der Punkte
   )
```

Parameter

chart_id

[out] Chart ID.

name

[in] Name des grafischen Objektes.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

points

[in] Die Anzahl der Ankerpunkte des grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Objekt nicht gebunden werden kann.

Beispiel:

```
//--- example for CChartObject::Attach
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- attach chart object
   if(!object.Attach(ChartID(),"MyObject",0,2))
     {
      printf("Object attach error");
      return;
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SetPoint

Setzt die neue Koordinaten des angegebenen Ankerpunktes des grafischen Objektes.

```
bool  SetPoint(
   int       point,          // Punktnummer
   datetime  new_time,       // Zeitkoordinate
   double    new_price       // Preiskoordinate
   )
```

Parameter

point

[in] Die Nummer des Ankerpunktes eines grafischen Objektes.

new_time

[in] Der neue Wert der Zeitkoordinate des angegebenen Ankerpunktes des grafischen Objektes.

new_price

[in] Der neue Wert der Preiskoordinate des angegebenen Ankerpunktes des grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Koordinaten des Punktes nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::SetPoint
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   double       price;
   //---
   if(object.NumPoints()>0)
     {
      //--- set point of chart object
      object.SetPoint(0,CurrTime(),price);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein gebundenes grafisches Objekt des Charts.

```
bool  Delete()
```

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Objekt nicht gelöscht werden kann.

Beispiel:

```
//--- example for CChartObject::Delete
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- detach chart object
   if(!object.Delete())
     {
      printf("Object delete error");
      return;
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Detach

Löst ein grafisches Objekt des Charts.

```
void  Detach()
```

Rückgabewert

Nichts.

Beispiel:

```
//--- example for CChartObject::Detach
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- detach chart object
   object.Detach();
  }
```

© 2000-2025, MetaQuotes Ltd.

ShiftObject

Verschiebt ein grafisches Objekt des Charts.

```
bool  ShiftObject(
   datetime  d_time,      // Inkrement der Zeitkoordinate
   double    d_price      // Inkrement der Preiskoordinate
   )
```

Parameter

d_time

[in] Inkrement der Zeitkoordinate aller Ankerpunkten

d_price

[in] Inkrement der Preiskoordinate aller Ankerpunkten

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das Objekt nicht verschiebt werden konnte.

Beispiel:

```
//--- example for CChartObject::ShiftObject
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   datetime     d_time;
   double       d_price;
   //--- shift chart object
   object.ShiftObject(d_time,d_price);
  }
```

© 2000-2025, MetaQuotes Ltd.

ShiftPoint

Verschiebt den angegebenen Ankerpunkt eines grafischen Objektes.

```
bool  ShiftPoint(
   int       point,       // Punktnummer
   datetime  d_time,      // Inkrement der Zeitkoordinate
   double    d_price      // Inkrement der Preiskoordinate
   )
```

Parameter

point

[in] Die Nummer des Ankerpunktes eines grafischen Objektes.

d_time

[in] Inkrement der Zeitkoordinate des angegebenen Ankerpunktes des grafischen Objektes.

d_price

[in] Inkrement der Preiskoordinate des angegebenen Ankerpunktes des grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Punkt nicht verschiebt werden konnte.

Beispiel:

```
//--- example for CChartObject::ShiftPoint
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   datetime     d_time;
   double       d_price;
   //---
   if(object.NumPoints()>0)
     {
      //--- shift point of chart object
      object.ShiftPoint(0,d_time,d_price);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Time (Get-Methode)

Erhält die Zeitkoordinate des angegebenen Ankerpunktes des grafischen Objektes.

```
datetime  Time(
   int  point      // die Nummer des Punktes
   ) const
```

Parameter

point

[in] Die Nummer des Ankerpunktes eines grafischen Objektes.

Rückgabewert

Die Zeitkoordinate des angegebenen Ankerpunktes eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat diesen Punkt nicht, wird 0 zurückgegeben.

Time (Set-Methode)

Setzt die Zeitkoordinate des angegebenen Ankerpunktes des grafischen Objektes.

```
bool  Time(
   int       point,        // Punktnummer
   datetime  new_time      // Zeit
   )
```

Parameter

point

[in] Die Nummer des Ankerpunktes eines grafischen Objektes.

new_time

[in] Der neue Wert der Zeitkoordinate des angegebenen Ankerpunktes des grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Zeitkoordinate nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Time
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.NumPoints();i++)
     {
      //--- get time of point chart object
      datetime point_time=object.Time(i);
```

© 2000-2025, MetaQuotes Ltd.

```
      if(point_time==0)
        {
         //--- set time of point chart object
         object.Time(i,TimeCurrent());
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Price (Get-Methode)

Erhält die Preiskoordinate des angegebenen Ankerpunktes des grafischen Objektes.

```
double  Price(
   int  point      // die Nummer des Punktes
   ) const
```

Parameter

point

[in] Die Nummer des Ankerpunktes eines grafischen Objektes.

Rückgabewert

Die Preiskoordinate des angegebenen Ankerpunktes eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat diesen Punkt nicht, wird EMPTY_VALUE zurückgegeben.

Price (Set-Methode)

Setzt die Preiskoordinate des angegebenen Ankerpunktes des grafischen Objektes.

```
bool  Price(
   int     point,          // Punktnummer
   double  new_price       // Preis
   )
```

Parameter

point

[in] Die Nummer des Ankerpunktes eines grafischen Objektes.

new_price

[in] Der neue Wert der Preiskoordinate des angegebenen Ankerpunktes des grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Preiskoordinate nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Price
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   double       price;
   //---
   for(int i=0;i<object.NumPoints();i++)
     {
      //--- get price of point chart object
```

© 2000-2025, MetaQuotes Ltd.

```
      double point_price=object.Price(i);
      if(point_price!=price)
        {
         //--- set price of point chart object
         object.Price(i,price);
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Color (Get-Methode)

Erhält die Linienfarbe eines grafischen Objektes.

```
color  Color() const
```

Rückgabewert

Die Linienfarbe eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird CLR_NONE zurückgegeben.

Color (Set-Methode)

Setzt die Linienfarbe eines grafischen Objektes.

```
bool  Color(
   color  new_color      // die neue Farbe
   )
```

Parameter

new_color

[in] Der neue Wert der Linienfarbe eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Color
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get color of chart object
   color object_color=object.Color();
   if(object_color!=clrRed)
     {
     //--- set color of chart object
     object.Color(clrRed);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Style (Get-Methode)

Erhält den Linienstil eines grafischen Objektes.

```
ENUM_LINE_STYLE  Style() const
```

Rückgabewert

Der Linienstil eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Style (Set-Methode)

Setzt den Linienstil eines grafischen Objektes.

```
bool  Style(
   ENUM_LINE_STYLE  new_style      // Stil
   )
```

Parameter

new_style

[in] Der neue Wert des Linienstils eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Stil nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Style
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get style of chart object
   ENUM_LINE_STYLE style=object.Style();
   if(style!=STYLE_SOLID)
     {
      //--- set style of chart object
      object.Style(STYLE_SOLID);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Width (Get-Methode)

Erhält die Linienbreite eines grafischen Objektes.

```
int  Width() const
```

Rückgabewert

Die Linienbreite eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird -1 zurückgegeben.

Width (Set-Methode)

Setzt die Linienbreite eines grafischen Objektes.

```
bool  Width(
   int  new_width      // Breite
   )
```

Parameter

new_width

[in] Der neue Wert der Linienbreite eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Breite nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Width
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get width of chart object
   int width=object.Width();
   if(width!=1)
     {
      //--- set width of chart object
      object.Width(1);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Background (Get-Methode)

Erhält ein Flag der Objektzeichnung im Hintergrund.

```
bool  Background() const
```

Rückgabewert

Ein Flag der Zeichnung im Hintergrund eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Background (Set-Methode)

Setzt ein Flag der Objektzeichnung im Hintergrund.

```
bool  Background(
   bool  background      // Flagwert
   )
```

Parameter

background

[in] Der neue Wert des Flags der Objektzeichnung im Hintergrund.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Background
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get background flag of chart object
   bool background_flag=object.Background();
   if(!background_flag)
     {
     //--- set background flag of chart object
     object.Background(true);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Selected (Get-Methode)

Erhält das Flag der Objektwahl. Mit anderen Worten, ob ein grafisches Objekt gewählt ist oder nicht.

```
bool  Selected() const
```

Rückgabewert

Das Flag der Wahl eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Selected (Set-Methode)

Setzt das Flag der Objektwahl.

```
bool  Selected(
   bool  selected      // Flagwert
   )
```

Parameter

selected

[in] Der neue Wert des Flags der Objektwahl.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Selected
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get the "selected" flag of chart object
   bool selected_flag=object.Selected();
   if(selected_flag)
     {
     //--- set the "selected" flag of chart object
     object.Selected(false);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Selectable (Get-Methode)

Erhält das Flag der "Wählbarkeit" eines grafischen Objektes. Mit anderen Worten, ob ein grafisches Objekt ausgewählt werden kann.

```
bool  Selectable() const
```

Rückgabewert

Das Flag der "Wählbarkeit" eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Selectable (Set-Methode)

Setzt das Flag der "Wählbarkeit" eines grafischen Objektes.

```
bool  Selectable(
   bool  selectable      // Flagwert
   )
```

Parameter

selectable

[in] Der neue Wert des Flags der "Wählbarkeit" eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Selectable
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get the "selectable" flag of chart object
   bool selectable_flag=object.Selectable();
   if(selectable_flag)
     {
     //--- set the "selectable" flag of chart object
     object.Selectable(false);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Description (Get-Methode)

Erhält die Beschreibung (Text) eines grafischen Objektes.

```
string  Description() const
```

Rückgabewert

Die Beschreibung (Text) eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird NULL zurückgegeben.

Description (Set-Methode)

Setzt die Beschreibung (Text) eines grafischen Objektes.

```
bool  Description(
   string  text      // Text
   )
```

Parameter

text

[in] Der neue Wert der Beschreibung (des Textes) eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Beschreibung (der Text) nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Description
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get description of chart object
   string description=object.Description();
   if(description=="")
     {
      //--- set description of chart object
      object.Description("MyObject");
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Tooltip (Get-Methode)

Erhält den Text von Tooltip.

```
string  Tooltip() const
```

Rückgabewert

Tooltip-Text eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird NULL zurückgegeben.

Tooltip (Set-Methode)

Setzt den Text von Tooltip.

```
bool  Tooltip(
   string  new_tooltip      // der neue Tooltip-Text
   )
```

Parameter

new_tooltip

[in] Der neue Wert des Tooltip-Textes eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Tooltip-Text nicht geändert werden konnte.

Bemerkung:

Wenn die Eigenschaft nicht gesetzt ist, wird ein durch den Terminal automatisch generierter Tooltip angezeigt. Ein Tooltip kann ausgeschaltet werden - setzen sie den Wert "\n" (Zeilenumbruch).

© 2000-2025, MetaQuotes Ltd.

Timeframes (Get-Methode)

Erhält das Flag der Sichtbarkeit eines grafischen Objektes.

```
int  Timeframes() const
```

Rückgabewert

Das Flag der Sichtbarkeit eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Timeframes (Set-Methode)

Setzt das Flag der Sichtbarkeit eines grafischen Objektes.

```
bool  Timeframes(
   int  new_timeframes      // Flags der Sichtbarkeit
   )
```

Parameter

new_timeframes

[in] Neue Sichtbarkeitsflage eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Flags nicht geändert werden konnten.

Beispiel:

```
//--- example for CChartObject::Timeframes
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get timeframes of chart object
   int timeframes=object.Timeframes();
   if(!(timeframes&OBJ_PERIOD_H1))
     {
      //--- set timeframes of chart object
      object.Timeframes(timeframes|OBJ_PERIOD_H1);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Z_Order (Get - Methode)

Erhält die Priorität eines grafischen Objektes für Erhaltung eines Ereignisses von Mausklick auf den Chart (CHARTEVENT_CLICK).

```
long  Z_Order() const
```

Rückgabewert

Die Priorität grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Z_Order (Set-Methode)

Setzt die Priorität eines grafischen Objektes für Erhaltung eines Ereignisses von Mausklick auf den Chart (CHARTEVENT_CLICK).

```
bool  Z_Order(
   long  value      // die Priorität eines grafischen Objektes
   )
```

Parameter

value

[in] Die neue Priorität des grafischen Objektes für Erhaltung des Ereignisses CHARTEVENT_CLICK.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Priorität nicht geändert werden konnte.

Hinweis

Eigenschaft Z_Order wird verwendet, um eine Priorität während der Verarbeitung vom Mausklick auf die Objekte zu steuern. Die Werte größer als der Standardwert (0) erhöhen die Priorität eines Objektes bei der Verarbeitung von Mausklicks.

© 2000-2025, MetaQuotes Ltd.

CreateTime

Erhält die Objekterstellungszeit.

```
datetime  CreateTime() const
```

Rückgabewert

Die Erstellungszeit eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Beispiel:

```
//--- example for CChartObject::CreateTime
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get create time of chart object
   datetime create_time=object.CreateTime();
  }
```

© 2000-2025, MetaQuotes Ltd.

LevelsCount (Get-Methode)

Erhält die Anzahl der Ebenen eines grafischen Objektes.

```
int  LevelsCount() const
```

Rückgabewert

Die Anzahl der Ebenen eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

LevelsCount (Set-Methode)

Setzt die Anzahl der Ebenen eines grafischen Objektes.

```
bool  LevelsCount(
   int  levels      // die Anzahl der Ebenen
   )
```

Parameter

levels

[in] Die neue Anzahl der Ebenen eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Anzahl der Ebenen nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::LevelsCount
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get levels count of chart object
   int levels_count=object.LevelsCount();
   //--- set levels count of chart object
   object.LevelsCount(levels_count+1);
  }
```

© 2000-2025, MetaQuotes Ltd.

LevelColor (Get-Methode)

Erhält die Linienfarbe der angegebenen Ebene eines grafischen Objektes.

```
color  LevelColor(
   int  level      // Nummer der Ebene
   ) const
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

Rückgabewert

Die Linienfarbe der angegebenen Ebene eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat keine Ebene, wird CLR_NONE zurückgegeben.

LevelColor (Set-Methode)

Setzt die Linienfarbe der angegebenen Ebene eines grafischen Objektes.

```
bool  LevelColor(
   int    level,          // Nummer der Ebene
   color  new_color       // die neue Farbe
   )
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

new_color

[in] Der neue Wert der Linienfarbe der angegebenen Ebene eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::LevelColor
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get level color of chart object
      color level_color=object.LevelColor(i);
```

© 2000-2025, MetaQuotes Ltd.

```
      if(level_color!=clrRed)
        {
         //--- set level color of chart object
         object.LevelColor(i,clrRed);
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

LevelStyle (Get-Methode)

Erhält den Linienstil der angegebenen Ebene eines grafischen Objektes.

```
ENUM_LINE_STYLE  LevelStyle(
   int  level      // Nummer der Ebene
   ) const
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

Rückgabewert

Der Linienstil der angegebenen Ebene eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat keine Ebene, wird WRONG_VALUE zurückgegeben.

LevelStyle (Set-Methode)

Setzt den Linienstil der angegebenen Ebene eines grafischen Objektes.

```
int  LevelStyle(
   int               level,     // Nummer der Ebene
   ENUM_LINE_STYLE   style      // Linienstil
   )
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

style

[in] Der neue Wert des Linienstils der angegebenen Ebene eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Stil nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::LevelStyle
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get level style of chart object
      ENUM_LINE_STYLE level_style=object.LevelStyle(i);
```

© 2000-2025, MetaQuotes Ltd.

```
      if(level_style!=STYLE_SOLID)
        {
         //--- set level style of chart object
         object.LevelStyle(i,STYLE_SOLID);
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

LevelWidth (Get-Methode)

Erhält die Linienbreite der angegebenen Ebene eines grafischen Objektes.

```
int  LevelWidth(
   int  level      // Nummer der Ebene
   ) const
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

Rückgabewert

Die Linienbreite der angegebenen Ebene eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat keine Ebene, wird -1 zurückgegeben.

LevelWidth (Set-Methode)

Setzt die Linienbreite der angegebenen Ebene eines grafischen Objektes.

```
bool  LevelWidth(
   int  level,          // Nummer der Ebene
   int  new_width       // die neue Breite
   )
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

new_width

[in] Der neue Wert der Linienbreite der angegebenen Ebene eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Breite nicht geändert werden konnte

Beispiel:

```
//--- example for CChartObject::LevelWidth
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get level width of chart object
      int level_width=object.LevelWidth(i);
```

© 2000-2025, MetaQuotes Ltd.

```
      if(level_width!=1)
        {
         //--- set level width of chart object
         object.LevelWidth(i,1);
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

LevelValue (Get-Methode)

Erhält den Wert der angegebenen Ebene eines grafischen Objektes.

```
double  LevelValue(
   int  level      // Nummer der Ebene
   ) const
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

Rückgabewert

Der Wert der angegebenen Ebene eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat keine Ebene, wird EMPTY_VALUE zurückgegeben.

LevelValue (Set-Methode)

Setzt den Wert der angegebenen Ebene eines grafischen Objektes.

```
bool  LevelValue(
   int     level,          // Nummer der Ebene
   double  new_value       // der neue Wert
   )
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

new_value

[in] Der neue Wert der angegebenen Ebene eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Ebenen nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::LevelValue
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get level value of chart object
      double level_value=object.LevelValue(i);
```

© 2000-2025, MetaQuotes Ltd.

```
      if(level_value!=0.1*i)
        {
         //--- set level value of chart object
         object.LevelValue(i,0.1*i);
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

LevelDescription (Get-Methode)

Erhält die Beschreibung (den Text) der angegebenen Ebene eines grafischen Objektes.

```
string  LevelDescription(
   int  level       // Nummer der Ebene
   ) const
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

Rückgabewert

Die Beschreibung (der Text) der angegebenen Ebene eines grafischen Objektes, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, oder das Objekt hat keine Ebene, wird NULL zurückgegeben.

LevelDescription (Set-Methode)

Setzt die Beschreibung (den Text) der angegebenen Ebene eines grafischen Objektes.

```
bool  LevelDescription(
   int     level,      // Nummer der Ebene
   string  text        // Text
   )
```

Parameter

level

[in] Nummer der Ebene eines grafischen Objektes.

text

[in] Der neue Wert der Beschreibung (des Textes) der angegebenen Ebene eines grafischen Objektes.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Beschreibung (der Text) nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::LevelDescription
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get level description of chart object
```

© 2000-2025, MetaQuotes Ltd.

```
      string level_description=object.LevelDescription(i);
      if(level_description=="")
        {
         //--- set level description of chart object
         object.LevelDescription(i,"Level_"+IntegerToString(i));
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

GetInteger

Bietet einen einfachen Zugriff auf Funktionen API MQL5 ObjectGetInteger() zum Erhalten der integer-Eigenschaften (der Typen bool, char, uchar, short, ushort, int, uint, long, ulong, datetime, color) eines an die Klasseninstanz gebundenen Objektes. Es gibt zwei Versionen des Funktionsaufrufs:

Erhalten von Eigenschaftswert ohne Validierung

```
long  GetInteger(
   ENUM_OBJECT_PROPERTY_INTEGER  prop_id,          // Identifikator der integer-Eigens
   int                           modifier=-1       // Modifikator
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft eines grafischen Objektes.

modifier=-1

[in] Modifikator (Index) der double-Eigenschaft.

Rückgabewert

Ein Wert der integer-Eigenschaft beim Erfolg, 0 - wenn double-Eigenschaft nicht erhalten werden kann.

Erhalten von Eigenschaftswert mit Validierung

```
bool  GetInteger(
   ENUM_OBJECT_PROPERTY_INTEGER  prop_id,       // Identifikator der integer-Eigenschaf
   int                           modifier,      // Modifikator
   long&                         value          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer integer-Eigenschaft eines grafischen Objektes.

modifier

[in] Modifikator (Index) der integer-Eigenschaft.

value

[out] Die Referenz an die Variable um den Wert der integer-Eigenschaft zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die integer-Eigenschaft nicht erhalten werden konnte.

Beispiel:

```
//--- example for CChartObject::GetInteger
#include <ChartObjects\ChartObject.mqh>
//---
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   CChartObject object;
   //--- get color of chart object by easy method
   printf("Objects color is %s",ColorToString(object.GetInteger(OBJPROP_COLOR),true));
   //--- get color of chart object by classic method
   long color_value;
   if(!object.GetInteger(OBJPROP_COLOR,0,color_value))
     {
      printf("Get integer property error %d",GetLastError());
      return;
     }
   else
      printf("Objects color is %s",color_value);
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get levels width by easy method
      printf("Level %d width is %d",i,object.GetInteger(OBJPROP_LEVELWIDTH,i));
      //--- get levels width by classic method
      long width_value;
      if(!object.GetInteger(OBJPROP_LEVELWIDTH,i,width_value))
        {
         printf("Get integer property error %d",GetLastError());
         return;
        }
      else
         printf("Level %d width is %d",i,width_value);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SetInteger

Bietet einen einfachen Zugriff auf Funktionen API MQL5 ObjectSetInteger() für die Änderung der integer-Eigenschaften (der Typen bool, char, uchar, short, ushort, int, uint, long, ulong, datetime, color) eines an die Klasseninstanz gebundenen Objektes. Es gibt zwei Versionen des Funktionsaufrufs:

Setzt den Wert der Eigenschaft die keinen Modifikator erfordert

```
bool  SetInteger(
   ENUM_OBJECT_PROPERTY_INTEGER   prop_id,      // Identifikator der integer-Eigenschaf
   long                           value         // Wert
   )
```

Parameter

prop_id

[in] Die Identifikator einer integer-Eigenschaft eines grafischen Objektes.

value

[in] Der neue Wert der integer-Eigenschaft.

Setzt den Wert der Eigenschaft mit einem Modifikator

```
bool  SetInteger(
   ENUM_OBJECT_PROPERTY_INTEGER   prop_id,      // Identifikator der integer-Eigenscha
   int                            modifier,     // Modifikator
   long                           value         // Wert
   )
```

Parameter

prop_id

[in] Die Identifikator einer integer-Eigenschaft eines grafischen Objektes.

modifier

[in] Modifikator (Index) der integer-Eigenschaft.

value

[in] Der neue Wert der integer-Eigenschaft.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die integer-Eigenschaft nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::SetInteger
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- set new color of chart object
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!object.SetInteger(OBJPROP_COLOR,clrRed))
     {
      printf("Set integer property error %d",GetLastError());
      return;
     }
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- set levels width
      if(!object.SetInteger(OBJPROP_LEVELWIDTH,i,i))
        {
         printf("Set integer property error %d",GetLastError());
         return;
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

GetDouble

Bietet einen einfachen Zugriff auf Funktionen API MQL5 ObjectGetDouble() zum Erhalten der double-Eigenschaften (der Typen float und double) eines an die Klasseninstanz gebundenen Objektes. Es gibt zwei Versionen vom Funktionsaufruf:

Erhalten von Eigenschaftswert ohne Validierung

```
double  GetDouble(
   ENUM_OBJECT_PROPERTY_DOUBLE  prop_id,          // Identifikator der double-Eigenscha
   int                          modifier=-1      // Modifikator
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft eines grafischen Objektes.

modifier=-1

[in] Modifikator (Index) der double-Eigenschaft.

Rückgabewert

Ein Wert der double-Eigenschaft beim Erfolg, EMPTY_VALUE - wenn double-Eigenschaft nicht erhalten werden kann.

Erhalten von Eigenschaftswert mit Validierung

```
bool  GetDouble(
   ENUM_OBJECT_PROPERTY_DOUBLE  prop_id,      // Identifikator der double-Eigenschaft
   int                          modifier,     // Modifikator
   double&                      value         // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft eines grafischen Objektes.

modifier

[in] Modifikator (Index) der double-Eigenschaft.

value

[out] Die Referenz an die Variable um den Wert der double-Eigenschaft zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die double-Eigenschaft nicht erhalten werden konnte.

Beispiel:

```
//--- example for CChartObject::GetDouble
#include <ChartObjects\ChartObject.mqh>
//---
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   CChartObject object;
   //---
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get levels value by easy method
      printf("Level %d value=%f",i,object.GetDouble(OBJPROP_LEVELVALUE,i));
      //--- get levels value by classic method
      double value;
      if(!object.SetDouble(OBJPROP_LEVELVALUE,i,value))
        {
         printf("Get double property error %d",GetLastError());
         return;
        }
      else
         printf("Level %d value=%f",i,value);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SetDouble

Bietet einen einfachen Zugriff auf Funktionen API MQL5 ObjectSetDouble() für die Änderung der double-Eigenschaften (der Typen float und double) eines an die Klasseninstanz gebundenen Objektes. Es gibt zwei Versionen des Funktionsaufrufs:

Setzt den Wert der Eigenschaft die keinen Modifikator erfordert

```
bool  SetDouble(
   ENUM_OBJECT_PROPERTY_DOUBLE   prop_id,      // Identifikator der double-Eigenschaft
   double                        value         // Wert
   )
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft eines grafischen Objektes.

value

[in] Der neue Wert der double-Eigenschaft.

Setzt den Wert der Eigenschaft mit einem Modifikator

```
bool  SetDouble(
   ENUM_OBJECT_PROPERTY_DOUBLE   prop_id,      // Identifikator der double-Eigenschaft
   int                           modifier,     // Modifikator
   double                        value         // Wert
   )
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft eines grafischen Objektes.

modifier

[in] Modifikator (Index) der double-Eigenschaft.

value

[in] Der neue Wert der double-Eigenschaft.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die double-Eigenschaft nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::SetDouble
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //---
```

© 2000-2025, MetaQuotes Ltd.

```
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- set level value of chart object
      if(!object.SetDouble(OBJPROP_LEVELVALUE,i,0.1*i))
        {
         printf("Set double property error %d",GetLastError());
         return;
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

GetString

Bietet einen einfachen Zugriff auf Funktionen API MQL5 ObjectGetString() zum Erhalten der string-Eigenschaften eines an die Klasseninstanz gebundenen Objektes. Es gibt zwei Versionen vom Funktionsaufruf:

Erhalten von Eigenschaftswert ohne Validierung

```
string  GetString(
   ENUM_OBJECT_PROPERTY_STRING  prop_id,          // Identifikator der string-Eigenscha
   int                          modifier=-1      // Modifikator
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer string-Eigenschaft eines grafischen Objektes.

modifier=-1

[in] Modifikator (Index) der string-Eigenschaft.

Rückgabewert

Ein Wert der integer-Eigenschaft beim Erfolg, "" - wenn double-Eigenschaft nicht erhalten werden kann.

Erhalten von Eigenschaftswert mit Validierung

```
bool  GetString(
   ENUM_OBJECT_PROPERTY_STRING  prop_id,      // Identifikator der string-Eigenschaft
   int                          modifier,     // Modifikator
   string&                      value         // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer string-Eigenschaft eines grafischen Objektes.

modifier

[in] Modifikator (Index) der string-Eigenschaft.

value

[out] Die Referenz an die Variable um den Wert der string-Eigenschaft zu platzieren.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die string-Eigenschaft nicht erhalten werden konnte.

Beispiel:

```
//--- example for CChartObject::GetString
#include <ChartObjects\ChartObject.mqh>
//---
```

© 2000-2025, MetaQuotes Ltd.

```
void OnStart()
  {
   CChartObject object;
   string       value;
   //--- get name of chart object by easy method
   printf("Object name is '%s'",object.GetString(OBJPROP_NAME));
   //--- get name of chart object by classic method
   if(!object.GetString(OBJPROP_NAME,0,value))
     {
      printf("Get string property error %d",GetLastError());
      return;
     }
   else
      printf("Object name is '%s'",value);
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- get levels description by easy method
      printf("Level %d description is '%s'",i,object.GetString(OBJPROP_LEVELTEXT,i));
      //--- get levels description by classic method
      if(!object.GetString(OBJPROP_LEVELTEXT,i,value))
        {
         printf("Get string property error %d",GetLastError());
         return;
        }
      else
         printf("Level %d description is '%s'",i,value);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

SetString

Bietet einen einfachen Zugriff auf Funktionen API MQL5 ObjectSetString() für die Änderung der string-Eigenschaften eines an die Klasseninstanz gebundenen Objektes. Es gibt zwei Versionen vom Funktionsaufruf:

Setzt den Wert der Eigenschaft die keinen Modifikator erfordert

```
bool  SetString(
   ENUM_OBJECT_PROPERTY_STRING   prop_id,      // Identifikator der string-Eigenschaft
   string                        value         // Wert
   )
```

Parameter

prop_id

[in] Die Identifikator einer string-Eigenschaft eines grafischen Objektes.

value

[in] Der neue Wert der string-Eigenschaft.

Setzt den Wert der Eigenschaft mit einem Modifikator

```
bool  SetString(
   ENUM_OBJECT_PROPERTY_STRING   prop_id,       // Identifikator der string-Eigenschaft
   int                           modifier,      // Modifikator
   string                        value          // Wert
   )
```

Parameter

prop_id

[in] Die Identifikator einer string-Eigenschaft eines grafischen Objektes.

modifier

[in] Modifikator (Index) der string-Eigenschaft.

value

[in] Der neue Wert der string-Eigenschaft.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die string-Eigenschaft nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::SetString
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- set new name of chart object
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!object.SetString(OBJPROP_NAME,"MyObject"))
     {
      printf("Set string property error %d",GetLastError());
      return;
     }
   for(int i=0;i<object.LevelsCount();i++)
     {
      //--- set levels description
      if(!object.SetString(OBJPROP_LEVELTEXT,i,"Level_"+IntegerToString(i)))
        {
         printf("Set string property error %d",GetLastError());
         return;
        }
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CChartObject::Save
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   int          file_handle;
   CChartObject object=new CChartObject;
   //--- set object parameters
   //--- . . .
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!object.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CChartObject::Load
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   int           file_handle;
   CChartObject  object;
   //--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!object.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
   //--- use object
   //--- . . .
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObject - 0x8888).

Beispiel:

```
//--- example for CChartObject::Type
#include <ChartObjects\ChartObject.mqh>
//---
void OnStart()
  {
   CChartObject object;
   //--- get object type
   int type=object.Type();
  }
```

© 2000-2025, MetaQuotes Ltd.

Objekte "Linien"

Eine Gruppe von Grafikobjekten "Linien".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Linien" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klassenname	Objekt
CChartObjectVLine	Grafisches Objekt "vertikale Linie".
CChartObjectHLine	Grafisches Objekt "horizontale Linie".
CChartObjectTrend	Grafisches Objekt "Trendlinie"
CChartObjectTrendByAngle	Grafisches Objekt "Trendlinie mit Winkel".
CChartObjectCycles	Grafisches Objekt "Zykluslinien"

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectVLine

Klasse CChartObjectVLine ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "vertikale Linie".

Beschreibung

Klasse CChartObjectVLine bietet Zugriff auf die Eigenschaften des Objekts "vertikale Linie".

Deklaration

```
class CChartObjectVLine : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsLines.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectVLine

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "vertikale Linie".
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "vertikale Linie".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time           // Zeitkoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time

[in] Die Zeitkoordinate des Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectVLine - OBJ_VLINE).

© 2000-2025, MetaQuotes Ltd.

CChartObjectHLine

Klasse CChartObjectHLine ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "horizontale Linie".

Beschreibung

Klasse CChartObjectHLine bietet Zugriff auf die Eigenschaften des Objekts "horizontale Linie".

Deklaration

```
class CChartObjectHLine : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsLines.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectHLine

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "horizontale Linie"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "horizontale Linie".

```
bool  Create(
   long    chart_id,      // Chart ID
   string  name,          // Objektname
   long    window,        // Chartfenster
   double  price          // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

price

[in] Die Preiskoordinate des Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectHLine – OBJ_HLINE).

© 2000-2025, MetaQuotes Ltd.

CChartObjectTrend

Klasse CChartObjectTrend ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Trendlinie".

Beschreibung

Klasse CChartObjectTrend bietet Zugriff auf die Eigenschaften des Objekts "Trendlinie".

Deklaration

```
class CChartObjectTrend : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsLines.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend

Direkte Ableitungen

CChartObjectChannel, CChartObjectFibo, CChartObjectFiboChannel, CChartObjectFiboExpansion, CChartObjectGannFan, CChartObjectGannGrid, CChartObjectPitchfork, CChartObjectRegression, CChartObjectStdDevChannel, CChartObjectTrendByAngle

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Trendlinie"
Eigenschaften	
RayLeft	Erhält/setzt ein Eigenschaft "Strahl nach links"
RayRight	Erhält/setzt ein Eigenschaft "Strahl nach rechts"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Trendlinie".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

RayLeft (Get-Methode)

Erhält den Wert des Flags "Strahl nach links".

```
bool  RayLeft() const
```

Rückgabewert

Der wert des Flags "Strahl nach links" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

RayLeft (Set-Methode)

Setzt den Wert des Flags "Strahl nach links".

```
bool  RayLeft(
   bool  ray      // Flagwert
   )
```

Parameter

ray

[in] Der neue Wert des Flags "Strahl nach links".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RayRight (Get-Methode)

Erhält den Wert des Flags "Strahl nach rechts".

```
bool  RayRight() const
```

Rückgabewert

Der wert des Flags "Strahl nach rechts" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

RayRight (Set-Methode)

Setzt den Wert des Flags "Strahl nach rechts".

```
bool  RayRight(
   bool  ray      // Flagwert
   )
```

Parameter

ray

[in] Der neue Wert des Flags "Strahl nach links".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectTrend – OBJ_TREND).

© 2000-2025, MetaQuotes Ltd.

CChartObjectTrendByAngle

Klasse CChartObjectTrendByAngle ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Trendlinie mit Winkel".

Beschreibung

Klasse CChartObjectTrendByAngle bietet Zugriff auf die Eigenschaften des Objekts "Trendlinie mit Winkel".

Deklaration

```
class CChartObjectTrendByAngle : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsLines.mqh>
```

Vererbungshierarchie

CObject
CChartObject
CChartObjectTrend
CChartObjectTrendByAngle

Direkte Ableitungen

CChartObjectGannLine

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Trendlinie" mit Winkel"
Eigenschaften	
Angle	Erhält/setzt die Eigenschaft "Winkel"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Trendlinie mit Winkel".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   long      window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Angle (Get-Methode)

Erhält die Eigenschaft "Winkel".

```
double  Angle() const
```

Rückgabewert

Der Eigenschaftswert "Winkel" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

Angle (Set-Methode)

Setzt die Eigenschaft "Winkel".

```
bool  Angle(
   double  angle      // Winkel
   )
```

Parameter

angle

[in] Der neue Wert der Eigenschaft "Winkel".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectTrendByAngle - OBJ_TRENDBYANGLE).

© 2000-2025, MetaQuotes Ltd.

CChartObjectCycles

Klasse CChartObjectCycles ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Zykluslinien".

Beschreibung

Klasse CChartObjectCycles bietet Zugriff auf die Eigenschaften des Objekts "Zykluslinien".

Deklaration

```
class CChartObjectCycles : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsLines.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectCycles

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Zykluslinien"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Zykluslinien".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   long      window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectCycles - OBJ_CYCLES).

© 2000-2025, MetaQuotes Ltd.

Objekte "Kanäle"

Eine Gruppe von Grafikobjekten "Kanäle".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Kanäle" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klassenname	Objekt
CChartObjectChannel	Grafisches Objekt "Äquidistantkanal"
CChartObjectRegression	Grafisches Objekt "Linearer Regressionskanal"
CChartObjectStdDevChannel	Grafisches Objekt "Standardabweichung-Kanal"
CChartObjectPitchfork	Grafisches Objekt "Andrews Forke"

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectChannel

Klasse CChartObjectChannel ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Äquidistantkanal".

Beschreibung

Klasse CChartObjectChannel bietet Zugriff auf die Eigenschaften des Objekts "Äquidistantkanal".

Deklaration

```
class CChartObjectChannel : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsChannels.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectChannel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Äquidistantkanal"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Äquidistantkanal".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate des ersten Ankerpunktes
   double    price1,        // Preiskoordinate des ersten Ankerpunktes
   datetime  time2,         // Zeitkoordinate des zweiten Ankerpunktes
   double    price2,        // Preiskoordinate des zweiten Ankerpunktes
   datetime  time3,         // Zeitkoordinate des dritten Ankerpunktes
   double    price3         // Preiskoordinate des dritten Ankerpunktes
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectChannel - OBJ_CHANNEL).

© 2000-2025, MetaQuotes Ltd.

CChartObjectRegression

Klasse CChartObjectRegression ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Linearer Regressionskanal".

Beschreibung

Klasse CChartObjectRegression bietet Zugriff auf die Eigenschaften des Objekts "Linearer Regressionskanal".

Deklaration

```
class CChartObjectRegression : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsChannels.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectRegression

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Linearer Regressionskanal"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Linearer Regressionskanal".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   long      window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   datetime  time2          // Zeitkoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectRegression - OBJ_REGRESSION).

© 2000-2025, MetaQuotes Ltd.

CChartObjectStdDevChannel

Klasse CChartObjectStdDevChannel ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Standardabweichung-Kanal".

Beschreibung

Klasse CChartObjectStdDevChannel bietet Zugriff auf die Eigenschaften des Objekts "Standardabweichung-Kanal".

Deklaration

```
class CChartObjectStdDevChannel : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsChannels.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectStdDevChannel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Standardabweichung-Kanal"
Eigenschaften	
Deviations	Erhält/setzt die Eigenschaft "Abweichung"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Standardabweichung-Kanal".

```
bool  Create(
   long      chart_id,       // Chart ID
   string    name,           // Objektname
   int       window,         // Chartfenster
   datetime  time1,          // Zeitkoordinate
   datetime  time2,          // Zeitkoordinate
   double    deviation       // Abweichung
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

deviation

[in] Wert der Eigenschaft "Abweichung".

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Deviation (Get-Methode)

Erhält den Wert der Eigenschaft "Abweichung".

```
double  Deviation() const
```

Rückgabewert

Der Eigenschaftswert "Abweichung" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

Deviation (Set-Methode)

Setzt den Wert der "Abweichung".

```
bool  Deviation(
   double  deviation      // Abweichung
   )
```

Parameter

deviation

[in] Der neue Wert der Eigenschaft "Abweichung".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectStdDevChannel - OBJ_STDDEVCHANNEL).

© 2000-2025, MetaQuotes Ltd.

CChartObjectPitchfork

Klasse CChartObjectPitchfork ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Andrews Forke".

Beschreibung

Klasse CChartObjectPitchfork bietet Zugriff auf die Eigenschaften des Objekts "Andrews' Forke".

Deklaration

```
class CChartObjectPitchfork : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsChannels.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectPitchfork

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Andrews Forke"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Andrews Forke".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   long      window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate des ersten Ankerpunktes
   double    price1,        // Preiskoordinate des ersten Ankerpunktes
   datetime  time2,         // Zeitkoordinate des zweiten Ankerpunktes
   double    price2,        // Preiskoordinate des zweiten Ankerpunktes
   datetime  time3,         // Zeitkoordinate des dritten Ankerpunktes
   double    price3         // Preiskoordinate des dritten Ankerpunktes
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectPitchfork - OBJ_PITCHFORK).

© 2000-2025, MetaQuotes Ltd.

Gann-Werkzeuge

Eine Gruppe von Grafikobjekten "Gann-Werkzeuge".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Gann-Werkzeuge" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klassenname	Objekt
CChartObjectGannLine	Grafisches Objekt "Gann-Linie"
CChartObjectGannFan	Grafisches Objekt "Gann-Fächer"
CChartObjectGannGrid	Grafisches Objekt "Gann-Gitter"

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectGannLine

Klasse CChartObjectGannLine ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Gann-Linie".

Beschreibung

Klasse CChartObjectGannLine bietet Zugriff auf die Eigenschaften des Objekts "Gann-Linie".

Deklaration

```
class CChartObjectGannLine : public CChartObjectTrendByAngle
```

Kopf

```
#include <ChartObjects\ChartObjectsGann.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectTrendByAngle
        CChartObjectGannLine

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Gann-Linie"
Eigenschaften	
PipsPerBar	Erhält/setzt die Eigenschaft "Maßstab"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create

Methoden geerbt von der Klasse CChartObjectTrendByAngle

Angle, Angle, Create

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Gann-Linie".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    ppb            // Maßstab
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

ppb

[in] Maßstab

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

PipsPerBar (Get-Methode)

Erhält den Wert der Eigenschaft "Maßstab".

```
double  PipsPerBar() const
```

Rückgabewert

Der Eigenschaftswert "Maßstab" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

PipsPerBar (Set-Methode)

Setzt den Wert der Eigenschaft "Maßstab".

```
bool  PipsPerBar(
   double  ppb      // Skala
   )
```

Parameter

ppb

[in] Der neue Wert der Eigenschaft "Maßstab".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectGannLine - OBJ_GANNLINE).

© 2000-2025, MetaQuotes Ltd.

CChartObjectGannFan

Klasse CChartObjectGannFan ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Gann-Fächer".

Beschreibung

Klasse CChartObjectGannFan bietet Zugriff auf die Eigenschaften des Objekts "Gann-Fächer".

Deklaration

```
class CChartObjectGannFan : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsGann.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectGannFan

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Gann-Fächer"
Eigenschaften	
PipsPerBar	Erhält/setzt die Eigenschaft "Maßstab"
Downtrend	Erhält/setzt ein Eigenschaft "Trend unten"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Gann-Fächer".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    ppb            // Maßstab
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

ppb

[in] Maßstab

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

PipsPerBar (Get-Methode)

Erhält den Wert der Eigenschaft "Maßstab".

```
double  PipsPerBar() const
```

Rückgabewert

Der Eigenschaftswert "Maßstab" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

PipsPerBar (Set-Methode)

Setzt den Wert der Eigenschaft "Maßstab".

```
bool  PipsPerBar(
   double  ppb      // Skala
   )
```

Parameter

ppb

[in] Der neue Wert der Eigenschaft "Maßstab".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Downtrend (Get-Methode)

Erhält den Wert des Flags "Abwärtstrend".

```
bool  Downtrend() const
```

Rückgabewert

Der wert des Flags "Abwärtstrend" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Downtrend (Set-Methode)

Setzt den Wert des Flags "Abwärtstrend".

```
bool  Downtrend(
   bool  downtrend      // Flagwert
   )
```

Parameter

downtrend

[in] Der neue Wert des Flags "Abwärtstrend".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Daten des Listenelements in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectGannFan - OBJ_GANNFAN).

© 2000-2025, MetaQuotes Ltd.

CChartObjectGannGrid

Klasse CChartObjectGannGrid ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Gann-Gitter".

Beschreibung

Klasse CChartObjectGannGrid bietet Zugriff auf die Eigenschaften des Objekts "Gann-Gitter".

Deklaration

```
class CChartObjectGannGrid : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsGann.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectGannGrid

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Gann-Gitter"
Eigenschaften	
PipsPerBar	Erhält/setzt die Eigenschaft "Maßstab"
Downtrend	Erhält/setzt ein Eigenschaft "Trend unten"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Gann-Gitter".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    ppb            // Maßstab
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

ppb

[in] Maßstab

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

PipsPerBar (Get-Methode)

Erhält den Wert der Eigenschaft "Maßstab".

```
double  PipsPerBar() const
```

Rückgabewert

Der Eigenschaftswert "Maßstab" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

PipsPerBar (Set-Methode)

Setzt den Wert der Eigenschaft "Maßstab".

```
bool  PipsPerBar(
   double  ppb      // Skala
   )
```

Parameter

ppb

[in] Der neue Wert der Eigenschaft "Maßstab".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Downtrend (Get-Methode)

Erhält den Wert des Flags "Abwärtstrend".

```
bool  Downtrend() const
```

Rückgabewert

Der wert des Flags "Abwärtstrend" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Downtrend (Set-Methode)

Setzt den Wert des Flags "Abwärtstrend".

```
bool  Downtrend(
   bool  downtrend      // Flagwert
   )
```

Parameter

downtrend

[in] Der neue Wert des Flags "Abwärtstrend".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectGannGrid - OBJ_GANNGRID).

© 2000-2025, MetaQuotes Ltd.

Fibonacci-Werkzeuge

Eine Gruppe von Grafikobjekten "Fibonacci-Werkzeuge".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Fibonacci-Werkzeuge" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klassenname	Objekt
CChartObjectFibo	Grafisches Objekt "Fibonacci-Linien"
CChartObjectFiboTimes	Grafisches Objekt "Fibonacci-Zeitzonen"
CChartObjectFiboFan	Grafisches Objekt "Fibonacci-Fächer"
CChartObjectFiboArc	Grafisches Objekt "Fibonacci-Bogen"
CChartObjectFiboChannel	Grafisches Objekt "Fibonacci-Kanal"
CChartObjectFiboExpansion	Grafisches Objekt "Fibonacci-Erweiterung"

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectFibo

Klasse CChartObjectFibo ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Fibonacci-Linien".

Beschreibung

Klasse CChartObjectFibo bietet Zugriff auf die Eigenschaften des Objekts "Fibonacci-Linien".

Deklaration

```
class CChartObjectFibo : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsFibo.mqh>
```

Vererbungshierarchie

CObject
CChartObject
CChartObjectTrend
CChartObjectFibo

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Fibonacci-Linien"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fibonacci-Linien".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectFibo - OBJ_FIBO).

© 2000-2025, MetaQuotes Ltd.

CChartObjectFiboTimes

Klasse CChartObjectFiboExpansion ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Fibonacci-Zeitzonen".

Beschreibung

Klasse CChartObjectFiboTimes bietet Zugriff auf die Eigenschaften des Objekts "Fibonacci-Zeitzonen".

Deklaration

```
class CChartObjectFiboTimes : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsFibo.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectFiboTimes

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Fibonacci-Zeitzonen"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fibonacci-Zeitzonen".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectFiboTimes – OBJ_FIBOTIMES).

© 2000-2025, MetaQuotes Ltd.

CChartObjectFiboFan

Klasse CChartObjectFiboFan ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Fibonacci-Fächer".

Beschreibung

Klasse CChartObjectFiboFan bietet Zugriff auf die Eigenschaften des Objekts "Fibonacci-Fächer".

Deklaration

```
class CChartObjectFiboFan : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsFibo.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectFiboFan

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Fibonacci-Fächer"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fibonacci-Fächer".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectFiboFan - OBJ_FIBOFAN).

© 2000-2025, MetaQuotes Ltd.

CChartObjectFiboArc

Klasse CChartObjectFibo ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Fibonacci-Bogen".

Beschreibung

Klasse CChartObjectFiboArc bietet Zugriff auf die Eigenschaften des Objekts "Fibonacci-Bogen".

Deklaration

```
class CChartObjectFiboArc : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsFibo.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectFiboArc

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Fibonacci-Bogen"
Eigenschaften	
Scale	Erhält/setzt die Eigenschaft "Maßstab"
Ellipse	Erhält/setzt die Eigenschaft "Ellipse"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fibonacci-Bogen".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Preiskoordinate
   double    scale          // Maßstab
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

scale

[in] Skala.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Scale (Get-Methode)

Erhält den Wert der Eigenschaft "Maßstab".

```
double  Scale() const
```

Rückgabewert

Der Eigenschaftswert "Maßstab" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

Scale (Set-Methode)

Setzt den Wert der Eigenschaft "Maßstab".

```
bool  Scale(
   double  scale      // Maßstab
   )
```

Parameter

scale

[in] Der neue Wert der Eigenschaft "Maßstab".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Ellipse (Get-Methode)

Erhält den Wert des Flags "Ellipse".

```
bool  Ellipse() const
```

Rückgabewert

Der wert des Flags "Ellipse" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Ellipse (Set-Methode)

Setzt den Wert des Flags "Ellipse".

```
bool  Ellipse(
   bool  ellipse      // Flagwert
   )
```

Parameter

ellipse

[in] Der neue Wert des Flags "Ellipse".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectFiboArc - OBJ_FIBOARC).

© 2000-2025, MetaQuotes Ltd.

CChartObjectFiboChannel

Klasse CChartObjectFiboChannel ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Fibonacci-Kanal".

Beschreibung

Klasse CChartObjectFiboChannel bietet Zugriff auf die Eigenschaften des Objekts "Fibonacci-Kanal".

Deklaration

```
class CChartObjectFiboChannel : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsFibo.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectFiboChannel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Fibonacci-Kanal"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fibonacci-Kanal".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Preiskoordinate
   datetime  time3,         // Zeitkoordinate
   double    price3         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectFiboChannel - OBJ_FIBOCHANNEL).

© 2000-2025, MetaQuotes Ltd.

CChartObjectFiboExpansion

Klasse CChartObjectFiboExpansion ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Fibonacci-Erweiterung".

Beschreibung

Klasse CChartObjectFiboExpansion bietet Zugriff auf die Eigenschaften des Objekts "Fibonacci-Erweiterung".

Deklaration

```
class CChartObjectFiboExpansion : public CChartObjectTrend
```

Kopf

```
#include <ChartObjects\ChartObjectsFibo.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTrend
      CChartObjectFiboExpansion

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Fibonacci-Erweiterung"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectTrend

RayLeft, RayLeft, RayRight, RayRight, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fibonacci-Erweiterung".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Koordinate des 2. Preises
   datetime  time3,         // Zeitkoordinate
   double    price3         // Koordinate des 3. Preises
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des zweiten Ankerpunktes.

price3

[in] Preiskoordinate des ersten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectFiboExpansion - OBJ_EXPANSION).

© 2000-2025, MetaQuotes Ltd.

Elliott Werkzeuge

Eine Gruppe von Grafikobjekten "Elliott-Werkzeuge".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Elliott-Werkzeuge".

Klassenname	Objekt
CChartObjectElliottWave3	Grafisches Objekt "Korrektur-Welle"
CChartObjectElliottWave5	Grafisches Objekt "Impulswelle"

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectElliottWave3

Klasse CChartObjectElliottWave3 ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Korrektur-Welle".

Beschreibung

Klasse CChartObjectElliottWave3 bietet Zugriff auf die Eigenschaften des Objekts "Korrektur-Welle".

Deklaration

```
class CChartObjectElliottWave3 : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsElliott.mqh>
```

Vererbungshierarchie

CObject

CChartObject

CChartObjectElliottWave3

Direkte Ableitungen

CChartObjectElliottWave5

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Korrektur-Welle"
Eigenschaften	
Degree	Erhält/setzt die Eigenschaft "Grad"
Lines	Erhält/setzt ein Eigenschaft "Linienanzeige"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Direkte abgeleitete Klasse:

- CChartObjectElliottWave5

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Korrektur-Welle" (Elliott-3-Welle)

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Preiskoordinate
   datetime  time3,         // Zeitkoordinate
   double    price3         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Degree (Get-Methode)

Erhält den Wert der Eigenschaft "Methode der Wellen-Kennzeichnung"

```
ENUM_ELLIOT_WAVE_DEGREE  Degree() const
```

Rückgabewert

Der Eigenschaftswert "Methode der Wellen-Kennzeichnung" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Degree (Set-Methode)

Setzt den Wert der Eigenschaft "Methode der Wellen-Kennzeichnung"

```
bool  Degree(
   ENUM_ELLIOT_WAVE_DEGREE  degree      // Eigenschaftswert
   )
```

Parameter

degree

[in] Der neue Wert der Eigenschaft "Methode der Wellen-Kennzeichnung".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Lines (Get-Methode)

Erhält den Wert des Flags "Linienanzeige".

```
bool  Lines() const
```

Rückgabewert

Der wert des Flags "Linienanzeige" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

Lines (Set-Methode)

Setzt den Wert des Flags "Linienanzeige".

```
bool  Lines(
   bool  lines      // Flagwert
   )
```

Parameter

lines

[in] Der neue Wert des Flags "Linienanzeige".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Daten des Listenelements in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectElliottWave3 - OBJ_ELLIOTWAVE3).

© 2000-2025, MetaQuotes Ltd.

CChartObjectElliottWave5

Klasse CChartObjectElliottWave5 ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Impulswelle".

Beschreibung

Klasse CChartObjectElliottWave5 bietet Zugriff auf die Eigenschaften des Objekts "Impulswelle".

Deklaration

```
class CChartObjectElliottWave5 : public CChartObjectElliottWave3
```

Kopf

```
#include <ChartObjects\ChartObjectsElliott.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectElliottWave3
      CChartObjectElliottWave5

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Impulswelle"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectElliottWave3

Degree, Degree, Lines, Lines, Create, Save, Load

Sehen Sie auch

© 2000-2025, MetaQuotes Ltd.

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Impulswelle" (Eliott-5-Welle").

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Preiskoordinate
   datetime  time3,         // Zeitkoordinate
   double    price3,        // Preiskoordinate
   datetime  time4,         // Zeitkoordinate
   double    price4,        // Preiskoordinate
   datetime  time5          // Zeitkoordinate
   double    price5,        // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

time4

[in] Zeitkoordinate des vierten Ankerpunktes.

© 2000-2025, MetaQuotes Ltd.

price4

[in] Preiskoordinate des vierten Ankerpunktes.

time5

[in] Zeitkoordinate des fünften Ankerpunktes.

price5

[in] Preiskoordinate des fünften Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectElliottWave5 - OBJ_ELLIOTWAVE5).

© 2000-2025, MetaQuotes Ltd.

Objekte "Figuren"

Eine Gruppe von Grafikobjekten "Figuren".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Figuren" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klassenname	Objekt
CChartObjectRectangle	Grafisches Objekt "Rechteck"
CChartObjectTriangle	Grafisches Objekt "Dreieck"
CChartObjectEllipse	Grafisches Objekt "Ellipse"

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectRectangle

Klasse CChartObjectRectangle ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Rechteck".

Beschreibung

Klasse CChartObjectRectangle bietet Zugriff auf die Eigenschaften des Objekts "Rechteck".

Deklaration

```
class CChartObjectRectangle : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsShapes.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectRectangle

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Rechteck"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Rechteck".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   long      window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectRectangle - OBJ_RECTANGLE).

© 2000-2025, MetaQuotes Ltd.

CChartObjectTriangle

Klasse CChartObjectTriangle ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Dreieck".

Beschreibung

Klasse CChartObjectTriangle bietet Zugriff auf die Eigenschaften des Objekts "Dreieck".

Deklaration

```
class CChartObjectTriangle : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsShapes.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectTriangle

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Dreieck"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Dreieck".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   long      window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Preiskoordinate
   datetime  time3,         // Zeitkoordinate
   double    price3         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectTriangle - OBJ_TRIANGLE).

© 2000-2025, MetaQuotes Ltd.

CChartObjectEllipse

Klasse CChartObjectEllipse ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Ellipse".

Beschreibung

Klasse CChartObjectEllipse bietet Zugriff auf die Eigenschaften des Objekts "Ellipse".

Deklaration

```
class CChartObjectEllipse : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsShapes.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectEllipse

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Ellipse"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint, Save, Load

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Ellipse".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time1,         // Zeitkoordinate
   double    price1,        // Preiskoordinate
   datetime  time2,         // Zeitkoordinate
   double    price2,        // Preiskoordinate
   datetime  time3,         // Zeitkoordinate
   double    price3         // Preiskoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time1

[in] Zeitkoordinate des ersten Ankerpunktes.

price1

[in] Preiskoordinate des ersten Ankerpunktes.

time2

[in] Zeitkoordinate des zweiten Ankerpunktes.

price2

[in] Preiskoordinate des zweiten Ankerpunktes.

time3

[in] Zeitkoordinate des dritten Ankerpunktes.

price3

[in] Preiskoordinate des dritten Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectEllipse - OBJ_ELLIPSE).

© 2000-2025, MetaQuotes Ltd.

Objekte "Pfeile"

Eine Gruppe von Grafikobjekten "Pfeile".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Pfeile" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5. Ein Pfeile ist im Wesentlichen ein dem Benutzer angezeigtes Symbol, dem ein spezifischer Code gegenübergestellt wird. Es gibt zwei Arten von Grafikobjekten "Pfeile" für die Anzeige der Symbole auf dem Chart:

- Objekt "Pfeil" für Angabe des Codes eines durch das Objekt angezeigten Symbols.
- Objektengruppe für die Anzeige eines bestimmten Symbols (das einem bestimmten Festcode entspricht).

Klasse für die Arbeit mit die Pfeile, die Symbole mit dem beliebigen Code anzeigen

Klassenname	Pfeilobjektname
CChartObjectArrow	"Pfeil"

Klassen für die Arbeit mit die Pfeile, die Symbole mit dem Festcode anzeigen

Klassenname	Pfeilobjektname
CChartObjectArrowCheck	"Prüfung" ("Häkchen")
CChartObjectArrowDown	"Pfeil nach unten"
CChartObjectArrowUp	"Pfeil nach oben"
CChartObjectArrowStop	"Stopp"
CChartObjectArrowThumbUp	"Daumen hoch"
CChartObjectArrowThumbDown	"Daumen runter"
CChartObjectArrowLeftPrice	Linkes Preisschild
CChartObjectArrowRightPrice	Rechtes Preisschild

Sehen Sie auch

Objekttypen, Objektbindungsmethoden, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectArrow

Klasse CChartObjectArrow ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften von grafischen Objekten "Pfeile".

Beschreibung

Klasse CChartObjectArrow bietet ihren abgeleiteten Klassen den Zugriff auf die allgemeine Eigenschaften von Objekten "Pfeile".

Deklaration

```
class CChartObjectArrow : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsArrows.mqh>
```

Vererbungshierarchie

CObject

CChartObject

CChartObjectArrow

Direkte Ableitungen

CChartObjectArrowCheck, CChartObjectArrowDown, CChartObjectArrowLeftPrice, CChartObjectArrowRightPrice, CChartObjectArrowStop, CChartObjectArrowThumbDown, CChartObjectArrowThumbUp, CChartObjectArrowUp

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Pfeil"
Eigenschaften	
ArrowCode	Erhält/setzt den Text den Zeichencode
Anchor	Erhält/setzt den Bindungstyp
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Sehen Sie auch

Objekttypen, Objektbindungsmethoden, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Pfeil".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time,          // Zeit
   double    price,         // Preis
   char      code           // Code
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time

[in] Zeitkoordinate.

price

[in] Preiskoordinate.

code

[in] Pfeilcode (Wingdings).

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CChartObjectArrow::Create
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
   CChartObjectArrow arrow;
//--- set object parameters
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(!arrow.Create(0,"Arrow",0,TimeCurrent(),price,181))
     {
      //--- arrow create error
      printf("Arrow create: Error %d!",GetLastError());
      //---
```

© 2000-2025, MetaQuotes Ltd.

```
      return;
     }
//--- use arrow
//--- . . .
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrowCode (Get-Methode)

Erhält den Zeichencode "Pfeil".

```
char  ArrowCode() const
```

Rückgabewert

Zeichencode "Pfeil" des Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

ArrowCode (Set-Methode)

Setzt den Zeichencode "Pfeil".

```
bool  ArrowCode(
   char  code      // Codewert
   )
```

Parameter

code

[in] Der neue Codewert für "Pfeil" (Wingdings).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Code nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObjectArrow::ArrowCode
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
   CChartObjectArrow arrow;
   char              code=181;
//--- set object parameters
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(!arrow.Create(0,"Arrow",0,TimeCurrent(),price,code))
     {
      //--- arrow create error
      printf("Arrow create: Error %d!",GetLastError());
      //---
      return;
     }
//--- use arrow to possible changes code
//--- . . .
//--- get code of arrow
   if(arrow.ArrowCode()!=code)
     {
     //--- set code of arrow
```

© 2000-2025, MetaQuotes Ltd.

```
      arrow.ArrowCode(code);
      }
//--- use arrow
//--- . . .
   }
```

© 2000-2025, MetaQuotes Ltd.

Anchor (Get-Methode)

Erhält die Methode der Bindung eines Objektes "Pfeil" an einen Chart.

```
ENUM_ARROW_ANCHOR  Anchor() const
```

Rückgabewert

Bindungsmethode des Objekts "Pfeil", das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Anchor (Set-Methode)

Setzt die Methode der Bindung eines Objektes "Pfeil" an einen Chart.

```
bool  Anchor(
   ENUM_ARROW_ANCHOR  anchor      // Typ der Bindung
   )
```

Parameter

anchor

[in] Der neue Wert der Bindungsmethode.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Bindungsmethode nicht geändert werden konnte.

Beispiel:

```
//--- example for CChartObject::Anchor
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
   CChartObjectArrow arrow;
   ENUM_ARROW_ANCHOR anchor=ANCHOR_BOTTOM;
//--- set object parameters
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(!arrow.Create(0,"Arrow",0,TimeCurrent(),price,181))
     {
      //--- arrow create error
      printf("Arrow create: Error %d!",GetLastError());
      //---
      return;
     }
//--- get anchor of arrow
   if(arrow.Anchor()!=anchor)
     {
     //--- set anchor of arrow
     arrow.Anchor(anchor);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- use arrow
//--- . . .
  }
```

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CChartObjectArrow::Save
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
   int               file_handle;
   CChartObjectArrow arrow;
//--- set object parameters
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(!arrow.Create(0,"Arrow",0,TimeCurrent(),price,181))
     {
      //--- arrow create error
      printf("Arrow create: Error %d!",GetLastError());
      //---
      return;
     }
//--- open file
   file_handle=FileOpen("MyFile.bin",FILE_WRITE|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!arrow.Save(file_handle))
        {
         //--- file save error
         printf("File save: Error %d!",GetLastError());
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
  }
```

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CChartObjectArrow::Load
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
   int               file_handle;
   CChartObjectArrow arrow;
//--- open file
   file_handle=FileOpen("MyFile.bin",FILE_READ|FILE_BIN|FILE_ANSI);
   if(file_handle>=0)
     {
      if(!arrow.Load(file_handle))
        {
         //--- file load error
         printf("File load: Error %d!",GetLastError());
         FileClose(file_handle);
         //---
         return;
        }
      FileClose(file_handle);
     }
//--- use arrow
//--- . . .
  }
```

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectArrow - OBJ_ARROW).

Beispiel:

```
//--- example for CChartObjectArrow::Type
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
   CChartObjectArrow arrow;
   //--- get arrow type
   int type=arrow.Type();
  }
```

© 2000-2025, MetaQuotes Ltd.

Fest codierte Pfeile

Klassen "Fest codierte Pfeile" sind die Klassen für vereinfachten Zugriff auf die Eigenschaften von grafischen Objekten:

Klassenname	Pfeilobjektname
CChartObjectArrowCheck	"Prüfung" ("Häkchen")
CChartObjectArrowDown	"Pfeil nach unten"
CChartObjectArrowUp	"Pfeil nach oben"
CChartObjectArrowStop	"Stopp"
CChartObjectArrowThumbUp	"Daumen hoch"
CChartObjectArrowThumbDown	"Daumen runter"
CChartObjectArrowLeftPrice	Linkes Preisschild
CChartObjectArrowRightPrice	Rechtes Preisschild

Beschreibung

Klassen "Fest codierte Pfeile" ermöglichen Zugriff auf die Eigenschaften der entsprechenden Objekten.

Deklarationen

```
class CChartObjectArrowCheck      : public CChartObjectArrow;
class CChartObjectArrowDown       : public CChartObjectArrow;
class CChartObjectArrowUp         : public CChartObjectArrow;
class CChartObjectArrowStop       : public CChartObjectArrow;
class CChartObjectArrowThumbDown  : public CChartObjectArrow;
class CChartObjectArrowThumbUp    : public CChartObjectArrow;
class CChartObjectArrowLeftPrice  : public CChartObjectArrow;
class CChartObjectArrowRightPrice : public CChartObjectArrow;
```

Kopf

```
#include <ChartObjects\ChartObjectsArrows.mqh>
```

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt, das der Klasse entspricht
Eigenschaften	
ArrowCode	"Stub" für die Zeichencodeänderungmethode
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Sehen Sie auch

Objekttypen, Objektbindungsmethoden, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Fest codierter Pfeil"

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time,          // Zeit
   double    price          // Preis
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time

[in] Zeitkoordinate.

price

[in] Preiskoordinate.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Beispiel:

```
//--- example for CChartObjectArrowCheck::Create
//--- example for CChartObjectArrowDown::Create
//--- example for CChartObjectArrowUp::Create
//--- example for CChartObjectArrowStop::Create
//--- example for CChartObjectArrowThumbDown::Create
//--- example for CChartObjectArrowThumbUp::Create
//--- example for CChartObjectArrowLeftPrice::Create
//--- example for CChartObjectArrowRightPrice::Create
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
//--- for example, take CChartObjectArrowCheck
   CChartObjectArrowCheck arrow;
//--- set object parameters
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(!arrow.Create(0,"ArrowCheck",0,TimeCurrent(),price))
```

© 2000-2025, MetaQuotes Ltd.

```
      {
       //--- arrow create error
       printf("Arrow create: Error %d!",GetLastError());
       //---
       return;
      }
//--- use arrow
//--- . . .
  }
```

© 2000-2025, MetaQuotes Ltd.

ArrowCode

Verbietet die Änderung der Zeichencode "Pfeil".

```
bool  ArrowCode(
   char  code      // Codewert
   )
```

Parameter

code

[in] Jeder Wert.

Rückgabewert

Immer false.

Beispiel:

```
//--- example for CChartObjectArrowCheck::ArrowCode
//--- example for CChartObjectArrowDown::ArrowCode
//--- example for CChartObjectArrowUp::ArrowCode
//--- example for CChartObjectArrowStop::ArrowCode
//--- example for CChartObjectArrowThumbDown::ArrowCode
//--- example for CChartObjectArrowThumbUp::ArrowCode
//--- example for CChartObjectArrowLeftPrice::ArrowCode
//--- example for CChartObjectArrowRightPrice::ArrowCode
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
//--- for example, take CChartObjectArrowCheck
   CChartObjectArrowCheck arrow;
//--- set object parameters
   double price=SymbolInfoDouble(Symbol(),SYMBOL_BID);
   if(!arrow.Create(0,"ArrowCheck",0,TimeCurrent(),price))
     {
      //--- arrow create error
      printf("Arrow create: Error %d!",GetLastError());
      //---
      return;
     }
//--- set code of arrow
   if(!arrow.ArrowCode(181))
     {
      //--- it is not error
      printf("Arrow code can not be changed");
     }
//--- use arrow
//--- . . .
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Die Identifikator des Objekttyps:
CChartObjectArrowCheck - OBJ_ARROW_CHECK,
CChartObjectArrowDown - OBJ_ARROW_DOWN,
CChartObjectArrowUp - OBJ_ARROW_UP,
CChartObjectArrowStop - OBJ_ARROW_STOP,
CChartObjectArrowThumbDown - OBJ_ARROW_THUMB_DOWN,
CChartObjectArrowThumbUp - OBJ_ARROW_THUMB_UP,
CChartObjectArrowLeftPrice - OBJ_ARROW_LEFT_PRICE,
CChartObjectArrowRightPrice - OBJ_ARROW_RIGHT_PRICE.

Beispiel:

```
//--- example for CChartObjectArrowCheck::Type
//--- example for CChartObjectArrowDown::Type
//--- example for CChartObjectArrowUp::Type
//--- example for CChartObjectArrowStop::Type
//--- example for CChartObjectArrowThumbDown::Type
//--- example for CChartObjectArrowThumbUp::Type
//--- example for CChartObjectArrowLeftPrice::Type
//--- example for CChartObjectArrowRightPrice::Type
#include <ChartObjects\ChartObjectsArrows.mqh>
//---
void OnStart()
  {
//--- for example, take CChartObjectArrowCheck
   CChartObjectArrowCheck arrow;
//--- get arrow type
   int type=arrow.Type();
  }
```

© 2000-2025, MetaQuotes Ltd.

Steuerelemente

Eine Gruppe von Grafikobjekten "Steuerelemente".

Dieser Abschnitt enthält die technischen Details der Arbeit mit einer Gruppe von Klassen von Grafikobjekten "Steuerelemente" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klassenname	Objekt
CChartObjectText	Grafisches Objekt "Text"
CChartObjectLabel	Grafisches Objekt "Text-Label"
CChartObjectEdit	Grafisches Objekt "Eingabefeld"
CChartObjectButton	Grafisches Objekt "Taste"
CChartObjectSubChart	Grafisches Objekt "Chart"
CChartObjectBitmap	Grafisches Objekt "Bitmap"
CChartObjectBmpLabel	Grafisches Objekt "Bitmap Label"
CChartObjectRectLabel	Grafisches Objekt "Rechteck-Label".

Sehen Sie auch

Objekttypen, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

CChartObjectText

Klasse CChartObjectText ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Text".

Beschreibung

Klasse CChartObjectText bietet Zugriff auf die Eigenschaften des Objekts "Text".

Deklaration

```
class CChartObjectText : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsTxtControls.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectText

Direkte Ableitungen

CChartObjectLabel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Text"
Eigenschaften	
Angle	Erhält/setzt die Eigenschaft "Winkel"
Font	Erhält/setzt die Eigenschaft "Schrift"
FontSize	Erhält/setzt die Eigenschaft "Schriftgröße"
Anchor	Erhält/setzt die Eigenschaft "Ankerpunkt"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Direkte abgeleitete Klasse:

- CChartObjectLabel

Sehen Sie auch

Objekttypen, Objekteigenschaften, Objektbindung-Methoden, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Text".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time,          // Zeitkoordinate
   double    price          // Zeitkoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time

[in] Die Zeitkoordinate des Ankerpunktes.

price

[in] Die Preiskoordinate des Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Angle (Get-Methode)

Erhält die Eigenschaft "Neigungswinkel".

```
double  Angle() const
```

Rückgabewert

Der Eigenschaftswert "Neigungswinkel" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

Angle (Set-Methode)

Setzt die Eigenschaft "Neigungswinkel".

```
bool  Angle(
   double  angle      // Eigenschaftswert
   )
```

Parameter

angle

[in] Der neue Wert der Eigenschaft "Neigungswinkel".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Font (Get-Methode)

Erhält den Wert der Eigenschaft "Schrift".

```
string  Font() const
```

Rückgabewert

Der Eigenschaftswert "Schrift" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird "" zurückgegeben.

Font (Set-Methode)

Setzt den Wert der Eigenschaft "Schrift".

```
bool  Font(
   string  font      // Eigenschaftswert
   )
```

Parameter

font

[in] Der neue Wert der Eigenschaft "Schrift".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

FontSize (Get-Methode)

Erhält den Wert der Eigenschaft "Schriftgröße".

```
int  FontSize() const
```

Rückgabewert

Der Eigenschaftswert "Schriftgröße" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

FontSize (Set-Methode)

Setzt den Wert der Eigenschaft "Schriftgröße".

```
bool  FontSize(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Schriftgröße".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Anchor (Get-Methode)

Erhält die Eigenschaft "Position des Ankerpunktes".

```
ENUM_ANCHOR_POINT  Anchor() const
```

Rückgabewert

Der Eigenschaftswert "Position des Ankerpunktes" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Anchor (Set-Methode)

Setzt die Eigenschaft "Position des Ankerpunktes".

```
bool  Anchor(
   ENUM_ANCHOR_POINT  anchor      // Eigenschaftswert
   )
```

Parameter

anchor

[in] Der neue Wert der Eigenschaft "Position des Ankerpunktes".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectText – OBJ_TEXT).

© 2000-2025, MetaQuotes Ltd.

CChartObjectLabel

Klasse CChartObjectLabel ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Text-Label".

Beschreibung

Klasse CChartObjectLabel bietet Zugriff auf die Eigenschaften des Objekts "Text-Label".

Deklaration

```
class CChartObjectLabel : public CChartObjectText
```

Kopf

```
#include <ChartObjects\ChartObjectsTxtControls.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectText
      CChartObjectLabel

Direkte Ableitungen

CChartObjectEdit, CChartObjectRectLabel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Text-Label"
Eigenschaften	
X_Distance	Erhält/setzt die Eigenschaft "X-Koordinate"
Y_Distance	Erhält/setzt die Eigenschaft "Y-Koordinate"
X_Size	Erhält die Eigenschaft "Größe entlang der X-Achse"
Y_Size	Erhält die Eigenschaft "Größe entlang der Y-Achse"
Corner	Erhält/setzt die Eigenschaft "Chartecke zur Bindung"
Time	"Stub" für Zeitkoordinateänderung
Price	"Stub" für Preiskoordinateänderung
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei

© 2000-2025, MetaQuotes Ltd.

Erstellung	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectText

Angle, Angle, Font, Font, FontSize, FontSize, Anchor, Anchor, Create

Sehen Sie auch

Objekttypen, Objekteigenschaften, Objektbindung-Methoden, Bindung-Ecke, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Text-Label".

```
bool  Create(
   long     chart_id,      // Chart ID
   string   name,          // Objektname
   int      window,        // Chartfenster
   int      X,             // X-Koordinate
   int      Y             // Y-Koordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

X

[in] Der Abstand entlang der Achse X

Y

[in] Der Abstand entlang der Achse Y

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

X_Distance (Get-Methode)

Erhält den Wert der Eigenschaft "Abstand entlang der Achse X".

```
int  X_Distance() const
```

Rückgabewert

Der Eigenschaftswert "Abstand entlang der Achse X" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

X_Distance (Set-Methode)

Setzt den Wert der Eigenschaft "Abstand entlang der Achse X".

```
bool  X_Distance(
   int  X      // Eigenschaftswert
   )
```

Parameter

X

[in] Der neue Wert der Eigenschaft "Abstand entlang der Achse X".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Distance (Get-Methode)

Erhält den Wert der Eigenschaft "Abstand entlang der Achse Y".

```
int  Y_Distance() const
```

Rückgabewert

Der Eigenschaftswert "Abstand entlang der Achse Y" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Y_Distance (Set-Methode)

Setzt den Wert der Eigenschaft "Abstand entlang der Achse Y".

```
bool  Y_Distance(
   int  Y      // Eigenschaftswert
   )
```

Parameter

Y

[in] Der neue Wert der Eigenschaft "Abstand entlang der Achse Y".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

X_Size

Erhält den Wert der Eigenschaft "Größe entlang der Achse X".

```
int  X_Size() const
```

Rückgabewert

Der Eigenschaftswert "Größe entlang der Achse X" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Y_Size

Erhält den Wert der Eigenschaft "Größe entlang der Achse Y".

```
int  Y_Size() const
```

Rückgabewert

Der Eigenschaftswert "Größe entlang der Achse Y" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Corner (Get-Methode)

Erhält die Eigenschaft "Chartecke zur Bindung".

```
ENUM_BASE_CORNER  Corner() const
```

Rückgabewert

Der Eigenschaftswert "Chartecke zur Bindung" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Corner (Set-Methode)

Setzt die Eigenschaft "Chartecke zur Bindung".

```
bool  Corner(
   ENUM_BASE_CORNER  corner      // Eigenschaftswert
   )
```

Parameter

corner

[in] Der neue Wert der Eigenschaft "Chartecke zur Bindung".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Time

Verbietet Änderung der Zeitkoordinaten.

```
bool  Time(
   datetime  time      // beliebiger Wert
   )
```

Parameter

time

[in] Beliebiger datetime-Wert.

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Price

Verbietet Änderung der Preiskoordinaten.

```
bool  Price(
   double  price      // beliebiger Wert
   )
```

Parameter

price

[in] Beliebiger double-Wert.

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectLabel - OBJ_LABEL).

© 2000-2025, MetaQuotes Ltd.

CChartObjectEdit

Klasse CChartObjectEdit ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Eingabefeld".

Beschreibung

Klasse CChartObjectEdit bietet Zugriff auf die Eigenschaften des Objekts "Eingabefeld".

Deklaration

```
class CChartObjectEdit : public CChartObjectLabel
```

Kopf

```
#include <ChartObjects\ChartObjectsTxtControls.mqh>
```

Vererbungshierarchie

CObject
CChartObject
CChartObjectText
CChartObjectLabel
CChartObjectEdit

Direkte Ableitungen

CChartObjectButton

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Objekt "Eingabefeld"
Eigenschaften	
TextAlign	Erhält/setzt die Eigenschaft "Typ der Textausrichtung"
X_Size	Erhält die Eigenschaft "Größe entlang der X-Achse"
Y_Size	Erhält die Eigenschaft "Größe entlang der Y-Achse"
BackColor	Erhält/setzt die Eigenschaft "Hintergrundfarbe"
BorderColor	Erhält/setzt die Eigenschaft "Rahmenfarbe"
ReadOnly	Erhält/setzt die Eigenschaft "Read Only"
Angle	"Stub" für Änderung der "Winkel"-Eigenschaft
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei

© 2000-2025, MetaQuotes Ltd.

Erstellung	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectText

Angle, Angle, Font, Font, FontSize, FontSize, Anchor, Anchor, Create

Methoden geerbt von der Klasse CChartObjectLabel

X_Distance, X_Distance, Y_Distance, Y_Distance, X_Size, Y_Size, Corner, Corner, Time, Price, Create

Sehen Sie auch

Objekttypen, Objekteigenschaften, Objektbindung-Methoden, Bindung-Ecke, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Objekt "Eingabefeld".

```
bool  Create(
   long    chart_id,      // Chart ID
   string  name,          // Objektname
   int     window,        // Chartfenster
   int     X,             // X-Koordinate
   int     Y,             // Y-Koordinate
   int     sizeX,         // X-Größe
   int     sizeY          // Y-Größe
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

X

[in] Der Abstand entlang der Achse X

Y

[in] Der Abstand entlang der Achse Y

sizeX

[in] Der Größe entlang der Achse X

sizeY

[in] Der Größe entlang der Achse Y

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

TextAlign (Get-Methode)

Erhält den wert der Eigenschaft "Typ der horizontalen Textausrichtung" des Objekts "Eingabefeld".

```
ENUM_ALIGN_MODE  TextAlign() const
```

Rückgabewert

Ein Wert der Eigenschaft "Typ der horizontalen Textausrichtung" eines Objekts, das an eine Klasseninstanz gebunden ist.

TextAlign (Set-Methode)

Setzt den wert der Eigenschaft "Typ der horizontalen Textausrichtung" des Objekts "Eingabefeld".

```
bool  TextAlign(
   ENUM_ALIGN_MODE  align      // Eigenschaftswert
   )
```

Parameter

align

[in] Der neue Wert der Eigenschaft "Typ der horizontalen Textausrichtung".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

X_Size

Setzt den Wert der Eigenschaft "Größe entlang der Achse X".

```
bool  X_Size(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Größe entlang der Achse X".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Size

Setzt den Wert der Eigenschaft "Größe entlang der Achse Y".

```
bool  Y_Size(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Größe entlang der Achse Y".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

BackColor (Get-Methode)

Erhält den Wert der Eigenschaft "Hintergrundfarbe".

```
color  BackColor() const
```

Rückgabewert

Der Eigenschaftswert "Hintergrundfarbe" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird CLR_NONE zurückgegeben.

BackColor (Set-Methode)

Setzt den Wert der Eigenschaft "Hintergrundfarbe".

```
bool  BackColor(
   color  new_color      // Eigenschaftswert
   )
```

Parameter

new_color

[in] Der neue Wert der Eigenschaft "Hintergrundfarbe".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

BorderColor (Get-Methode)

Erhält den Wert der Eigenschaft "Rahmenfarbe".

```
color  BorderColor() const
```

Rückgabewert

Der Eigenschaftswert "Rahmenfarbe" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird CLR_NONE zurückgegeben.

BorderColor (Set-Methode)

Setzt den Wert der Eigenschaft "Rahmenfarbe".

```
bool  BorderColor(
   color  new_color      // der neue Eigenschaftswert
   )
```

Parameter

new_color

[in] Der neue Wert der Eigenschaft "Rahmenfarbe".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ReadOnly (Get-Methode)

Erhält den Wert der Eigenschaft "Read Only".

```
bool  ReadOnly() const
```

Rückgabewert

Der Wert der "Read Only"-Eigenschaft eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

ReadOnly (Set-Methode)

Setzt den Wert der Eigenschaft "Read Only".

```
bool  ReadOnly(
   const  bool  flag      // der neue Eigenschaftswert
   )
```

Parameter

flag

[in] Der neue Wert der "Read Only"-Eigenschaft (true bedeutet dass Textbearbeitungsfunktionen verboten sind"<t2>.</t2>

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Angle

Verbietet die Änderung der Eigenschaft "Neigungswinkel".

```
bool  Angle(
   double  angle      // beliebiger Wert
   )
```

Parameter

angle

[in] beliebiger double-Wert.

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectEdit - OBJ_EDIT).

© 2000-2025, MetaQuotes Ltd.

CChartObjectButton

Klasse CChartObjectButton ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Taste".

Beschreibung

Klasse CChartObjectButton bietet Zugriff auf die Eigenschaften des Objekts "Taste".

Deklaration

```
class CChartObjectButton : public CChartObjectEdit
```

Kopf

```
#include <ChartObjects\ChartObjectsTxtControls.mqh>
```

Vererbungshierarchie

CObject
CChartObject
CChartObjectText
CChartObjectLabel
CChartObjectEdit
CChartObjectButton

Direkte Ableitungen

CChartObjectPanel

Gruppen der Klassenmethode

Erstellung	
Create	Geerbt von der Klasse CChartObjectEdit
Eigenschaften	
State	Erhält/setzt die Tastenzustand (gedrückt/nicht gedrückt)
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectText

Angle, Angle, Font, Font, FontSize, FontSize, Anchor, Anchor, Create

Methoden geerbt von der Klasse CChartObjectLabel

X_Distance, X_Distance, Y_Distance, Y_Distance, X_Size, Y_Size, Corner, Corner, Time, Price, Create

Methoden geerbt von der Klasse CChartObjectEdit

X_Size, Y_Size, BackColor, BackColor, BorderColor, BorderColor, ReadOnly, ReadOnly, TextAlign, TextAlign, Angle, Create

Sehen Sie auch

Objekttypen, Objekteigenschaften, Objektbindung-Methoden, Bindung-Ecke, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

State (Get-Methode)

Erhält den Wert der Eigenschaft "Zustand".

```
bool  State() const
```

Rückgabewert

Der Eigenschaftswert "Zustand" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

State (Set-Methode)

Setzt den Wert der Eigenschaft "Zustand".

```
bool  State(
   bool  state      // Eigenschaftswert
   )
```

Parameter

state

[in] Der neue Wert der Eigenschaft "Zustand".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectButton - OBJ_BUTTON).

© 2000-2025, MetaQuotes Ltd.

CChartObjectSubChart

Klasse CChartObjectSubChart ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Chart".

Beschreibung

Klasse CChartObjectSubChart bietet Zugriff auf die Eigenschaften des Objekts "Chart".

Deklaration

```
class CChartObjectSubChart : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectSubChart.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectSubChart

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Chart"
Eigenschaften	
X_Distance	Erhält/setzt die Eigenschaft "X-Koordinate"
Y_Distance	Erhält/setzt die Eigenschaft "Y-Koordinate"
Corner	Erhält/setzt die Eigenschaft "Chartecke zur Bindung"
X_Size	Erhält/setzt die Eigenschaft "Größe entlang der X-Achse"
Y_Size	Erhält/setzt die Eigenschaft "Größe entlang der Y-Achse"
Symbol	Erhält/setzt die Eigenschaft "Symbol"
Period	Erhält/setzt die Eigenschaft "Periode"
Scale	Erhält/setzt die Eigenschaft "Maßstab"
DateScale	Erhält/setzt die Eigenschaft "Zeit-Skala einblenden"
PriceScale	Erhält/setzt die Eigenschaft "Preis-Skala einblenden"
Time	"Stub" für Zeitkoordinateänderung
Price	"Stub" für Preiskoordinateänderung
Eingabe/Ausgabe	

© 2000-2025, MetaQuotes Ltd.

Erstellung	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Sehen Sie auch

Objekttypen, Objekteigenschaften, Bindung-Ecke, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Chart auf dem Chart".

```
bool  Create(
   long     chart_id,      // Chart ID
   string   name,          // Objektname
   int      window,        // Chartfenster
   int      X,             // X-Koordinate
   int      Y,             // Y-Koordinate
   int      sizeX,         // X-Größe
   int      sizeY          // Y-Größe
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

X

[in] Der Abstand entlang der Achse X

Y

[in] Der Abstand entlang der Achse Y

sizeX

[in] Der Größe entlang der Achse X

sizeY

[in] Der Größe entlang der Achse Y

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

X_Distance (Get-Methode)

Erhält den Wert der Eigenschaft "Abstand entlang der Achse X".

```
int  X_Distance() const
```

Rückgabewert

Der Eigenschaftswert "Abstand entlang der Achse X" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

X_Distance (Set-Methode)

Setzt den Wert der Eigenschaft "Abstand entlang der Achse X".

```
bool  X_Distance(
   int  X      // Eigenschaftswert
   )
```

Parameter

X

[in] Der neue Wert der Eigenschaft "Abstand entlang der Achse X".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Distance (Get-Methode)

Erhält den Wert der Eigenschaft "Abstand entlang der Achse Y".

```
int  Y_Distance() const
```

Rückgabewert

Der Eigenschaftswert "Abstand entlang der Achse Y" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Y_Distance (Set-Methode)

Setzt den Wert der Eigenschaft "Abstand entlang der Achse Y".

```
bool  Y_Distance(
   int  Y      // Eigenschaftswert
   )
```

Parameter

Y

[in] Der neue Wert der Eigenschaft "Abstand entlang der Achse Y".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Corner (Get-Methode)

Erhält die Eigenschaft "Chartecke zur Bindung".

```
ENUM_BASE_CORNER  Corner() const
```

Rückgabewert

Der Eigenschaftswert "Chartecke zur Bindung" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Corner (Set-Methode)

Setzt die Eigenschaft "Chartecke zur Bindung".

```
bool  Corner(
   ENUM_BASE_CORNER  corner      // Eigenschaftswert
   )
```

Parameter

corner

[in] Der neue Wert der Eigenschaft "Chartecke zur Bindung".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

X_Size (Get-Methode)

Erhält den Wert der Eigenschaft "Größe entlang der Achse X".

```
int  X_Size() const
```

Rückgabewert

Der Eigenschaftswert "Größe entlang der Achse X" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

X_Size (Set-Methode)

Setzt den Wert der Eigenschaft "Größe entlang der Achse X".

```
bool  X_Size(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Größe entlang der Achse X".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Size (Get-Methode)

Erhält den Wert der Eigenschaft "Größe entlang der Achse Y".

```
int  Y_Size() const
```

Rückgabewert

Der Eigenschaftswert "Größe entlang der Achse Y" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Y_Size (Set-Methode)

Setzt den Wert der Eigenschaft "Größe entlang der Achse Y".

```
bool  Y_Size(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Größe entlang der Achse Y".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Symbol (Get-Methode)

Erhält den Wert der Eigenschaft "Symbol".

```
string  Symbol() const
```

Rückgabewert

Der Eigenschaftswert "Symbol" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird "" zurückgegeben.

Symbol (Set-Methode)

Setzt den Wert der Eigenschaft "Symbol".

```
bool  Symbol(
   string  symbol       // Symbol
   )
```

Parameter

symbol

[in] Der neue Wert der Eigenschaft "Symbol".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Period (Get-Methode)

Erhält den Wert der Eigenschaft "Periode".

```
int  Period() const
```

Rückgabewert

Der Eigenschaftswert "Periode" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Period (Set-Methode)

Setzt den Wert der Eigenschaft "Periode".

```
bool  Period(
   int  period      // Periode
   )
```

Parameter

period

[in] Der neue Wert der Eigenschaft "Periode".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Scale (Get-Methode)

Erhält den Wert der Eigenschaft "Maßstab".

```
double  Scale() const
```

Rückgabewert

Der Eigenschaftswert "Maßstab" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird EMPTY_VALUE zurückgegeben.

Scale (Set-Methode)

Setzt den Wert der Eigenschaft "Maßstab".

```
bool  Scale(
   double  scale      // Eigenschaftswert
   )
```

Parameter

scale

[in] Der neue Wert der Eigenschaft "Maßstab".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

DateScale (Get-Methode)

Erhält den Wert des Flags "Anzeige der Zeitskala"-

```
bool  DateScale() const
```

Rückgabewert

Der wert des Flags "Zeitskala einblenden" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

DateScale (Set-Methode)

Setzt den Wert des Flags "Zeitskala einblenden".

```
bool  DateScale(
   bool  scale      // Flagwert
   )
```

Parameter

scale

[in] Der neue Wert des Flags "Zeitskala einblenden".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

PriceScale (Get-Methode)

Erhält den Wert des Flags "Preisskala einblenden".

```
bool  1_PriceScale() const
```

Rückgabewert

Der wert des Flags "Preisskala einblenden" für ein Objekt, das an die Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

PriceScale (Set-Methode)

Setzt den Wert des Flags "Preisskala einblenden".

```
bool  2_PriceScale(
   bool  scale      // Flagwert
   )
```

Parameter

scale

[in] Der neue Wert des Flags "Preisskala einblenden".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Time

Verbietet Änderung der Zeitkoordinaten.

```
bool  Time(
   datetime  time      // beliebiger Wert
   )
```

Parameter

time

[in]

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Price

Verbietet Änderung der Preiskoordinaten.

```
bool  Price(
   double  price      // beliebiger Wert
   )
```

Parameter

price

[in]

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectSubChart - OBJ_CHART).

© 2000-2025, MetaQuotes Ltd.

CChartObjectBitmap

Klasse CChartObjectBirmap ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften eines grafischen Objekts "Bitmap".

Beschreibung

Klasse CChartObjectBitmap bietet Zugriff auf die Eigenschaften des Objekts "Bitmap".

Deklaration

```
class CChartObjectBitmap : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsBmpControls.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectBitmap

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Bitmap"
Eigenschaften	
BmpFile	Erhält/setzt die Eigenschaft "Name der BMP-Datei"
X_Offset	Erhält/setzt die Eigenschaft "X-Koordinate des Gültigkeitsbereichs"
Y_Offset	Erhält/setzt die Eigenschaft "Y-Koordinate des Gültigkeitsbereichs"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Sehen Sie auch

Objekttypen, Objekteigenschaften, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Bitmap".

```
bool  Create(
   long      chart_id,      // Chart ID
   string    name,          // Objektname
   int       window,        // Chartfenster
   datetime  time,          // Zeitkoordinate
   double    price          // Zeitkoordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

time

[in] Die Zeitkoordinate des Ankerpunktes.

price

[in] Die Preiskoordinate des Ankerpunktes.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

BmpFile (Get-Methode)

Erhält die Eigenschaft "Name der BMP-Datei".

```
string  BmpFile() const
```

Rückgabewert

Der Eigenschaftswert "Name der BMP-Datei" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird "" zurückgegeben.

BmpFile (Set-Methode)

Setzt die Eigenschaft "Name der BMP-Datei".

```
bool  BmpFile(
   string  name      // Eigenschaftswert
   )
```

Parameter

name

[in] Der neue Wert der Eigenschaft ""Name der BMP-Datei".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

X_Offset (Get-Methode)

Erhält die Eigenschaft "X-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) des grafischen Objekts CChartObjectBitmap.

```
int  X_Offset() const
```

Rückgabewert

Ein Wert der Elgenschaft "X-Koordinate des Gültigkeitsbereichs" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

X_Offset (Set-Methode)

Setzt die Eigenschaft "X-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) des grafischen Objekts CChartObjectBitmap. Der Wert wird in Pixel relativ zur oberen linken Ecke des Originalbildes gegeben.

```
bool  X_Offset(
   int  X      // Eigenschaftswert
   )
```

Parameter

X

[in] Der neue Wert der Eigenschaft "X-Koordinate des Gültigkeitsbereichs".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Offset (Get-Methode)

Erhält die Eigenschaft "Y-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) eines grafischen Objekts CChartObjectBitmap.

```
int  Y_Offset() const
```

Rückgabewert

Ein Wert der Elgenschaft "Y-Koordinate des Gültigkeitsbereichs" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Y_Offset (Set-Methode)

Setzt die Eigenschaft "Y-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) des grafischen Objekts CChartObjectBitmap. Der Wert wird in Pixel relativ zur oberen linken Ecke des Originalbildes gegeben.

```
bool  Y_Offset(
   int  Y      // Eigenschaftswert
   )
```

Parameter

Y

[in] Der neue Wert der Eigenschaft "Y-Koordinate des Gültigkeitsbereichs".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectBitmap - OBJ_BITMAP).

© 2000-2025, MetaQuotes Ltd.

CChartObjectBmpLabel

Klasse CChartObjectBmpLabel ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Bitmap Label".

Beschreibung

Klasse CChartObjectBmpLabel bietet Zugriff auf die Eigenschaften des Objekts "Bitmap Label".

Deklaration

```
class CChartObjectBmpLabel : public CChartObject
```

Kopf

```
#include <ChartObjects\ChartObjectsBmpControls.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectBmpLabel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Bitmap Label"
Eigenschaften	
X_Distance	Erhält/setzt die Eigenschaft "X-Koordinate"
Y_Distance	Erhält/setzt die Eigenschaft "Y-Koordinate"
X_Offset	Erhält/setzt die Eigenschaft "X-Koordinate des Gültigkeitsbereichs"
Y_Offset	Erhält/setzt die Eigenschaft "Y-Koordinate des Gültigkeitsbereichs"
Corner	Erhält/setzt die Eigenschaft "Chartecke zur Bindung"
X_Size	Erhält die Eigenschaft "Größe entlang der X-Achse"
Y_Size	Erhält die Eigenschaft "Größe entlang der Y-Achse"
BmpFileOn	Erhält/setzt die Eigenschaft "Name der BMP-Datei" für den On-Zustand (gedrückt)
BmpFileOff	Erhält/setzt die Eigenschaft "Name der BMP-Datei" für den Off-Zustand (nicht gedrückt)
State	Erhält/setzt die Tastenzustand (gedrückt/nicht gedrückt)
Time	"Stub" für Zeitkoordinateänderung
Price	"Stub" für Preiskoordinateänderung

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Sehen Sie auch

Objekttypen, Objekteigenschaften, Bindung-Ecke, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Bitmap Label".

```
bool  Create(
   long     chart_id,      // Chart ID
   string   name,          // Objektname
   int      window,        // Chartfenster
   int      X,             // X-Koordinate
   int      Y              // Y-Koordinate
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

X

[in] Der Abstand entlang der Achse X

Y

[in] Der Abstand entlang der Achse Y

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

X_Distance (Get-Methode)

Erhält den Wert der Eigenschaft "Abstand entlang der Achse X".

```
int  X_Distance() const
```

Rückgabewert

Der Eigenschaftswert "Abstand entlang der Achse X" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

X_Distance (Set-Methode)

Setzt den Wert der Eigenschaft "Abstand entlang der Achse X".

```
bool  X_Distance(
   int  X      // Eigenschaftswert
   )
```

Parameter

X

[in] Der neue Wert der Eigenschaft "Abstand entlang der Achse X".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Distance (Get-Methode)

Erhält den Wert der Eigenschaft "Abstand entlang der Achse Y".

```
int  Y_Distance() const
```

Rückgabewert

Der Eigenschaftswert "Abstand entlang der Achse Y" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Y_Distance (Set-Methode)

Setzt den Wert der Eigenschaft "Abstand entlang der Achse Y".

```
bool  Y_Distance(
   int  Y      // Eigenschaftswert
   )
```

Parameter

Y

[in] Der neue Wert der Eigenschaft "Abstand entlang der Achse Y".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

X_Offset (Get-Methode)

Erhält die Eigenschaft "X-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) des grafischen Objekts CChartObjectBmpLabel.

```
int  X_Offset() const
```

Rückgabewert

Ein Wert der Elgenschaft "X-Koordinate des Gültigkeitsbereichs" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

X_Offset (Set-Methode)

Setzt die Eigenschaft "X-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) des grafischen Objekts CChartObjectBmpLabel. Der Wert wird in Pixel relativ zur oberen linken Ecke des Originalbildes gegeben.

```
bool  X_Offset(
   int  X      // Eigenschaftswert
   )
```

Parameter

X

[in] Der neue Wert der Eigenschaft "X-Koordinate des Gültigkeitsbereichs".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Y_Offset (Get-Methode)

Erhält die Eigenschaft "Y-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) eines grafischen Objekts CChartObjectBmpLabel.

```
int  Y_Offset() const
```

Rückgabewert

Ein Wert der Elgenschaft "Y-Koordinate des Gültigkeitsbereichs" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

Y_Offset (Set-Methode)

Setzt die Eigenschaft "Y-Koordinate des Gültigkeitsbereichs" (der oberen linken Ecke) des grafischen Objekts CChartObjectBmpLabel. Der Wert wird in Pixel relativ zur oberen linken Ecke des Originalbildes gegeben.

```
bool  Y_Offset(
   int  Y      // Eigenschaftswert
   )
```

Parameter

Y

[in] Der neue Wert der Eigenschaft "Y-Koordinate des Gültigkeitsbereichs".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Corner (Get-Methode)

Erhält die Eigenschaft "Chartecke zur Bindung".

```
ENUM_BASE_CORNER  Corner() const
```

Rückgabewert

Der Eigenschaftswert "Chartecke zur Bindung" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird WRONG_VALUE zurückgegeben.

Corner (Set-Methode)

Setzt die Eigenschaft "Chartecke zur Bindung".

```
bool  Corner(
   ENUM_BASE_CORNER  corner      // Eigenschaftswert
   )
```

Parameter

corner

[in] Der neue Wert der Eigenschaft "Chartecke zur Bindung".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

X_Size

Erhält den Wert der Eigenschaft "Größe entlang der Achse X".

```
int  X_Size() const
```

Rückgabewert

Der Eigenschaftswert "Größe entlang der Achse X" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Y_Size

Erhält den Wert der Eigenschaft "Größe entlang der Achse Y".

```
int  Y_Size() const
```

Rückgabewert

Der Eigenschaftswert "Größe entlang der Achse Y" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

BmpFileOn (Get-Methode)

Erhält die Eigenschaft "Name der BMP-Datei ON".

```
string  BmpFileOn() const
```

Rückgabewert

Der Eigenschaftswert "Name der BMP-Datei ON" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird "" zurückgegeben.

BmpFileOn (Set-Methode)

Setzt die Eigenschaft "Name der BMP-Datei ON".

```
bool  BmpFileOn(
   string  name      // Dateiname
   )
```

Parameter

name

[in] Der neue Wert der Eigenschaft ""Name der BMP-Datei ON".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

BmpFileOff (Get-Methode)

Erhält die Eigenschaft "Name der BMP-Datei OFF".

```
string  BmpFileOff() const
```

Rückgabewert

Der Eigenschaftswert "Name der BMP-Datei OFF" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird "" zurückgegeben.

BmpFileOff (Set-Methode)

Setzt die Eigenschaft "Name der BMP-Datei OFF".

```
bool  BmpFileOff(
   string  name      // Dateiname
   )
```

Parameter

name

[in] Der neue Wert der Eigenschaft ""Name der BMP-Datei OFF".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

State (Get-Methode)

Erhält den Wert der Eigenschaft "Zustand".

```
bool  State() const
```

Rückgabewert

Der Eigenschaftswert "Zustand" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird false zurückgegeben.

State (Set-Methode)

Setzt den Wert der Eigenschaft "Zustand".

```
bool  State(
   bool  state      // Eigenschaftswert
   )
```

Parameter

state

[in] Der neue Wert der Eigenschaft "Zustand".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Time

Verbietet Änderung der Zeitkoordinaten.

```
bool  Time(
   datetime  time      // beliebiger Wert
   )
```

Parameter

time

[in] Beliebiger datetime-Wert.

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Price

Verbietet Änderung der Preiskoordinaten.

```
bool  Price(
   double  price      // beliebiger Wert
   )
```

Parameter

price

[in] beliebiger double-Wert.

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectBmpLabel - OBJ_BITMAP_LABEL).

© 2000-2025, MetaQuotes Ltd.

CChartObjectRectLabel

Klasse CChartObjectRectLabel ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des grafischen Objekts "Rechteck-Label".

Beschreibung

Klasse CChartObjectRectLabel bietet Zugriff auf die Eigenschaften des Objekts "Rechteck-Label".

Deklaration

```
class CChartObjectRectLabel : public CChartObjectLabel
```

Kopf

```
#include <ChartObjects\ChartObjectsTxtControls.mqh>
```

Vererbungshierarchie

CObject
  CChartObject
    CChartObjectText
      CChartObjectLabel
        CChartObjectRectLabel

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein grafisches Objekt "Rechteck-Label"
Eigenschaften	
X_Size	Setzt die Eigenschaft "Größe entlang der X-Achse"
Y_Size	Setzt die Eigenschaft "Größe entlang der Y-Achse"
BackColor	Erhält/setzt die Eigenschaft "Hintergrundfarbe"
Angle	Verbietet die Änderung der Eigenschaft "Neigungswinkel"
BorderType	Erhält/setzt die Eigenschaft "Rahmentyp"
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CChartObject

ChartId, Window, Name, Name, NumPoints, Attach, SetPoint, Delete, Detach, Time, Time, Price, Price, Color, Color, Style, Style, Width, Width, Background, Background, Fill, Fill, Z_Order, Z_Order, Selected, Selected, Selectable, Selectable, Description, Description, Tooltip, Tooltip, Timeframes, Timeframes, CreateTime, LevelsCount, LevelsCount, LevelColor, LevelColor, LevelStyle, LevelStyle, LevelWidth, LevelWidth, LevelValue, LevelValue, LevelDescription, LevelDescription, GetInteger, GetInteger, SetInteger, SetInteger, GetDouble, GetDouble, SetDouble, SetDouble, GetString, GetString, SetString, SetString, ShiftObject, ShiftPoint

Methoden geerbt von der Klasse CChartObjectText

Angle, Angle, Font, Font, FontSize, FontSize, Anchor, Anchor, Create

Methoden geerbt von der Klasse CChartObjectLabel

X_Distance, X_Distance, Y_Distance, Y_Distance, X_Size, Y_Size, Corner, Corner, Time, Price, Create

Sehen Sie auch

Objekttypen, Objekteigenschaften, Grafische Objekte

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein grafisches Objekt "Rechteck-Label".

```
bool  Create(
   long     chart_id,      // ID des Charts
   string   name,          // Objektname
   int      window,        // Chartfenster
   int      X,             // X-Koordinate
   int      Y,             // Y-Koordinate
   int      sizeX,         // X-Größe
   int      sizeY          // Y-Größe
   )
```

Parameter

chart_id

[in] Chart ID (0 - der aktuelle Chart).

name

[in] Der eindeutige Name des Objekts.

window

[in] Nummer des Chartfensters (0 - das Hauptfenster).

X

[in] Der Abstand entlang der Achse X

Y

[in] Der Abstand entlang der Achse Y

sizeX

[in] Der Größe entlang der Achse X

sizeY

[in] Der Größe entlang der Achse Y

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

X_Size

Setzt den Wert der Eigenschaft "Größe entlang der Achse X".

```
bool  X_Size(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Größe entlang der Achse X".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

Hinweis

Die Werte der Eigenschaften "Größe entlang Achse X" und "Größe entlang Achse X" werden durch die Methoden X_Size und Y_Size der Basisklasse CChartObjectLabel erhalten.

© 2000-2025, MetaQuotes Ltd.

Y_Size

Setzt den Wert der Eigenschaft "Größe entlang der Achse Y".

```
bool  Y_Size(
   int  size      // Eigenschaftswert
   )
```

Parameter

size

[in] Der neue Wert der Eigenschaft "Größe entlang der Achse Y".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

Hinweis

Die Werte der Eigenschaften "Größe entlang Achse X" und "Größe entlang Achse X" werden durch die Methoden X_Size und Y_Size der Basisklasse CChartObjectLabel erhalten.

© 2000-2025, MetaQuotes Ltd.

BackColor

Erhält den Wert der Eigenschaft "Hintergrundfarbe".

```
color  BackColor() const
```

Rückgabewert

Der Eigenschaftswert "Hintergrundfarbe" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

BackColor

Setzt den Wert der Eigenschaft "Hintergrundfarbe".

```
bool  BackColor(
   color  new_color      // Eigenschaftswert
   )
```

Parameter

new_color

[in] Der neue Wert der Eigenschaft "Hintergrundfarbe".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Angle

Verbietet die Änderung der Eigenschaft "Neigungswinkel".

```
bool  Angle(
   double  angle      // beliebiger Wert
   )
```

Parameter

angle

[in] Beliebiger double-Wert.

Rückgabewert

Immer false.

© 2000-2025, MetaQuotes Ltd.

BorderType

Erhält den Wert der Eigenschaft "Rahmentyp".

```
int  BorderType() const
```

Rückgabewert

Der Eigenschaftswert "Rahmentyp" eines Objekts, das an eine Klasseninstanz gebunden ist. Wenn es kein gebundenes Objekt gibt, wird 0 zurückgegeben.

BorderType

Setzt den Wert der Eigenschaft "Rahmentyp".

```
bool  BorderType(
   int  type      // Eigenschaftswert
   )
```

Parameter

type

[in] Der neue Wert der Eigenschaft "Rahmentyp".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert die Parameter des Objektes in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die mit der Funktion FileOpen früher geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Identifikator des Typs eines grafischen Objektes.

```
virtual int  Type() const
```

Rückgabewert

Identifikator des Objekttyps (für CChartObjectRectLabel - OBJ_RECTANGLE_LABEL).

© 2000-2025, MetaQuotes Ltd.

Benutzerdefinierte Grafik

In diesem Abschnitt sind Werkzeuge für die benutzerdefinierte Grafik beschrieben.

Die Anwendung dieser Tools erleichtert wesentlich das Zeichnen benutzerdefinierter Charts, Abbildungen und die Visualisierung von Daten.

Es wurden Klassen für die Erstellung grafischer Objekte und Primitive sowie für das Zeichnen von Diagrammen und Kurven entwickelt. Es wurden unterschiedliche Möglichkeiten der Darstellung von Objekten implementiert: Änderung des Stils und der Farbe von Linien, Füllung, Operationen mit Datenreihen auf dem Chart usw.

Klasse	Beschreibung
CCanvas	Klasse für eine vereinfachte Erstellung benutzerdefinierter Zeichnungen
CChartCanvas	Basisklasse für die Umsetzung der Klassen für das Zeichnen von Charts und deren Elemente
CHistogramChart	Klasse für das Zeichnen von Säulenhistogrammen
CLineChart	Klasse für das Zeichnen von Kurven
CPieChart	Klasse für das Zeichnen von Kuchendiagrammen

© 2000-2025, MetaQuotes Ltd.

CCanvas

Die CCanvas Klasse stellt eine Klasse für eine einfache Erstellung benutzerdefinierter Zeichnungen.

Beschreibung

Die CCanvas Klasse sichert die Erstellung einer grafischen Ressource (mit Bindung zum Chartobjekt oder ohne) und das Zeichnen grafischer Grundelemente.

Deklaration

```
class CCanvas
```

Überschrift

```
#include <Canvas\Canvas.mqh>
```

Vererbungshierarchie

CCanvas

Direkte Ableitungen

CChartCanvas, CFlameCanvas

Klassenmethoden nach Gruppen

Erstellung	
Attach	Bindet das Objekt OBJ_BITMAP_LABEL an eine Instanz der CCanvas Klasse
Create	Erstellt eine grafische Ressource ohne Bindung an ein Objekt des Charts
CreateBitmap	Erstellt eine grafische Ressource, die mit einem Chart-Objekt verknüpft ist
CreateBitmapLabel	Erstellt eine grafische Ressource, die mit einem Chart-Objekt verknüpft ist
Destroy	Vernichtet die grafische Ressource
Eigenschaften	
ChartObjectName	Erhält den Namen eines gebundenen Objekts
ResourceName	Erhält den Namen der grafischen Ressource
Width	Erhält die Breite der grafischen Ressourcea
Height	Erhält die Höhe der grafischen Ressource
LineStyleSet	Setzt den Linienstil

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Aktualisiert das Objekt auf dem Bildschirm	
Update	Gibt Änderungen auf dem Bildschirm aus
Resize	Ändert die Größe der grafischen Ressource
Очистка/заполнение цветом	
Erase	Löscht oder füllt mit der angegebenen Farbe
Zugang zu Daten	
PixelGet	Erhält die Farbe des Punktes mit den angegebenen Koordinaten
PixelSet	Setzt die Farbe des Punktes mit den angegebenen Koordinaten
Zeichnen grafischer Grundelemente	
LineVertical	Zeichnet eine vertikale Linie
LineHorizontal	Zeichnet eine horizontale Linie
Line	Zeichnet eine zufällige Linie
Polyline	Zeichnet eine gebrochene Linie
Polygon	Zeichnet ein Vieleck
Rectangle	Zeichnet einen Rechteck
Circle	Zeichnet einen Kreis
Triangle	Zeichnet ein Dreieck
Ellipse	Zeichnet eine Ellipse
Arc	Zeichnet einen Ellipsenbogen
Pie	Zeichnet einen Ellipsensektor
Zeichnet mit Farbe gefüllte Grundelemente	
FillRectangle	Zeichnet einen mit Farbe gefüllten Rechteck
FillCircle	Zeichnet einen mit Farbe gefüllten Kreis
FillTriangle	Zeichnet ein mit Farbe gefülltes Dreieck
FillPolygon	Zeichnet ein mit Farbe gefülltes Vieleck
FillEllipse	Zeichnet eine mit Farbe gefüllte Ellipse
Fill	Füllt einen Bereich mit Farbe

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Zeichnung grafischer Grundelement unter Verwendung von Glättung	
PixelSetAA	Zeichnet einen Punkt
LineAA	Zeichnet eine Linie
PolylineAA	Zeichnet eine gebrochene Linie
PolygonAA	Zeichnet ein Vieleck
TriangleAA	Zeichnet ein Dreieck
CircleAA	Zeichnet einen Kreis
EllipseAA	Zeichnet eine Ellipse
LineWu	Zeichnet eine Linie
PolylineWu	Zeichnet eine gebrochene Linie
PolygonWu	Zeichnet ein Vieleck
TriangleWu	Zeichnet ein Dreieck
CircleWu	Zeichnet einen Kreis
EllipseWu	Zeichnet eine Ellipse
LineThick	Zeichnet eine Strecke einer zufälligen Linie mit der angegebenen Größe mithilfe eines Glättungsalgorithmus
LineThickVertical	Zeichnet eine vertikale Strecke einer zufälligen Linie mit der angegebenen Dicke mithilfe eines Glättungsalgorithmus
LineThickHorizontal	Zeichnet eine horizontale Strecke einer zufälligen Linie mit der angegebene Dicke mithilfe eines Glättungsalgorithmus
PolygonSmooth	Zeichnet ein Vieleck mit der angegebenen Dicke unter Verwendung zweier Glättungsalgorithmen
PolygonThick	Zeichnet ein Vieleck mit der angegebenen Dicke unter Verwendung des Glättungsalgorithmus
PolylineSmooth	Zeichnet eine gebrochene Linie mit der angegebenen Dicke unter Verwendung zweier Glättungsalgorithmen
PolylineThick	Zeichnet eine gebrochen Linie mit der angegebenen Dicke unter Verwendung eines Glättungsalgorithmus
Text	
FontSet	Setzt Schriftparameter
FontNameSet	Setzt den Schriftnamen
FontSizeSet	Setzt die Schriftgröße

© 2000-2025, MetaQuotes Ltd.

Erstellung	
FontFlagsSet	Setzt Flags der Schrift
FontAngleSet	Setzt den Neigungswinkel der Schrift
FontGet	Erhält Schriftparameter
FontNameGet	Erhält den Schriftnamen
FontSizeGet	Erhält die Schriftgröße
FontFlagsGet	Erhält Flags der Schrift
FontAngleGet	Erhält den Neigungswinkel der Schrift
TextOut	Gibt den Text aus
TextWidth	Erhält die Textbreite
TextHeight	Erhält die Texthöhe
TextSize	Erhält die Textgröße
Transparenz	
TransparentLevelSet	Stellt Transparenz ein
Eingabe/Ausgabe	
LoadFromFile	Liest das Bild auf einer BMP-Datei

© 2000-2025, MetaQuotes Ltd.

Attach

Erhält eine grafische Ressource aus dem Objekt OBJ_BITMAP_LABEL und bindet diese an eine Instanz der CCanvas Klasse.

```
bool  Attach(
   const long          chart_id,                          // Chart-ID
   const string        objname,                           // Objektname
   ENUM_COLOR_FORMAT   clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Methode zur Verarbeitu
   )
```

Erstellt eine grafische Ressource für das Objekt OBJ_BITMAP_LABEL und bindet diese an eine Instanz der CCanvas Klasse.

```
bool  Attach(
   const long          chart_id,                          // Chart-ID
   const string        objname,                           // Objektname
   const int           width,                             // Bildbreite in Pixel
   const int           height,                            // Bildhöhe in Pixel
   ENUM_COLOR_FORMAT   clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Methode zur Verarbeitu
   )
```

Parameter

chart_id

[out] Chart-ID.

objname

[in] Bezeichnung (Name) des grafischen Objekts.

width

[in] Bildbreite in der Ressource.

height

[in] Bildhöhe in der Ressource.

clrfmt=COLOR_FORMAT_XRGB_NOALPHA

[in] Methode zur Verarbeitung des Alpha-Kanals. Standardmäßig wird der Alpha-Kanal ignoriert.

Rückgabewert

true - wenn erfolgreich, false - wenn fehlgeschlagen.

© 2000-2025, MetaQuotes Ltd.

Arc

Zeichnet einen Ellipsenbogen, der einem Rechteck mit den Ecken (x1,y1) und (x2,y2) eingeschrieben ist. Die Grenzen des Bogens werden durch die Linien aus dem Zentrum der Ellipse abgeschnitten, die zu zwei Punkten mit den Koordinaten (x3,y3) und (x4,y4) gezogen werden.

```
void  Arc(
   int          x1,      // X-Koordinate der linken oberen Ecke des Rechtecks
   int          y1,      // Y-Koordinate der linken oberen Ecke des Rechtecks
   int          x2,      // X-Koordinate der rechten unteren Ecke des Rechtecks
   int          y2,      // Y-Koordinate der rechten unteren Ecke des Rechtecks
   int          x3,      // X-Koordinate des ersten Punktes für die Bestimmung der Grer
   int          y3,      // Y-Koordinate des ersten Punktes für die Bestimmung der Grer
   int          x4,      // X-Koordinate des zweiten Punktes für die Bestimmung der Gre
   int          y4,      // Y-Koordinate des zweiten Punktes für die Bestimmung der Gre
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate der linken oberen Ecke, die das Rechteck bestimmt.

y1

[in] Y-Koordinate der linken oberen Ecke, die das Rechteck bestimmt.

x2

[in] X-Koordinate der rechten unteren Ecke, die das Rechteck bestimmt.

y2

[in] Y-Koordinate der rechten unteren Ecke, die das Rechteck bestimmt.

x3

[in] X-Koordinate des ersten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

y3

[in] Y-Koordinate des ersten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

x4

[in] X-Koordinate des zweiten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

y4

[in] Y-Koordinate des zweiten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

clr

[in] Farbe im ARGB-Format. Für die Umwandlung einer Farbe ins ARGB-Format verwenden Sie bitte die Funktion ColorToARGB().

© 2000-2025, MetaQuotes Ltd.

Zeichnet einen Ellipsenbogen mit dem Zentrum im Punkt (x,y), der in das Rechteck mit den Radien rx und ry eingeschrieben ist. Die Grenzen des Bogens werden durch die Strahlen aus dem Zentrum der Ellipse angeschnitten, die durch die Winkel fi3 und fi4 gesetzt werden.

```
void  Arc(
   int          x,        // X-Koordinate des Zentrums der Ellipse
   int          y,        // Y-Koordinate des Zentrums der Ellipse
   int          rx,       // Radius der Ellipse nach der X-Koordinate
   int          ry,       // Radius der Ellipse nach der Y-Koordinate
   int          fi3,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die e
   int          fi4,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die z
   const uint   clr       // Farbe
   );
```

Zeichnet einen Ellipsenbogen mit dem Zentrum im Punkt (x,y), der in das Rechteck mit den Radien rx und ry eingeschrieben ist, und gibt die Koordinaten der Grenze des Bogens zurück. Die Grenzen des Bogens werden durch die Strahlen aus dem Zentrum der Ellipse angeschnitten, die durch die Winkel fi3 und fi4 gesetzt werden.

```
void  Arc(
   int          x,        // X-Koordinate des Zentrums der Ellipse
   int          y,        // Y-Koordinate des Zentrums der Ellipse
   int          rx,       // Radius der Ellipse nach der X-Koordinate
   int          ry,       // Radius der Ellipse nach der Y-Koordinate
   int          fi3,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die e
   int          fi4,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die z
   int&         x3,       // X-Koordinate der ersten Grenze des Bogens
   int&         y3,       // Y-Koordinate der ersten Grenze des Bogens
   int&         x4,       // X-Koordinate der zweiten Grenze des Bogens
   int&         y4,       // Y-Koordinate der zweiten Grenze des Bogens
   const uint   clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Zentrums der Ellipse.

y

[in] Y-Koordinate des Zentrums der Ellipse.

rx

[in] Radius der Ellipse nach der X-Koordinate, in Pixel.

ry

[in] Radius der Ellipse nach der Y-Koordinate, in Pixel.

fi3

[in] Winkel in Radianten, der die erste Grenze des Bogens definiert

© 2000-2025, MetaQuotes Ltd.

fi4

[in] Winkel in Radianten, der die zweite Grenze des Bogens definiert

x3

[out] Variable für das Erhalten der X-Koordinate der ersten Grenze des Bogens.

y3

[out] Variable für das Erhalten der Y-Koordinate der ersten Grenze des Bogens.

x4

[out] Variable für das Erhalten der X-Koordinate der zweiten Grenze des Bogens.

y4

[out] Variable für das Erhalten der Y-Koordinate der zweiten Grenze des Bogens.

clr

[in] Farbe im ARGB-Format. Für die Umwandlung einer Farbe ins ARGB-Format verwenden Sie bitte die Funktion ColorToARGB().

Beispiele für das Aufruf der Methoden der Klasse:

```
#include <Canvas\Canvas.mqh>
CCanvas canvas;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int      Width=600;
   int      Height=400;
//--- create canvas
   if(!canvas.CreateBitmapLabel(0,0,"CirclesCanvas",30,30,Width,Height))
     {
      Print("Error creating canvas: ",GetLastError());
     }
//--- clear canvas
   canvas.Erase(clrWhite);
//--- draw rectangle
   canvas.Rectangle(215-190,215-120,215+190,215+120,clrGray);
//--- draw first arc
   canvas.Arc(215,215, 190,120,M_PI_4,2*M_PI-M_PI_4,ColorToARGB(clrRed));
   int x1,y1,x2,y2;
//--- draw second arc
   canvas.Arc(215,215, 190,120,2*M_PI-M_PI_4,2*M_PI+M_PI_4,x1,y1,x2,y2,ColorToARGB(cl
//--- print coordinates of arc
   PrintFormat("First point of arc at (%G,%G), second point of arc at (%G,%G)",x1,y1,x
   canvas.CircleAA(x1,y1,3, ColorToARGB(clrRed));
   canvas.CircleAA(x2,y2,3, ColorToARGB(clrBlue));
//--- show updated canvas
   canvas.Update();
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Pie

Zeichnet einen gefüllten Ellipsensektor, der einem Rechteck mit den Ecken (x1,y1) und (x2,y2) eingeschrieben ist. Die Sektorgrenzen werden durch die Linien aus dem Zentrum der Ellipse abgeschnitten, die zu zwei Punkten mit den Koordinaten (x3,y3) und (x4,y4) gezogen werden.

```
void  Pie(
   int          x1,       // X-Koordinate der linken oberen Ecke des Rechtecks
   int          y1,       // Y-Koordinate der linken oberen Ecke des Rechtecks
   int          x2,       // X-Koordinate der rechten unteren Ecke des Rechtecks
   int          y2,       // Y-Koordinate der rechten unteren Ecke des Rechtecks
   int          x3,       // X-Koordinate des ersten Punktes für die Bestimmung der Grer
   int          y3,       // Y-Koordinate des ersten Punktes für die Bestimmung der Grer
   int          x4,       // X-Koordinate des zweiten Punktes für die Bestimmung der Gre
   int          y4,       // Y-Koordinate des zweiten Punktes für die Bestimmung der Gre
   const uint   clr,      // Farbe der Linie
   const uint   fill_clr // Farbe der Füllung
   );
```

Parameter

x1

[in] X-Koordinate der linken oberen Ecke, die das Rechteck bestimmt.

y1

[in] Y-Koordinate der linken oberen Ecke, die das Rechteck bestimmt.

x2

[in] X-Koordinate der rechten unteren Ecke, die das Rechteck bestimmt.

y2

[in] Y-Koordinate der rechten unteren Ecke, die das Rechteck bestimmt.

x3

[in] X-Koordinate des ersten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

y3

[in] Y-Koordinate des ersten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

x4

[in] X-Koordinate des zweiten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

y4

[in] Y-Koordinate des zweiten Punktes, zu welchem eine Linie aus dem Zentrum des Rechtecks gezogen wurde, um die Grenze des Bogens zu erhalten.

clr

[in] Farbe der Sektorgrenze im ARGB-Format.

fill_clr

© 2000-2025, MetaQuotes Ltd.

[in] Farbe der Sektorfüllung im ARGB-Format. Für die Umwandlung einer Farbe ins ARGB-Format verwenden Sie bitte die Funktion ColorToARGB().

Zeichnet einen gefüllten Ellipsensektor mit dem Zentrum im Punkt (x,y), der in das Rechteck mit den Radien rx und ry eingeschrieben ist. Die Sektorgrenzen werden durch die Linien aus dem Zentrum der Ellipse angeschnitten, die durch die Winkel fi3 und fi4 bestimmt werden.

```
void  Pie(
   int          x,        // X-Koordinate des Zentrums der Ellipse
   int          y,        // Y-Koordinate des Zentrums der Ellipse
   int          rx,       // Radius der Ellipse nach der X-Koordinate
   int          ry,       // Radius der Ellipse nach der Y-Koordinate
   int          fi3,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die e
   int          fi4,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die z
   const uint   clr,      // Farbe der Linie
   const uint   fill_clr // Farbe der Füllung
   );
```

Zeichnet einen gefüllten Ellipsensektor mit dem Zentrum im Punkt (x,y), der in ein Rechteck mit den Radien rx und ry eingeschrieben ist, und gibt die Koordinaten der Grenze des Bogens zurück. Die Sektorgrenzen werden durch die Linien aus dem Zentrum der Ellipse angeschnitten, die durch die Winkel fi3 und fi4 bestimmt werden.

```
void  Pie(
   int          x,        // X-Koordinate des Zentrums der Ellipse
   int          y,        // Y-Koordinate des Zentrums der Ellipse
   int          rx,       // Radius der Ellipse nach der X-Koordinate
   int          ry,       // Radius der Ellipse nach der Y-Koordinate
   int          fi3,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die e
   int          fi4,      // Winkel eines Strahls aus dem Zentrum der Ellipse, der die z
   int&         x3,       // X-Koordinate der ersten Grenze des Bogens
   int&         y3,       // Y-Koordinate der ersten Grenze des Bogens
   int&         x4,       // X-Koordinate der zweiten Grenze des Bogens
   int&         y4,       // Y-Koordinate der zweiten Grenze des Bogens
   const uint   clr,      // Farbe der Linie
   const uint   fill_clr // Farbe der Füllung
   );
```

Parameter

x

[in] X-Koordinate des Zentrums der Ellipse.

y

[in] Y-Koordinate des Zentrums der Ellipse.

rx

[in] Radius der Ellipse nach der X-Koordinate, in Pixel.

ry

© 2000-2025, MetaQuotes Ltd.

[in] Radius der Ellipse nach der X-Koordinate, in Pixel.

fi3

[in] Winkel in Radianten, der die erste Grenze des Bogens definiert

fi4

[in] Winkel in Radianten, der die zweite Grenze des Bogens definiert

x3

[out] Variable für das Erhalten der X-Koordinate der ersten Grenze des Bogens.

y3

[out] Variable für das Erhalten der Y-Koordinate der ersten Grenze des Bogens.

x4

[out] Variable für das Erhalten der X-Koordinate der zweiten Grenze des Bogens.

y4

[out] Variable für das Erhalten der Y-Koordinate der zweiten Grenze des Bogens.

clr

[in] Farbe der Sektorgrenze im ARGB-Format.

fill_clr

[in] Farbe der Sektorfüllung im ARGB-Format. Für die Umwandlung einer Farbe ins ARGB-Format verwenden Sie bitte die Funktion ColorToARGB().

Beispiele für das Aufruf der Methoden der Klasse:

```
#include <Canvas\Canvas.mqh>
CCanvas canvas;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int      Width=600;
   int      Height=400;
//--- create canvas
   if(!canvas.CreateBitmapLabel(0,0,"CirclesCanvas",30,30,Width,Height))
     {
      Print("Error creating canvas: ",GetLastError());
     }
//--- clear canvas
   canvas.Erase(clrWhite);
//--- draw rectangle
   canvas.Rectangle(215-190,215-120,215+190,215+120,clrGray);
//--- draw first pie
   canvas.Pie(215,215, 190,120,M_PI_4,2*M_PI-M_PI_4,ColorToARGB(clrBlue),ColorToARGB(c
//--- draw second pie
   canvas.Pie(215,215, 190,120,2*M_PI-M_PI_4,2*M_PI+M_PI_4,ColorToARGB(clrGreen),Colo
//--- show updated canvas
```

© 2000-2025, MetaQuotes Ltd.

```
   canvas.Update();
   DebugBreak();
  }
```

© 2000-2025, MetaQuotes Ltd.

FillPolygon

Zeichnet ein gefülltes Vieleck.

```
void  FillPolygon(
   int&          x,      // ein Array mit X-Koordinaten der Punkte des Vielecks
   int&          y,      // ein Array mit X-Koordinaten der Punkte des Vielecks
   const uint    clr     // Farbe
   );
```

Parameter

x

[in] ein Array mit X-Koordinaten der Punkte des Vielecks.

y

[in] ein Array mit Y-Koordinaten der Punkte des Vielecks.

clr

[in] Farbe im ARGB-Format.

© 2000-2025, MetaQuotes Ltd.

FillEllipse

Zeichnet eine gefüllte Ellipse, die einem Rechteck mit angegebenen Koordinaten eingeschrieben ist.

```
void  FillPolygon(
   int          x1,      // X-Koordinate der linken oberen Ecke des Rechtecks
   int          y1,      // Y-Koordinate der linken oberen Ecke des Rechtecks
   int          x2,      // X-Koordinate der rechten unteren Ecke des Rechtecks
   int          y2,      // Y-Koordinate der rechten unteren Ecke des Rechtecks
   const uint   clr       // Farbe der Ellipse
   );
```

Parameter

x1

[in] X-Koordinate der linken oberen Ecke, die das Rechteck bestimmt.

y1

[in] Y-Koordinate der linken oberen Ecke, die das Rechteck bestimmt.

x2

[in] X-Koordinate der rechten unteren Ecke, die das Rechteck bestimmt.

y2

[in] Y-Koordinate der rechten unteren Ecke, die das Rechteck bestimmt.

clr

[in] Farbe im ARGB-Format.

© 2000-2025, MetaQuotes Ltd.

GetDefaultColor

Gibt eine voreingestellte Farbe nach ihrem Index zurück.

```
static uint  GetDefaultColor(
   const uint  i      // Index
   );
```

Parameter

i

[in] Index für das Erhalten der Farbe.

Rückgabewert

Farbe.

© 2000-2025, MetaQuotes Ltd.

ChartObjectName

Erhält den Namen eines gebundenen Objekts des Charts.

```
string  ChartObjectName();
```

Rückgabewert

Der Name eines gebundenen Objekts des Charts

© 2000-2025, MetaQuotes Ltd.

Circle

Zeichnet einen Kreis

```
void  Circle(
   int          x,        // X-Koordinate
   int          y,        // Y-Koordinate
   int          r,        // Radius
   const uint   clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Kreismittelpunktes.

y

[in] Y-Koordinate des Kreismittelpunktes.

r

[in] Radius des Kreises.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

CircleAA

Zeichnet einen Kreis mit Anti-Aliasing

```
void  CircleAA(
   const int     x,        // X-Koordinate
   const int     y,        // Y-Koordinate
   const double  r,        // Radius
   const uint    clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Kreismittelpunktes.

y

[in] Y-Koordinate des Kreismittelpunktes.

r

[in] Radius des Kreises.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

CircleWu

Zeichnet einen Kreis mit Wu's Anti-Aliasing

```
void  CircleWu(
   const int     x,        // X-Koordinate
   const int     y,        // Y-Koordinate
   const double  r,        // Radius
   const uint    clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Kreismittelpunktes.

y

[in] Y-Koordinate des Kreismittelpunktes.

r

[in] Radius des Kreises.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine grafische Ressource ohne Bindung an das Objekt des Charts.

```
virtual bool  Create(
   const string       name,                                  // Name
   const int          width,                                 // Breite
   const int          height,                                // Höhe
   ENUM_COLOR_FORMAT  clrfmt=COLOR_FORMAT_XRGB_NOALPHA       // Format
   );
```

Parameter

name

[in] Die Basis für den Namen der grafischen Ressource. Der Ressourcenname wird durch Hinzufügen einer pseudo-zufälligen Zeichenfolge während Erstellung gebildet.

width

[in] Die Breite (Größe entlang der Achse X) in Pixel.

height

[in] Die Höhe (Größe entlang der Achse Y) in Pixel.

clrfmt=COLOR_FORMAT_XRGB_NOALPHA

[in] Farbeverarbeitungsmethode. Lesen Sie mehr darüber in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

CreateBitmap

Erstellt eine grafische Ressource mit der Bindung an Objekt des Charts.

1. Erstellt eine grafische Ressource im Hauptfenster des aktuellen Charts.

```
bool  CreateBitmap(
   const string       name,                              // Name
   const datetime     time,                              // Zeit
   const double       price,                             // Preis
   const int          width,                             // Breite
   const int          height,                            // Höhe
   ENUM_COLOR_FORMAT  clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Format
   );
```

2. Erstellt eine grafische Ressource unter Verwendung der Chartidentifikator und Nummer des Unterfensters.

```
bool  CreateBitmap(
   const long         chart_id,                          // ID des Charts
   const int          subwin,                            // Nummer des Unterfenstes
   const string       name,                              // Name
   const datetime     time,                              // Zeit
   const double       price,                             // Preis
   const int          width,                             // Breite
   const int          height,                            // Höhe
   ENUM_COLOR_FORMAT  clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Format
   );
```

Parameter

chart_id

[in] Die ID des Charts für Objekterstellung.

subwin

[in] Nummer des Unterfensters für Objekterstellung.

name

[in] Objektnamen und die Grundlage für den Namen der grafischen Ressource.

time

[in] Zeitkoordinate des Objektbezugspunktes auf dem Chart.

price

[in] Preiskoordinate des Objektbezugspunktes auf dem Chart.

width

[in] Breite der grafischen Ressource (Größe entlang der Achse X) in Pixel.

height

[in] Höhe der grafischen Ressource (Größe entlang der Achse Y) in Pixel.

clrfmt=COLOR_FORMAT_XRGB_NOALPHA

© 2000-2025, MetaQuotes Ltd.

[in] Farbeverarbeitungsmethode. Lesen Sie mehr darüber in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Wenn Sie die erste Version der Funktion verwenden, wird das Objekt im Hauptfenster des aktuellen Charts erstellt.

Objektgröße ist dieselbe wie die Größe einer grafischen Ressource.

© 2000-2025, MetaQuotes Ltd.

CreateBitmapLabel

Erstellt eine grafische Ressource mit der Bindung an Objekt des Charts.

1. Erstellt eine grafische Ressource im Hauptfenster des aktuellen Charts.

```
bool  CreateBitmapLabel(
   const string       name,                              // Name
   const int          x,                                 // X-Koordinate
   const int          y,                                 // Y-Koordinate
   const int          width,                             // Breite
   const int          height,                            // Höhe
   ENUM_COLOR_FORMAT  clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Format
   );
```

2. Erstellt eine grafische Ressource unter Verwendung der Chartidentifikator und Nummer des Unterfensters.

```
bool  CreateBitmapLabel(
   const long         chart_id,                          // ID des Charts
   const int          subwin,                            // Nummer des Unterfenster
   const string       name,                              // Name
   const int          x,                                 // X-Koordinate
   const int          y,                                 // Y-Koordinate
   const int          width,                             // Breite
   const int          height,                            // Höhe
   ENUM_COLOR_FORMAT  clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Format
   );
```

Parameter

chart_id

[in] Die ID des Charts für Objekterstellung.

subwin

[in] Nummer des Unterfensters für Objekterstellung.

name

[in] Objektnamen und die Grundlage für den Namen der grafischen Ressource.

x

[in] X-Koordinate des Objektbezugspunktes auf dem Chart..

y

[in] Y-Koordinate des Objektbezugspunktes auf dem Chart.

width

[in] Breite der grafischen Ressource (Größe entlang der Achse X) in Pixel.

height

[in] Höhe der grafischen Ressource (Größe entlang der Achse Y) in Pixel.

clrfmt=COLOR_FORMAT_XRGB_NOALPHA

© 2000-2025, MetaQuotes Ltd.

[in] Farbeverarbeitungsmethode. Lesen Sie mehr darüber in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Wenn Sie die erste Version der Funktion verwenden, wird das Objekt im Hauptfenster des aktuellen Charts erstellt.

Objektgröße ist dieselbe wie die Größe einer grafischen Ressource.

© 2000-2025, MetaQuotes Ltd.

Destroy

Löscht eine grafische Ressource.

```
void  Destroy();
```

Hinweis

Wenn eine grafische Ressource ist an ein Chartobjekt gebunden, wird das Chartobjekt gelöscht.

© 2000-2025, MetaQuotes Ltd.

Ellipse

Zeichnet ein Ellipse aufgrund zwei Punkten.

```
void  Ellipse(
   int          x1,      // X-Koordinate
   int          y1,      // Y-Koordinate
   int          x2,      // X-Koordinate
   int          y2,      // Y-Koordinate
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes der Ellipse.

y1

[in] Y-Koordinate des ersten Punktes der Ellipse.

x2

[in] X-Koordinate des zweiten Punktes der Ellipse.

y2

[in] X-Koordinate des zweiten Punktes der Ellipse.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

EllipseAA

Zeichnet ein Ellipse aufgrund zwei Punkten mit Anti-Aliasing.

```
void  EllipseAA(
   int          x1,      // X-Koordinate
   int          y1,      // Y-Koordinate
   int          x2,      // X-Koordinate
   int          y2,      // Y-Koordinate
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes der Ellipse.

y1

[in] Y-Koordinate des ersten Punktes der Ellipse.

x2

[in] X-Koordinate des zweiten Punktes der Ellipse.

y2

[in] X-Koordinate des zweiten Punktes der Ellipse.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

EllipseWu

Zeichnet ein Ellipse aufgrund zwei Punkten mit Wu's Anti-Aliasing.

```
void  EllipseWu(
   int          x1,       // X-Koordinate
   int          y1,       // Y-Koordinate
   int          x2,       // X-Koordinate
   int          y2,       // Y-Koordinate
   const uint   clr       // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes der Ellipse.

y1

[in] Y-Koordinate des ersten Punktes der Ellipse.

x2

[in] X-Koordinate des zweiten Punktes der Ellipse.

y2

[in] X-Koordinate des zweiten Punktes der Ellipse.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

Erase

Füllfarbe löscht oder füllt mit einer Farbe.

```
void  Erase(
   const uint  clr=0      // Farbe
   );
```

Parameter

clr=0

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

Fill

Füllt den Bereich.

```
void  Fill(
   int          x,        // X-Koordinate
   int          y,        // Y-Koordinate
   const uint   clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Startpunktes der Füllung.

y

[in] Y-Koordinate des Startpunktes der Füllung.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

FillCircle

Zeichnet einen gefüllten Kreis.

```
void  FillCircle(
   int          x,        // X-Koordinate
   int          y,        // Y-Koordinate
   int          r,        // Radius
   const uint   clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Kreismittelpunktes.

y

[in] Y-Koordinate des Kreismittelpunktes.

r

[in] Radius des Kreises.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

FillRectangle

Zeichnet ein gefülltes Rechteck.

```
void  FillRectangle(
   int          x1,      // X-Koordinate
   int          y1,      // Y-Koordinate
   int          x2,      // X-Koordinate
   int          y2,      // Y-Koordinate
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes des Rechtecks.

y1

[in] Y-Koordinate des ersten Punktes des Rechtecks

x2

[in] X-Koordinate des zweiten Punktes des Rechtecks.

y2

[in] X-Koordinate des zweiten Punktes des Rechtecks.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

FillTriangle

Zeichnet ein gefülltes Dreieck.

```
void  FillTriangle(
   int          x1,       // X-Koordinate
   int          y1,       // Y-Koordinate
   int          x2,       // X-Koordinate
   int          y2,       // Y-Koordinate
   int          x3,       // X-Koordinate
   int          y3,       // Y-Koordinate
   const uint   clr       // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Scheitelpunktes des Dreiecks.

y1

[in] Y-Koordinate des ersten Scheitelpunktes des Dreiecks.

x2

[in] X-Koordinate des zweiten Scheitelpunktes des Dreiecks.

y2

[in] Y-Koordinate des zweiten Scheitelpunktes des Dreiecks.

x3

[in] X-Koordinate des dritten Scheitelpunktes des Dreiecks.

y3

[in] Y-Koordinate des dritten Scheitelpunktes des Dreiecks.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

FontAngleGet

Erhält den Schriftwinkel.

```
uint  FontAngleGet();
```

Rückgabewert

Schriftwinkel

© 2000-2025, MetaQuotes Ltd.

FontAngleSet

Setzt den Schriftwinkel.

```
bool  FontAngleSet(
   uint  angle      // Winkel
   );
```

Parameter

angle

[in] Schriftwinkel in zehntel Grads.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

FontFlagsGet

Erhält die Schriftflags.

```
uint  FontFlagsGet();
```

Rückgabewert

Schriftflags

© 2000-2025, MetaQuotes Ltd.

FontFlagsSet

Setzt die Schriftflags.

```
bool  FontFlagsSet(
   uint  flags      // Flags
   );
```

Parameter

flags

[in] Flags von Schrifterstellung. Weitere Informationen zu Flags finden Sie in der Funktionsbeschreibung TextSetFont().

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

FontGet

Erhält die Schrifteigenschaften.

```
void  FontGet(
   string&  name,       // Name
   int&     size,       // Größe
   uint&    flags,      // Flags
   uint&    angle       // Winkel
   );
```

Parameter

name

[out] Ein Verweis auf Variable, um den Namen der Schrift zurückzugeben.

size

[out] Ein Verweis auf Variable, um die Größe der Schrift zurückzugeben.

flags

[out] Ein Verweis auf Variable, um die Schriftflags zurückzugeben.

angle

[out] Ein Verweis auf Variable, um den Schriftwinkel zurückzugeben.

© 2000-2025, MetaQuotes Ltd.

FontNameGet

Erhält den Schriftnamen.

```
string  FontNameGet();
```

Rückgabewert

Schriftnamen

© 2000-2025, MetaQuotes Ltd.

FontNameSet

Setzt den Schriftnamen.

```
bool  FontNameSet(
   string  name      // Name
   );
```

Parameter

name

[in] Schriftname Zum Beispiel "Arial".

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

FontSet

Setzt die aktuelle Schriftart.

```
bool  FontSet(
   const string  name,         // Name
   const int     size,         // Größe
   const uint    flags=0,      // Flags
   const uint    angle=0       // Winkel
   );
```

Parameter

name

[in] Schriftname Zum Beispiel "Arial".

size

[in] Schriftgröße. Weitere Informationen zu den Besonderheiten der Größenangabe finden Sie in der Funktionsbeschreibung TextSetFont().

flags=0

[in] Flags von Schrifterstellung. Weitere Informationen zu Flags finden Sie in der Funktionsbeschreibung TextSetFont().

angle=0

[in] Schriftwinkel in zehntel Grads.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

FontSizeGet

Erhält die Schriftgröße.

```
int  FontSizeGet();
```

Rückgabewert

Schriftgröße

© 2000-2025, MetaQuotes Ltd.

FontSizeSet

Setzt die Schriftgröße.

```
bool  FontSizeSet(
   int  size      // Größe
   );
```

Parameter

size

[in] Schriftgröße. Weitere Informationen zu den Besonderheiten der Größenangabe finden Sie in der Funktionsbeschreibung TextSetFont().

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Height

Erhält die Höhe der grafischen Ressource.

```
int  Height();
```

Rückgabewert

Die Höhe der grafischen Ressource

© 2000-2025, MetaQuotes Ltd.

Line

Zeichnet ein Liniensegment.

```
void  Line(
   int          x1,      // X-Koordinate
   int          y1,      // Y-Koordinate
   int          x2,      // X-Koordinate
   int          y2,      // Y-Koordinate
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes des Segments.

y1

[in] Y-Koordinate des ersten Punktes des Segments.

x2

[in] X-Koordinate des zweiten Punktes des Segments.

y2

[in] Y-Koordinate des zweiten Punktes des Segments.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

LineAA

Zeichnet einen Liniensegment mit Anti-Aliasing

```
void  LineAA(
   const int   x1,                  // X-Koordinate
   const int   y1,                  // Y-Koordinate
   const int   x2,                  // X-Koordinate
   const int   y2,                  // Y-Koordinate
   const uint  clr,                 // Farbe
   const uint  style=UINT_MAX       // Linienstil
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes des Segments.

y1

[in] Y-Koordinate des ersten Punktes des Segments.

x2

[in] X-Koordinate des zweiten Punktes des Segments.

y2

[in] Y-Koordinate des zweiten Punktes des Segments.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

LineWu

Zeichnet einen Liniensegment mit Wu's Anti-Aliasing.

```
void  LineWu(
   const int   x1,                  // X-Koordinate
   const int   y1,                  // Y-Koordinate
   const int   x2,                  // X-Koordinate
   const int   y2,                  // Y-Koordinate
   const uint  clr,                 // Farbe
   const uint  style=UINT_MAX       // Linienstil
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes des Segments.

y1

[in] Y-Koordinate des ersten Punktes des Segments.

x2

[in] X-Koordinate des zweiten Punktes des Segments.

y2

[in] Y-Koordinate des zweiten Punktes des Segments.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

LineHorizontal

Zeichne ein horizontales Liniensegment.

```
void  LineHorizontal(
   int          x1,       // X-Koordinate
   int          x2,       // X-Koordinate
   int          y,        // Y-Koordinate
   const uint   clr       // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes des Segments.

x2

[in] X-Koordinate des zweiten Punktes des Segments.

y

[in] Y-Koordinate des Segments.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

LineVertical

Zeichne ein vertikales Liniensegment.

```
void  LineVertical(
   int          x,        // X-Koordinate
   int          y1,       // Y-Koordinate
   int          y2,       // Y-Koordinate
   const uint   clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Segments.

y1

[in] Y-Koordinate des ersten Punktes des Segments.

y2

[in] Y-Koordinate des zweiten Punktes des Segments.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

LineStyleSet

Setzt den Linienstil.

```
void  LineStyleSet(
   const uint  style      // Stil
   );
```

Parameter

style

[in] Linienstil.

Hinweis

Der Eingangsparameter kann einer der Enumerationswerte ENUM_LINE_STYLE sein. Darüber hinaus ist es möglich, einen benutzerdefinierten Linienstil zu erstellen.

© 2000-2025, MetaQuotes Ltd.

LineThick

Zeichnet die Strecke einer zufälligen Linie mit der angegebenen Linienstärke mithilfe eines Glättungsalgorithmus.

```
void  LineThick(
   const int      x1,            // X-Koordinate des ersten Punktes der Strecke
   const int      y1,            // Y-Koordinate des ersten Punktes der Strecke
   const int      x2,            // X-Koordinate des zweiten Punktes der Strecke
   const int      y2,            // Y-Koordinate des zweiten Punktes der Strecke
   const uint     clr,           // Farbe
   const int      size,          // Linienstärke
   const uint     style,         // Linienstil
   ENUM_LINE_END  end_style      // Stil der Linienenden
   )
```

Parameter

x1

[in] X-Koordinate des ersten Punktes der Strecke.

y1

[in] Y-Koordinate des ersten Punktes der Strecke.

x2

[in] X-Koordinate des zweiten Punktes der Strecke.

y2

[in] Y-Koordinate des zweiten Punktes der Strecke.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style

[in] Stil der Enden der Linie — einer der Werte der ENUM_LINE_END Aufzählung

ENUM_LINE_END

Identifier	Beschreibung
LINE_END_ROUND	Linienenden werden abgerundet.
LINE_END_BUTT	Linienenden werden abgeschnitten.

© 2000-2025, MetaQuotes Ltd.

Identifier	Beschreibung
LINE_END_SQUARE	Die Linie endet mit einem mit Farbe gefüllten Rechteck.

© 2000-2025, MetaQuotes Ltd.

LineThickVertical

Zeichnet eine vertikale Strecke einer zufälligen Linie mit der angegebenen Linienstärke mithilfe eines Glättungsalgorithmus.

```
void  LineThickVertical(
   const int       x,              // X-Koordinate der Strecke
   const int       y1,             // Y-Koordinate des ersten Punktes der Strecke
   const int       y2,             // Y-Koordinate des zweiten Punktes der Strecke
   const uint      clr,            // Farbe
   const int       size,           // Linienstärke
   const uint      style,          // Linienstil
   ENUM_LINE_END   end_style       // Stil der Linienenden
   )
```

Parameter

x

[in] X-Koordinate der Strecke.

y1

[in] Y-Koordinate des ersten Punktes der Strecke.

y2

[in] Y-Koordinate des zweiten Punktes der Strecke.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style

[in] Stil der Linienenden — einer der Werte der ENUM_LINE_END Aufzählung.

© 2000-2025, MetaQuotes Ltd.

LineThickHorizontal

Zeichnet eine horizontale Strecke einer zufälligen Linie mit der angegebenen Linienstärke mit Glättung.

```
void  LineThickHorizontal(
   const int       x1,            // X-Koordinate des ersten Punktes der Strecke
   const int       x2,            // X-Koordinate des zweiten Punktes der Strecke
   const int       y,             // Y-Koordinate des ersten Punktes der Strecke
   const uint      clr,           // Farbe
   const int       size,          // Linienstärke
   const uint      style,         // Linienstil
   ENUM_LINE_END   end_style      // Stil der Linienenden
   )
```

Parameter

x1

[in] X-Koordinate des ersten Punktes der Strecke.

x2

[in] X-Koordinate des zweiten Punktes der Strecke.

y

[in] Y-Koordinate der Strecke.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style

[in] Stil der Linienenden — einer der Werte der ENUM_LINE_END Aufzählung.

© 2000-2025, MetaQuotes Ltd.

LoadFromFile

Liest eine Zeichnung aus einer BMP-Datei.

```
bool  LoadFromFile(
   const string  filename      // Dateiname
   );
```

Parameter

filename

[in] Dateiname (einschließlich der Erweiterung "BMP").

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

PixelGet

Liest die Farbe des Punktes mit den angegebenen Koordinaten.

```
uint  PixelGet(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   );
```

Parameter

x

[in] X-Koordinate des Punktes.

y

[in] Y-Koordinate des Punktes.

Rückgabewert

Punktfarbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

PixelSet

Setzt die Farbe des Punktes mit den angegebenen Koordinaten.

```
void  PixelSet(
   const int   x,       // X-Koordinate
   const int   y,       // Y-Koordinate
   const uint  clr      // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Punktes.

y

[in] Y-Koordinate des Punktes.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

PixelSetAA

Zeichnet einen Punkt mit Anti-Aliasing

```
void  PixelSetAA(
   const double  x,        // X-Koordinate
   const double  y,        // Y-Koordinate
   const uint    clr       // Farbe
   );
```

Parameter

x

[in] X-Koordinate des Punktes.

y

[in] Y-Koordinate des Punktes.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

Polygon

Zeichnet ein Polygon.

```
void  Polygon(
   int&         x[],     // Array der X-Koordinaten
   int&         y[],     // Array der Y-Koordinaten
   const uint   clr      // Farbe
   );
```

Parameter

x[]

[in] Array der X-Koordinaten des Polygons.

y[]

[in] Array der Y-Koordinaten des Polygons.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

PolygonAA

Zeichnet ein Polygon mit Anti-Aliasing

```
void  PolygonAA(
   int&         x[],                // Array der X-Koordinaten
   int&         y[],                // Array der Y-Koordinaten
   const uint   clr,                // Farbe
   const uint   style=UINT_MAX      // Linienstil
   );
```

Parameter

x[]

[in] Array der X-Koordinaten des Polygons.

y[]

[in] Array der Y-Koordinaten des Polygons.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

PolygonWu

Zeichnet ein Polygon mit Wu's Anti-Aliasing.

```
void  PolygonWu(
   int&         x[],                // Array der X-Koordinaten
   int&         y[],                // Array der Y-Koordinaten
   const uint   clr,                // Farbe
   const uint   style=UINT_MAX      // Linienstil
   );
```

Parameter

x[]

[in] Array der X-Koordinaten des Polygons.

y[]

[in] Array der Y-Koordinaten des Polygons.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

PolygonThick

Zeichnet ein Vieleck mit der angegebenen Dicke unter Verwendung eines Glättungsalgorithmus.

```
void  PolygonThick(
   const int&      x[],          // Array der X-Koordinaten der Punkte des Vielecks
   const int&      y[],          // Array der Y-Koordinaten der Punkte des Vielecks
   const uint      clr,          // Farbe
   const int       size,         // Linienstärke
   const uint      style,        // Linienstil
   ENUM_LINE_END   end_style     // Stil der Linienenden
   )
```

Parameter

x[]

[in] Array für X-Koordinaten der Punkte des Vielecks.

y[]

[in] Array für Y-Koordinaten der Punkte des Vielecks.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style

[in] Stil der Linienenden — einer der Werte der ENUM_LINE_END Aufzählung.

© 2000-2025, MetaQuotes Ltd.

PolygonSmooth

Zeichnet ein Vieleck mit der angegebenen Dicke unter Verwendung zweier Glättungsalgorithmen konsekutiv. Zuerst werden einzelne Strecken basierend auf Bézierkurven geglättet. Danach wird zwecks der Verbesserung der Qualität der Zeichnung auf das mit diesen Strecken gezeichnete Vieleck ein Raster-Algorithmus der Glättung angewandt.

```
void  PolygonSmooth(
   int&             x[],                        // Array der X-Koordinaten der Punkt
   int&             y[],                        // Array der Y-Koordinaten der Punkt
   const uint       clr,                        // Farbe
   const int        size,                       // Linienstärke
   ENUM_LINE_STYLE  style=STYLE_SOLID,          // Linienstil
   ENUM_LINE_END    end_style=LINE_END_ROUND,   // Stil der Linienenden
   double           tension=0.5,                // Wert des Glättungsparameters
   double           step=10                     // Länge der Näherungslinien
   )
```

Parameter

&x[]

[in] Array für X-Koordinaten der Punkte des Vielecks.

&y[]

[in] Array für Y-Koordinaten der Punkte des Vielecks.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style=STYLE_SOLID

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style=LINE_END_ROUND

[in] Stil der Linienenden — einer der Werte der ENUM_LINE_END Aufzählung.

tension=0.5

[in] Wert des Glättungsparameters.

step=10

[in] Länge der Näherungslinien.

© 2000-2025, MetaQuotes Ltd.

Polyline

Zeichnet eine Polylinie.

```
void  Polyline(
   int&         x[],     // Array der X-Koordinaten
   int&         y[],     // Array der Y-Koordinaten
   const uint   clr      // Farbe
   );
```

Parameter

x[]

[in] Array der X-Koordinaten des Polylinie.

y[]

[in] Array der Y-Koordinaten des Polylinie.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

PolylineSmooth

Zeichnet eine gebrochene Linie mit der angegebenen Linienstärke unter Verwendung zweier Glättungsalgorithmen konsekutiv. Zuerst werden einzelne Strecken der Linie basierend auf Bézierkurven geglättet. Danach wird zwecks der Verbesserung der Qualität der Zeichnung auf das mit diesen Strecken der gebrochenen Linie gezeichneten Geraden ein Raster-Algorithmus der Glättung angewandt.

```
void  PolylineSmooth(
   const int&        x[],                        // Array der X-Koordinate der Punkte
   const int&        y[],                        // Array der Y-Koordinate der Punkte
   const uint        clr,                        // Farbe
   const int         size,                       // Linienstärke
   ENUM_LINE_STYLE   style=STYLE_SOLID,          // Linienstil
   ENUM_LINE_END     end_style=LINE_END_ROUND,   // Stil Linienenden
   double            tension=0.5,                // Wert der Glättungsparameters
   double            step=10                     // Schritt der Approximation
   )
```

Parameter

&x[]

[in] Array der X-Koordinaten der Punkte der gebrochenen Linie.

&y[]

[in] Array der Y-Koordinaten der Punkte der gebrochenen Linie.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style=STYLE_SOLID

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style=LINE_END_ROUND

[in] Stil der Linienenden — einer der Werte der ENUM_LINE_END Aufzählung.

tension=0.5

[in] Wert des Glättungsparameters.

step=10

[in] Schritt der Approximation.

© 2000-2025, MetaQuotes Ltd.

PolylineThick

Zeichnet eine gebrochene Linie mit der angegebenen Stärke unter Verwendung eines Glättungsalgorithmus.

```
void  PolylineThick(
   const int      &x[],          // Array der X-Koordinaten der Punkte der gebrochener
   const int      &y[],          // Array der Y-Koordinaten der Punkte der gebrochener
   const uint     clr,           // Farbe
   const int      size,          // Linienstärke
   const uint     style,         // Linienstil
   ENUM_LINE_END  end_style      // Stil der Linienenden
   )
```

Parameter

&x[]

[in] Array der X-Koordinaten der Punkte der gebrochenen Linie.

&y[]

[in] Array der Y-Koordinaten der Punkte der gebrochenen Linie.

clr

[in] Farbe im ARGB-Format.

size

[in] Linienstärke.

style

[in] Linienstil — einer der Werte der ENUM_LINE_STYLE Aufzählung oder ein benutzerdefinierter Wert.

end_style

[in] Stil der Linienende — einer der Werte der ENUM_LINE_END Aufzählung

© 2000-2025, MetaQuotes Ltd.

PolylineWu

Zeichnet eine Polylinie mit Wu's Anti-Aliasing

```
void  PolylineWu(
   int&         x[],                // Array der X-Koordinaten
   int&         y[],                // Array der Y-Koordinaten
   const uint   clr,                // Farbe
   const uint   style=UINT_MAX      // Linienstil
   );
```

Parameter

x[]

[in] Array der X-Koordinaten des Polylinie.

y[]

[in] Array der Y-Koordinaten des Polylinie.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

PolylineAA

Zeichnet eine Polylinie mit Anti-Aliasing

```
void  PolylineAA(
   int&         x[],                // Array der X-Koordinaten
   int&         y[],                // Array der Y-Koordinaten
   const uint   clr,                // Farbe
   const uint   style=UINT_MAX      // Linienstil
   );
```

Parameter

x[]

[in] Array der X-Koordinaten des Polylinie.

y[]

[in] Array der Y-Koordinaten des Polylinie.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

Rectangle

Zeichnet ein Rechteck aufgrund zwei Punkten.

```
void  Rectangle(
   int          x1,      // X-Koordinate
   int          y1,      // Y-Koordinate
   int          x2,      // X-Koordinate
   int          y2,      // Y-Koordinate
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Punktes des Rechtecks.

y1

[in] Y-Koordinate des ersten Punktes des Rechtecks

x2

[in] X-Koordinate des zweiten Punktes des Rechtecks.

y2

[in] X-Koordinate des zweiten Punktes des Rechtecks.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

Resize

Ändert die Größe der grafischen Ressource.

```
bool  Resize(
   const int  width,       // Breite
   const int  height       // Höhe
   );
```

Parameter

width

[in] Die neue Breite der grafischen Ressource.

height

[in] Die neue Höhe der grafischen Ressource.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Wenn Sie die Größe ändern, wird das vorherige Bild nicht gespeichert.

© 2000-2025, MetaQuotes Ltd.

ResourceName

Erhält den Namen der grafischen Ressource.

```
string  ResourceName();
```

Rückgabewert

Der Name der grafischen Ressource

© 2000-2025, MetaQuotes Ltd.

TextHeight

Erhält die Höhe des Text.

```
int  TextHeight(
   const string  text      // Text
   );
```

Parameter

text

[in] Der Text, den Sie messen wollen.

Rückgabewert

Texthöhe in Pixel

Hinweis

Die aktuelle Schriftart wird für Textmessen verwendet.

© 2000-2025, MetaQuotes Ltd.

TextOut

Zeigt den Text.

```
void  TextOut(
   int          x,               // X-Koordinate
   int          y,               // Y-Koordinate
   string       text,            // Text
   const uint   clr,             // Farbe
   uint         alignment=0      // Alignment
   );
```

Parameter

x

[in] X-Koordinate des Ankerpunktes des Textes.

y

[in] Y-Koordinate des Ankerpunktes des Textes.

text

[in] Der Text, den Sie zeigen wollen.

clr

[in] Farbe in ARGB.

alignment=0

[in] Textbindungsmethode. Weitere Informationen zu Bindungsmethoden finden Sie in der Funktionsbeschreibung TextOut().

Hinweis

Die aktuelle Schriftart wird für Textanzeige verwendet.

© 2000-2025, MetaQuotes Ltd.

TextSize

Erhält die Größe des Textes.

```
void  TextSize(
   const string  text,        // Text
   int&          width,       // Breite
   int&          height       // Höhe
   );
```

Parameter

text

[in] Der Text, den Sie messen wollen.

width

[out] Ein Verweis auf Variable, um die Textbreite zurückzugeben.

height

[out] Ein Verweis auf Variable, um die Texthöhe zurückzugeben.

Hinweis

Die aktuelle Schriftart wird für Textmessen verwendet.

© 2000-2025, MetaQuotes Ltd.

TextWidth

Erhält die Breite des Text.

```
int  TextWidth(
   const string  text      // Text
   );
```

Parameter

text

[in] Der Text, den Sie messen wollen.

Rückgabewert

Textbreite in Pixel

Hinweis

Die aktuelle Schriftart wird für Textmessen verwendet.

© 2000-2025, MetaQuotes Ltd.

TransparentLevelSet

Setzt den Grad der Transparenz.

```
void  TransparentLevelSet(
   const uchar  value      // Wert
   );
```

Parameter

value

[in] Der neue Transparenzwert.

Hinweis

Ein Wert von 0 ist völlig transparent, 255 - die absolute Opazität.

Der gesetzte Grad der Transparenz gilt allen zuvor gezeichneten. Aber er gilt nicht allen anschließenden Bilder.

© 2000-2025, MetaQuotes Ltd.

Triangle

Zeichnet ein Dreieck.

```
void  Triangle(
   int          x1,      // X-Koordinate
   int          y1,      // Y-Koordinate
   int          x2,      // X-Koordinate
   int          y2,      // Y-Koordinate
   int          x3,      // X-Koordinate
   int          y3,      // Y-Koordinate
   const uint   clr      // Farbe
   );
```

Parameter

x1

[in] X-Koordinate des ersten Scheitelpunktes des Dreiecks.

y1

[in] Y-Koordinate des ersten Scheitelpunktes des Dreiecks.

x2

[in] X-Koordinate des zweiten Scheitelpunktes des Dreiecks.

y2

[in] Y-Koordinate des zweiten Scheitelpunktes des Dreiecks.

x3

[in] X-Koordinate des dritten Scheitelpunktes des Dreiecks.

y3

[in] Y-Koordinate des dritten Scheitelpunktes des Dreiecks.

clr

[in] Farbe in ARGB.

© 2000-2025, MetaQuotes Ltd.

TriangleAA

Zeichnet ein Dreieck mit Anti-Aliasing.

```
void  TriangleAA(
   const int   x1,                  // X-Koordinate
   const int   y1,                  // Y-Koordinate
   const int   x2,                  // X-Koordinate
   const int   y2,                  // Y-Koordinate
   const int   x3,                  // X-Koordinate
   const int   y3,                  // Y-Koordinate
   const uint  clr,                 // Farbe
   const uint  style=UINT_MAX       // Linienstil
   );
```

Parameter

x1

[in] X-Koordinate des ersten Scheitelpunktes des Dreiecks.

y1

[in] Y-Koordinate des ersten Scheitelpunktes des Dreiecks.

x2

[in] X-Koordinate des zweiten Scheitelpunktes des Dreiecks.

y2

[in] Y-Koordinate des zweiten Scheitelpunktes des Dreiecks.

x3

[in] X-Koordinate des dritten Scheitelpunktes des Dreiecks.

y3

[in] Y-Koordinate des dritten Scheitelpunktes des Dreiecks.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

TriangleWu

Zeichnet ein Dreieck mit Wu's Anti-Aliasing.

```
void  TriangleWu(
   const int   x1,                  // X-Koordinate
   const int   y1,                  // Y-Koordinate
   const int   x2,                  // X-Koordinate
   const int   y2,                  // Y-Koordinate
   const int   x3,                  // X-Koordinate
   const int   y3,                  // Y-Koordinate
   const uint  clr,                 // Farbe
   const uint  style=UINT_MAX       // Linienstil
   );
```

Parameter

x1

[in] X-Koordinate des ersten Scheitelpunktes des Dreiecks.

y1

[in] Y-Koordinate des ersten Scheitelpunktes des Dreiecks.

x2

[in] X-Koordinate des zweiten Scheitelpunktes des Dreiecks.

y2

[in] Y-Koordinate des zweiten Scheitelpunktes des Dreiecks.

x3

[in] X-Koordinate des dritten Scheitelpunktes des Dreiecks.

y3

[in] Y-Koordinate des dritten Scheitelpunktes des Dreiecks.

clr

[in] Farbe in ARGB.

style=UINT_MAX

[in] Linienstil - einer der Enumerationswerte ENUM_LINE_STYLE oder einen benutzerdefinierten Wert.

© 2000-2025, MetaQuotes Ltd.

Update

Zeigt Änderungen auf dem Bildschirm.

```
void  Update(
   const bool  redraw=true      // Flag
   );
```

Parameter

redraw=true

Ein Flag das bedeutet, dass der Chart neu gezeichnet werden muss.

© 2000-2025, MetaQuotes Ltd.

Width

Erhält die Breite der grafischen Ressource.

```
int  Width();
```

Rückgabewert

Die Breite der grafischen Ressource

© 2000-2025, MetaQuotes Ltd.

CChartCanvas

Basisklasse für die Umsetzung der Klassen für das Zeichnen von Charts und deren Elemente.

Beschreibung

Diese Klasse beinhaltet Methoden für das Arbeiten mit den Grundelementen jedes Charts: Koordinatenachsen und deren Beschriftung, Legende, Gitter, Hintergrund usw. Hier werden die Parameter der Anzeige von Elementen gesetzt: Sichtbarkeit/Unsichtbarkeit, Textfarbe usw.

Deklaration

```
class CChartCanvas : public CCanvas
```

Überschrift

```
#include <Canvas\Charts\ChartCanvas.mqh>
```

Vererbungshierarchie

CCanvas

CChartCanvas

Direkte Ableitungen

CHistogramChart, CLineChart, CPieChart

Methoden der Klasse

Methode	Aktion
ColorBackground	Liefert und setzt die Hintergrundfarbe.
ColorBorder	Liefert und setzt die Farbe der Grenzen.
ColorText	Liefert und setzt die Textfarbe.
ColorGrid	Liefert und setzt die Gitterfarbe.
MaxData	Liefert und setzt die maximale Anzahl zulässiger Daten (Reihen).
MaxDescrLen	Liefert und setzt die maximale Länge von Deskriptoren.
ShowFlags	Liefert und setzt das Flag der Sichtbarkeit von Elementen eines Charts.
IsShowLegend	Gibt das Flag der Sichtbarkeit der Legende auf dem Chart zurück.
IsShowScaleLeft	Gibt das Flag der Sichtbarkeit der Werteskala links zurück.

© 2000-2025, MetaQuotes Ltd.

Methode	Aktion
IsShowScaleRight	Gibt das Flag der Sichtbarkeit der Werteskala rechts zurück.
IsShowScaleTop	Gibt das Flag der Sichtbarkeit der Werteskala oben zurück.
IsShowScaleBottom	Gibt das Flag der Sichtbarkeit der Werteskala unten zurück.
IsShowGrid	Gibt das Flag der Sichtbarkeit des Gitters auf dem Chart zurück.
IsShowDescriptors	Gibt das Flag der Sichtbarkeit von Deskriptoren auf dem Chart zurück.
IsShowPercent	Gibt das Flag der Sichtbarkeit von Prozenten auf dem Chart zurück.
VScaleMin	Liefert und setzt das Minimum auf der vertikalen Werteskala.
VScaleMax	Liefert und setzt das Maximum auf der vertikalen Werteskala.
NumGrid	Liefert und setzt die Anzahl vertikaler Striche auf der Skala beim Zeichnen des Gitters des Charts.
DataOffset	Liefert und setzt den Wert des Datenoffsets.
DataTotal	Gibt die Gesamtzahl der Datenreihen im Chart zurück.
DrawDescriptors	Virtuelle Methode für das Zeichnen von Deskriptoren.
DrawData	Virtuelle Methode für das Zeichnen von Datenreihen nach dem angegebenen Index.
Create	Virtuelle Methode, die eine grafische Ressource erstellt.
AllowedShowFlags	Setzt den Set zulässiger Flags der Sichtbarkeit für Elemente des Charts.
ShowLegend	Setzt das Flag der Sichtbarkeit für die Legende.
ShowScaleLeft	Setzt das Flag der Sichtbarkeit für die linke Skala.
ShowScaleRight	Setzt das Flag der Sichtbarkeit für die rechte Skala.

© 2000-2025, MetaQuotes Ltd.

Methode	Aktion
ShowScaleTop	Setzt das Flag der Sichtbarkeit für die obere Skala.
ShowScaleBottom	Setzt das Flag der Sichtbarkeit für die untere Skala.
ShowGrid	Setzt das Flag der Sichtbarkeit für das Gitter.
ShowDescriptors	Setzt das Flag der Sichtbarkeit für Deskriptoren.
ShowValue	Setzt das Flag der Sichtbarkeit für Werte.
ShowPercent	Setzt das Flag der Sichtbarkeit für Prozente.
LegendAlignment	Setzt die Textausrichtung für die Legende.
Accumulative	Setzt das Flag für das Abspeichern von Werten für Reihen.
VScaleParams	Setzt die Parameter für die vertikale Werteskala.
DescriptorUpdate	Aktualisiert den Wert des Deskriptors der Reihe (nach der angegebenen Position).
ColorUpdate	Aktualisiert die Farben der Reihen (nach der angegebenen Position).
ValuesCheck	Führt interne Berechnungen für das Zeichnen einer Grafik durch.
Redraw	Zeichnet den Chart neu.
DrawBackground	Zeichnet den Hintergrund.
DrawLegend	Zeichnet die Legende neu.
DrawLegendVertical	Zeichnet die vertikale Legende.
DrawLegendHorizontal	Zeichnet die horizontale Legende.
CalcScales	Berechnet die Koordinaten der Skala.
DrawScales	Zeichnet alle Werteskalen neu.
DrawScaleLeft	Zeichnet die linke Werteskala neu.
DrawScaleRight	Zeichnet die rechte Werteskala neu.
DrawScaleTop	Zeichnet die obere Werteskala neu.
DrawScaleBottom	Zeichnet die untere Werteskala neu.
DrawGrid	Zeichnet das Gitter neu.
DrawChart	Zeichnet die Daten neu.

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CCanvas

CreateBitmap, CreateBitmap, CreateBitmapLabel, CreateBitmapLabel, Attach, Attach, Destroy, ChartObjectName, ResourceName, Width, Height, Update, Resize, Erase, PixelGet, PixelSet, LineVertical, LineHorizontal, Line, Polyline, Polygon, Rectangle, Triangle, Circle, Ellipse, Arc, Arc, Arc, Pie, Pie, FillRectangle, FillTriangle, FillPolygon, FillCircle, FillEllipse, Fill, Fill, PixelSetAA, LineAA, PolylineAA, PolygonAA, TriangleAA, CircleAA, EllipseAA, LineWu, PolylineWu, PolygonWu, TriangleWu, CircleWu, EllipseWu, LineThickVertical, LineThickHorizontal, LineThick, PolylineThick, PolygonThick, PolylineSmooth, PolygonSmooth, FontSet, FontNameSet, FontSizeSet, FontFlagsSet, FontAngleSet, FontGet, FontNameGet, FontSizeGet, FontFlagsGet, FontAngleGet, TextOut, TextWidth, TextHeight, TextSize, GetDefaultColor, TransparentLevelSet, LoadFromFile, LineStyleGet, LineStyleSet

© 2000-2025, MetaQuotes Ltd.

ColorBackground (Get-Methode)

Liefert die Hintergrundfarbe.

```
uint  ColorBackground()
```

Rückgabewert

Hintergrundfarbe.

ColorBackground (Set-Methode)

Setzt die Hintergrundfarbe.

```
void  ColorBackground(
  const uint  value,  // Hintergrundfarbe
  )
```

Parameter

value

[in] Hintergrundfarbe.

© 2000-2025, MetaQuotes Ltd.

ColorBorder (Get-Methode)

Gibt die Farbe der Grenzen zurück.

```
uint  ColorBorder()
```

Rückgabewert

Farbe der Grenzen.

ColorBorder (Set-Methode)

Setzt die Farbe der Grenzen.

```
void  ColorBorder(
  const uint  value,  // Farbe der Grenzen
  )
```

Parameter

value

[in] Farbe der Grenzen.

© 2000-2025, MetaQuotes Ltd.

ColorText (Get-Methode)

Gibt die Textfarbe zurück.

```
uint  ColorText()
```

Rückgabewert

Textfarbe.

ColorText (Set-Methode)

Setzt die Textfarbe.

```
void  ColorText(
  const uint  value,  // Textfarbe
  )
```

Parameter

value

[in] Farbe des Textes.

© 2000-2025, MetaQuotes Ltd.

ColorGrid (Get-Methode)

Gibt die Farbe des Gitters zurück.

```
uint  ColorGrid()
```

Rückgabewert

Farbe des Gitters.

ColorGrid (Set-Methode)

Setzt die Farbe des Gitters.

```
void  ColorGrid(
  const uint  value,  // Farbe des Gitters
  )
```

Parameter

value

[in] Farbe des Gitters.

© 2000-2025, MetaQuotes Ltd.

MaxData (Get-Methode)

Gibt die maximale Anzahl zulässiger Daten (Reihen) zurück.

```
uint  MaxData()
```

Rückgabewert

Maximale Anzahl der Daten (Reihen).

MaxData (Set-Methode)

Setzt die maximale Anzahl zulässiger Daten (Reihen).

```
void  MaxData(
  const uint  value,  // Anzahl der Daten
  )
```

Parameter

value

[in] Maximale Anzahl der Daten (Reihen).

© 2000-2025, MetaQuotes Ltd.

MaxDescrLen (Get-Methode)

Gibt die maximale Länge der Deskriptoren zurück.

```
uint  MaxDescrLen()
```

Rückgabewert

Wert der maximalen Länge der Deskriptoren.

MaxDescrLen (Set-Methode)

Setzt die maximale Länge der Deskriptoren.

```
void  MaxDescrLen(
  const uint  value,  // maximale Länge
  )
```

Parameter

value

[in] Wert der maximalen Länge der Deskriptoren.

© 2000-2025, MetaQuotes Ltd.

ShowFlags (Get-Methode)

Gibt das Flag der Sichtbarkeit von Elementen des Charts zurück.

```
bool  ShowFlags()
```

Rückgabewert

Wert des Flags der Sichtbarkeit von Elementen des Charts.

ShowFlags (Set-Methode)

Setzt das Flag der Sichtbarkeit von Elementen des Charts.

```
void  ShowFlags(
  const uint  flags,  // Flag
  )
```

Parameters

flags

[in] Wert des Flags der Sichtbarkeit der Elemente des Charts.

© 2000-2025, MetaQuotes Ltd.

IsShowLegend

Gibt das Flag der Sichtbarkeit der Legende auf dem Chart zurück.

```
bool  IsShowLegend()
```

Rückgabewert

true, wenn die Legende sichtbar ist, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

IsShowScaleLeft

Gibt das Flag der Sichtbarkeit der Werteskala links zurück.

```
bool  IsShowScaleLeft()
```

Rückgabewert

true, wenn die Werteskala sichtbar ist, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

IsShowScaleRight

Gibt das Flag der Sichtbarkeit der Werteskala rechts zurück.

```
bool  IsShowScaleRight()
```

Rückgabewert

true, wenn die Werteskala sichtbar ist, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

IsShowScaleTop

Gibt das Flag der Sichtbarkeit der Werteskala oben zurück.

```
bool  IsShowScaleTop()
```

Rückgabewert

true, wenn die Werteskala sichtbar ist, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

IsShowScaleBottom

Gibt das Flag der Sichtbarkeit der Werteskala unten zurück.

```
bool  IsShowScaleBottom()
```

Rückgabewert

true, wenn die Werteskala sichtbar ist, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

IsShowGrid

Gibt das Flag der Sichtbarkeit des Gitters auf dem Chart zurück.

```
bool  IsShowGrid()
```

Rückgabewert

true, wenn der Gitter sichtbar ist, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

IsShowDescriptors

Gibt das Flag der Sichtbarkeit von Deskriptoren auf dem Chart zurück.

```
bool  IsShowDescriptors()
```

Rückgabewert

true, wenn die Deskriptoren sichtbar sind, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

IsShowPercent

Gibt das Flag der Sichtbarkeit von Prozenten auf dem Chart zurück.

```
bool  IsShowPercent()
```

Rückgabewert

true, wenn die Prozente sichtbar sind, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

VScaleMin (Get-Methode)

Gibt das Minimum auf der vertikalen Werteskala zurück.

```
double  VScaleMin()
```

Rückgabewert

Mindestwert auf der vertikalen Skala.

VScaleMin (Set-Methode)

Setzt den Mindestwert auf der vertikalen Werteskala.

```
void  VScaleMin(
  const double  value,    // Wert auf der vertikalen Skala
  )
```

Parameter

value

[in] Mindestwert.

© 2000-2025, MetaQuotes Ltd.

VScaleMax

Gibt das Maximum auf der vertikalen Werteskala zurück.

```
double  VScaleMax()
```

Rückgabewert

Höchstwert auf der vertikalen Skala.

VScaleMax

Setzt das Maximum auf der vertikalen Werteskala.

```
void  VScaleMax(
  const double  value,  //   Wert auf der vertikalen Skala
  )
```

Parameter

value

[in] Maximaler Wert.

© 2000-2025, MetaQuotes Ltd.

NumGrid

Gibt die Anzahl der vertikalen Striche beim Zeichnen des Gitters des Charts zurück.

```
uint  NumGrid()
```

Rückgabewert

Anzahl der Striche.

NumGrid

Setzt die Anzahl der vertikalen Striche beim Zeichnen des Gitters des Charts.

```
void  NumGrid(
  const uint  value,  // Anzahl der Striche
  )
```

Parameter

value

[in] Anzahl der Striche.

© 2000-2025, MetaQuotes Ltd.

DataOffset

Gibt den Wert des Daten-Offsets zurück.

```
int  DataOffset()
```

Rückgabewert

Daten-Offset.

DataOffset

Setzt den Wert für den Daten-Offset.

```
void  DataOffset(
  const int  value,  // Wert
  )
```

Parameter

value

[in] Daten-Offset.

© 2000-2025, MetaQuotes Ltd.

DataTotal

Gibt die Gesamtzahl der Datenreihen im Chart zurück.

```
uint  DataTotal()
```

Rückgabewert

Anzahl der Reihen.

© 2000-2025, MetaQuotes Ltd.

DrawDescriptors

Virtuelle Methode für das Zeichnen von Deskriptoren.

```
virtual void  DrawDescriptors()
```

© 2000-2025, MetaQuotes Ltd.

DrawData

Virtuelle Methode für das Zeichnen von Datenreihen nach dem angegebenen Index.

```
virtual void  DrawData(
   const uint  idx=0,  // Index
   )
```

Parameter

idx=0

[in] Index der Reihe.

© 2000-2025, MetaQuotes Ltd.

Create

Virtuelle Methode, die eine grafische Ressource erstellt.

```
virtual bool  Create(
  const string       name,      // Name der Ressource
  const int          width,     // Breite
  const int          height,    // Höhe
  ENUM_COLOR_FORMAT  clrfmt,    // Format
  )
```

Parameter

name

[in] Basis für den Namen der grafischen Ressource. Der Name einer Ressource wird bei der Erstellung durch das Hinzufügen eines Pseudozufallsstrings gebildet.

width

[in] Breite (Größe an der X-Achse) in Pixel.

height

[in] Höhe (Größe an der Y-Achse) in Pixel.

clrfmt

[in] Methode der Verarbeitung der Farbe. Mehr Informationen über Methoden der Verarbeitung der Farbe finden Sie in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Wenn erfolgreich — true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

AllowedShowFlags

Setzt den Set zulässiger Flags der Sichtbarkeit für Elemente des Charts.

```
void  AllowedShowFlags(
  const uint  flags,  // Flags
  )
```

Parameter

flags

[in] Zulässige Flags.

© 2000-2025, MetaQuotes Ltd.

ShowLegend

Setzt den Wert des Flags der Sichtbarkeit für die Legende (FLAG_SHOW_LEGEND).

```
void  ShowLegend(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — die Legende ist sichtbar.
- false — die Legende ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowScaleLeft

Setzt den Wert des Flags der Sichtbarkeit für die linke Skala (FLAG_SHOW_SCALE_LEFT).

```
void  ShowScaleLeft(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — die linke Skala ist sichtbar.
- false — die linke Skala ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowScaleRight

Setzt den Wert des Flags der Sichtbarkeit für die rechte Skala (FLAG_SHOW_SCALE_RIGHT).

```
void  ShowScaleRight(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — die rechte Skala ist sichtbar.
- false — die rechte Skala ist sichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowScaleTop

Setzt den Wert des Flags der Sichtbarkeit für die obere Skala (FLAG_SHOW_SCALE_TOP).

```
void  ShowScaleTop(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — die obere Skala ist sichtbar.
- false — die obere Skala ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowScaleBottom

Setzt den Wert des Flags der Sichtbarkeit für die untere Skala(FLAG_SHOW_SCALE_BOTTOM).

```
void  ShowScaleBottom(
  const bool  flag,     // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — die untere Skala ist sichtbar.
- false — die untere Skala ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowGrid

Setzt den Wert des Flags der Sichtbarkeit für das Gitter (FLAG_SHOW_GRID).

```
void  ShowGrid(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — das Gitter ist sichtbar.
- false — das Gitter ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowDescriptors

Setzt den Wert des Flags der Sichtbarkeit für Deskriptoren (FLAG_SHOW_DESCRIPTORS).

```
void  ShowDescriptors(
  const bool  flag,    // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — der Deskriptor ist sichtbar.
- false — der Deskriptor ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowValue

Setzt das Flag der Sichtbarkeit für Werte (FLAG_SHOW_VALUE).

```
void  ShowValue(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — der Wert ist sichtbar.
- false — der Wert ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

ShowPercent

Setzt den Wert des Flags der Sichtbarkeit für Prozente (FLAG_SHOW_PERCENT).

```
void  ShowPercent(
  const bool  flag,  // Wert des Flags
  )
```

Parameter

flag

[in] Wert des Flags:

- true — das Prozent ist sichtbar.
- false — das Prozent ist unsichtbar.

© 2000-2025, MetaQuotes Ltd.

LegendAlignment

Setzt die Textausrichtung für die Legende.

```
void  LegendAlignment(
  const ENUM_ALIGNMENT  value,  // Flag
  )
```

Parameter

value

[in] Nimmt einen der Werte der Aufzählung ENUM_ALIGNMENT an:

- ALIGNMENT_LEFT — Ausrichtung links.
- ALIGNMENT_TOP — Ausrichtung oben.
- ALIGNMENT_RIGHT — Ausrichtung rechts.
- ALIGNMENT_BOTTOM — Ausrichtung unten.

© 2000-2025, MetaQuotes Ltd.

Accumulative

Setzt das Flag für das Abspeichern von Werten für Reihen.

```
void  Accumulative(
  const bool  flag=true,  // Wert des Flags
  )
```

Parameter

flag=true

[in] Wert des Flags:

- true — der aktuelle Wert der Reihe wird durch die Summe aller vorherigen Werte ersetzt.
- false — der Standardmodus des Zeichnens der Reihen.

© 2000-2025, MetaQuotes Ltd.

VScaleParams

Setzt die Parameter für die vertikale Werteskala.

```
void  VScaleParams(
  const double  max,   // Maximum
  const double  min,   // Minimum
  const uint    grid,  // Anzahl der Striche
  )
```

Parameter

max

[in] Minimaler Wert.

min

[in] Maximaler Wert.

grid

[in] Anzahl der Striche.

© 2000-2025, MetaQuotes Ltd.

DescriptorUpdate

Aktualisiert den Wert des Deskriptors der Reihe (nach der angegebenen Position).

```
bool  DescriptorUpdate(
  const uint    pos,    // Index
  const string  descr,  // Wert
  )
```

Parameter

pos

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

descr

[in] Wert des Deskriptors.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ColorUpdate

Aktualisiert die Farben der Reihen (nach der angegebenen Position).

```
bool  ColorUpdate(
  const uint  pos,  // Index
  const uint  clr,  // Farbe
  )
```

Parameter

pos

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

clr

[in] Farbwert.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValuesCheck

Virtuelle Hilfsmethode, führt interne Berechnungen für das Zeichnen eines Charts durch.

```
virtual void  ValuesCheck()
```

© 2000-2025, MetaQuotes Ltd.

Redraw

Virtuelle Methode für das Neuchzeichnen des Charts.

```
virtual void  Redraw()
```

© 2000-2025, MetaQuotes Ltd.

DrawBackground

Virtuelle Methode für das Neuzeichnen des Hintergrunds.

```
virtual void  DrawBackground()
```

© 2000-2025, MetaQuotes Ltd.

DrawLegend

Virtuelle Methode für das Neuzeichnen der Legende.

```
virtual void  DrawLegend()
```

© 2000-2025, MetaQuotes Ltd.

DrawLegendVertical

Zeichnet die vertikale Legende.

```
int  DrawLegendVertical(
  const int  w,  // Breite
  const int  h,  // Höhe
  )
```

Parameter

w

[in] Maximale Textbreite in der Legende.

h

[in] Maximale Texthöhe in der Legende.

Rückgabewert

Breite der Legende in Pixel.

© 2000-2025, MetaQuotes Ltd.

DrawLegendHorizontal

Zeichnet eine horizontale Legende.

```
int  DrawLegendHorizontal(
  const int  w,  //
  const int  h,  //
  )
```

Parameter

w

[in] Maximale Textbreite in der Legende.

h

[in] Maximale Texthöhe in der Legende.

Rückgabewert

Höhe der Legende in Pixel.

© 2000-2025, MetaQuotes Ltd.

CalcScales

Virtuelle Methode für die Berechnung von Koordinaten der Beschriftungen für die Werteskala.

```
virtual void  CalcScales()
```

© 2000-2025, MetaQuotes Ltd.

DrawScales

Virtuelle Methode für das Neuzeichnen aller Werteskalen.

```
virtual void  DrawScales()
```

© 2000-2025, MetaQuotes Ltd.

DrawScaleLeft

Virtuelle Methode für das Neuzeichnen der linken Werteskala.

```
virtual int  DrawScaleLeft(
  const bool  draw,  // Flag
  )
```

Parameter

draw

[in] Das Flag gibt an, ob die Skala neu gezeichnet werden muss.

Rückgabewert

Breite der Werteskala.

© 2000-2025, MetaQuotes Ltd.

DrawScaleRight

Virtuelle Methode für das Neuzeichnen der rechten Werteskala.

```
virtual int  DrawScaleRight(
  const bool  draw,  // Flag
  )
```

Parameter

draw

[in] Das Flag gibt an, ob die Skala neu gezeichnet werden muss.

Rückgabewert

Breite der Werteskala.

© 2000-2025, MetaQuotes Ltd.

DrawScaleTop

Virtuelle Methode für das Neuzeichnen der oberen Werteskala.

```
virtual int  DrawScaleTop(
  const bool  draw,  // Flag
  )
```

Parameter

draw

[in] Das Flag gibt an, ob die Skala neu gezeichnet werden muss.

Rückgabewert

Höhe der Werteskala.

© 2000-2025, MetaQuotes Ltd.

DrawScaleBottom

Virtuelle Methode für das Neuzeichnen der unteren Werteskala.

```
virtual int  DrawScaleBottom(
  const bool  draw,  // Flag
  )
```

Parameter

draw

[in] Das Flag gibt an, ob die Skala neu gezeichnet werden muss.

Rückgabewert

Höhe der Werteskala.

© 2000-2025, MetaQuotes Ltd.

DrawGrid

Virtuelle Methode für das Neuzeichnen des Gitters.

```
virtual void  DrawGrid()
```

© 2000-2025, MetaQuotes Ltd.

DrawChart

Virtuelle Methode für das Neuzeichnen des Charts.

```
virtual void  DrawChart()
```

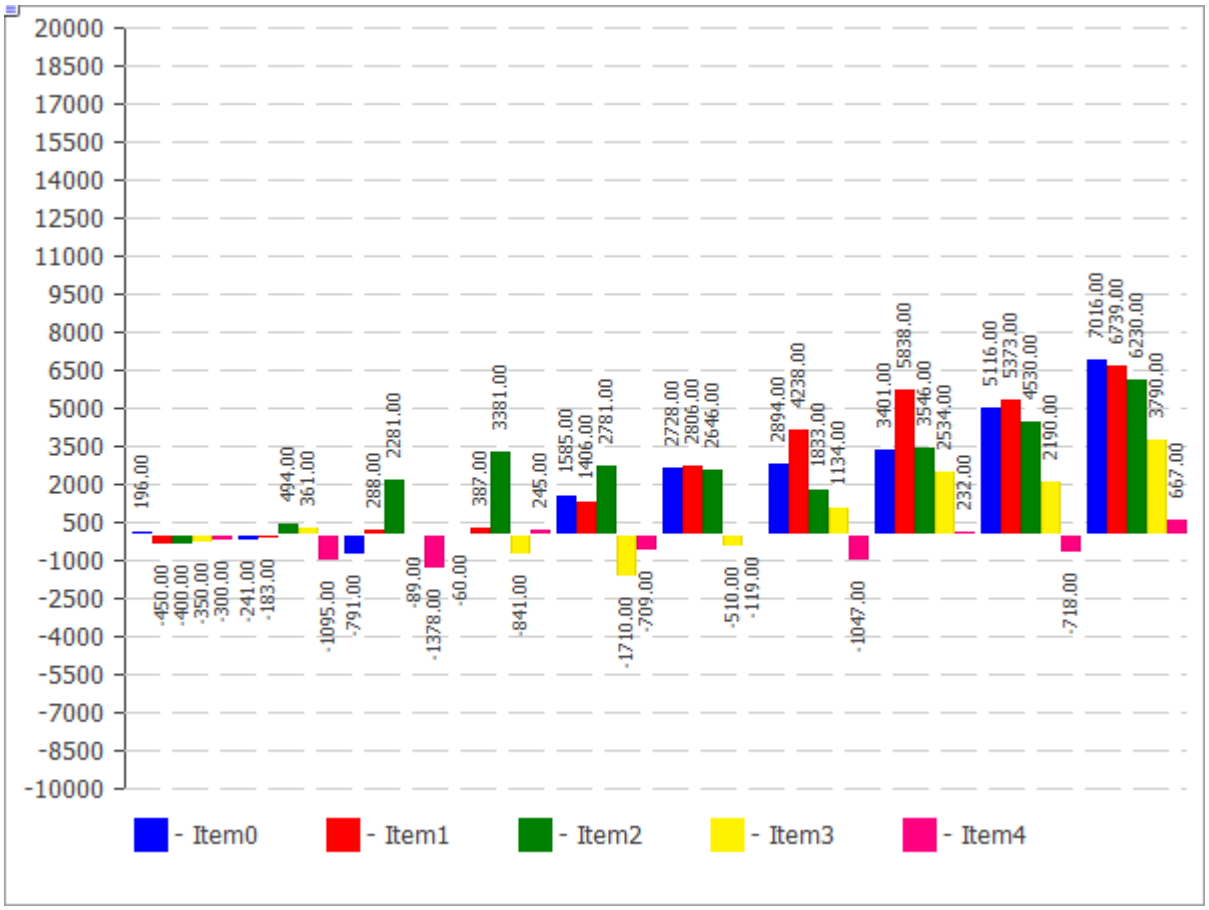
© 2000-2025, MetaQuotes Ltd.

CHistogramChart

Klasse für das Zeichnen von Histogrammen.

Beschreibung

In dieser Klasse sind alle Methoden für das Zeichnen von Balkenhistogrammen implementiert. Sie setzen die Breite der Balken und stellen das Arbeiten mit Datenreihen ein. Es sind die Methoden für das Arbeiten mit der graduellen Füllung von Balken des Histogramms einbezogen.



Der Code des oben angeführten Bildes ist unten dargestellt.

Deklaration

```
class CHistogramChart : public CChartCanvas
```

Überschrift

```
#include <Canvas\Charts\HistogramChart.mqh>
```

Vererbungshierarchie

CCanvas

CChartCanvas

CHistogramChart

© 2000-2025, MetaQuotes Ltd.

Methoden der Klasse

Methode	Aktion
Gradient	Setzt das Flag, das angibt, ob die graduelle Füllung für die Balken des Histogramms angewandt wird.
BarGap	Setzt den Wert des Abstands des Histogramms bis zum Koordinatenursprung.
BarMinSize	Setzt die minimale Breite der Balken des Histogramms.
BarBorder	Setzt das Flag, das auf die Notwendigkeit hinweist, Grenzen für jeden Balken zu zeichnen.
Create	Virtuelle Methode, die eine grafische Ressource erstellt.
SeriesAdd	Fügt eine neue Datenreihe hinzu.
SeriesInsert	Fügt eine Datenreihe auf den Chart ein.
SeriesUpdate	Aktualisiert die Datenreihe auf dem Chart.
SeriesDelete	Löscht eine Datenreihe vom Chart.
ValueUpdate	Aktualisiert den Wert des Elements in der angegebenen Reihe.
DrawData	Virtuelle Methode, die ein Histogramm für die angegebene Reihe zeichnet.
DrawBar	Zeichnet den Balken des Histogramms als ein mit Farbe gefülltes Rechteck.
GradientBrush	Erstellt einen Pinsel für graduelle Füllung.

Methoden geerbt von der Klasse CCanvas

CreateBitmap, CreateBitmap, CreateBitmapLabel, CreateBitmapLabel, Attach, Attach, Destroy, ChartObjectName, ResourceName, Width, Height, Update, Resize, Erase, PixelGet, PixelSet, LineVertical, LineHorizontal, Line, Polyline, Polygon, Rectangle, Triangle, Circle, Ellipse, Arc, Arc, Arc, Pie, Pie, FillRectangle, FillTriangle, FillPolygon, FillCircle, FillEllipse, Fill, Fill, PixelSetAA, LineAA, PolylineAA, PolygonAA, TriangleAA, CircleAA, EllipseAA, LineWu, PolylineWu, PolygonWu, TriangleWu, CircleWu, EllipseWu, LineThickVertical, LineThickHorizontal, LineThick, PolylineThick, PolygonThick, PolylineSmooth, PolygonSmooth, FontSet, FontNameSet, FontSizeSet, FontFlagsSet, FontAngleSet, FontGet, FontNameGet, FontSizeGet, FontFlagsGet, FontAngleGet, TextOut, TextWidth, TextHeight, TextSize, GetDefaultColor, TransparentLevelSet, LoadFromFile, LineStyleGet, LineStyleSet

Methoden geerbt von der Klasse CChartCanvas

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CCanvas

CreateBitmap, CreateBitmap, CreateBitmapLabel, CreateBitmapLabel, Attach, Attach, Destroy, ChartObjectName, ResourceName, Width, Height, Update, Resize, Erase, PixelGet, PixelSet, LineVertical, LineHorizontal, Line, Polyline, Polygon, Rectangle, Triangle, Circle, Ellipse, Arc, Arc, Arc, Pie, Pie, FillRectangle, FillTriangle, FillPolygon, FillCircle, FillEllipse, Fill, Fill, PixelSetAA, LineAA, PolylineAA, PolygonAA, TriangleAA, CircleAA, EllipseAA, LineWu, PolylineWu, PolygonWu, TriangleWu, CircleWu, EllipseWu, LineThickVertical, LineThickHorizontal, LineThick, PolylineThick, PolygonThick, PolylineSmooth, PolygonSmooth, FontSet, FontNameSet, FontSizeSet, FontFlagsSet, FontAngleSet, FontGet, FontNameGet, FontSizeGet, FontFlagsGet, FontAngleGet, TextOut, TextWidth, TextHeight, TextSize, GetDefaultColor, TransparentLevelSet, LoadFromFile, LineStyleGet, LineStyleSet

ColorBackground, ColorBackground, ColorBorder, ColorBorder, ColorText, ColorText, ColorGrid, ColorGrid, MaxData, MaxData, MaxDescrLen, MaxDescrLen, AllowedShowFlags, ShowFlags, ShowFlags, IsShowLegend, IsShowScaleLeft, IsShowScaleRight, IsShowScaleTop, IsShowScaleBottom, IsShowGrid, IsShowDescriptors, IsShowPercent, ShowLegend, ShowScaleLeft, ShowScaleRight, ShowScaleTop, ShowScaleBottom, ShowGrid, ShowDescriptors, ShowValue, ShowPercent, LegendAlignment, Accumulative, VScaleMin, VScaleMin, VScaleMax, VScaleMax, NumGrid, NumGrid, VScaleParams, DataOffset, DataOffset, DataTotal, DescriptorUpdate, ColorUpdate

Beispiel

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                          HistogramChartSample.mq5 |
//|                  Copyright 2009-2017, MetaQuotes Software Corp. |
//|                                              http://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "2009-2017, MetaQuotes Software Corp."
#property link        "http://www.mql5.com"
#property description "Example of using histogram"
//---
#include <Canvas\Charts\HistogramChart.mqh>
//+------------------------------------------------------------------+
//| inputs                                                           |
//+------------------------------------------------------------------+
input bool Accumulative=true;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
int OnStart(void)
  {
   int k=100;
   double arr[10];
//--- create chart
   CHistogramChart chart;
   if(!chart.CreateBitmapLabel("SampleHistogramChart",10,10,600,450))
     {
      Print("Error creating histogram chart: ",GetLastError());
      return(-1);
     }
   if(Accumulative)
     {
      chart.Accumulative();
      chart.VScaleParams(20*k*10,-10*k*10,20);
     }
   else
      chart.VScaleParams(20*k,-10*k,20);
   chart.ShowValue(true);
   chart.ShowScaleTop(false);
   chart.ShowScaleBottom(false);
   chart.ShowScaleRight(false);
   chart.ShowLegend();
   for(int j=0;j<5;j++)
     {
      for(int i=0;i<10;i++)
        {
         k=-k;
         if(k>0)
            arr[i]=k*(i+10-j);
         else
            arr[i]=k*(i+10-j)/2;
        }
      chart.SeriesAdd(arr,"Item"+IntegerToString(j));
     }
//--- play with values
   while(!IsStopped())
     {
      int i=rand()%5;
      int j=rand()%10;
      k=rand()%3000-1000;
      chart.ValueUpdate(i,j,k);
      Sleep(200);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- finish
   chart.Destroy();
   return(0);
  }
```

© 2000-2025, MetaQuotes Ltd.

Gradient

Setzt das Flag, das angibt, ob die graduelle Füllung für die Balken des Histogramms angewandt wird.

```
void  Gradient(
  const bool  flag=true,  // Wert des Flags
  )
```

Parameter

flag=true

Der Wert des Flags: true, wenn die graduelle Füllung aktiviert ist, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

BarGap

Setzt den Wert des Abstands des Histogramms bis zum Koordinatenursprung.

```
void  BarGap(
  const uint  value,  // Abstand
  )
```

Parameter

value

[in] Abstand des Histogramms.

© 2000-2025, MetaQuotes Ltd.

BarMinSize

Setzt die minimale Breite der Balken des Histogramms.

```
void  BarMinSize(
  const uint  value,  // minimale Breite
  )
```

Parameter

value

[in] Minimale Breite.

© 2000-2025, MetaQuotes Ltd.

BarBorder

Setzt das Flag, das auf die Notwendigkeit hinweist, Grenzen für jeden Balken zu zeichnen.

```
void  BarBorder(
  const uint  value,  // Flag
  )
```

Parameter

value

[in] Wert des Flags:

- true — die Grenzen werden gezeichnet
- false — die Grenzen werden nicht gezeichnet

© 2000-2025, MetaQuotes Ltd.

Create

Virtuelle Methode, die eine grafische Ressource erstellt.

```
virtual bool  Create(
  const string       name,    // Name
  const int          width,   // Breite
  const int          height,  // Höhe
  ENUM_COLOR_FORMAT  clrfmt,  // Format
  )
```

Parameter

name

[in] Basis für den Namen der grafischen Ressource. Der Name einer Ressource wird bei der Erstellung durch das Hinzufügen eines Pseudozufallsstrings gebildet.

width

[in] Breite (Größe an der X-Achse) in Pixel.

height

[in] Höhe (Größe an der Y-Achse) in Pixel.

clrfmt

[in] Methode der Verarbeitung der Farbe. Mehr Informationen über Methoden der Verarbeitung der Farbe finden Sie in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Wenn erfolgreich — true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesAdd

Fügt eine neue Datenreihe hinzu.

```
bool  SeriesAdd(
  const double&  value[],  // Werte
  const string   descr,    // Label
  const uint     clr,      // Farbe
  )
```

Parameter

value[]

[in] Datenreihe.

descr

[in] Label der Reihe.

clr

[in] Anzeigefarbe der Reihe.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesInsert

Fügt eine Datenreihe auf den Chart ein.

```
bool  SeriesInsert(
  const uint    pos,        // Index
  const double&  value[],   // Werte
  const string  descr,      // Label
  const uint    clr,        // Farbe
  )
```

Parameter

pos

[in] Index für das Einfügen.

value[]

[in] Datenreihe.

descr

[in] Label der Reihe.

clr

[in] Anzeigefarbe der Reihe.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesUpdate

Aktualisiert die Datenreihe auf dem Chart.

```
bool  SeriesUpdate(
  const uint    pos,        // Index
  const double  &value[],   // Werte
  const string  descr,      // Label
  const uint    clr,        // Farbe
  )
```

Parameter

pos

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

&value[]

[in] Neue Werte für die Datenreihe.

descr

[in] Label der Reihe.

clr

[in] Anzeigefarbe der Reihe.

Rückgabewert

true — wenn erfolgreich, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesDelete

Löscht eine Datenreihe vom Chart.

```
bool  SeriesDelete(
  const uint  pos,  // Index
  )
```

Parameter

pos

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValueUpdate

Aktualisiert den angegebenen Wert in der angegebenen Reihe.

```
bool  ValueUpdate(
  const uint   series,  // Index der Reihe
  const uint   pos,     // Index des Elements
  double       value,   // Wert
  )
```

Parameter

series

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

pos

[in] Index des Elements in der Reihe.

value

[in] Neuer Wert.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

DrawData

Virtuelle Methode, die ein Histogramm für die angegebene Reihe zeichnet.

```
virtual void  DrawData(
  const uint  index,  // Index
  )
```

Parameter

index

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

© 2000-2025, MetaQuotes Ltd.

DrawBar

Zeichnet den Balken des Histogramms als ein mit Farbe gefülltes Rechteck.

```
void  DrawBar(
  const int   x,    // x-Koordinate
  const int   y,    // y-Koordinate
  const int   w,    // Breite
  const int   h,    // Höhe
  const uint  clr,  // Farbe
  )
```

Parameter

x

[in] X-Koordinate des linken oberen Punktes des Rechtecks.

y

[in] Y-Koordinate des linken oberen Punktes des Rechtecks.

w

[in] Breite des Rechtecks.

h

[in] Höhe des Rechtecks.

clr

[in] Farbe des Rechtecks.

© 2000-2025, MetaQuotes Ltd.

GradientBrush

Erstellt einen Pinsel für graduelle Füllung.

```
void  GradientBrush(
  const int   size,      // Größe
  const uint  fill_clr,  // Füllfarbe
  )
```

Parameter

size

[in] Pinselstärke

fill_clr

[in] Füllfarbe

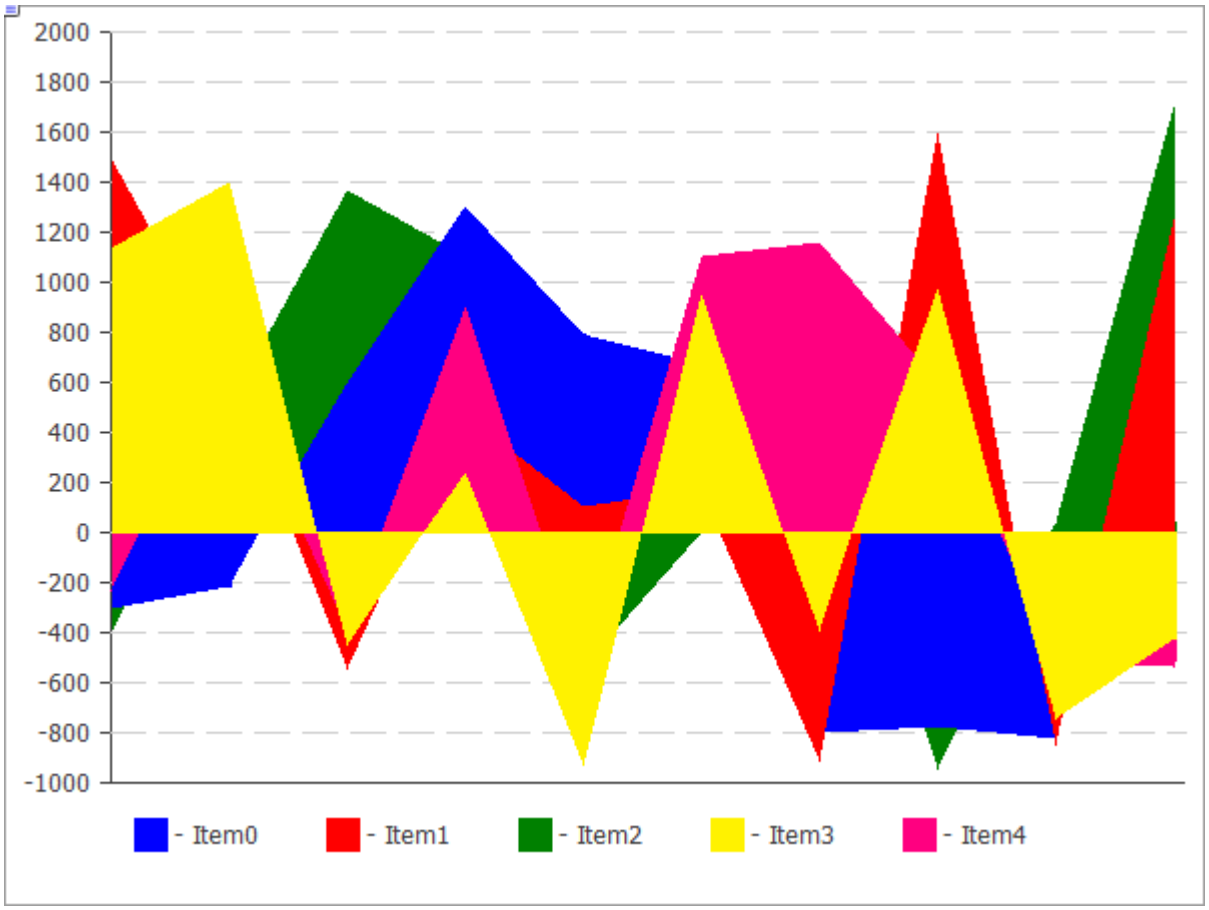
© 2000-2025, MetaQuotes Ltd.

CLineChart

Klasse für das Zeichnen von Kurven.

Beschreibung

Die Methoden, die zu dieser Klasse gehören, sind für Operationen mit Kurven auf dem Chart bestimmt. Es ist die Option aktiviert, den durch die Kurve begrenzten Bereich mit Farbe zu füllen.



Der Code des oben angeführten Bildes ist unten angeführt.

Deklaration

```
class CLineChart : public CChartCanvas
```

Überschrift

```
#include <Canvas\Charts\LineChart.mqh>
```

Vererbungshierarchie

CCanvas
  CChartCanvas
    CLineChart

Methoden der Klasse

© 2000-2025, MetaQuotes Ltd.

Method	Action
Filled	Setzt das Flag der Füllung für den Bereich unterhalb der Kurve, die durch eine Datenreihe gesetzt wurde.
Create	Erstellt eine grafische Ressource.
SeriesAdd	Fügt eine neue Datenreihe hinzu.
SeriesInsert	Fügt eine Datenreihe auf den Chart ein.
SeriesUpdate	Aktualisiert die Datenreihe auf dem Chart.
SeriesDelete	Löscht die Datenreihe vom Chart.
ValueUpdate	Aktualisiert den angegebenen Wert in der angegebenen Reihe.
DrawChart	Virtuelle Methode, die die Kurve und alle deren Elemente zeichnet.
DrawData	Virtuelle Methode, die die Kurve für die angegebene Reihe zeichnet.
CalcArea	Berechnet die Fläche unterhalb der Kurve, die durch eine Datenreihe gesetzt wurde.

Methoden geerbt von der Klasse CCanvas

CreateBitmap, CreateBitmap, CreateBitmapLabel, CreateBitmapLabel, Attach, Attach, Destroy, ChartObjectName, ResourceName, Width, Height, Update, Resize, Erase, PixelGet, PixelSet, LineVertical, LineHorizontal, Line, Polyline, Polygon, Rectangle, Triangle, Circle, Ellipse, Arc, Arc, Arc, Pie, Pie, FillRectangle, FillTriangle, FillPolygon, FillCircle, FillEllipse, Fill, Fill, PixelSetAA, LineAA, PolylineAA, PolygonAA, TriangleAA, CircleAA, EllipseAA, LineWu, PolylineWu, PolygonWu, TriangleWu, CircleWu, EllipseWu, LineThickVertical, LineThickHorizontal, LineThick, PolylineThick, PolygonThick, PolylineSmooth, PolygonSmooth, FontSet, FontNameSet, FontSizeSet, FontFlagsSet, FontAngleSet, FontGet, FontNameGet, FontSizeGet, FontFlagsGet, FontAngleGet, TextOut, TextWidth, TextHeight, TextSize, GetDefaultColor, TransparentLevelSet, LoadFromFile, LineStyleGet, LineStyleSet

Methoden geerbt von der Klasse CChartCanvas

ColorBackground, ColorBackground, ColorBorder, ColorBorder, ColorText, ColorText, ColorGrid, ColorGrid, MaxData, MaxData, MaxDescrLen, MaxDescrLen, AllowedShowFlags, ShowFlags, ShowFlags, IsShowLegend, IsShowScaleLeft, IsShowScaleRight, IsShowScaleTop, IsShowScaleBottom, IsShowGrid, IsShowDescriptors, IsShowPercent, ShowLegend, ShowScaleLeft, ShowScaleRight, ShowScaleTop, ShowScaleBottom, ShowGrid, ShowDescriptors, ShowValue, ShowPercent, LegendAlignment, Accumulative, VScaleMin, VScaleMin, VScaleMax, VScaleMax, NumGrid, NumGrid, VScaleParams, DataOffset, DataOffset, DataTotal, DescriptorUpdate, ColorUpdate

Beispiel

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                LineChartSample.mq5 |
//|                  Copyright 2009-2017, MetaQuotes Software Corp. |
//|                                              http://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "2009-2017, MetaQuotes Software Corp."
#property link        "http://www.mql5.com"
#property description "Example of using line chart"
//---
#include <Canvas\Charts\LineChart.mqh>
//+------------------------------------------------------------------+
//| inputs                                                           |
//+------------------------------------------------------------------+
input bool Accumulative=false;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
int OnStart(void)
  {
   int k=100;
   double arr[10];
//--- create chart
   CLineChart chart;
//--- create chart
   if(!chart.CreateBitmapLabel("SampleHistogrammChart",10,10,600,450))
     {
      Print("Error creating line chart: ",GetLastError());
      return(-1);
     }
   if(Accumulative)
     {
      chart.Accumulative();
      chart.VScaleParams(20*k*10,-10*k*10,20);
     }
   else
      chart.VScaleParams(20*k,-10*k,15);
   chart.ShowScaleTop(false);
   chart.ShowScaleRight(false);
   chart.ShowLegend();
   chart.Filled();
   for(int j=0;j<5;j++)
     {
      for(int i=0;i<10;i++)
        {
         k=-k;
         if(k>0)
            arr[i]=k*(i+10-j);
         else
            arr[i]=k*(i+10-j)/2;
        }
      chart.SeriesAdd(arr,"Item"+IntegerToString(j));
     }
//--- play with values
   while(!IsStopped())
     {
      int i=rand()%5;
      int j=rand()%10;
      k=rand()%3000-1000;
      chart.ValueUpdate(i,j,k);
      Sleep(200);
     }
```

© 2000-2025, MetaQuotes Ltd.

```
//--- finish
   chart.Destroy();
   return(0);
  }
```

© 2000-2025, MetaQuotes Ltd.

Filled

Setzt das Flag, das angibt, ob der Bereich unterhalb der Kurve, welche durch eine Datenreihe gesetzt wurde, gefärbt werden muss.

```
void  Filled(
  const bool  flag=true,  // Flag
  )
```

Parameter

flag=true

[in] Wert des Flags:

- true — den Bereich unterhalb der Kurve färben
- false — den Bereich unterhalb der Kurve nicht färben

© 2000-2025, MetaQuotes Ltd.

Create

Virtuelle Methode, die eine grafische Ressource erstellt.

```
virtual bool  Create(
  const string       name,    // Name
  const int          width,   // Breite
  const int          height,  // Höhe
  ENUM_COLOR_FORMAT  clrfmt,  // Format
  )
```

Parameter

name

[in] Basis für den Namen der grafischen Ressource. Der Name einer Ressource wird bei der Erstellung durch das Hinzufügen eines Pseudozufallsstrings gebildet.

width

[in] Breite (Größe an der X-Achse) in Pixel.

height

[in] Höhe (Größe an der Y-Achse) in Pixel.

clrfmt

[in] Methode der Verarbeitung der Farbe. Mehr Informationen über Methoden der Verarbeitung der Farbe finden Sie in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Wenn erfolgreich — true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesAdd

Fügt eine neue Datenreihe hinzu.

```
bool  SeriesAdd(
  const double&  value[],  // Werte
  const string   descr,    // Label
  const uint     clr,      // Farbe
  )
```

Parameter

value[]

[in] Datenreihe.

descr

[in] Label der Reihe.

clr

[in] Anzeigefarbe der Reihe.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesInsert

Fügt eine Datenreihe auf den Chart ein.

```
bool  SeriesInsert(
  const uint     pos,        // Index
  const double&  value[],    // Werte
  const string   descr,      // Label
  const uint     clr,        // Farbe
  )
```

Parameter

pos

[in] Index für das Einfügen.

value[]

[in] Datenreihe.

descr

[in] Label der Reihe.

clr

[in] Anzeigefarbe der Reihe.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesUpdate

Aktualisiert die Datenreihe auf dem Chart.

```
bool  SeriesUpdate(
  const uint     pos,        // Index
  const double&  value[],    // Werte
  const string   descr,      // Label
  const uint     clr,        // Farbe
  )
```

Parameter

pos

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

value[]

[in] Neue Werte für die Datenreihe.

descr

[in] Label der Reihe.

clr

[in] Anzeigefarbe der Reihe.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesDelete

Löscht die Datenreihe vom Chart.

```
bool  SeriesDelete(
  const uint  pos,  // Index
  )
```

Parameter

pos

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValueUpdate

Aktualisiert den angegebenen Wert in der angegebenen Reihe.

```
bool  ValueUpdate(
  const uint   series,  // Index der Reihe
  const uint   pos,     // Index des Elements
  double       value,   // Wert
  )
```

Parameter

series

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

pos

[in] Index des Elements in der Reihe.

value

[in] Neuer Wert.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

DrawChart

Virtuelle Methode, die die Kurve und alle deren Elemente zeichnet.

```
virtual void  DrawChart()
```

© 2000-2025, MetaQuotes Ltd.

DrawData

Virtuelle Methode, die die Kurve für die angegebene Reihe zeichnet.

```
virtual void  DrawData(
  const uint  index,  // Index
  )
```

Parameters

index

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

© 2000-2025, MetaQuotes Ltd.

CalcArea

Berechnet die Fläche unterhalb der Kurve, die durch eine Datenreihe gesetzt wurde.

```
double  CalcArea(
  const uint  index,  // Index
  )
```

Parameter

index

[in] Index der Reihe — die Nummer, unter welcher sie hinzugefügt wurde, ab 0.

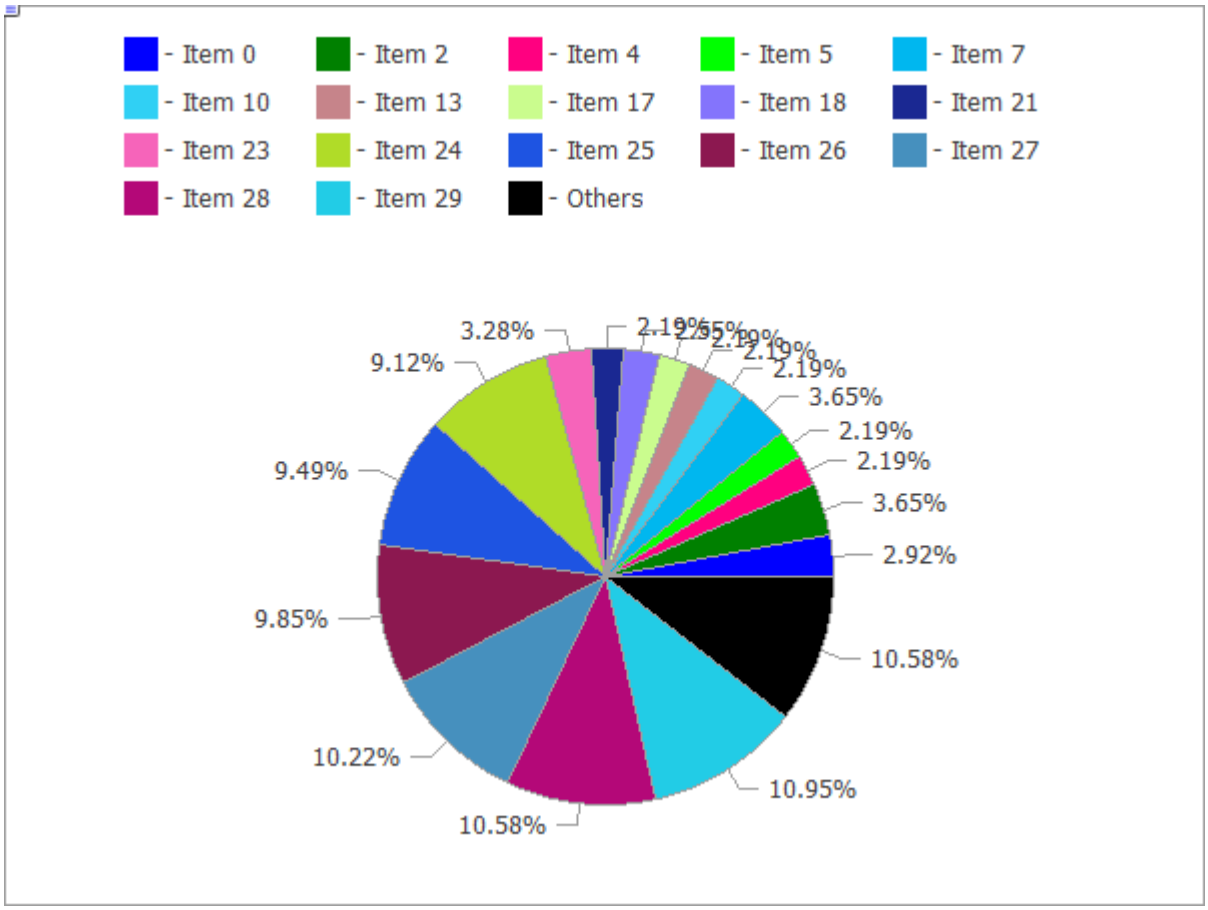
Rückgabewert

Die Fläche der Figur, die durch eine Datenreihe gesetzte Kurve begrenzt.

© 2000-2025, MetaQuotes Ltd.

CPieChart

Klasse für das Zeichnen von Kuchendiagrammen.



Der Code des oben angeführten Bildes ist unten dargestellt.

Beschreibung

Die Methoden, die zu dieser Klasse gehören, sind für Operationen mit Kuchendiagrammen bestimmt: von der Erstellung einer grafischen Ressource bis zur Beschriftung von Segmenten.

Deklaration

```
class CPieChart : public CChartCanvas
```

Überschrift

```
#include <Canvas\Charts\PieChart.mqh>
```

Vererbungshierarchie

CCanvas
  CChartCanvas
    CPieChart

Methoden der Klasse

© 2000-2025, MetaQuotes Ltd.

Methode	Aktion
Create	Virtuelle Methode, die eine grafische Ressource erstellt.
SeriesSet	Setzt die Wertereihe, die auf dem Diagramm angezeigt werden.
ValueAdd	Fügt einen neuen Wert auf das Diagramm (am Ende) hinzu.
ValueInsert	Fügt einen neuen Wert in das Diagramm ein (nach der angegebenen Position).
ValueUpdate	Aktualisiert den Wert auf dem Diagramm (nach der angegebenen Position).
ValueDelete	Löscht einen Wert vom Diagramm (nach der angegebenen Position).
DrawChart	Virtuelle Methode, die das Diagramm und alle dessen Elemente zeichnet.
DrawPie	Zeichnet ein Segment des Diagramms, das einem bestimmten Wert entspricht.
LabelMake	Generiert das Label des Segments basierend auf seinem Wert und dem ursprünglichen Label.

Methoden geerbt von der Klasse CCanvas

CreateBitmap, CreateBitmap, CreateBitmapLabel, CreateBitmapLabel, Attach, Attach, Destroy, ChartObjectName, ResourceName, Width, Height, Update, Resize, Erase, PixelGet, PixelSet, LineVertical, LineHorizontal, Line, Polyline, Polygon, Rectangle, Triangle, Circle, Ellipse, Arc, Arc, Arc, Pie, Pie, FillRectangle, FillTriangle, FillPolygon, FillCircle, FillEllipse, Fill, Fill, PixelSetAA, LineAA, PolylineAA, PolygonAA, TriangleAA, CircleAA, EllipseAA, LineWu, PolylineWu, PolygonWu, TriangleWu, CircleWu, EllipseWu, LineThickVertical, LineThickHorizontal, LineThick, PolylineThick, PolygonThick, PolylineSmooth, PolygonSmooth, FontSet, FontNameSet, FontSizeSet, FontFlagsSet, FontAngleSet, FontGet, FontNameGet, FontSizeGet, FontFlagsGet, FontAngleGet, TextOut, TextWidth, TextHeight, TextSize, GetDefaultColor, TransparentLevelSet, LoadFromFile, LineStyleGet, LineStyleSet

Methoden geerbt von der Klasse CChartCanvas

ColorBackground, ColorBackground, ColorBorder, ColorBorder, ColorText, ColorText, ColorGrid, ColorGrid, MaxData, MaxData, MaxDescrLen, MaxDescrLen, AllowedShowFlags, ShowFlags, ShowFlags, IsShowLegend, IsShowScaleLeft, IsShowScaleRight, IsShowScaleTop, IsShowScaleBottom, IsShowGrid, IsShowDescriptors, IsShowPercent, ShowLegend, ShowScaleLeft, ShowScaleRight, ShowScaleTop, ShowScaleBottom, ShowGrid, ShowDescriptors, ShowValue, ShowPercent, LegendAlignment, Accumulative, VScaleMin, VScaleMin, VScaleMax, VScaleMax, NumGrid, NumGrid, VScaleParams, DataOffset, DataOffset, DataTotal, DescriptorUpdate, ColorUpdate

© 2000-2025, MetaQuotes Ltd.

Beispiel

```
//+------------------------------------------------------------------+
//|                                                PieChartSample.mq5 |
//|                  Copyright 2009-2017, MetaQuotes Software Corp. |
//|                                              http://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright   "2009-2017, MetaQuotes Software Corp."
#property link        "http://www.mql5.com"
#property description "Example of using pie chart"
//---
#include <Canvas\Charts\PieChart.mqh>
//+------------------------------------------------------------------+
//| inputs                                                           |
//+------------------------------------------------------------------+
input int      Width=600;
input int      Height=450;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
int OnStart(void)
  {
//--- check
   if(Width<=0 || Height<=0)
     {
      Print("Too simple.");
      return(-1);
     }
//--- create chart
   CPieChart pie_chart;
   if(!pie_chart.CreateBitmapLabel("PieChart",10,10,Width,Height))
     {
      Print("Error creating pie chart: ",GetLastError());
      return(-1);
     }
   pie_chart.ShowPercent();
//--- draw
   for(uint i=0;i<30;i++)
     {
      pie_chart.ValueAdd(100*(i+1),"Item "+IntegerToString(i));
      Sleep(10);
     }
   Sleep(2000);
//--- disable legend
   pie_chart.LegendAlignment(ALIGNMENT_LEFT);
   Sleep(2000);
//--- disable legend
   pie_chart.LegendAlignment(ALIGNMENT_RIGHT);
   Sleep(2000);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- disable legend
   pie_chart.LegendAlignment(ALIGNMENT_TOP);
   Sleep(2000);
//--- disable legend
   pie_chart.ShowLegend(false);
   Sleep(2000);
//--- disable percentage
   pie_chart.ShowPercent(false);
   Sleep(2000);
//--- disable descriptors
   pie_chart.ShowDescriptors(false);
   Sleep(2000);
//--- enable all
   pie_chart.ShowLegend();
   pie_chart.ShowValue();
   pie_chart.ShowDescriptors();
   Sleep(2000);
//--- or like this
   pie_chart.ShowFlags(FLAG_SHOW_LEGEND|FLAG_SHOW_DESCRIPTORS|FLAG_SHOW_PERCENT);
   uint total=pie_chart.DataTotal();
//--- play with values
   for(uint i=0;i<total && !IsStopped();i++)
     {
      pie_chart.ValueUpdate(i,100*(rand()%10+1));
      Sleep(1000);
     }
//--- play with colors
   for(uint i=0;i<total && !IsStopped();i++)
     {
      pie_chart.ColorUpdate(i%total,RandomRGB());
      Sleep(1000);
     }
//--- rotate
   while(!IsStopped())
     {
      pie_chart.DataOffset(pie_chart.DataOffset()+1);
      Sleep(200);
     }
//--- finish
   pie_chart.Destroy();
   return(0);
  }
//+------------------------------------------------------------------+
//| Random RGB color                                                 |
//+------------------------------------------------------------------+
uint RandomRGB(void)
  {
   return(XRGB(rand()%255,rand()%255,rand()%255));
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Virtuelle Methode, die eine grafische Ressource erstellt.

```
virtual bool  Create(
  const string       name,    // Name
  const int          width,   // Breite
  const int          height,  // Höhe
  ENUM_COLOR_FORMAT  clrfmt,  // Format
  )
```

Parameter

name

[in] Basis für den Namen der grafischen Ressource. Der Name einer Ressource wird bei der Erstellung durch das Hinzufügen eines Pseudozufallsstrings gebildet.

width

[in] Breite (Größe an der X-Achse) in Pixel.

height

[in] Höhe (Größe an der Y-Achse) in Pixel.

clrfmt

[in] Methode der Verarbeitung der Farbe. Mehr Informationen über Methoden der Verarbeitung der Farbe finden Sie in der Beschreibung der Funktion ResourceCreate().

Rückgabewert

Wenn erfolgreich — true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

SeriesSet

Setzt die Wertereihe, die auf dem Diagramm angezeigt werden.

```
bool  SeriesSet(
  const double&  value[],  // Werte
  const string&  text[],   // Label
  const uint&    clr[],    // Farben
  )
```

Parameter

value[]

[in] Array der Werte.

text[]

[in] Array der Labels der Werte.

clr[]

[in] Array der Farben der Werte.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValueAdd

Fügt einen neuen Wert auf das Diagramm (am Ende) hinzu.

```
bool  ValueAdd(
  const double  value,  // Wert
  const string  descr,  // Label
  const uint    clr,    // Farbe
  )
```

Parameter

value

[in] Wert.

descr

[in] Label des Wertes.

clr

[in] Farbe des Wertes.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValueInsert

Fügt einen neuen Wert in das Diagramm ein (nach der angegebenen Position).

```
bool  ValueInsert(
  const uint    pos,    // Index
  const double  value,  // Wert
  const string  descr,  // Label
  const uint    clr,    // Farbe
  )
```

Parameter

pos

[in] Index für das Einfügen.

value

[in] Wert.

descr

[in] Label der Werte.

clr

[in] Farbe des Wertes.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValueUpdate

Aktualisiert den Wert auf dem Diagramm (nach der angegebenen Position).

```
bool  ValueUpdate(
  const uint    pos,    // Index
  const double  value,  // Wert
  const string  descr,  // Label
  const uint    clr,    // Farbe
  )
```

Parameter

pos

[in] Index des Wertes — die Nummer, unter welcher er hinzugefügt wurde, ab 0.

value

[in] Wert.

descr

[in] Label des Wertes.

clr

[in] Farbe des Wertes.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

ValueDelete

Löscht einen Wert vom Diagramm (nach der angegebenen Position).

```
bool  ValueDelete(
  const uint  pos,  // Index
  )
```

Parameter

pos

[in] Index des Wertes — die Nummer, unter welcher er hinzugefügt wurde, ab 0.

Rückgabewert

Wenn erfolgreich true, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

DrawChart

Virtuelle Methode, die das Diagramm und alle dessen Elemente zeichnet.

```
virtual void  DrawChart()
```

© 2000-2025, MetaQuotes Ltd.

DrawPie

Zeichnet ein Segment des Diagramms, das einem bestimmten Wert entspricht.

```
void  DrawPie(
  double      fi3,   // Winkel eines Strahls aus dem Zentrum des Diagramms, der die e
  double      fi4,   // Winkel eines Strahls aus dem Zentrum des Diagramms, der die z
  int         idx,   // Index
  CPoint&     p[],  //
  const uint  clr,   //
  )
```

Parameter

fi3

[in] Winkel in Radianten, der die erste Grenze des Bogens definiert.

fi4

[in] Winkel in Radianten, der die zweite Grenze des Bogens definiert.

idx

[in] Index des Wertes, welchem das Segment entspricht.

p[]

[in] Array für Referenzpunkte (x, y) für das Zeichnen von Segmenten.

clr

[in] Farbe des Segments.

© 2000-2025, MetaQuotes Ltd.

LabelMake

Generiert das Label des Segments basierend auf seinem Wert und dem ursprünglichen Label.

```
string  LabelMake(
  const string  text,     // Label
  const double  value,    // Wert
  const bool    to_left,  // Flag
  )
```

Parameter

text

[in] Label.

value

[in] Wert.

to_left

[in] Bestimmt die Reihenfolge der Zusammensetzung des Labels:

- true — Label, danach der Wert.
- false — Wert, danach das Label.

Rückgabewert

Label des Segments.

© 2000-2025, MetaQuotes Ltd.

3D Grafik

Der Abschnitt bietet die Klassen für die Entwicklung dreidimensionaler Grafiken. Die Klassen basieren auf den Funktionen Funktionen für die Arbeit mit DirectX. CCnavas3D ist eine Basisklasse, die die Methoden für die Verwaltung von Kamera und Beleuchtung sowie den Manager der Grafikressourcen - Texturen, Shader, Vertex-Puffer, Indizes und Shader-Parameter - enthält.

Um die Arbeit mit der Bibliothek zu beginnen, lesen Sie einfach den Artikel Wie man 3D-Grafiken mit DirectX in MetaTrader 5 erstellt.

Daneben gibt es die Klassen der Szenenbasisobjekte, wie z.B. eine Box, eine dreidimensionale Oberfläche für die Benutzerdaten und ein beliebiges Raster.

© 2000-2025, MetaQuotes Ltd.

CCanvas3D

CCanvas3D ist eine Klasse zur vereinfachten Erstellung und Visualisierung von 3D-Objekten auf einem Chart.

Beschreibung

CCanvas3D vereinfacht erheblich das Erstellen und Visualisieren großer Datenmengen in Form von animierten 3D-Grafiken. Die Klasse enthält die Methoden zur Verwaltung von Kamera und Beleuchtung sowie den Ressourcenmanager zur Erstellung von Grafikressourcen: Texturen, Shader, Vertex-Puffer, Indizes und Shader-Parameter.

Außerdem enthält die Bibliothek die Klassen der Szenenbasisobjekte, wie z.B. eine Box, eine dreidimensionale Oberfläche auf Benutzerdaten oder ein beliebiges Gitter.

Eine Grafikkarte sollte DX 11 und Shader Model 5.0 unterstützen, damit die Funktionen funktionieren.

Deklaration

```
class CCanvas
```

Titel

```
#include <Canvas\Canvas.mqh>
```

Vererbungshierarchie

CCanvas

CCanvas3D

Gruppierte Klassenmethoden

Erstellen/Löschen	Beschreibung
Erstellen	Erstellt eine grafische Ressource zum Rendern einer 3D-Szene ohne Bindung an ein Chartobjekt.
Anhängen	Ruft eine grafische Ressource aus dem Objekt OBJ_BITMAP_LABEL ab und hängt sie an eine Instanz der Klasse CCanvas an.
ObjectAdd	Fügt für das spätere Rendern ein Objekt in eine 3D-Szene ein.
Löschen	Zerstört eine grafische Ressource und gibt den grafischen 3D-Kontext frei.
Licht	
AmbientColorSet	Legt die Farbe und Intensität der Umgebungsbeleuchtung fest.
AmbientColorGet	Bestimmt die Farbe und Intensität der Umgebungsbeleuchtung.
LightDirectionSet	Legt die Richtung einer gerichteten Lichtquelle fest.
LightDirectionGet	Ermittelt die Richtung einer gerichteten Lichtquelle.

© 2000-2025, MetaQuotes Ltd.

Erstellen/Löschen	Beschreibung
LightColorSet	Legt die Farbe und Intensität einer gerichteten Lichtquelle fest.
LightColorGet	Ermittelt die Farbe und Intensität einer gerichteten Lichtquelle.
Kamera	
ProjectionMatrixSet	Berechnet und setzt eine 3D-Koordinatenprojektionsmatrix auf einen 2D-Rahmen.
ProjectionMatrixGet	Ruft eine 3D-Szenenprojektionsmatrix auf einen 2D-Rahmen ab.
ViewMatrixSet	Legt eine 3D-Szenenansichtsmatrix fest.
ViewMatrixGet	Gibt eine 3D-Szenenansichtsmatrix zurück.
ViewPositionSet	Setzt einen Betrachtungsstandpunkt auf eine 3D-Szene.
ViewRotationSet	Legt die Richtung eines Blicks auf eine 3D-Szene fest.
ViewTargetSet	Legt die Koordinaten des Punktes fest, auf den ein Blick gerichtet ist.
ViewUpDirectionSet	Legt die Richtung des oberen Rahmenrandes im 3D-Raum fest.
Rendern	
Render	Rendert alle Szenenobjekte im Frame-Innenpuffer für die spätere Anzeige.
RenderBegin	Bereitet einen grafischen Kontext für das Rendern eines neuen Rahmens vor.
RenderEnd	Kopiert einen gerenderten Rahmen in den inneren Puffer und aktualisiert gegebenenfalls ein Bild auf dem Chart.
Abrufen von Ressourcen	
DXContext	Ruft das Handle des grafische Kontext ab.
DXDispatcher	Ruft das Handle des Ressourcen-Dispatchers ab.
InputScene	Ruft den Zeiger auf den Puffer der Szenenparameter ab.

© 2000-2025, MetaQuotes Ltd.

AmbientColorGet

Bestimmt die Farbe und Intensität der Umgebungsbeleuchtung.

```
void  AmbientColorGet(
   DXColor  &ambient_color      // Farben und Intensität der Umgebungsbeleuchtung
   );
```

Parameter

&ambient_color

[out] Farben der Umgebungsbeleuchtung.

Rückgabewert

Keiner

Hinweis

Die Intensität wird im Alphakanal der Struktur DXColor gespeichert.

© 2000-2025, MetaQuotes Ltd.

AmbientColorSet

Legt die Farbe und Intensität der Umgebungsbeleuchtung fest.

```
void  AmbientColorSet(
   const DXColor  &ambient_color      // Farben und Intensität der Umgebungsbeleuchtur
   );
```

Parameter

&ambient_color

[in] Farben der Umgebungsbeleuchtung.

Rückgabewert

Nr

Hinweis

Die Intensität wird im Alphakanal der Struktur DXColor bestimmt.

© 2000-2025, MetaQuotes Ltd.

Anhängen

Ruft die grafische Ressource des Objekts OBJ_BITMAP_LABEL ab und hängt sie an eine Instanz der Klasse CCanvas an.

```
bool  Attach(
   const long          chart_id,                          // Chart-ID
   const string        objname,                           // Objektname
   ENUM_COLOR_FORMAT   clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Methode zur Farbbehand
   )
```

Erzeugt eine grafische Ressource für ein Objekt OBJ_BITMAP_LABEL und hängt es an eine Instanz der Klasse CCanvas an.

```
bool  Attach(
   const long          chart_id,                          // Chart-ID
   const string        objname,                           // Objektname
   const int           width,                             // Bildbreite in Pixel
   const int           height,                            // Bildhöhe in Pixel
   ENUM_COLOR_FORMAT   clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Methode zur Farbbehand
   )
```

Parameter

chart_id

[in] Chart-ID.

objname

[in] Name des Grafikobjekts.

width

[in] Frame-Breite in einer Ressource.

height

[in] Frame-Höhe.

clrfmt=COLOR_FORMAT_XRGB_NOALPHA

[in] Methode zur Farbbehandlung. In der Funktionsbeschreibung zu ResourceCreate() finden Sie mehr über die Methoden der Farbhandhabung.

Hinweis

true - bei Erfolg, false - wenn das Grafikobjekt nicht hinzugefügt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Erstellen

Erstellt eine grafische Ressource zum Rendern einer 3D-Szene ohne Bindung an ein Chartobjekt.

```
virtual bool  Create(
   const string       name,                              // Name des Grafikobjekts
   const int          width,                             // Breite
   const int          height,                            // Höhe
   ENUM_COLOR_FORMAT  clrfmt=COLOR_FORMAT_XRGB_NOALPHA   // Farbformat
   );
```

Parameter

name

[in] Name des Grafikobjekts.

width

[in] Frame-Breite.

height

[in] Frame-Höhe.

clrfmt=COLOR_FORMAT_XRGB_NOALPHA

[in] Methode zur Farbbehandlung. In der Funktionsbeschreibung zu ResourceCreate() finden Sie mehr über die Methoden der Farbhandhabung.

Hinweis

true - wenn eine Ressource erstellt wurde, andernfalls - false.

© 2000-2025, MetaQuotes Ltd.

Löschen

Zerstört eine grafische Ressource und gibt den grafischen 3D-Kontext frei.

```
virtual void  Destroy()
```

Rückgabewert

Nr

Hinweis

Wenn eine grafische Ressource an ein Chartobjekt gebunden wurde, wird dieses gelöscht.

© 2000-2025, MetaQuotes Ltd.

DXContext

Ruft das Handle des grafische Kontext ab.

```
int  DXContext()
```

Rückgabewert

Handle des grafischen Kontext.

© 2000-2025, MetaQuotes Ltd.

DXDispatcher

Ruft das Handle des Ressourcen-Dispatchers ab.

```
CDXDispatcher*  DXDispatcher()
```

Rückgabewert

Handle des Resource-Dispatchers.

© 2000-2025, MetaQuotes Ltd.

InputScene

Ruft den Zeiger auf den Puffer der Szenenparameter ab.

```
CDXInput*  InputScene()
```

Rückgabewert

Pointer auf den Puffer mit den Parametern der Szene.

© 2000-2025, MetaQuotes Ltd.

LightColorGet

Ermittelt die Farbe und Intensität einer gerichteten Lichtquelle.

```
void  LightColorGet(
   DXColor  &light_color      // Richtung des Lichts der Farben und Intensität
   );
```

Parameter

&light_color

[out] Richtung des Lichts der Farben und Intensität.

Rückgabewert

Keiner.

Hinweis

Die Intensität wird im Alphakanal der Struktur DXColor gespeichert.

© 2000-2025, MetaQuotes Ltd.

LightColorSet

Legt die Farbe und Intensität einer gerichteten Lichtquelle fest.

```
void  LightColorSet(
   const DXColor  &light_color      // Richtuing des Lichts der Farbe und Intensität
   );
```

Parameter

&light_color

[in] Richtung des Lichts der Farben und Intensität.

Rückgabewert

Keiner.

Hinweis

Die Intensität wird im Alphakanal der Struktur DXColor bestimmt.

© 2000-2025, MetaQuotes Ltd.

LightDirectionGet

Ermittelt die Richtung einer gerichteten Lichtquelle.

```
void  LightDirectionGet(
   DXVector3  &light_direction      // Richtungsvektor
   );
```

Parameter

&light_direction

[out] Richtungsvektor.

Rückgabewert

Keiner.

© 2000-2025, MetaQuotes Ltd.

LightDirectionSet

Legt die Richtung einer gerichteten Lichtquelle fest.

```
void  LightDirectionSet(
   const DXVector3  &light_direction      // Richtungsvektor
   );
```

Parameter

&light_direction

[in] Richtungsvektor.

Rückgabewert

Keiner.

© 2000-2025, MetaQuotes Ltd.

ObjectAdd

Fügt für das spätere Rendern ein Objekt in eine 3D-Szene ein.

```
bool  ObjectAdd(
   CDXObject  *object      // Pointer auf das Objekt
   );
```

Parameter

object

[in] Pointer auf eine Instanz der Klasse, abgeleitet von der abstrakten Klasse CDXObject.

Rückgabewert

true - bei Erfolg, false - wenn das Grafikobjekt nicht hinzugefügt werden konnte.

© 2000-2025, MetaQuotes Ltd.

ProjectionMatrixGet

Ruft eine 3D-Szenenprojektionsmatrix auf einen 2D-Rahmen ab.

```
void  ProjectionMatrixGet(
   DXMatrix  &projection_matrix      // Projektionsmatrix
   );
```

Parameter

&projection_matrix

[out] Projektionsmatrix.

Rückgabewert

Keiner.

© 2000-2025, MetaQuotes Ltd.

ProjectionMatrixSet

Berechnet und setzt eine 3D-Koordinatenprojektionsmatrix auf einen 2D-Rahmen.

```
void  ProjectionMatrixSet(
   float  fov,              // Sichtfeld
   float  aspect_ratio,     // Seitenverhältnis des Frames
   float  z_near,           //
   float  z_far             //
   );
```

Parameter

fov

[in] Breite des Sichtfelds in Bogenmaß, um die Szenenprojektion zu erstellen.

aspect_ratio

[in] Seitenverhältnis des 2D-Frames.

z_near

[in] Abstand zur nahen Schnittebene.

z_far

[in] Abstand zur entfernten Schnittebene.

Rückgabewert

Keiner.

Hinweis

Der 2D-Rahmen zeigt nur Projektionen von 3D-Objekten an, die in das angegebene Sichtfeld fallen und sich zwischen der nahen und der fernen Schnittebene befinden.

© 2000-2025, MetaQuotes Ltd.

Render

Rendert alle Szenenobjekte im Frame-Innenpuffer für die spätere Anzeige.

```
bool  Render(
   uint  flags,                  // Kombination der Flags
   uint  background_color=0      // Hintergrundfarbe
   );
```

Parameter

flags

[in] Kombination der Flags, um die Render-Methode festzulegen. Mögliche Werte:
DX_CLEAR_COLOR - löschen des Bildpuffers mittels *background_color*.
DX_CLEAR_DEPTH - löschen des Tiefen-Puffers.

background_color=0

[in] Hintergrundfarbe der 3D-Szene.

Rückgabewert

true - bei Erfolg, false - wenn das Rendern fehlgeschlagen ist.

Hinweis

Der Aufruf von Render() aktualisiert nicht die Szene auf dem Chart. Stattdessen wird nur der innere Puffer des Bildes aktualisiert. Die Methode Update() sollte explizit aufgerufen werden, um den Frame zu aktualisieren.

Render() beinhaltet die Aufrufe von RenderBegin und RenderEnd().

© 2000-2025, MetaQuotes Ltd.

RenderBegin

Bereitet einen grafischen Kontext für das Rendern eines neuen Rahmens vor.

```
virtual bool  RenderBegin(
   uint  flags,                  // Kombination der Flags
   uint  background_color=0      // Hintergrundfarbe
   );
```

Parameter

flags

[in] Kombination der Flags, um die Render-Methode festzulegen. Mögliche Werte:
DX_CLEAR_COLOR - löschen des Bildpuffers mittels *background_color*.
DX_CLEAR_DEPTH - löschen des Tiefen-Puffers.

background_color=0

[in] Hintergrundfarbe der 3D-Szene.

Rückgabewert

true - bei Erfolg, false - wenn das Aktualisieren der Shader-Eingaben fehlgeschlagen ist.

© 2000-2025, MetaQuotes Ltd.

RenderEnd

Kopiert einen gerenderten Rahmen in den inneren Puffer und aktualisiert gegebenenfalls ein Bild auf dem Chart.

```
virtual bool  RenderEnd(
   bool  redraw=false      // aktualisieren der Flags
   );
```

Parameter

redraw=false

[in] Flag für die Notwendigkeit den Chart neu zu zeichnen.

Rückgabewert

true - bei Erfolg, andernfalls - false.

© 2000-2025, MetaQuotes Ltd.

ViewMatrixGet

Gibt eine 3D-Szenenansichtsmatrix zurück.

```
void  ViewMatrixGet(
   DXMatrix  &view_matrix      // Matrix der Ansicht
   );
```

Parameter

&view_matrix

[out] Die Einstellungen der Kameraposition der Matrix der Ansicht im 3D-Raum.

Rückgabewert

Keiner.

© 2000-2025, MetaQuotes Ltd.

ViewMatrixSet

Legt eine 3D-Szenenansichtsmatrix fest.

```
void  ViewMatrixSet(
   const DXMatrix  &view_matrix      // Matrix der Ansicht
   );
```

Parameter

&view_matrix

[in] Festlegen der Kameraposition und -richtung der Matrix der Ansicht.

Rückgabewert

Keiner.

© 2000-2025, MetaQuotes Ltd.

ViewPositionSet

Setzt einen Betrachtungsstandpunkt auf eine 3D-Szene.

```
void  ViewPositionSet(
   const DXVector3  &position      // Standpunkt der Betrachtung
   );
```

Parameter

&position

[in] Festlegen des Standpunkts der Betrachtung in einer 3D-Szene.

Rückgabewert

Keiner.

Hinweis

Das Bestimmen des Standpunkts der Betrachtung mittels ViewPositionSet() ändert die in ViewMatrixGet() erhaltene Ansichtsmatrix.

© 2000-2025, MetaQuotes Ltd.

ViewRotationSet

Legt die Richtung eines Blicks auf eine 3D-Szene fest.

```
void  ViewRotationSet(
   const DXVector3  &rotation      // Vektor der gedrehten Winkle
   );
```

Parameter

&rotation

[in] Vektorielle Einstellung der Euler-Winkel zur Berechnung der Blickrichtung auf eine 3D-Szene.

Rückgabewert

Keiner.

Hinweis

Die Einstellung der Blickrichtung mittels ViewRotationSet() ändert die in ViewMatrixGet() erhaltene Ansichtsmatrix.

© 2000-2025, MetaQuotes Ltd.

ViewTargetSet

Legt die Koordinaten des Punktes fest, auf den ein Blick gerichtet ist.

```
void  ViewTargetSet(
   const DXVector3  &target      // Zielkoordinaten
   );
```

Parameter

&target

[in] Koordinaten des Punktes auf die ein Blick gerichtet ist.

Rückgabewert

Keiner.

Hinweis

Sie wird verwendet, um den Blick auf einen Szenenpunkt zu fixieren, wenn der Blickpunkt bewegt wird.

Das Setzen einer neuen Zielkoordinate mit ViewRotationSet() ändert die Ansichtsmatrix, die in ViewMatrixGet() erhalten wurde.

ViewTargetSet() wird zusammen mit ViewUpDirectionSet() verwendet, um die Blickrichtung zu definieren.

© 2000-2025, MetaQuotes Ltd.

ViewUpDirectionSet

Legt die Richtung des oberen Rahmenrandes im 3D-Raum fest.

```
void  ViewUpDirectionSet(
   const DXVector3  &up_direction      // obere Richtung
   );
```

Parameter

&up_direction

[in] Richtung des oberen Teils des Frames im 3D-Raum.

Rückgabewert

Keiner.

Hinweis

Das Setzen einer neuen Richtung mit ViewUpDirectionSet() ändert die Ansichtsmatrix, die in ViewMatrixGet() erhalten wurde.

ViewUpDirectionSet() wird zusammen mit ViewTargetSet() verwendet, um die Blickrichtung zu definieren.

© 2000-2025, MetaQuotes Ltd.

CChart

CChart ist eine Klasse für den vereinfachten Zugriff auf die Charteigenschaften.

Beschreibung

Die Klasse CChart bietet Zugriff auf Charteigenschaften.

Deklaration

```
class CChart : public CObject
```

Kopf

```
#include <Charts\Chart.mqh>
```

Vererbungshierarchie

CObject

CChart

Gruppen der Klassenmethode

Zugriff auf geschützte Daten	
ChartID	Erhält die ID des Charts
Gemeinsame Eigenschaften	
Mode	Erhalten/setzen den Typ des Charts (Kerzen, Balken oder Linie)
Foreground	Erhält/setzt ein Flag "Preischart im Hintergrund"
Shift	Erhält/setzt ein Flag "Verschiebung des Preischarts vom rechten Rand"
ShiftSize	Erhält/setzt Größe der Verschiebung des Nullbalkens vom rechten Rand als Prozentsatz
AutoScroll	Erhält/setzt ein Flag "Automatisches Wechseln zum rechten Chartrand"
Scale	Erhält/setzt die Skala
ScaleFix	Erhält/setzt ein Flag "Feste Skala"
ScaleFix_11	Erhält/setzt ein Flag "Skala 1:1"
FixedMax	Erhält/setzt festes Maximum des Charts
FixedMin	Erhält/setzt festes Minimum des Charts
ScalePPB	Erhält/setzt ein Flag "Skala als Punkte pro Balken"
PointsPerBar	Erhält/setzt die Skala als Punkte pro Bar

© 2000-2025, MetaQuotes Ltd.

Zugriff auf geschützte Daten	
Eigenschaften "Show"	
ShowOHLC	Erhält/setzt ein Flag "Anzeige von OHLC"
ShowLineBid	Erhält/setzt ein Flag "Anzeige von Bid als eine horizontale Linie"
ShowLineAsk	Erhält/setzt ein Flag "Anzeige von Ask als eine horizontale Linie"
ShowLastLine	Erhält/setzt ein Flag "Anzeige von Last als eine horizontale Linie"
ShowPeriodSep	Erhält/setzt ein Flag "Anzeige von vertikalen Trenner zwischen benachbarten Perioden"
ShowGrid	Erhält/setzt ein Flag "Gitteranzeige"
ShowVolumes	Erhält/setzt ein Flag "Volumenanzeige"
ShowObjectDescr	Erhält/setzt ein Flag "Anzeige von Pop-Up-Beschreibungen von grafischen Objekten"
ShowDateScale	Setzt ein Flag "Anzeige der Zeitskala"
ShowPriceScale	Setzt ein Flag "Anzeige der Preisskala"
Eigenschaften "Colors"	
ColorBackground	Erhält/setzt die Hintergrundfarbe
ColorForeground	Erhält/setzt die Farbe der Achsen, Skala und OHLC
ColorGrid	Erhält/setzt die Farbe des Gitters
ColorBarUp	Erhält/setzt die Farbe der Balken nach oben, des Schattens und Körpers von Bullenkerze
ColorBarDown	Erhält/setzt die Farbe der Balken nach unten, des Schattens und Körpers von Bärenkerze
ColorCandleBull	Erhält/setzt die Farbe des Körpers der Bullenkerze
ColorCandleBear	Erhält/setzt die Farbe des Körpers der Bärenkerze
ColorChartLine	Erhält/setzt die Farbe der Chartlinie und Doji-Kerzen
ColorVolumes	Erhält/setzt die Farbe des Volumens und Positioneröffnungsebene
ColorLineBid	Erhält/setzt die Farbe der Bid-Preislinie
ColorLineAsk	Erhält/setzt die Farbe der Ask-Preislinie
ColorLineLast	Erhält/setzt die Farbe der Preislinie der letzte Transaktion
ColorStopLevels	Erhält/setzt die Farbe der Stopebenen (Stop Loss und Take Profit)

© 2000-2025, MetaQuotes Ltd.

Zugriff auf geschützte Daten	
Eigenschaften "Read only"	
VisibleBars	Erhält die Anzahl der Balken auf dem Chart, die angezeigt werden können
WindowsTotal	Erhält die Anzahl der Chartfenster, einschließlich der Unterfenster von Indikatoren
WindowIsVisible	Erhält die Sichtbarkeit der Unterfenster
WindowHandle	Erhält Handle des Charts (HWND)
FirstVisibleBar	Erhält die Nummer des ersten sichtbaren Balkens auf dem Chart
WidthInBars	Erhält die Breite des Charts in Balken
WidthInPixels	Erhält die Breite des Charts in Pixel
HeightInPixels	Erhält die Fensterhöhe in Pixel
PriceMin	Erhält das Minimum des Fensters
PriceMax	Erhält das Maximum des Fensters
Eigenschaften	
Attach	Bindet einen Chart an eine Klasseninstanz
FirstChart	Erhält die ID des ersten Charts
NextChart	Erhält die ID des nächsten Charts
Open	Einen neuen Chart öffnen
Detach	Löst einen Chart aus der Klasseninstanz
Close	Schließt den angegebenen Chart
BringToTop	Zeigt einen Chart über allen anderen Fenstern
EventObjectCreate	Setzt ein Flag vom Senden einer Nachricht über Ereignisse der Objekterstellung
EventObjectDelete	Setzt ein Flag vom Senden einer Nachricht über Ereignisse der Objektlöschung
Indikatoren	
IndicatorAdd	Fügt einen Indikator mit dem angegebenen Handle auf das angegebene Chartfenster hinzu
IndicatorDelete	Löscht den Indikator mit dem angegebenen Namen aus dem angegebene Fenster
IndicatorsTotal	Gibt die Anzahl der Indikatoren auf dem Chartfenster zurück

© 2000-2025, MetaQuotes Ltd.

Zugriff auf geschützte Daten	
IndicatorName	Gibt den Kurznamen des Indikators aufgrund der Nummer in der Indikatorenliste im angegebenen Chartfenster zurück
Navigation	
Navigate	Chartvershiebung
Zugriff an API MQL5	
Symbol	Erhält den Namen des Chartsymbols
Period	Erhält den Wert des Chat-Zeitrahmens
Redraw	Zwangs-Neuzeichnung des Charts
GetInteger	Erhält den Wert des angegebenen Ganzzahl-Eigenschaften
SetInteger	Setzt den Wert des angegebenen Ganzzahl-Eigenschaften
GetDouble	Erhält den Wert des angegebenen Double-Eigenschaften
SetDouble	Setzt den Wert des angegebenen Double-Eigenschaften
GetString	Erhält den Wert des angegebenen String-Eigenschaften
SetString	Setzt den Wert des angegebenen String-Eigenschaften
SetSymbolPeriod	Setzt den Wert des Symbols und Zeitraums des angegebenen Charts
ApplyTemplate	Gilt die angegebene Vorlage für den Chart
ScreenShot	Macht ein Bildschirmfoto des angegebenen Charts
WindowOnDropped	Erhält die Nummer des Unterfensters, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde
PriceOnDropped	Erhält die Preiskoordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde
TimeOnDropped	Erhält die Zeitkoordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde
XOnDropped	Erhält die X-Koordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde
YOnDropped	Erhält die Y-Koordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde
Eingabe/Ausgabe	
virtual Save	Virtuelle Methode zum Schreiben in eine Datei
virtual Load	Virtuelle Methode zum Lesen aus einer Datei
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

© 2000-2025, MetaQuotes Ltd.

Prev, Prev, Next, Next, Compare

© 2000-2025, MetaQuotes Ltd.

ChartID

Erhält die ID des Charts

```
long  ChartID() const
```

Rückgabewert

Die ID des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird -1 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Mode (Get-Methode)

Erhält den Typ des Charts (Kerzen, Balken oder Linie).

```
ENUM_CHART_MODE  Mode() const
```

Rückgabewert

Typ des Charts, der an die Klasseninstanz gebunden ist (Kerzen, Balken oder Linie). Wenn es keinen gebundenen Chart gibt, wird WRONG_VALUE zurückgegeben.

Mode (Set-Methode)

Setzt den Typ des Charts (Kerzen, Balken oder Linie).

```
bool  Mode(
   ENUM_CHART_MODE  mode      // Typ des Charts
   )
```

Parameter

mode

[in] Typ des Charts (Kerzen, Balken oder Linie) aus der Enumeration ENUM_CHART_MODE.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Typ nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Foreground (Get-Methode)

Erhält den Wert des Flags "Preischart im Hintergrund".

```
bool  Foreground() const
```

Rückgabewert

Wert des Flags "Preischart im Hintergrund" für den an eine Klasseninstanz gebundenen Chart. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

Foreground (Set-Methode)

Setzt den Wert des Flags "Preischart im Hintergrund".

```
bool  Foreground(
   bool  foreground      // Flagwert
   )
```

Parameter

foreground

[in] Der neue Wert des Flags "Preischart im Hintergrund".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Shift (Get-Methode)

Erhält den Wert des Flags "Verschiebung des Preischarts vom rechten Rand".

```
bool  Shift() const
```

Rückgabewert

Wert des Flags "Verschiebung des Preischarts vom rechten Rand" des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

Shift (Set-Methode)

Setzt den Wert des Flags "Verschiebung des Preischarts vom rechten Rand".

```
bool  Shift(
   bool  shift      // Flagwert
   )
```

Parameter

shift

[in] Der neue Wert des Flags "Verschiebung des Preischarts vom rechten Rand".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShiftSize (Get-Methode)

Erhält die Größe der Verschiebung des Nullbalkens vom rechten Rand als Prozentsatz.

```
double  ShiftSize() const
```

Rückgabewert

Größe der Verschiebung des Nullbalkens vom rechten Rand als Prozentsatz. für den Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

ShiftSize (Set-Methode)

Setzt die Größe der Verschiebung des Nullbalkens vom rechten Rand als Prozentsatz.

```
bool  ShiftSize(
   double  shift_size      // Eigenschaftswert
   )
```

Parameter

shift_size

[in] Die neue Größe der Verschiebung des Nullbalkens vom rechten Rand als Prozentsatz.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

AutoScroll (Get-Methode)

Erhält das Flag "Automatisches Wechseln zum rechten Chartrand".

```
bool  AutoScroll() const
```

Rückgabewert

Wert des Flags "Automatisches Wechseln zum rechten Rand" für den an eine Klasseninstanz gebundenen Chart. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

AutoScroll (Set-Methode)

Setzt das Flag "Automatisches Wechseln zum rechten Chartrand".

```
bool  AutoScroll(
   bool  autoscroll      // Flagwert
   )
```

Parameter

autoscroll

[in] Der neue Wert des Flags "Automatisches Wechseln zum rechten Chartrand".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Scale (Get-Methode)

Erhält den Wert der Eigenschaft "Skala".

```
int  Scale() const
```

Rückgabewert

Wert der Eigenschaft "Skala" des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

Scale (Set-Methode)

Setzt den Wert der Eigenschaft "Skala".

```
bool  Scale(
   int  scale      // Eigenschaftswert
   )
```

Parameter

scale

[in] Der neue Wert der Eigenschaft "Skala".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ScaleFix (Get-Methode)

Erhält den Wert des Flags "Feste Skala".

```
bool  ScaleFix() const
```

Rückgabewert

Wert des Flags "Feste Skala" des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ScaleFix (Set-Methode)

Setzt den Wert des Flags "Feste Skala".

```
bool  ScaleFix(
   bool  scale_fix      // Flagwert
   )
```

Parameter

scale_fix

[in] Der neue Wert des Flags "Feste Skala".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ScaleFix_11 (Get-Methode)

Erhält den Wert des Flags "Skala 1:1".

```
bool  ScaleFix_11() const
```

Rückgabewert

Wert des Flags "Skala 1:1" des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ScaleFix_11 (Set-Methode)

Setzt den Wert des Flags "Skala 1:1".

```
bool  ScaleFix_11(
   string  scale_11      // Flagwert
   )
```

Parameter

scale_11

[in] Der neue Wert des Flags "Skala 1:1".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

FixedMax (Get-Methode)

Erhält festes Maximum des Charts

```
double  FixedMax() const
```

Rückgabewert

Das feste Maximum des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

FixedMax (Set-Methode)

Setzt ein festes Maximum des Charts

```
bool  FixedMax(
   double  max       // Maximum
   )
```

Parameter

max

[in] Das neue festes Maximum des Charts

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das feste Maximum nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

FixedMin (Get-Methode)

Erhält festes Minimum des Charts

```
double  FixedMin() const
```

Rückgabewert

Das feste Minimum des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

FixedMin (Set-Methode)

Setzt ein festes Minimum des Charts

```
bool  FixedMin(
   double  min       // Minimum
   )
```

Parameter

min

[in] Das neue festes Minimum des Charts

Rückgabewert

Gibt bei Erfolg true zurück, false wenn das feste Minimum nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

PointsPerBar (Get-Methode)

Erhält die Skala als Punkte pro Bar.

```
double  PointsPerBar() const
```

Rückgabewert

Die Skala als Punkte pro Bar des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

PointsPerBar (Set-Methode)

Setzt die Skala als Punkte pro Bar.

```
bool  PointsPerBar(
   double  ppb      // Skala
   )
```

Parameter

ppb

[in] Die neue Skala als Punkte pro Bar.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Skala nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ScalePPB (Get-Methode)

Erhält den Wert des Flags "Skala als Punkte pro Balken".

```
bool  ScalePPB() const
```

Rückgabewert

Wert des Flags "Skala als Punkte pro Balken" des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ScalePPB (Set-Methode)

Setzt den Wert des Flags "Skala als Punkte pro Balken".

```
bool  ScalePPB(
   bool  scale_ppb      // Flagwert
   )
```

Parameter

scale_ppb

[in] Der neue Wert des Flags "Skala als Punkte pro Balken".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowOHLC (Get-Methode)

Erhält den Wert des Flags "OHLC einblenden".

```
bool  ShowOHLC() const
```

Rückgabewert

Wert des Flags "Anzeige von OHLC" auf dem Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowOHLC (Set-Methode)

Setzt den Wert des Flags "OHLC einblenden".

```
bool  ShowOHLC(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Die neue wert des Flags "Anzeige von OHLC".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowLineBid (Get-Methode)

Erhält den Wert des Flags "Anzeige von Bid als eine horizontale Linie"

```
bool  ShowLineBid() const
```

Rückgabewert

Wert des Flags "Anzeige von Bid als eine horizontale Linie" für einen Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowLineBid (Set-Methode)

Setzt den Wert des Flags "Anzeige von Bid als eine horizontale Linie"

```
bool  ShowLineBid(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Anzeige von Bid als eine horizontale Linie".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowLineAsk (Get-Methode)

Erhält den Wert des Flags "Anzeige von Ask als eine horizontale Linie"

```
bool  ShowLineAsk() const
```

Rückgabewert

Wert des Flags "Anzeige von Ask als eine horizontale Linie" für einen Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowLineAsk (Set-Methode)

Setzt den Wert des Flags "Anzeige von Ask als eine horizontale Linie"

```
bool  ShowLineAsk(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Anzeige von Ask als eine horizontale Linie".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowLastLine (Get-Methode)

Erhält den Wert des Flags "Anzeige von Last als eine horizontale Linie"

```
bool  ShowLastLine() const
```

Rückgabewert

Wert des Flags "Anzeige von Last als eine horizontale Linie" für einen Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowLastLine (Set-Methode)

Setzt den Wert des Flags "Anzeige von Last als eine horizontale Linie"

```
bool  ShowLastLine(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Anzeige von Last als eine horizontale Linie".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowPeriodSep (Get-Methode)

Erhält den Wert des Flags "Anzeige von vertikalen Trennlinien zwischen benachbarten Perioden"

```
bool  ShowPeriodSep() const
```

Rückgabewert

Wert des Flags "Anzeige von vertikalen Trennlinien zwischen benachbarten Perioden" für einen Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowPeriodSep (Set-Methode)

Setzt den Wert des Flags "Anzeige von vertikalen Trennlinien zwischen benachbarten Perioden"

```
bool  ShowPeriodSep(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Anzeige von vertikalen Trennlinien zwischen benachbarten Perioden".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowGrid (Get-Methode)

Erhält den Wert des Flags "Gitteranzeige".

```
bool  ShowGrid() const
```

Rückgabewert

Wert des Flags "Gitteranzeige" des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowGrid (Set-Methode)

Setzt den Wert des Flags "Gitteranzeige".

```
bool  ShowGrid(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Gitteranzeige".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowVolumes (Get-Methode)

Erhält den Wert des Flags "Volumenanzeige".

```
bool  ShowVolumes()
```

Rückgabewert

Der wert des Flags "Volumenanzeige" für den Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowVolumes (Set-Methode)

Setzt den Wert des Flags "Volumenanzeige".

```
bool  ShowVolumes(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Volumenanzeige".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowObjectDescr (Get-Methode)

Erhält den Wert des Flags "Anzeige von Pop-Up-Beschreibungen von grafischen Objekten"

```
bool  ShowObjectDescr() const
```

Rückgabewert

Wert des Flags "Anzeige von Pop-Up-Beschreibungen von grafischen Objekten" für einen Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

ShowObjectDescr (Set-Methode)

Setzt den Wert des Flags "Anzeige von Pop-Up-Beschreibungen von grafischen Objekten"

```
bool  ShowObjectDescr(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Anzeige von Pop-Up-Beschreibungen von grafischen Objekten".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowDateScale

Setzt den Wert des Flags "Zeitskala einblenden".

```
bool  ShowDateScale(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Zeitskala einblenden".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ShowPriceScale

Setzt den Wert des Flags "Preisskala einblenden".

```
bool  ShowPriceScale(
   bool  show      // Flagwert
   )
```

Parameter

show

[in] Der neue Wert des Flags "Preisskala einblenden".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorBackground (Get-Methode)

Erhält die Hintergrundfarbe.

```
color  ColorBackground() const
```

Rückgabewert

Der Wert der Hintergrundfarbe des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorBackground (Set-Methode)

Setzt die Hintergrundfarbe.

```
bool  ColorBackground(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Hintergrundfarbe.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorForeground (Get-Methode)

Erhält die Farbe der Achsen, Skala und OHLC.

```
color  ColorForeground() const
```

Rückgabewert

Die Farbe der Achsen, Skala und OHLC des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorForeground (Set-Methode)

Setzt die Farbe der Achsen, Skala und OHLC.

```
bool  ColorForeground(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Achsen, Skala und OHLC.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorGrid (Get-Methode)

Erhält die Farbe des Gitters.

```
color  ColorGrid() const
```

Rückgabewert

Die Farbe des Gitters des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorGrid (Set-Methode)

Setzt die Farbe des Gitters.

```
bool  ColorGrid(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe des Gitters.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorBarUp (Get-Methode)

Erhält die Farbe der Balken nach oben, des Schattens und Körpers einer Bullenkerze.

```
color  ColorBarUp() const
```

Rückgabewert

Die Farbe der Balken nach oben, des Schattens und Körpers der Bullenkerze des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorBarUp (Set-Methode)

Setzt die Farbe der Balken nach oben, des Schattens und Körpers einer Bullenkerze.

```
bool  ColorBarUp(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Balken nach oben, des Schattens und Körpers einer Bullenkerze.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorBarDown (Get-Methode)

Erhält die Farbe der Balken nach unten, des Schattens und Körpers der Bärenkerze.

```
color  ColorBarDown() const
```

Rückgabewert

Die Farbe der Balken nach unten, des Schattens und Körpers der Bärenkerze des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorBarDown (Set-Methode)

Setzt die Farbe der Balken nach unten, des Schattens und Körpers der Bärenkerze.

```
bool  ColorBarDown(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Balken nach unten, des Schattens und Körpers der Bärenkerze.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorCandleBull (Get-Methode)

Erhält die Farbe des Körpers einer Bullenkerze.

```
color  ColorCandleBull() const
```

Rückgabewert

Die Farbe des Körpers einer Bullenkerze des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorCandleBull (Set-Methode)

Setzt die Farbe des Körpers einer Bullenkerze.

```
bool  ColorCandleBull(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe des Körpers einer Bullenkerze.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorCandleBear (Get-Methode)

Erhält die Farbe des Körpers einer Bärenkerze.

```
color  ColorCandleBear() const
```

Rückgabewert

Die Farbe des Körpers einer Bärenkerze des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorCandleBear (Set-Methode)

Setzt die Farbe des Körpers einer Bärenkerze.

```
bool  ColorCandleBear(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe des Körpers einer Bärenkerze.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorChartLine (Get-Methode)

Erhält die Farbe der Chartlinie und Doji-Kerzen.

```
color  ColorChartLine() const
```

Rückgabewert

Die Farbe der Chartlinie und Doji-Kerzen des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorChartLine (Set-Methode)

Setzt die Farbe der Chartlinie und Doji-Kerzen.

```
bool  ColorChartLine(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Chartlinie und Doji-Kerzen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorVolumes (Get-Methode)

Erhält die Farbe des Volumens und Positioneröffnungsebenen.

```
color  ColorVolumes() const
```

Rückgabewert

Die Farbe des Volumens und Positioneröffnungsebenen des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorVolumes (Set-Methode)

Setzt die Farbe des Volumens und Positioneröffnungsebenen.

```
bool  ColorVolumes(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe des Volumens und Positioneröffnungsebenen.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorLineBid (Get-Methode)

Erhält die Farbe der Bid-Preislinie.

```
color  ColorLineBid() const
```

Rückgabewert

Die Farbe der Bid-Preislinie des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorLineBid (Set-Methode)

Setzt die Farbe der Bid-Preislinie.

```
bool  ColorLineBid(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Bid-Linie.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorLineAsk (Get-Methode)

Erhält die Farbe der Ask-Preislinie.

```
color  ColorLineAsk() const
```

Rückgabewert

Die Farbe der Ask-Linie des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorLineAsk (Set-Methode)

Setzt die Farbe der Ask-Preislinie.

```
bool  ColorLineAsk(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Ask-Linie.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorLineLast (Get-Methode)

Erhält die Farbe der Preislinie der letzte Transaktion.

```
color  ColorLineLast() const
```

Rückgabewert

Die Farbe der Preislinie der letzte Transaktion des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorLineLast (Set-Methode)

Setzt die Farbe der Preislinie der letzte Transaktion.

```
bool  ColorLineLast(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Preislinie der letzte Transaktion.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ColorStopLevels (Get-Methode)

Erhält die Farbe der Stopebenen (Stop Loss und Take Profit).

```
color  ColorStopLevels() const
```

Rückgabewert

Die Farbe der Stopebenen (Stop Loss und Take Profit) des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird CLR_NONE zurückgegeben.

ColorStopLevels (Set-Methode)

Setzt die Farbe der Stopebenen (Stop Loss und Take Profit).

```
bool  ColorStopLevels(
   color  new_color      // die Farbe
   )
```

Parameter

new_color

[in] Die neue Farbe der Stopebenen (Stop Loss und Take Profit).

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Farbe nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

VisibleBars

Erhält die Anzahl der Balken auf dem Chart, die angezeigt werden können.

```
int  VisibleBars() const
```

Rückgabewert

Die Anzahl der Balken, die angezeigt werden können, auf dem Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

WindowsTotal

Erhält die Anzahl der Chartfenster, einschließlich der Unterfenster von Indikatoren.

```
int  WindowsTotal() const
```

Rückgabewert

Die Gesamtzahl der Fenster einschließlich der Unterfenster mit Indikatoren auf dem Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

WindowIsVisible

Erhält die Sichtbarkeit der Unterfenster.

```
bool  WindowIsVisible(
   int  num      // Unterfenster
   ) const
```

Parameter

num

[in] Nummer des Unterfensters (0 bedeutet das Hauptfenster).

Rückgabewert

Die Sichtbarkeit der Fenster des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird false zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

WindowHandle

Erhält Handle des Charts (HWND).

```
int  WindowHandle() const
```

Rückgabewert

Handle (HWND) des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird INVALID_HANDLE zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

FirstVisibleBar

Erhält die Nummer des ersten sichtbaren Balkens auf dem Chart.

```
int  FirstVisibleBar() const
```

Rückgabewert

Nummer des ersten sichtbaren Balkens auf dem Chart, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird -1 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

WidthInBars

Erhält die Breite des Charts in Balken.

```
int  WidthInBars() const
```

Rückgabewert

Die Breite in Balken des CHarts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

WidthInPixels

Erhält die Breite des Charts in Pixeln.

```
int  WidthInPixels() const
```

Rückgabewert

Die Breite in Pixeln des CHarts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

HeightInPixels

Erhält die Fensterhöhe in Pixel.

```
int  HeightInPixels(
   int  num      // Unterfenster
   ) const
```

Parameter

num

[in] Nummer des geprüften Unterfensters (0 bedeutet das Hauptfenster).

Rückgabewert

Die Höhe in Pixel des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

PriceMin

Erhält das Minimum des Fensters.

```
double  PriceMin(
   int  num      // Unterfenster
   ) const
```

Parameter

num

[in] Nummer des Unterfensters (0 bedeutet das Hauptfenster).

Rückgabewert

Fensterminimum des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

PriceMax

Erhält das Maximum des Fensters.

```
double  PriceMax(
   int  num      // Unterfenster
   ) const
```

Parameter

num

[in] Nummer des Unterfensters (0 bedeutet das Hauptfenster).

Rückgabewert

Fenstermaximum des Charts, der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Attach

Bindet den aktuellen Chart an die Klasseninstanz.

```
void  Attach()
```

Attach

Bindet den angegebenen Chart an die Klasseninstanz.

```
void  Attach(
   long  chart       // ID des Charts
   )
```

Parameter

chart

[in] ID des Charts den Sie verbinden.

© 2000-2025, MetaQuotes Ltd.

FirstChart

Bindet den ersten Chart in Client-Terminal auf eine Instanz der Klasse.

```
void  FirstChart()
```

© 2000-2025, MetaQuotes Ltd.

NextChart

Bindet den nächsten Chart an die Klasseninstanz.

```
void  NextChart()
```

© 2000-2025, MetaQuotes Ltd.

Open

Öffnet den Chart mit den angegebenen Parameter und bindet ihn an eine Klasseninstanz.

```
long  Open(
   const string      symbol_name,      // Symbol
   ENUM_TIMEFRAMES   timeframe         // Periode
   )
```

Parameter

symbol_name

[in] Das Symbol des Charts. NULL bedeutet das Symbol des aktuellen Charts (auf dem der Expert Advisor arbeitet).

timeframe

[in] Zeitrahmen des Charts (Enumeration ENUM_TIMEFRAMES). 0 bedeutet die Periode des aktuellen Charts.

Rückgabewert

Chart ID

© 2000-2025, MetaQuotes Ltd.

Detach

Löst einen Chart aus der Klasseninstanz.

```
void  Detach()
```

© 2000-2025, MetaQuotes Ltd.

Close

Schließt den auf Klasseninstanz gebundenen Chart.

```
void  Close()
```

© 2000-2025, MetaQuotes Ltd.

BringToTop

Zeigt einen Chart über allen anderen Fenstern.

```
bool  BringToTop() const
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

EventObjectCreate

Setzt ein Flag vom Senden einer Nachricht über die Ereignisse der Objekterstellung.

```
bool  EventObjectCreate(
   bool  flag      // ein Flag
   )
```

Parameter

flag

[in] Die neue Wert des Flags vom Senden einer Nachricht über Ereignisse der Objekterstellung.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

EventObjectDelete

Setzt ein Flag vom Senden einer Nachricht über die Ereignisse der Objektlöschung.

```
bool  EventObjectDelete(
   bool  flag      // ein Flag
   )
```

Parameter

flag

[in] Die neue Wert des Flags vom Senden einer Nachricht über Ereignisse der Objektlöschung.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

IndicatorAdd

Fügt einen Indikator mit dem angegebenen Handle auf das angegebene Chartfenster hinzu.

```
bool  IndicatorAdd(
   int   sub_win          // Nummer des Unterfensters
   int   handle           // Indikator-Handle
   );
```

Parameter

sub_win

[in] Nummer des Unterfensters. 0 bedeutet das Hauptfenster des Charts. Wenn eine Nummer des nicht vorhandenen Fensters angegeben war, wird ein neues Fenster erstellt werden.

handle

[in] Indikator-Handle.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Sehen Sie auch

IndicatorDelete(), IndicatorsTotal(), IndicatorName().

© 2000-2025, MetaQuotes Ltd.

IndicatorDelete

Löscht den Indikator mit dem angegebenen Namen aus dem angegebene Fenster.

```
bool  IndicatorDelete(
   int            sub_win      // Nummer des Unterfensters
   const string   name         // Kurzname des Indikators
   );
```

Parameter

sub_win

[in] Nummer des Unterfensters. 0 bedeutet das Hauptfenster des Charts.

const name

[in] Kurzname des Indikators, der in der Eigenschaft INDICATOR_SHORTNAME durch die FunktionIndicatorSetString() angegeben wird. Um den Kurznamen des Indikators zu erhalten, verwenden Sie die Funktion IndicatorName().

Rückgabewert

Gibt true zurück wenn der Indikator erfolgreich gelöscht wurde, ansonsten false. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Wenn es mehrere Indikatoren mit dem gleichen Kurznamen im angegebene Chartfenster gibt, wird der erste gelöscht werden.

Wenn irgendwelche Indikatoren auf demselben Chart die Daten dieses Indikators verwenden, werden sie auch gelöscht.

Verwechseln Sie nicht den Kurznamen des Indikators und den Dateinamen, der beim Indikatorerstellung durch die Funktionen iCustom() und IndicatorCreate() angegeben wird. Wenn der Kurzname des Indikators nicht explizit festgelegt ist, dann wird der Name der Datei, die den Source-Code des Indikators enthält, beim Kompilation angegeben.

Entfernung des Indikators aus dem Chart bedeutet nicht, dass die Berechnungsteil des Indikators auch aus der Terminalspeicher gelöscht wird. Um das Handle des Indikators wieder frei zu geben, verwenden Sie die Funktion IndicatorRelease().

Man muss richtig den Kurznamen eines Indikators generieren, der mit der Funktion IndicatorSetString() in die Eigenschaft INDICATOR_SHORTNAME geschrieben wird. Ein kurzer Name soll die Werte der Eingangsparameter des Indikators enthalten, da ein aus dem Chart gelöschten Indikator in der Funktion IndicatorDelete() nach seinem Namen identifiziert wird.

Sehen Sie auch

IndicatorAdd(), IndicatorsTotal(), IndicatorName(), iCustom(), IndicatorCreate(), IndicatorSetString().

© 2000-2025, MetaQuotes Ltd.

IndicatorsTotal

Gibt die Anzahl der Indikatoren auf dem Chartfenster zurück.

```
int  IndicatorsTotal(
   long  chart_id,       // ID des Charts
   int   sub_window      // Nummer des Unterfensters
   );
```

Parameter

chart_id

[in] ID des Charts. 0 bedeutet den aktuellen Chart.

sub_window

[in] Nummer des Unterfensters. 0 bedeutet das Hauptfenster des Charts.

Rückgabewert

Anzahl der Indikatoren im angegebenen Chartfenster. Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Die Funktion erlaubt durch alle Indikatoren auf diesem Chart zu suchen. Um die Anzahl aller Chartfenster aus der Eigenschaft CHART_WINDOWS_TOTAL zu erhalten verwenden Sie die Funktion GetInteger().

Sehen Sie auch

IndicatorAd(), IndicatorDelete(), IndicatorsTotal(), iCustom(), IndicatorCreate(), IndicatorSetString().

© 2000-2025, MetaQuotes Ltd.

IndicatorName

Gibt den Kurznamen des Indikators aufgrund der Nummer in der Indikatorenliste im angegebenen Chartfenster zurück.

```
string  IndicatorName(
   int   sub_win     // Nummer des Unterfensters
   int   index       // Index des Indikators in der Liste der Indikatoren, die in dies
   );
```

Parameter

sub_win

[in] Nummer des Unterfensters. 0 bedeutet das Hauptfenster des Charts.

index

[in] Index des Indikators in der Indikatorenliste. Die Nummerierung beginnt mit Null, d.h. der erste Indikator in der Liste hat den Index Null. Um die Anzahl der Indikatoren zu erhalten verwenden Sie die Funktion IndicatorsTotal().

Rückgabewert

Der Kurzname des Indikators, der in der Eigenschaft INDICATOR_SHORTNAME durch die Funktion SetString(). Um Informationen über den Fehler zu erhalten, rufen Sie die Funktion GetLastError().

Hinweis

Verwechseln Sie nicht den Kurznamen des Indikators und den Dateinamen, der beim Indikatorerstellung durch die Funktionen iCustom() und IndicatorCreate() angegeben wird. Wenn der Kurzname des Indikators nicht explizit festgelegt ist, dann wird der Name der Datei mit dem Source-Code des Indikators beim Kompilation angegeben.

Man muss richtig den Kurznamen eines Indikators generieren, der mit der Funktion IndicatorSetString() in die Eigenschaft INDICATOR_SHORTNAME geschrieben wird. Ein kurzer Name soll die Werte der Eingangsparameter des Indikators enthalten, da ein aus dem Chart gelöschten Indikator in der Funktion IndicatorDelete() nach seinem Namen identifiziert wird.

Sehen Sie auch

IndicatorAdd(), IndicatorDelete, IndicatorsTotal, iCustom(), IndicatorCreate(), IndicatorSetString().

© 2000-2025, MetaQuotes Ltd.

Navigate

Chartnavigation.

```
bool  Navigate(
   ENUM_CHART_POSITION  position,     // die Position
   int                  shift=0       // Verschiebung
   )
```

Parameter

position

[in] Die Position des Charts (aus der Enumeration ENUM_CHART_POSITION).

shift=0

[in] Die Anzahl der Balken um den Chart zu verschieben.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Chart nicht verschiebt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Symbol

Erhält den Namen des Chartsymbols.

```
string  Symbol() const
```

Rückgabewert

Symbolname des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird "" zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Period

Erhält den Wert des Chat-Zeitrahmens.

```
ENUM_TIMEFRAMES  Period() const
```

Rückgabewert

Periode des Charts (aus der Enumeration ENUM_TIMEFRAMES), der an die Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Redraw

Zeichnet den auf Klasseninstanz gebundenen Chart neu.

```
void  Redraw()
```

© 2000-2025, MetaQuotes Ltd.

GetInteger

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück. Die Charteigenschaft muss ein integer-Wert sein. Es gibt zwei Varianten der Funktion.

1. Gibt den Wert der Eigenschaft zurück.

```
long  GetInteger(
   ENUM_CHART_PROPERTY_INTEGER  prop_id,          // Identifikator der Eigenschaft
   int                          sub_window=0     // Nummer des Unterfensters
   ) const
```

2. Beim Erfolg fügt den Eigenschaftswert in eine Empfangsvariable, die durch den letzten Parameter als Referenz übergeben wird, hinzu.

```
bool  GetInteger(
   ENUM_CHART_PROPERTY_INTEGER  prop_id,        // Identifikator der Eigenschaft
   int                          sub_window,     // Nummer des Unterfensters
   long&                        value           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Identifikator der Charteigenschaft (aus der Enumeration ENUM_CHART_PROPERTY_INTEGER).

sub_window

[in] Nummer des Unterfensters.

value

[in] Referenz an die Variable, die den Wert der angeforderten Eigenschaft empfängt.

Rückgabewert

Der Eigenschaftswert des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird -1 zurückgegeben.

Für die zweite Variante wird zurückgegeben, wenn diese Eigenschaft unterstützt wird und der Wert in die Variable value hinzugefügt wurde. Ansonsten wird false zurückgegeben. Für mehr Informationen über den Fehler rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

SetInteger

Setzt den Wert der integer-Eigenschaft des Charts.

```
bool  SetInteger(
   ENUM_CHART_PROPERTY_INTEGER  prop_id,      // Identifikator der Eigenschaft
   long                         value         // Wert
   )
```

Parameter

prop_id

[in] Identifikator der Charteigenschaft (aus der Enumeration ENUM_CHART_PROPERTY_INTEGER).

value

[in] Der neue Eigenschaftswert.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die integer-Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

GetDouble

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück. Die Charteigenschaft muss ein double-Wert sein. Es gibt zwei Varianten der Funktion.

1. Gibt den Wert der Eigenschaft zurück.

```
double  GetDouble(
   ENUM_CHART_PROPERTY_DOUBLE  prop_id,          // Identifikator der Eigenschaft
   int                         sub_window=0    // Nummer des Unterfensters
   ) const
```

2. Beim Erfolg fügt den Eigenschaftswert in eine Empfangsvariable, die durch den letzten Parameter als Referenz übergeben wird, hinzu.

```
bool  GetDouble(
   ENUM_CHART_PROPERTY_DOUBLE  prop_id,        // Identifikator der Eigenschaft
   int                         sub_window,     // Nummer des Unterfensters
   double&                     value           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Identifikator der Charteigenschaft (aus der Enumeration ENUM_CHART_PROPERTY_DOUBLE).

sub_window

[in] Nummer des Unterfensters.

value

[in] Referenz an die Variable, die den Wert der angeforderten Eigenschaft empfängt.

Rückgabewert

Der Eigenschaftswert des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird EMPTY_VALUE zurückgegeben.

Für die zweite Variante wird true beim Erfolg zurückgegeben oder false, wenn die Eigenschaftswert konnte nicht erhalten werden. Für mehr Informationen über den Fehler rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

SetDouble

Setzt den Wert der double-Eigenschaft des Charts.

```
bool  SetDouble(
   ENUM_CHART_PROPERTY_DOUBLE  prop_id,      // Identifikator der Eigenschaft
   double                      value         // Wert
   )
```

Parameter

prop_id

[in] Identifikator der Charteigenschaft (aus der Enumeration ENUM_CHART_PROPERTY_DOUBLE).

value

[in] Der neue Eigenschaftswert.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die double-Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

GetString

Gibt den Wert der entsprechenden Eigenschaft des angegebenen Charts zurück. Die Charteigenschaft muss ein string-Wert sein. Es gibt zwei Varianten der Funktion.

1. Gibt den Wert der Eigenschaft zurück.

```
string  GetString(
   ENUM_CHART_PROPERTY_STRING  prop_id      // Identifikator der Eigenschaft
   ) const
```

2. Beim Erfolg fügt den Eigenschaftswert in eine Empfangsvariable, die durch den letzten Parameter als Referenz übergeben wird, hinzu.

```
bool  GetString(
   ENUM_CHART_PROPERTY_STRING  prop_id,     // Identifikator der Eigenschaft
   string&                     value        // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Identifikator der Charteigenschaft (aus der Enumeration ENUM_CHART_PROPERTY_STRING).

value

[in] Referenz an die Variable, die den Wert der angeforderten Eigenschaft empfängt.

Rückgabewert

Der Eigenschaftswert des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird "" zurückgegeben.

Für die zweite Variante wird zurückgegeben, wenn diese Eigenschaft unterstützt wird und der Wert in die Variable value hinzugefügt wurde. Ansonsten wird false zurückgegeben. Für mehr Informationen über den Fehler rufen Sie die Funktion GetLastError().

© 2000-2025, MetaQuotes Ltd.

SetString

Setzt den Wert der string-Eigenschaft des Charts.

```
bool  SetString(
   ENUM_CHART_PROPERTY_STRING  prop_id,      // Identifikator der Eigenschaft
   string                      value         // Wert
   )
```

Parameter

prop_id

[in] Identifikator der Charteigenschaft (aus der Enumeration ENUM_CHART_PROPERTY_STRING).

value

[in] Der neue Eigenschaftswert.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die string-Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

SetSymbolPeriod

Ändert das Symbol und die Periode des Charts, der an eine Klasseninstanz gebunden ist.

```
bool  SetSymbolPeriod(
   const string      symbol_name,      // Symbol
   ENUM_TIMEFRAMES   timeframe         // Periode
   )
```

Parameter

symbol_name

[in] Das neue Symbol des Charts. NULL bedeutet das Symbol des aktuellen Charts (auf dem der Expert Advisor arbeitet).

timeframe

[in] Die neue Periode des Charts (aus der Enumeration ENUM_TIMEFRAMES). 0 bedeutet die Periode des aktuellen Charts.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

ApplyTemplate

Gilt die angegebene Vorlage für den Chart.

```
bool  ApplyTemplate(
   const string  filename      // Vorlage
   )
```

Parameter

filename

[in] Name der Datei, in der die Vorlage liegt

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Vorlage nicht verwendet werden konnte.

© 2000-2025, MetaQuotes Ltd.

ScreenShot

Macht ein Bildschirmfoto des aktuellen Zustands des angegebenen Charts im Format gif.

```
bool  ScreenShot(
   string           filename,                  // Dateiname
   int              width,                     // Breite
   int              height,                    // Höhe
   ENUM_ALIGN_MODE  align_mode=ALIGN_RIGHT     // Ausrichtung
   ) const
```

Parameter

filename

[in] Dateiname des Bildschirmfotos.

width

[in] Die Breite des Bildschirmfotos in Pixel.

height

[in] Die Höhe des Bildschirmfotos in Pixel.

align_mode=ALIGN_RIGHT

[in] Ausgabemodus für ein schmales Bildschirmfoto.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

WindowOnDropped

Erhält die Nummer des Unterfensters, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

```
int  WindowOnDropped() const
```

Rückgabewert

Die Nummer des Unterfensters, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde. 0 bedeutet das Hauptfenster des Charts.

© 2000-2025, MetaQuotes Ltd.

PriceOnDropped

Erhält die Preiskoordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

```
double  PriceOnDropped() const
```

Rückgabewert

Die Preiskoordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

© 2000-2025, MetaQuotes Ltd.

TimeOnDropped

Erhält die Zeitkoordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

```
datetime  TimeOnDropped() const
```

Rückgabewert

Die Zeitkoordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

© 2000-2025, MetaQuotes Ltd.

XOnDropped

Erhält die X-Koordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

```
int  XOnDropped() const
```

Rückgabewert

Die X-Koordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

© 2000-2025, MetaQuotes Ltd.

YOnDropped

Erhält die Y-Koordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

```
int  YOnDropped() const
```

Rückgabewert

Die Y-Koordinate des Punktes, auf dem der Expert Advisor oder das Skript mit dem Maus hinzugefügt wurde.

© 2000-2025, MetaQuotes Ltd.

Save

Speichert Daten des Listenelements in einer Datei.

```
virtual bool  Save(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Load

Lädt die Objektparameter aus einer Datei.

```
virtual bool  Load(
   int  file_handle      // Datei-Handle
   )
```

Parameter

file_handle

[in] Handle einer binären Datei, die mit der Funktion FileOpen(...) früher geöffnet wurde

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Type

Erhält die Typenidentifikator.

```
virtual int  Type() const
```

Rückgabewert

Typenidentifikator (für CChart - 0x1111).

© 2000-2025, MetaQuotes Ltd.

Wissenschaftliche Grafiken

Eine grafische Bibliothek, die Klassen und globale Funktionen für schnelles Zeichnen benutzerdefinierter Charts beinhaltet. Die Bibliothek bietet einfache fertige Lösungen für das Zeichnen von Achsen und Kurven sowie Methoden des schnellen Zugangs zur Änderung allgemeiner Eigenschaften eines benutzerdefinierten Charts.

Um die Arbeit mit der Bibliothek zu beginnen, lesen Sie einfach den Artikel Visualisierung! Eine grafische MQL5 Bibliothek ähnlich 'plot' der Sprache R.

Die grafische Bibliothek ist im Ordner Include\Graphics im Verzeichnis des Terminals gespeichert.

Klasse	Beschreibung
CAxis	Eine Klasse für Operationen mit den Koordinatenachsen
CColorGenerator	Eine Klasse, die das Farbschema standardmäßig setzt
CCurve	Eine Klasse für Operationen mit Kurven
CGraphic	Eine Basisklasse für die Erstellung benutzerdefinierter Grafiken

© 2000-2025, MetaQuotes Ltd.

GraphPlot

Funktionen für schnelles Zeichnen von Kurven.

Die Version für das Zeichnen einer Kurve nach Y-Koordinaten.

```
string  GraphPlot(
   const double      &y[],                  // Y-Koordinate
   ENUM_CURVE_TYPE   type=CURVE_POINTS      // Typ der Kurve
   )
```

Hinweis

Als X-Koordinaten dienen für diese Kurve die Indexe des Y-Arrays.

Die Version für das Zeichnen einer Kurve anhand des Koordinatenpaars X und Y

```
string  GraphPlot(
   const double      &x[],                  // X-Koordinaten
   const double      &y[],                  // Y-Koordinate
   ENUM_CURVE_TYPE   type=CURVE_POINTS      // Typ der Kurve
   )
```

Die Version für das Zeichnen zweier Kurven anhand des Koordinatenpaares X und Y

```
string  GraphPlot(
   const double      &x1[],                 // X-Koordinaten
   const double      &y1[],                 // Y-Koordinaten
   const double      &x2[],                 // X-Koordinaten
   const double      &y2[],                 // Y-Koordinaten
   ENUM_CURVE_TYPE   type=CURVE_POINTS      // Typ der Kurve
   )
```

Die Version für das Zeichnen von drei Kurven anhand des Koordinatenpaares X und Y.

```
string  GraphPlot(
   const double      &x1[],                 // X-Koordinaten
   const double      &y1[],                 // Y-Koordinaten
   const double      &x2[],                 // X-Koordinaten
   const double      &y2[],                 // Y-Koordinaten
   const double      &x3[],                 // X-Koordinaten
   const double      &y3[],                 // Y-Koordinaten
   ENUM_CURVE_TYPE   type=CURVE_POINTS      // Typ der Kurve
   )
```

© 2000-2025, MetaQuotes Ltd.

Die Version für das Zeichnen einer Kurve anhand der Koordinaten der Punkte CPoint2D

```
string  GraphPlot(
   const CPoint2D   &points[],            // Koordinaten der Kurve
   ENUM_CURVE_TYPE  type=CURVE_POINTS     // Typ der Kurve
   )
```

Die Version für das Zeichnen zweier Kurven anhand der Koordinaten der Punkte CPoint2D

```
string  GraphPlot(
   const CPoint2D   &points1[],           // Koordinaten der Kurve
   const CPoint2D   &points2[],           // Koordinaten der Kurve
   ENUM_CURVE_TYPE  type=CURVE_POINTS     // Typ der Kurve
   )
```

Die Version für das Zeichnen von drei Kurven anhand der Koordinaten der Punkte CPoint2D

```
string  GraphPlot(
   const CPoint2D   &points1[],           // Koordinaten der Kurve
   const CPoint2D   &points2[],           // Koordinaten der Kurve
   const CPoint2D   &points3[],           // Koordinaten der Kurve
   ENUM_CURVE_TYPE  type=CURVE_POINTS     // Typ der Kurve
   )
```

Die Version für das Zeichnen einer Kurve anhand des Bezeichners auf die Funktion CurveFunction

```
string  GraphPlot(
   CurveFunction    function,             // Bezeichner auf die Funktion
   const double     from,                 // Anfangswert des Arguments
   const double     to,                   // Endwert des Arguments
   const double     step,                 // Inkrement des Arguments
   ENUM_CURVE_TYPE  type=CURVE_POINTS     // Typ der Kurve
   )
```

Die Version für das Zeichnen von zwei Kurven anhand der Bezeichner auf die CurveFunction Funktion

```
string  GraphPlot(
   CurveFunction    function1,            // Bezeichner auf die Funktion
   CurveFunction    function2,            // Bezeichner auf die Funktion
   const double     from,                 // Anfangswert des Arguments
   const double     to,                   // Endwert des Arguments
   const double     step,                 // Inkrement des Arguments
```

© 2000-2025, MetaQuotes Ltd.

```
   ENUM_CURVE_TYPE   type=CURVE_POINTS      // Typ der Kurve
   )
```

Die Version für das Zeichnen von drei Kurven anhand der Bezeichner auf die Funktion CurveFunction

```
string  GraphPlot(
   CurveFunction    function1,             // Bezeichner auf die Funktion
   CurveFunction    function2,             // Bezeichner auf die Funktion
   CurveFunction    function3,             // Bezeichner auf die Funktion
   const double     from,                  // Anfangswert des Arguments
   const double     to,                    // Endwert des Arguments
   const double     step,                  // Inkrement des Arguments
   ENUM_CURVE_TYPE  type=CURVE_POINTS      // Typ der Kurve
   )
```

Parameter

&x[]

[in] X-Koordinaten.

&y[]

[in] Y-Koordinaten.

&x1[]

[in] X-Koordinaten für die erste Kurve.

&y1[]

[in] X-Koordinaten für die zweite Kurve.

&x2[]

[in] X-Koordinaten für die zweite Kurve.

&y2[]

[in] Y-Koordinaten für die zweite Kurve.

&x3[]

[in] X-Koordinaten für die dritte Kurve.

&y3[]

[in] Y-Koordinaten für die dritte Kurve.

&points[]

[in] Koordinaten der Punkte der Kurve.

&points1[]

[in] Koordinaten der Punkte der ersten Kurve.

&points2[]

© 2000-2025, MetaQuotes Ltd.

[in] Koordinaten der Punkte der zweiten Kurve.

&points3[]

[in] Koordinaten der Punkte der dritten Kurve.

function

[in] Bezeichner auf die Funktion CurveFunction.

function1

[in] Bezeichner auf die erste Funktion.

function2

[in] Bezeichner auf die zweite Funktion.

function3

[in] Bezeichner auf die dritte Funktion.

from

[in] Entspricht der ersten X-Koordinate.

to

[in] Entspricht der letzten X-Koordinate.

step

[in] Parameter für die Berechnung der X-Koordinaten.

type=CURVE_POINTS

[in] Typ der Kurve.

Rückgabewert

Name der grafischen Ressource.

© 2000-2025, MetaQuotes Ltd.

CAxis

Die CAxis Klasse ist eine Hilfsklasse der grafischen Bibliothek für Operationen mit Koordinatenachsen.

Beschreibung

Die CAxis Klasse erhält und speichert verschiedene Parameter der Koordinatenachsen. In dieser Klasse wird die Möglichkeit einer automatischen dynamischen Skalierung der Koordinatenachsen umgesetzt.

Deklaration

```
class CAxis
```

Überschrift

```
#include <Graphics\Axis.mqh>
```

Methoden der Klasse

Methode	Beschreibung
AutoScale	Flag der automatischen Skalierung erhalten/setzen
Min	Mindestwert der Achse erhalten/setzen
Max	Höchstwert der Achse erhalten/setzen
Step	Liefert den Schrittwert auf der Achse
Name	Namen der Achse erhalten/setzen
Color	Farbe der Achse erhalten/setzen
ValuesSize	Größe der Ziffern auf der Achse erhalten/setzten
ValuesWidth	Maximal angezeigte Länge der Ziffern auf der Achse erhalten/setzen
ValuesFormat	Format der Ziffern auf der Achse erhalten/setzen
ValuesDateTimeMode	Gibt das Format der Umwandlung des Datums in einen String zurück.
ValuesFunctionFormat	Gibt den Bezeichner auf die Funktion zurück, die das Format der Ausgabe der Werte an die Achse bestimmt.
ValuesFunctionFormatCBData	Gibt den Bezeichner auf das Objekt zurück, welches zusätzliche Information für die Umwandlung des Achsenwertes beinhaltet.
NameSize	Schriftgröße für den Namen der Achse erhalten/setzten
ZeroLever	Wert des "Nullhebels" erhalten/setzen

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
DefaultStep	Anfangswert des Schritts auf der Achse erhalten/setzen
MaxLabels	Maximale Anzahl der Ziffern auf der Achse erhalten/setzen
MinGrace	Toleranzwert für das Minimum der Achse erhalten/setzen
MaxGrace	Toleranzwert für das Maximum der Achse erhalten/setzen
SelectAxisScale	Führt die automatische Skalierung der Achse aus.

© 2000-2025, MetaQuotes Ltd.

AutoScale (Get-Methode)

Liefert das Flag, das definiert, ob die automatische Skalierung benötigt wird.

```
bool  AutoScale()
```

Rückgabewert

Wert des Flags.

Hinweis

true — automatische Skalierung.

false — keine automatische Skalierung.

AutoScale (Set-Methode)

Setzt ein Flag, das definieren, ob die automatische Skalierung benötigt wird.

```
void  AutoScale(
   const bool  auto      // Wert des Flags
   )
```

Parameter

auto

[in]

Hinweis

true — automatische Skalierung.

false — keine automatische Skalierung.

© 2000-2025, MetaQuotes Ltd.

Min (Get-Methode)

Liefert den Mindestwert der Achse.

```
double  Min()
```

Rückgabewert

Mindestwert der Achse.

Min (Set-Methode)

Setzt den Mindestwert der Achse.

```
void  Min(
   const double  min      // Mindestwert
   )
```

Parameter

min

[in] Mindestwert.

© 2000-2025, MetaQuotes Ltd.

Max (Get-Methode)

Liefert den maximalen Wert der Achse.

```
double  Max()
```

Rückgabewert

Maximaler Wert der Achse.

Max (Set-Methode)

Setzt den maximalen Wert der Achse.

```
void  Max(
   const double  max       // maximaler Wert
   )
```

Parameter

max

[in] Maximaler Wert der Achse.

© 2000-2025, MetaQuotes Ltd.

Step (Get-Methode)

Liefert den Schrittwert.

```
double  Step()
```

Rückgabewert

Schrittwert.

© 2000-2025, MetaQuotes Ltd.

Name (Get-Methode)

Liefert den Namen der Achse.

```
string  Name()
```

Rückgabewert

Name der Achse.

Name (Set-Methode)

Setzt den Namen der Achse.

```
void  Name(
   const string  name      // Name der Achse
   )
```

Parameter

name

[in] Name der Achse.

© 2000-2025, MetaQuotes Ltd.

Color (Get-Methode)

Liefert die Farbe der Achse.

```
color  Color()
```

Rückgabewert

Farbe der Achse.

Color (Set-Methode)

Setzt die Farbe der Achse.

```
void  Color(
   const color  clr      // Farbe der Achse
   )
```

Parameter

clr

[in] Farbe der Achse.

© 2000-2025, MetaQuotes Ltd.

ValuesSize (Get-Methode)

Liefert die Größe der Ziffern auf der Achse.

```
int  ValuesSize()
```

Rückgabewert

Größe der Ziffern, die auf der Achse angezeigt werden.

ValuesSize (Set-Methode)

Setzt die Größe der Ziffern auf der Achse.

```
void  ValuesSize(
   const int  size      // Größe der Ziffern auf der Achse
   )
```

Parameter

size

[in] Größe der Ziffern auf der Achse

© 2000-2025, MetaQuotes Ltd.

ValuesWidth (Get method)

Returns the maximum allowed length in pixels for displaying the axis numbers.

```
int  ValuesWidth()
```

Return Value

Length of the axis numbers in pixels.

Note

If a length in pixels for a specified number exceeds the maximum allowed display length, it is truncated and ends in dots.

ValuesWidth (Set method)

Sets the maximum allowed length in pixels for displaying the axis numbers.

```
void  ValuesWidth(
   const int  width      // maximum allowed length in pixels
   )
```

Parameters

width

[in] Maximum allowed length of the axis numbers.

Note

If a length in pixels for a specified number exceeds the maximum allowed display length, it is truncated and ends in dots.

© 2000-2025, MetaQuotes Ltd.

ValuesFormat (Get-Methode)

Liefert das Format der Anzeige der Ziffern auf der Achse.

```
string  ValuesFormat()
```

Rückgabewert

Format der Anzeige der Ziffern.

ValuesFormat (Set-Methode)

Setzt das Anzeigeformat der Ziffern auf der Achse.

```
void  ValuesFormat(
   const string  format      // Anzeigeformat der Ziffern auf der Achse
   )
```

Parameter

format

[in] Anzeigeformat der Ziffern auf der Achse.

© 2000-2025, MetaQuotes Ltd.

ValuesDateTimeMode (Get-Methode)

Gibt das Format der Umwandlung des Datums in einen String zurück.

```
int  ValuesDateTimeMode()
```

Rückgabewert

Format der Umwandlung eines Datums in einen String.

ValuesDateTimeMode (Set-Methode)

Setzt das Format der Umwandlung eines Datums in einen String.

```
void  ValuesDateTimeMode(
   const int  mode      // Format der Umwandlung eines Datums in einen String
   )
```

Parameter

mode

[in] Umwandlungsformat.

Hinweis

Mehr Informationen über die Umwandlung eines Datums in einen String finden Sie in der Beschreibung der TimeToString() Funktion.

© 2000-2025, MetaQuotes Ltd.

ValuesFunctionFormat (Get-Methode)

Gibt den Bezeichner auf die Funktion zurück, die das Format der Ausgabe der Werte an die Achse bestimmt.

```
DoubleToStringFunction  ValuesFunctionFormat()
```

Rückgabewert

Bezeichner auf die Funktion, die das Format der Ausgabe der Werte an die Achse bestimmt.

ValuesFunctionFormat (Set-Methode)

Setzt den Bezeichner auf die Funktion, die das Format der Ausgabe von Werten an die Achse bestimmt.

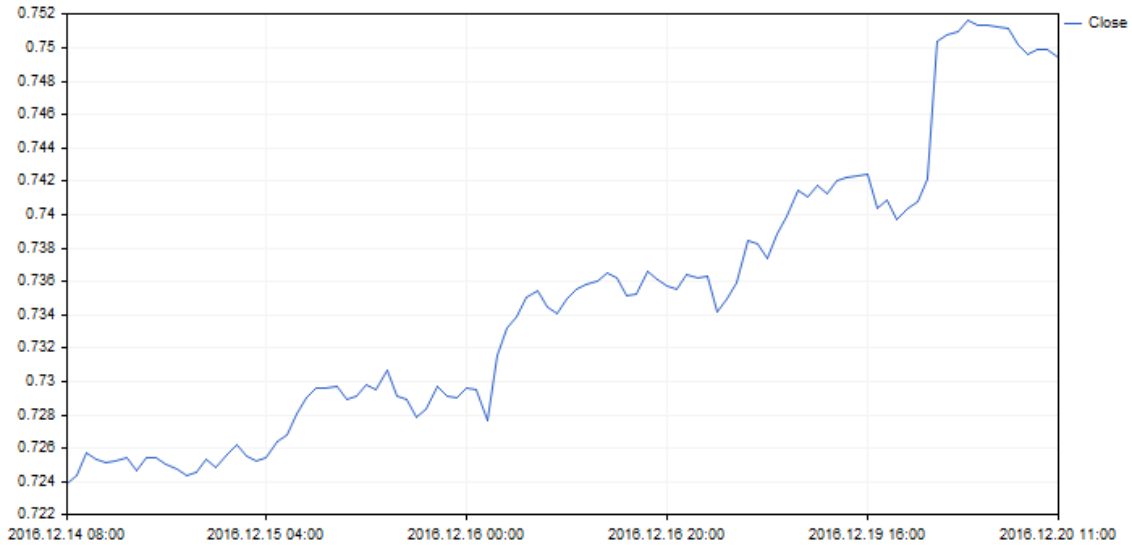
```
void  ValuesFunctionFormat(
   DoubleToStringFunction  func      // Funktion für die Umwandlung numerischer Werte
   )
```

Parameter

func

[in] Benutzerdefinierte Funktion für die Umwandlung numerischer Werte in einen String.

Beispiel:



Das Format der Ausgabe von Werten an die X-Achse wurde mithilfe des folgenden Codes geändert:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                              DateAxisGraphic.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Graphics\Graphic.mqh>
//--- array for store values
double arrX[];
double arrY[];
//+------------------------------------------------------------------+
//| Custom function for create values on X-axis                      |
//+------------------------------------------------------------------+
string TimeFormat(double x,void *cbdata)
  {
   return(TimeToString((datetime)arrX[ArraySize(arrX)-(int)x-1]));
  }
//+------------------------------------------------------------------+
void OnStart()
  {
   MqlRates rates[];
   CopyRates(Symbol(),Period(),0,100,rates);
   ArraySetAsSeries(rates,true);
   int size=ArraySize(rates);
   ArrayResize(arrX,size);
   ArrayResize(arrY,size);
   for(int i=0; i<size;++i)
     {
      arrX[i]=(double)rates[i].time;
      arrY[i]=rates[i].close;
     }
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"DateAxisGraphic",0,30,30,780,380))
     {
      graphic.Attach(0,"DateAxisGraphic");
     }
//--- create curve
   CCurve *curve=graphic.CurveAdd(arrY,CURVE_LINES);
//--- gets the X-axis
   CAxis *xAxis=graphic.XAxis();
//--- sets the X-axis properties
   xAxis.AutoScale(false);
   xAxis.Type(AXIS_TYPE_CUSTOM);
   xAxis.ValuesFunctionFormat(TimeFormat);
   xAxis.DefaultStep(20.0);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

© 2000-2025, MetaQuotes Ltd.

ValuesFunctionFormatCBData (Get-Methode)

Gibt den Bezeichner auf das Objekt zurück, welches zusätzliche Information für die Umwandlung des Achsenwertes beinhaltet.

```
void*  ValuesFunctionFormatCBData()
```

Rückgabewert

Bezeichner auf das Objekt, welches zusätzliche Information für die Umwandlung von Werten der Achse beinhalten kann.

ValuesFunctionFormatCBData (Set-Methode)

Setzt den Bezeichner auf das Objekt der Klasse, welche zusätzliche Information für die Umwandlung von Achsenwerten beinhalten kann.

```
void  ValuesFunctionFormatCBData(
   void*  cbdata      // Bezeichner auf das Objekt der Klasse
   )
```

Parameter

cbdata

[in] Bezeichner auf das Objekt jeglicher Klasse, welches zusätzliche Informationen für die Umwandlung des Achsenwertes beinhaltet

© 2000-2025, MetaQuotes Ltd.

NameSize (Get-Methode)

Liefert die Schriftgröße für den Namen der Achse.

```
int  NameSize()
```

Rückgabewert

Schriftgröße, mit der der Name der Achse gesetzt wird.

NameSize (Set-Methode)

Setzt die Schriftgröße für den Namen der Achse.

```
void  NameSize(
   const int  size      // Schriftgröße für den Namen der Achse
   )
```

Parameter

size

[in] Schriftgröße, mit der der Name der Achse gesetzt wird.

© 2000-2025, MetaQuotes Ltd.

ZeroLever (Get-Methode)

Liefert den Wert des "Nullhebels".

```
double  ZeroLever()
```

Rückgabewert

"Nullhebel".

Hinweis

Dieser Wert definiert, wann der Skalenbereich der Achse erweitert werden soll, um den Nullwert hinzuzufügen.

ZeroLever (Set-Methode)

Setzt den Wert des "Nullhebels".

```
void  ZeroLever(
   const double  value      // Wert des "Nullhebels"
   )
```

Parameter

value

[in] Wert des "Nullhebels".

Hinweis

Dieser Wert definiert, wann der Skalenbereich der Achse erweitert werden soll, um den Nullwert hinzuzufügen.

© 2000-2025, MetaQuotes Ltd.

DefaultStep (Get-Methode)

Liefert den Anfangswert des Schritt auf der Achse.

```
double  DefaultStep()
```

Rückgabewert

Schritt auf der Achse.

DefaultStep (Set-Methode)

Setzt den Anfangswert des Schritts auf der Achse.

```
void  DefaultStep(
   const double  value      // Schritt
   )
```

Parameter

value

[in] Anfangswert des Schritts.

© 2000-2025, MetaQuotes Ltd.

MaxLabels (Get-Methode)

Liefert die maximal mögliche Anzahl der auf der Achse angezeigten Ziffern.

```
double  MaxLabels()
```

Rückgabewert

Maximale Anzahl der Ziffern auf der Achse.

MaxLabels (Set-Methode)

Setzt die maximal mögliche Anzahl der auf der Achse angezeigten Ziffern.

```
void  MaxLabels(
   const double  value      // maximale Anzahl
   )
```

Parameter

value

[in] Setzt die maximal mögliche Anzahl der auf der Achse angezeigten Ziffern

© 2000-2025, MetaQuotes Ltd.

MinGrace (Get-Methode)

Liefert den Toleranzwert für das Minimum der Achse.

```
double  MinGrace()
```

Rückgabewert

Toleranzwert für das Maximum der Achse.

Hinweis

Dieser Wert wird als Teil der Gesamtlänge der Achse ausgedrückt. Z.B. liegt der Wert der Achse im Bereich von 4.0 bis 16.0, dann beträgt die Länge der Achse 12.0. Wenn MinGrace gleich 0.1 ist, dann werden 10% der Länge der Achse (oder 1.2) von dem Wert des Minimums subtrahiert. Auf diese Weise wird die Achse den Bereich von 2.8 bis 16.0 abdecken.

MinGrace (Set-Methode)

Setzt den Toleranzwert für das Minimum der Achse.

```
void  MinGrace(
   const double  value      // Toleranzwert
   )
```

Parameter

value

[in] Toleranzwert für den Mindestwert der Achse.

Hinweis

Dieser Wert wird als Teil der Gesamtlänge der Achse ausgedrückt. Z.B. liegt der Wert der Achse im Bereich von 4.0 bis 16.0, dann beträgt die Länge der Achse 12.0. Wenn MinGrace gleich 0.1 ist, dann werden 10% der Länge der Achse (oder 1.2) von dem Wert des Minimums subtrahiert. Auf diese Weise wird die Achse den Bereich von 2.8 bis 16.0 abdecken.

© 2000-2025, MetaQuotes Ltd.

MaxGrace (Get-Methode)

Liefert den Toleranzwert für das Maximum der Achse.

```
double  MaxGrace()
```

Rückgabewert

Toleranzwert für das Maximum der Achse.

Hinweis

Dieser Wert wird als Teil der Gesamtlänge der Achse ausgedrückt. Z.B. liegt der Wert der Achse im Bereich von 4.0 bis 16.0, dann beträgt die Länge der Achse 12.0. Wenn MaxGrace gleich 0.1 ist, dann werden 10% der Länge der Achse (oder 1.2) zum Wert des Maximums addiert. Auf diese Weise wird die Achse den Bereich von 4.0 bis 17.2 abdecken.

MaxGrace (Set-Methode)

Setzt den Toleranzwert für das Maximum der Achse.

```
void  MaxGrace(
   const double  value      // Toleranzwert
   )
```

Parameter

value

[in] Toleranzwert für das Maximum der Achse.

Hinweis

Dieser Wert wird als Teil der Gesamtlänge der Achse ausgedrückt. Z.B. liegt der Wert der Achse im Bereich von 4.0 bis 16.0, dann beträgt die Länge der Achse 12.0. Wenn MinGrace gleich 0.1 ist, dann werden 10% der Länge der Achse (oder 1.2) von dem Wert des Minimums subtrahiert. Auf diese Weise wird die Achse den Bereich von 2.8 bis 16.0 abdecken.

© 2000-2025, MetaQuotes Ltd.

SelectAxisScale

Führt die automatische Skalierung der Achse aus.

```
void  SelectAxisScale()
```

© 2000-2025, MetaQuotes Ltd.

CColorGenerator

Die CColorGenerator Klasse ist eine Hilfsklasse der grafischen Bibliothek für Operationen mit der Farbpalette.

Beschreibung

Die CColorGenerator Klasse beinhaltet eine elementare Palette von Farben, welche standardmäßig für die Darstellung von Kurven verwendet werden (wenn keine Farben vom Nutzer definiert wurden).

Wenn alle Farben aus der Palette verwendet wurden, werden neue Farben automatisch generiert und die Palette wird neu gefüllt.

Deklaration

```
class CColorGenerator
```

Überschrift

```
#include <Graphics\ColorGenerator.mqh>
```

Methoden der Klasse

Methode	Beschreibung
Next	Liefert die nächste Farbe aus der Palette
Reset	Setzt den Generator zurück

© 2000-2025, MetaQuotes Ltd.

Next

Liefert die nächste Farbe aus der Palette.

```
+uint  Next()
```

Rückgabewert

Farbe.

Hinweis

Wenn alle Farben aus der Palette verwendet wurden, werden neue Farben automatisch generiert, um die alten in der Palette zu ersetzen.

© 2000-2025, MetaQuotes Ltd.

Reset

Setzt den Generator zurück.

```
void  Reset()
```

© 2000-2025, MetaQuotes Ltd.

CCurve

Die CCurve Klasse ist eine Klasse für Operationen mit Eigenschaften von Kurven, die auf dem Chart generiert werden.

Beschreibung

Die CCurve Klasse setzt, speichert und erhält Koordinaten und verschiedene Kurven bei Operationen mit der CGraphic Klasse.

Es gibt drei Modi für das Zeichnen von Kurven: Punkte, Linien und Histogramme. Für jeden Modus werden unterschiedliche Parameter in der Klasse umgesetzt.

Deklaration

```
class CCurve : public CObject
```

Überschrift

```
#include <Graphics\Curve.mqh>
```

Vererbungshierarchie

CObject

CCurve

Methoden der Klasse

Methode	Beschreibung
Type	Liefert den Typ der Kurve
Name	Liefert den Namen der Kurve
Color	Liefert die Farbe der Kurve
XMax	Liefert den höchsten Wert der Funktion auf der X-Achse
XMin	Liefert den niedrigsten Wert der Funktion auf der X-Achse
YMax	Liefert den höchsten Wert der Funktion auf der Y-Achse
YMin	Liefert den niedrigsten Wert der Funktion auf der Y-Achse
Size	Liefert die Anzahl der Punkte, die eine Kurve definieren
PointsSize	Größe der Punkte, die eine Kurve definieren, erhalten/setzen

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
PointsFill	Flag der Füllung der Punkte, die eine Kurve definieren, erhalten/setzen
PointsColor	Füllfarbe der Punkte erhalten/setzen
GetX	Erhält die X-Koordinaten aller Punkte der Kurve ins Array
GetY	Erhält die Y-Koordinaten aller Punkte der Kurve ins Array
LinesStyle	Stil für Linien beim Zeichnen mit Linien erhalten/setzen
LinesIsSmooth	Flag der Glättung beim Zeichnen mit Linien erhalten/setzen
LinesSmoothTension	Glättungsparameter der Kurve beim Zeichnen mit Linien erhalten/setzen
LinesSmoothStep	Länge der Annäherungslinien für Glättung beim Zeichnen mit Linien erhalten/setzen
LinesWidth	Linienstärke beim Zeichnen einer gebrochenen Linie mit Linien erhalten/setzen.
HistogramWidth	Breite der Spalten beim Zeichnen in Form eines Histogramms erhalten/setzen
CustomPlotCBData	Bezeichner auf das Objekt, das beim Zeichnen der Kurve im benutzerdefinierten Modus verwendet wird, setzen/erhalten.
CustomPlotFunction	Bezeichner auf die Funktion, die den benutzerdefinierten Modus beim Zeichnen der Kurve implementiert, setzen/erhalten.
PointsType	Flag erhalten/setzen, das den Punktentyp angibt, die beim Zeichnen der Kurve mit Punkten verwendet werden.
StepsDimension	Wert erhalten/setzen, der die Dimension angibt, die bei einem stufenartigen Zeichnen der Kurve verwendet wird.
TrendLineCoefficients	Koeffizienten der Trendlinie für das Schreiben ins Array erhalten/setzen.
TrendLineColor	Farbe der Trendlinie für die Kurve erhalten/setzen.
TrendLineVisible	Flag erhalten/setzen, das angibt, ob die Trendlinie auf dem Chart sichtbar sein wird.
Update	Aktualisiert Koordinaten der Kurve.

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
Visible	Flag erhalten/setzen, das angibt, ob die Funktion auf dem Chart sichtbar sein wird.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Compare

© 2000-2025, MetaQuotes Ltd.

Type

Liefert den Typ der Kurve.

```
ENUM_CURVE_TYPE   Type()
```

Rückgabewert

Typ der Kurve.

© 2000-2025, MetaQuotes Ltd.

Name

Liefert den Namen der Kurve.

```
string  Name()
```

Rückgabewert

Name der Kurve.

© 2000-2025, MetaQuotes Ltd.

Color

Liefert die Farbe der Kurve.

```
uint  Color()
```

Rückgabewert

Farbe der Kurve.

© 2000-2025, MetaQuotes Ltd.

XMax

Liefert den höchsten Wert der Funktion auf der X-Achse (nur reelle Zahlen).

```
double  XMax()
```

Rückgabewert

Die höchste reelle Zahl unter anderen Argumenten für diese Funktion.

© 2000-2025, MetaQuotes Ltd.

XMin

Liefert den niedrigsten Wert der Funktion auf der X-Achse (nur reelle Zahlen).

```
double  XMin()
```

Rückgabewert

Die niedrigste reelle Zahl unter anderen Argumenten für diese Funktion.

© 2000-2025, MetaQuotes Ltd.

YMax

Liefert den höchsten Wert der Funktion auf der Y-Achse (nur reelle Zahlen).

```
double  YMax()
```

Rückgabewert

Der höchste Wert dieser Funktion auf der Y-Achse (nur reelle Zahlen).

© 2000-2025, MetaQuotes Ltd.

YMin

Liefert den niedrigsten Wert der Funktion auf der Y-Achse (nur reelle Zahlen).

```
double  YMin()
```

Rückgabewert

Der niedrigste Wert dieser Funktion auf der Y-Achse (nur reelle Zahlen).

© 2000-2025, MetaQuotes Ltd.

Size

Liefert die Anzahl der Punkte, die eine Kurve definieren.

```
int  Size()
```

Rückgabewert

Anzahl der Punkte, die die Kurve definieren.

© 2000-2025, MetaQuotes Ltd.

PointsSize (Get-Methode)

Liefert die lineare Größe der Punkte in Pixel, die beim Zeichnen einer Kurve mit Punkten verwendet werden.

```
int  PointsSize()
```

Rückgabewert

Größe der Punkte in Pixel, die beim Zeichnen einer Kurve verwendet werden.

PointsSize (Set-Methode)

Setzt die Größe der Punkte in Pixel, die beim Zeichnen einer Kurve mit Punkten verwendet werden.

```
void  PointsSize(
   const int  size      // Größe der Punkte in Pixel
   )
```

Parameter

size

[in] Größe der Punkte in Pixel, die beim Zeichnen einer Kurve verwendet werden.

© 2000-2025, MetaQuotes Ltd.

PointsFill (Get-Methode)

Liefert das Flag, das festlegt, ob die eine Kurve definierenden Punkte mit Farbe gefüllt werden müssen.

```
bool  PointsFill ()
```

Rückgabewert

Wert des Flags.

Hinweis

true — mit Farbe füllen

false — nicht mit Farbe füllen

PointsFill (Set-Methode)

Setzt ein Flag, das festlegt, ob die eine Kurve definierenden Punkte mit Farbe gefüllt werden müssen..

```
void  PointsFill(
   const bool  fill      // Flagwert
   )
```

Parameter

fill

[in] Flag.

Hinweis

true — mit Farbe füllen

false — nicht mit Farbe füllen

© 2000-2025, MetaQuotes Ltd.

PointsColor (Get-Methode)

Liefert die Füllfarbe der Punkte.

```
uint  PointsColor ()
```

Rückgabewert

Füllfarbe für Punkte, die diese Kurve definieren.

PointsColor (Set-Methode)

Setzt die Füllfarbe der Punkte

```
void  PointsColor(
   const uint  clr      //Füllfarbe der Punkte
   )
```

Parameter

clr

[in] Füllfarbe der Punkte, die diese Kurve definieren.

© 2000-2025, MetaQuotes Ltd.

GetX

Speichert die X-Werte aller Punkte der Kurve in ein Array.

```
void  GetX(
   double&  x[]      // Array für das Schreiben der X-Werte
   )
```

Parameter

x[]

[out] Array für das Speichern der X-Werte aller Punkte der Kurve.

Hinweis

Jeder Punkt der Kurve wird mit dem Koordinatenpaar X und Y gesetzt. Diese Werte sind keine Koordinaten in Pixel für das Zeichnen in der Klasse CGraphic.

© 2000-2025, MetaQuotes Ltd.

GetY

Speichert die Y-Werte aller Punkte der Kurve in ein Array.

```
void  GetY(
   double&  y[]      // Array für das Schreiben der Y-Werten
   )
```

Parameter

y[]

[out] Array für das Speichern der Y-Werte aller Punkte der Kurve.

Hinweis

Jeder Punkt der Kurve wird mit dem Koordinatenpaar X und Y gesetzt. Diese Werte sind keine Koordinaten in Pixel für das Zeichnen in der Klasse CGraphic.

© 2000-2025, MetaQuotes Ltd.

LinesStyle (Get-Methode)

Liefert den Linienstil beim Zeichnen einer Kurve mit Linien.

```
ENUM_LINE_STYLE  LinesStyle()
```

Rückgabewert

Linienstil.

LinesStyle (Set-Methode)

Setzt einen Linienstil beim Zeichnen einer Kurve mit Linien.

```
void  LinesStyle (
   ENUM_LINE_STYLE  style      // Linienstil
   )
```

Parameter

style

[in] Linienstil.

© 2000-2025, MetaQuotes Ltd.

LinesIsSmooth (Get-Methode)

Liefert ein Flag, das definiert, ob eine Kurve beim Zeichen mit Linien geglättet werden muss.

```
bool  LinesIsSmooth()
```

Rückgabewert

Flag

Hinweis

true — glätten

false — nicht glätten

LinesIsSmooth (Set-Methode)

Setzt ein Flag, das definiert, ob eine Kurve beim Zeichen mit Linien geglättet werden muss.

```
void  LinesIsSmooth(
   const bool  smooth      // Flag-Wert
   )
```

Parameter

smooth

[in] Flag-Wert

Hinweis

true — glätten

false — nicht glätten

© 2000-2025, MetaQuotes Ltd.

LinesSmoothTension (Get-Methode)

Liefert den Glättungsparameter der Kurve beim Zeichnen mit Linien.

```
double  LinesSmoothTension()
```

Rückgabewert

Wert des Glättungsparameters

Hinweis

Der tension-Wert liegt im Bereich (0.0; 1.0].

LinesSmoothTension (Set-Methode)

Setzt den Glättungsparameter einer Kurve beim Zeichnen mit Linien.

```
void  LinesSmoothTension(
   const double  tension      // Wert des Parameters
   )
```

Parameter

tension

[in] Wert des Glättungsparameters.

Hinweis

Der tension-Wert liegt im Bereich (0.0; 1.0].

© 2000-2025, MetaQuotes Ltd.

LinesSmoothStep (Get-Methode)

Liefert die Länge der Näherungskurven für die Glättung beim Zeichnen mit Linien.

```
double  LinesSmoothStep()
```

Rückgabewert

Länge der Annäherungslinien in Pixel.

LinesSmoothStep (Set-Methode)

Setzt die Länge der Näherungslinien für die Glättung beim Zeichnen mit Linien.

```
void  LinesSmoothStep(
   const double  step      // Länge der Linien
   )
```

Parameter

step

[in] Länge der Näherungslinien

© 2000-2025, MetaQuotes Ltd.

LinesWidth (Get-Methode)

Gibt die Linienstärke beim Zeichnen einer Kurve mit Linien zurück.

```
int  LinesWidth()
```

Rückgabewert

Linienstärke.

LinesWidth (Set-Methode)

Setzt die Linienstärke beim Zeichnen einer Kurve mit Linien.

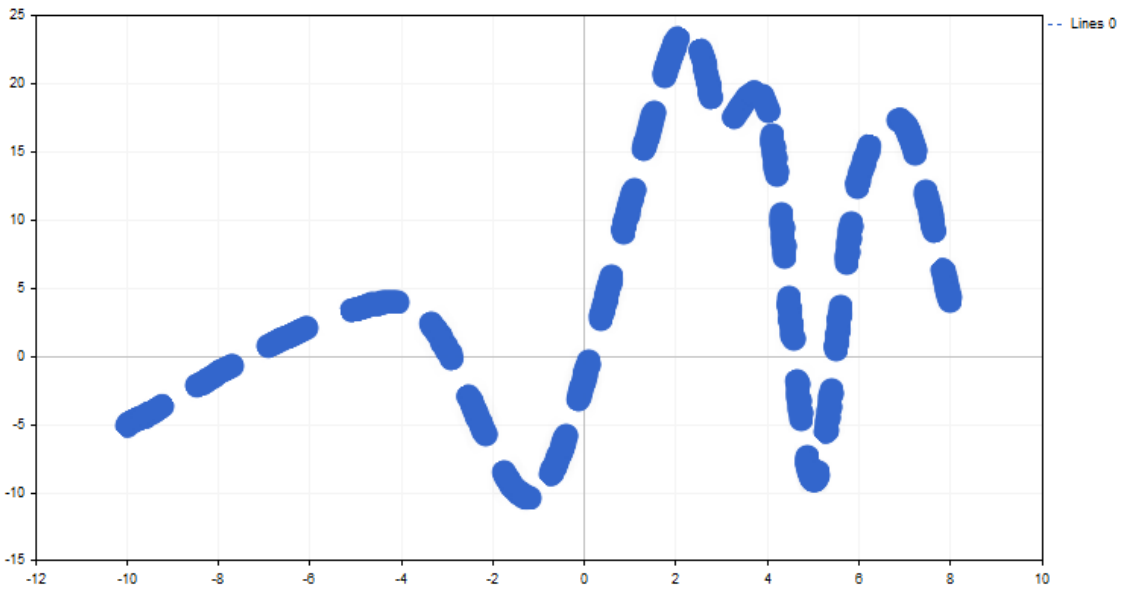
```
void  LinesWidth(
   const int  width      // Linienstärke
   )
```

Parameter

width

[in] Linienstärke beim Zeichnen einer Kurve mit Linien.

Beispiel:



Die Linienstärke wurde mithilfe des folgenden Codes geändert:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                CandleGraphic.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Graphics\Graphic.mqh>
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   double x[]= { -100,-40,-10,20,30,40,50,60,70,80,120 };
   double y[]= { -5,4,-10,23,17,18,-9,13,17,4,9 };
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"ThickLineGraphic",0,30,30,780,380))
     {
      graphic.Attach(0,"ThickLineGraphic");
     }
//--- create curve
   CCurve *curve=graphic.CurveAdd(x,y,CURVE_LINES);
//--- sets the curve properties
   curve.LinesSmooth(true);
   curve.LinesStyle(STYLE_DASH);
   curve.LinesEndStyle(LINE_END_ROUND);
   curve.LinesWidth(10);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

© 2000-2025, MetaQuotes Ltd.

LinesEndStyle (Set-Methode)

Gibt das Flag zurück, das auf den Stil der Enden der Linien beim Zeichnen einer Kurve mit Linien angibt.

```
ENUM_LINE_END  LinesEndStyle()
```

Rückgabewert

Flag, das den Stil der Linienenden beim Zeichnen einer Kurve mit Linien angibt.

LinesEndStyle (Get-Methode)

Setzt das Flag, das den Stil der Linienenden beim Zeichnen einer Kurve mit Linien angibt.

```
void  LinesEndStyle(
   ENUM_LINE_END  end_style      // Flag
   )
```

Parameter

end_style

[in] Flag, das den Stil der Enden der Linien beim Zeichnen einer Kurve mit Linien angibt.

© 2000-2025, MetaQuotes Ltd.

HistogramWidth (Get-Methode)

Liefert die Breite der Spalten beim Zeichnen als Histogramm.

```
int  HistogramWidth()
```

Rückgabewert

Breite der Spalten in Pixel.

HistogramWidth (Set-Methode)

Setzt die Breite der Spalten beim Zeichnen als Histogramm.

```
void  HistogramWidth(
   const int  width      // Breite der Spalten
   )
```

Parameter

width

[in] Breite der Spalten in Pixel.

© 2000-2025, MetaQuotes Ltd.

CustomPlotCBData (Get-Methode)

Gibt den Bezeichner auf das Objekt zurück, das beim Zeichnen einer Kurve im benutzerdefinierten Modus verwendet wird.

```
void*  CustomPlotCBData()
```

Rückgabewert

Bezeichner auf das Objekt für den benutzerdefinierten Modus beim Zeichnen der Kurve.

CustomPlotCBData (Set-Methode)

Setzt den Bezeichner auf das Objekt der Klasse, das beim Zeichnen der Kurve im benutzerdefinierten Modus verwendet wird.

```
void  CustomPlotCBData(
   void*  cbdata      // Bezeichner auf das Objekt
   )
```

Parameter

cbdata

[in] Bezeichner auf das Objekt der Klasse, das beim Zeichnen der Kurve im benutzerdefinierten Modus verwendet wird

© 2000-2025, MetaQuotes Ltd.

CustomPlotFunction (Get-Methode)

Gibt den Bezeichner auf die Funktion zurück, die den benutzerdefinierten Modus beim Zeichnen einer Kurve implementiert.

```
PlotFucntion  CustomPlotFunction()
```

Rückgabewert

Bezeichner auf die Funktion, die den benutzerdefinierten Modus beim Zeichnen der Kurve implementiert.

CustomPlotFunction (Set-Methode)

Gibt den Bezeichner auf die Funktion zurück, die den benutzerdefinierten Modus beim Zeichnen der Kurve implementiert.

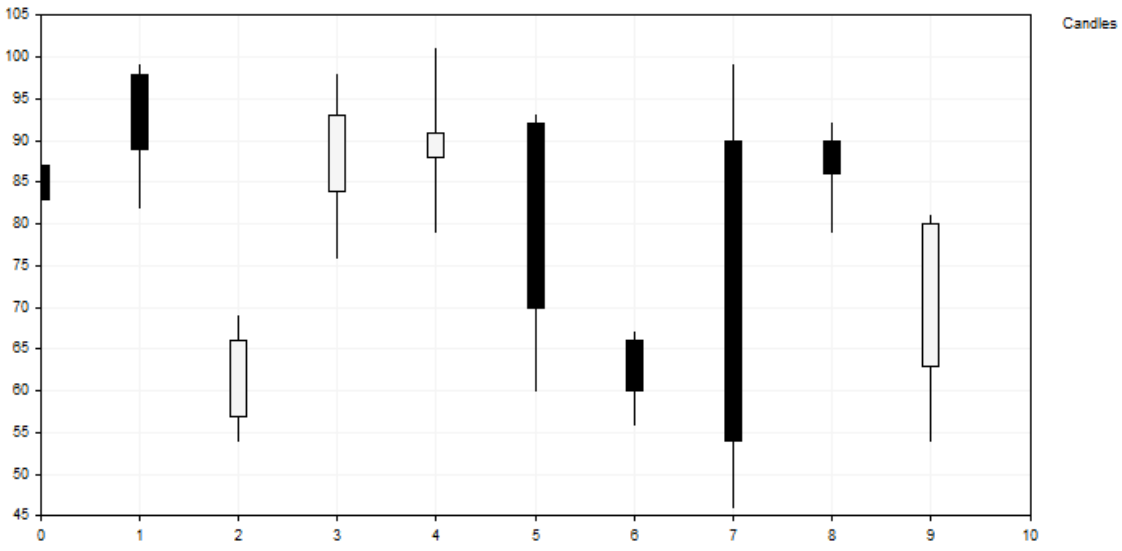
```
void  CustomPlotFunction(
   PlotFucntion  func      // Bezeichner auf die Funktion
   )
```

Parameter

func

[in] Bezeichner auf die Funktion, die den benutzerdefinierten Modus beim Zeichnen der Kurve implementiert

Beispiel:



Diese Kurve aus Balken wurde mithilfe des folgenden Codes gezeichnet:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                                CandleGraphic.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Graphics\Graphic.mqh>
//+------------------------------------------------------------------+
//| Class CCandle                                                    |
//| Usage: class to represent the candle                             |
//+------------------------------------------------------------------+
class CCandle: public CObject
  {
private:
   double            m_open;
   double            m_close;
   double            m_high;
   double            m_low;
   uint              m_clr_inc;
   uint              m_clr_dec;
   int               m_width;

public:
                     CCandle(const double open,const double close,const double high,co
                                                    const int width,const uint clr_
                    ~CCandle(void);
   double            OpenValue(void)            const { return(m_open);     }
   double            CloseValue(void)           const { return(m_close);    }
   double            HigthValue(void)           const { return(m_high);     }
   double            LowValue(void)             const { return(m_low);      }
   uint              CandleColorIncrement(void) const { return(m_clr_inc);  }
   uint              CandleColorDecrement(void) const { return(m_clr_dec);  }
   int               CandleWidth(void)          const { return(m_width);    }
  };
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CCandle::CCandle(const double open,const double close,const double high,const double l
                                 const int width,const uint clr_inc=0x000000,const uin
                                 m_open(open),m_close(close),m_high(high),m_low(low),
                                 m_clr_inc(clr_inc),m_clr_dec(clr_dec),m_width(width)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CCandle::~CCandle(void)
  {
  }
//+------------------------------------------------------------------+
//| Custom method for plot candles                                   |
//+------------------------------------------------------------------+
void PlotCandles(double &x[],double &y[],int size,CGraphic *graphic,CCanvas *canvas,vo
  {
//--- check obj
   CArrayObj *candles=dynamic_cast<CArrayObj*>(cbdata);
   if(candles==NULL || candles.Total()!=size)
      return;
//--- plot candles
   for(int i=0; i<size; i++)
     {
      CCandle *candle=dynamic_cast<CCandle*>(candles.At(i));
```

© 2000-2025, MetaQuotes Ltd.

```
      if(candle==NULL)
         return;
      //--- primary calculate
      int xc=graphic.ScaleX(x[i]);
      int width_2=candle.CandleWidth()/2;
      int open=graphic.ScaleY(candle.OpenValue());
      int close=graphic.ScaleY(candle.CloseValue());
      int high=graphic.ScaleY(candle.HigthValue());
      int low=graphic.ScaleY(candle.LowValue());
      uint clr=(open<=close) ? candle.CandleColorIncrement() :  candle.CandleColorDecr
      //--- plot candle
      canvas.LineVertical(xc,high,low,0x000000);
      //--- plot candle real body
      canvas.FillRectangle(xc+width_2,open,xc-width_2,close,clr);
      canvas.Rectangle(xc+width_2,open,xc-width_2,close,0x000000);
     }
  }
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int count=10;
   int width=10;
   double x[];
   double y[];
   ArrayResize(x,count);
   ArrayResize(y,count);
   CArrayObj candles();
   double max=0;
   double min=0;
//--- create values
   for(int i=0; i<count; i++)
     {
      x[i] = i;
      y[i] = i;
      //--- calculate values
      double open=MathRound(50.0+(MathRand()/32767.0)*50.0);
      double close=MathRound(50.0+(MathRand()/32767.0)*50.0);
      double high=MathRound(MathMax(open,close)+(MathRand()/32767.0)*10.0);
      double low=MathRound(MathMin(open,close) -(MathRand()/32767.0)*10.0);
      //--- find max and min
      if(i==0 || max<high)
         max=high;
      if(i==0 || min>low)
         min=low;
      //--- create candle
      CCandle *candle=new CCandle(open,close,high,low,width);
      candles.Add(candle);
     }
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"CandleGraphic",0,30,30,780,380))
     {
      graphic.Attach(0,"CandleGraphic");
     }
//--- create curve
   CCurve *curve=graphic.CurveAdd(x,y,CURVE_CUSTOM,"Candles");
//--- sets the curve properties
   curve.CustomPlotFunction(PlotCandles);
   curve.CustomPlotCBData(GetPointer(candles));
```

© 2000-2025, MetaQuotes Ltd.

```
//--- sets the graphic properties
   graphic.YAxis().Max((int)max);
   graphic.YAxis().Min((int)min);
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

© 2000-2025, MetaQuotes Ltd.

PointsType (Get-Methode)

Gibt das Flag zurück, das den Punktentyp angibt, die beim Zeichnen der Kurve mit Punkten verwendet werden.

```
ENUM_POINT_TYPE  PointsType()
```

Rückgabewert

Wert des Flags, das den Punktentyp angibt.

PointsType (Set-Methode)

Setzt das Flag, das den Punktentyp angibt, die beim Zeichnen der Kurve mit Punkten verwendet werden.

```
void  PointsType(
   ENUM_POINT_TYPE  type      // Wert des Flags
   )
```

Parameter

type

[in] Wert des Flags, das den Typ der beim Zeichnen der Kurve verwendeten Punkten angibt.

© 2000-2025, MetaQuotes Ltd.

StepsDimension (Get-Methode)

Gibt den Wert zurück, der die Dimension angibt, die bei einem stufenartigen Zeichnen der Kurve verwendet wird.

```
int  StepsDimension()
```

Rückgabewert

Dimension, die bei einem stufenartigen Zeichnen der Kurve verwendet wird.

StepsDimension (Set-Methode)

Setzt den Wert, der die Dimension angibt, die bei einem stufenartigen Zeichnen der Kurve verwendet wird.

```
void  StepsDimension(
   const int  dimension      // Dimension
   )
```

Parameter

dimension

[in] Dimension (0 oder 1).

Hinweis

0 — x (zuerst wird die horizontale Linie gezeichnet, dann die vertikale).

1 — y (zuerst wird die vertikale Linie gezeichnet, dann die horizontale).

© 2000-2025, MetaQuotes Ltd.

TrendLineCoefficients (Get-Methode)

Gibt die Koeffizienten der Trendlinie für das Schreiben ins Array zurück.

```
double&  TrendLineCoefficients()
```

Rückgabewert

Koeffizienten der Trendlinie.

TrendLineCoefficients (Set-Methode)

Erhält Koeffizienten der Trendlinie für das Schreiben ins Array.

```
void  TrendLineCoefficients(
   double&  coefficients[]      // Array für das Schreiben von Koeffizienten
   )
```

Parameter

coefficients[]

[out] Array für das Schreiben von Koeffizienten.

© 2000-2025, MetaQuotes Ltd.

TrendLineColor (Get-Methode)

Gibt die Farbe der Trendlinie für die Kurve zurück.

```
uint  TrendLineColor()
```

Rückgabewert

Farbe der Trendlinie.

TrendLineColor (Set-Methode)

Setzt die Farbe der Trendlinie für die Kurve.

```
void  TrendLineColor(
   const uint  clr      // Farbe der Trendlinie
   )
```

Parameter

clr

[in] Farbe der Linie

© 2000-2025, MetaQuotes Ltd.

TrendLineVisible (Get-Methode)

Gibt das Flag zurück, das angibt, ob die Trendlinie auf dem Chart sichtbar ist.

```
bool  TrendLineVisible()
```

Rückgabewert

Wert des Flags, das angibt ob die Trendlinie auf dem Chart sichtbar ist.

TrendLineVisible (Set-Methode)

Setzt das Flag das angibt, ob die Trendlinie auf dem Chart sichtbar ist.

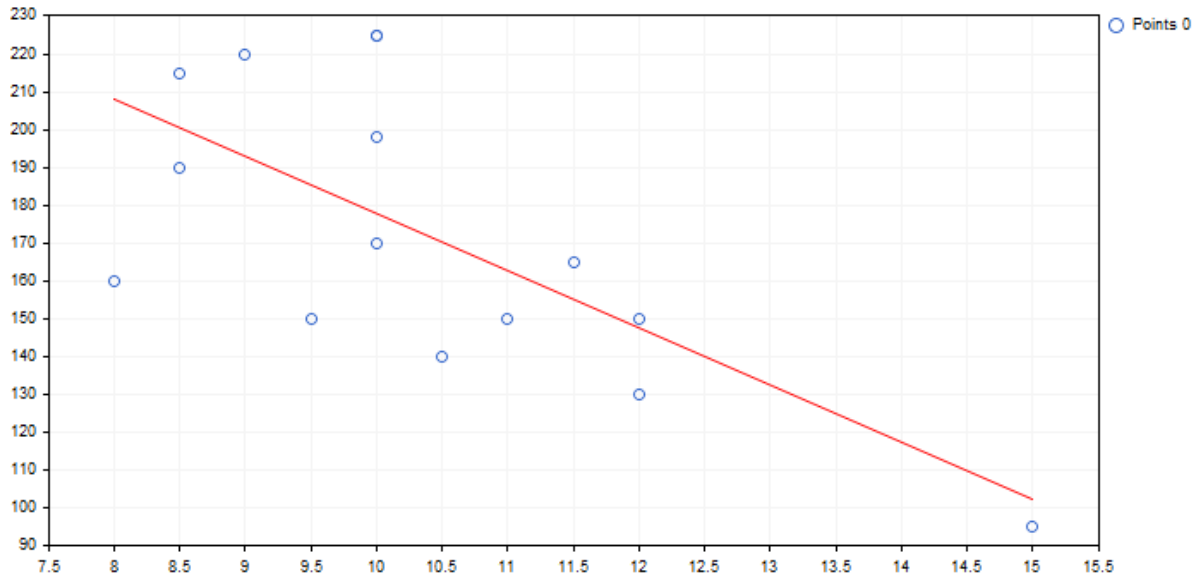
```
void  TrendLineVisible(
   const bool  visible      // Wert des Flags
   )
```

Parameter

visible

[in] Der Wert des Flags, das angibt, ob die Trendlinie auf dem Chart sichtbar ist.

Beispiel:



Der Code des Zeichnens der oben beschriebenen Trendlinie und ihrer Anzeige auf dem Chart:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//|                                              TrendLineGraphic.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#include <Graphics\Graphic.mqh>
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   double x[]={12.0,11.5,11.0,12.0,10.5,10.0,9.0,8.5,10.0,8.5,10.0,8.0,9.5,10.0,15.0};
   double y[]={130.0,165.0,150.0,150.0,140.0,198.0,220.0,215.0,225.0,190.0,170.0,160.0
//--- create graphic
   CGraphic graphic;
   if(!graphic.Create(0,"TrendLineGraphic",0,30,30,780,380))
     {
      graphic.Attach(0,"TrendLineGraphic");
     }
//--- create curve
   CCurve *curve=graphic.CurveAdd(x,y,CURVE_POINTS);
//--- sets the curve properties
   curve.TrendLineVisible(true);
   curve.TrendLineColor(ColorToARGB(clrRed));
//--- plot
   graphic.CurvePlotAll();
   graphic.Update();
  }
```

© 2000-2025, MetaQuotes Ltd.

Update

Aktualisiert Koordinaten der Kurve.

Version für das Arbeiten nach der Y-Koordinate In diesem Fall werden Indizes des übergebenen Arrays als X-Koordinaten auftreten.

```
void  Update(
   const double&  y[]   // Y-Koordinaten
   )
```

Version für das Arbeiten mit dem Koordinatenpaar X und Y.

```
void  Update(
   const double&  x[],     // X-Koordinaten
   const double&  y[]      // Y-Koordinaten
   )
```

Version für das Arbeiten mit den Punkten CPoint2D.

```
void  Update(
   const CPoint2D&  points[]      // Koordinaten der Kurve
   )
```

Version für das Arbeiten mit dem Bezeichner auf die CurveFunction Funktion.

```
void  Update(
   CurveFunction  function,     // Bezeichner auf die Funktion, die die Kurve beschrei
   const double   from,         // Anfangswert des Arguments
   const double   to,           // Endwert des Arguments
   const double   step          // Inkrement des Arguments
   )
```

Parameter

x[]

[in] X-Koordinaten.

y[]

[in] Y-Koordinaten.

points[]

[in] Koordinaten der Kurve.

function

[in] Bezeichner auf die Funktion, die die Kurve beschreibt

from

[in] Anfangswert des Arguments

to

[in] Endwert des Arguments

step

© 2000-2025, MetaQuotes Ltd.

[in] Inkrement des Arguments

© 2000-2025, MetaQuotes Ltd.

Visible (Get-Methode)

Gibt das Flag zurück, das angibt, ob die Funktion auf dem Chart sichtbar ist.

```
void  Visible(
   const bool  visible      //
   )
```

Rückgabewert

Wert des Flags, das angibt, ob die Trendlinie auf dem Chart sichtbar ist.

Visible (Set-Methode)

Setzt das Flag zurück, das angibt, ob die Funktion auf dem Chart sichtbar ist.

```
void  Visible(
   const bool  visible      // Wert des Flags
   )
```

Parameter

visible

[in] Wert des Flags, das angibt, ob die Trendlinie auf dem Chart sichtbar ist.

© 2000-2025, MetaQuotes Ltd.

CGraphic

Die CGraphic Klasse ist eine Basisklasse für die Erstellung benutzerdefinierter Grafiken.

Beschreibung

Die CGraphic Klasse ermöglicht Operationen mit benutzerdefinierten Grafiken.

Die Klasse speichert Basis-Elemente einer Grafik, setzt ihre Parameter und zeichnet sie.

Darüber hinaus speichert diese Klasse Kurven für die Grafik und ermöglicht verschiedene Anzeigemodi.

Deklaration

```
class CGraphic
```

Überschrift

```
#include <Graphics\Graphic.mqh>
```

Methoden der Klasse

Methode	Beschreibung
Create	Erstellt eine grafische Ressource, die mit einem Chart-Objekt verknüpft ist
Destroy	Entfernt die Grafik und löscht die grafische Ressource
Update	Zeigt vorgenommene Änderungen auf dem Bildschirm an
ChartObjectName	Liefert den Namen des verknüpften Objekts
ResourceName	Liefert den Namen der grafischen Ressource
XAxis	Liefert den Bezeichner auf die X-Achse
YAxis	Liefert den Bezeichner auf die Y-Achse.
GapSize	Einzug zwischen den Elementen der Grafik erhalten/setzen.
BackgroundColor	Hintergrundfarbe erhalten/setzen
BackgroundMain	Text der Überschrift erhalten/setzen
BackgroundMainSize	Schriftgröße der Überschrift erhalten/setzen
BackgroundMainColor	Farbe der Überschrift erhalten/setzen
BackgroundSub	Text der Zwischenüberschrift erhalten/setzen
BackgroundSubSize	Schriftgröße der Überschrift erhalten/setzen

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
BackgroundSubColor	Farbe der Zwischenüberschrift erhalten/setzen
GridLineColor	Farbe der Gitterlinien erhalten/setzen
GridBackgroundColor	Hintergrundfarbe des Gitters erhalten/setzen
GridCircleRadius	Halbmesser der Punkte in den Gitterpunkten erhalten/setzen
GridCircleColor	Farbe der Punkte in den Gitterpunkten erhalten/setzen
GridHasCircle	Flag des Zeichnens der Punkte in Gitterpunkten erhalten/setzen
GridAxisLineColor	Gibt den Wert der Farbe realer Achsen des Charts zurück.
HistoryNameWidth	Maximal erlaubte Länge für die Anzeige des Namens der Kurve erhalten/setzen
HistoryNameSize	Schriftgröße für den Namen der Kurve erhalten/setzten
HistorySymbolSize	Größe der verwendeten Symbole und Zeichen erhalten/setzen
TextAdd	Fügt Text hinzu
LineAdd	Fügt Linien hinzu
CurveAdd	Erstellt und fügt Kurven hinzu
CurvePlot	Zeichnet eine früher erstellte Kurve mit dem angegebenen Index
CurvePlotAll	Zeichnet alle früher erstellten Kurven
CurveGetByIndex	Erhält eine Kurve nach dem angegebenen Index
CurveGetByName	Erhält eine Kurve nach dem angegebenen Namen
CurveRemoveByIndex	Löscht eine Kurve nach dem angegebenen Index.
CurveRemoveByName	Löscht eine Kurve nach dem angegebenen Namen.
CurvesTotal	Gibt die Anzahl der Kurven für angegebenen Chart.
MarksToAxisAdd	Fügt Striche auf eine Chart-Achse
MajorMarkSize	Größe der Striche auf einer Chart-Achse erhalten/setzten
FontSet	Parameter der aktuellen Schriftart setzen
FontGet	Parameter der aktuellen Schriftart erhalten

© 2000-2025, MetaQuotes Ltd.

Methode	Beschreibung
Attach	Grafische Ressource erhalten/erstellen und sie an die Instanz der CGraphic Klasse binden
CalculateMaxMinValues	Berechnet die minimalen und maximalen Werte des Charts an beiden Achsen.
Height	Gibt die Höhe des Charts in Pixel zurück.
IndentDown	Einzug des Charts von der unteren Grenze erhalten/setzen.
IndentLeft	Einzug des Charts von der linken Grenze erhalten/setzen.
IndentRight	Einzug des Charts von der rechten Grenze erhalten/setzen.
IndentUp	Einzug des Charts von der oberen Grenze erhalten/setzen.
Redraw	Zeichnet den Chart neu.
ResetParameters	Setzt alle Parameter zurück, die für Neuzeichnung des Charts.
ScaleX	Skaliert den Wert nach der X-Achse.
ScaleY	Skaliert den Wert nach der Y-Achse.
SetDefaultParameters	Setzt Parameter des Charts als Standardparameter.
Width	Gibt die Chartbreite in Pixel zurück.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine grafische Ressource, die mit einem Chart-Objekt verknüpft ist .

```
bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Unterfenster-Index
   const int      x1,          // x1-Koordinate
   const int      y1,          // y1-Koordinate
   const int      x2,          // x2-Koordinate
   const int      y2           // y1-Koordinate
   )
```

Parameter

chart

[in] Chart-ID.

name

[in] Name.

subwin

[in] Index des Unterfensters.

x1

[in] X1-Koordinate.

y1

[in] Y1-Koordinate.

x2

[in] X2-Koordinate.

y2

[in] Y2-Koordinate.

© 2000-2025, MetaQuotes Ltd.

Destroy

Entfernt einen Chart und löscht eine grafische Ressource.

```
void  Destroy()
```

© 2000-2025, MetaQuotes Ltd.

Update

Zeigt vorgenommene Änderungen auf dem Bildschirm an.

```
void  Update(
   const bool  redraw=true      // Flag
   )
```

Parameter

redraw=true

[in] Flag-Wert

© 2000-2025, MetaQuotes Ltd.

ChartObjectName

Erhält den Namen eines mit dem Chart verknüpften Objekts.

```
string  ChartObjectName()
```

Rückgabewert

Name eines mit dem Chart verknüpften Objekts.

© 2000-2025, MetaQuotes Ltd.

ResourceName

Erhält den Namen einer grafischen Ressource.

```
string  ResourceName()
```

Rückgabewert

Name der grafischen Ressource.

© 2000-2025, MetaQuotes Ltd.

XAxis

Liefert den Bezeichner auf die X-Achse.

```
CAxis  *XAxis()
```

Rückgabewert

Bezeichner auf die X-Achse.

© 2000-2025, MetaQuotes Ltd.

YAxis

Liefert den Bezeichner auf die Y-Achse.

```
CAxis  *YAxis()
```

Rückgabewert

Bezeichner auf die Y-Achse.

© 2000-2025, MetaQuotes Ltd.

GapSize (Get-Methode)

Liefert die Größe des Einzugs zwischen den Elementen des Charts.

```
int  GapSize()
```

Rückgabewert

Einzug in Pixel.

GapSize (Set-Methode)

Setzt die Größe des Einzugs zwischen den Elementen des Charts.

```
void  GapSize(
   const int  size     // Einzug
   )
```

Parameter

size

[in] Einzug in Pixel.

© 2000-2025, MetaQuotes Ltd.

BackgroundColor (Get-Methode)

Liefert die Hintergrundfarbe.

```
color  BackgroundColor()
```

BackgroundColor (Set-Methode)

Setzt die Hintergrundfarbe.

```
void  BackgroundColor(
   const color  clr      // Hintergrundfarbe
   )
```

Parameter

clr

[in] Hintergrundfarbe.

© 2000-2025, MetaQuotes Ltd.

BackgroundMain (Get-Methode)

Liefert den Text der Überschrift eines Charts

```
string  BackgroundMain()
```

BackgroundMain (Set-Methode)

Setzt einen Text für die Überschrift.

```
void  BackgroundMain(
   const string  main      // Text der Überschrift
   )
```

Parameter

main

[in] Text der Chart-Überschrift

© 2000-2025, MetaQuotes Ltd.

BackgroundMainSize (Get-Methode)

Liefert die Größe der Überschrift.

```
int  BackgroundMainSize()
```

Rückgabewert

Schriftgröße der Überschrift.

BackgroundMainSize (Set-Methode)

Setzt die Größe der Überschrift.

```
void  BackgroundMainSize(
   const int  size     // Größe der Überschrift
   )
```

Parameter

size

[in] Schriftgröße der Überschrift.

© 2000-2025, MetaQuotes Ltd.

BackgroundMainColor (Get-Methode)

Liefert die Farbe der Überschrift.

```
color  BackgroundMainColor()
```

Rückgabewert

Farbe der Überschrift.

BackgroundMainColor (Set-Methode)

Setzt die Farbe der Überschrift.

```
void  BackgroundMainColor(
   const color  clr      // Farbe der Überschrift
   )
```

Parameter

clr

[in] Farbe der Überschrift.

© 2000-2025, MetaQuotes Ltd.

BackgroundSub (Get-Methode)

Liefert der Text der Zwischenüberschrift.

```
string  BackgroundSub()
```

Rückgabewert

Text der Zwischenüberschrift.

BackgroundSub (Set-Methode)

Setzt den Text der Zwischenüberschrift.

```
void  BackgroundSub (Set-Methode)(
   const string  sub      // Text der Zwischenüberschrift
   )
```

Parameter

sub

[in] Text der Zwischenüberschrift.

© 2000-2025, MetaQuotes Ltd.

BackgroundSubSize (Get-Methode)

Liefert die Schriftgröße der Zwischenüberschrift

```
int BackgroundSubSize()
```

BackgroundSubSize (Get-Methode)

Setzt die Größe der Zwischenüberschrift

```
void  BackgroundSubSize(
   const int  size      // Schriftgröße der Zwischenüberschrift
   )
```

Parameter

size

[in] Schriftgröße der Zwischenüberschrift

© 2000-2025, MetaQuotes Ltd.

BackgroundSubColor (Get-Methode)

Liefert die Farbe der Zwischenüberschrift.

```
color  BackgroundSubColor()
```

BackgroundSubColor (Set-Methode)

Setzt die Farbe der Zwischenüberschrift.

```
void  BackgroundSubColor(
   const color  clr      // Farbe der
   )
```

Parameter

clr

[in] Farbe der Zwischenüberschrift.

© 2000-2025, MetaQuotes Ltd.

GridLineColor (Get-Methode)

Liefert die Farbe der Gitterlinien

```
color  GridLineColor()
```

Rückgabewert

Farbe der Gitterlinien.

GridLineColor (Set-Methode)

Setzt die Farbe der Gitterlinien.

```
void  GridLineColor(
   const color  clr      // Farbe der Linien
   )
```

Parameter

clr

[in] Farbe der Gitterlinien.

© 2000-2025, MetaQuotes Ltd.

GridBackgroundColor (Get-Methode)

Liefert die Hintergrundfarbe des Gitters

```
color  GridBackgroundColor()
```

Rückgabewert

Hintergrundfarbe des Gitters

GridBackgroundColor (Set-Methode)

Setzt die Hintergrundfarbe des Gitters

```
void  GridBackgroundColor(
   const color  clr      // Hintergrundfarbe des Gitters
   )
```

Parameter

clr

[in] Hintergrundfarbe des Gitters

© 2000-2025, MetaQuotes Ltd.

GridCircleRadius (Get-Methode)

Liefert das Halbmesser der Punkte in den Gitterpunkten.

```
int  GridCircleRadius()
```

Rückgabewert

Halbmesser der Punkte in Pixel.

GridCircleRadius (Set-Methode)

Setzt das Halbmesser der Punkte in den Gitterpunkten

```
void  GridCircleRadius(
   const int  r      // Halbmesser
   )
```

Parameter

r

[in] Halbmesser der Punkte in Pixel.

© 2000-2025, MetaQuotes Ltd.

GridCircleColor (Get-Methode)

Liefert die Farbe der Gitterpunkte.

```
color  GridCircleColor()
```

Rückgabewert

Farbe der Punkte.

GridCircleColor (SetMethode)

Setzt die Farbe der Gitterpunkte.

```
void  GridCircleColor(
   const color  clr      // Farbe der Punkte
   )
```

Parameter

clr

[in] Farbe der Punkte.

© 2000-2025, MetaQuotes Ltd.

GridHasCircle (Get-Methode)

Liefert ein Flag, das festlegt, ob Punkte in den Gitterpunkten gezeichnet werden müssen.

```
bool  GridHasCircle()
```

Rückgabewert

Wert des Flags.

Hinweis

true — Punkte zeichnen

false — keine Punkte zeichnen

GridHasCircle (Set-Methode)

Setzt das Flag, dass festlegt, ob Punkte in den Gitterpunkten gezeichnet werden müssen.

```
void  GridHasCircle(
   const bool  has
   )
```

Parameter

has

[in] Flag.

Hinweis

true — Punkte zeichnen

false — keine Punkte zeichnen

© 2000-2025, MetaQuotes Ltd.

GridAxisLineColor (Get-Methode)

Gibt den Wert der Farbe realer Achsen des Charts zurück.

```
uint  GridAxisLineColor()
```

Rückgabewert

Farbe realer Achsen des Charts.

GridAxisLineColor (Set-Methode)

Setzt den Wert realer Achsen des Charts.

```
void  GridAxisLineColor(
   const uint  clr      // Farbe der Achsen des Chart
   )
```

Parameter

clr

[in] Farbe realer Achsen des Charts.

© 2000-2025, MetaQuotes Ltd.

HistoryNameWidth (Get-Methode)

Liefert die maximal erlaubte Länge für die Anzeige des Namens der Kurve.

```
int  HistoryNameWidth()
```

Rückgabewert

Maximale Länge in Pixel.

Hinweis

Wenn der Name der Kurve die maximal erlaubte Länge übersteigt, wird er abgeschnitten und am Ende werden die Auslassungspunkte gesetzt.

HistoryNameWidth (Set-Methode)

Setzt die maximal erlaubte Länge für die Anzeige des Namens der Kurve.

```
void  HistoryNameWidth(
   const int  width      // maximale Länge
   )
```

Parameter

width

[in] Maximale Länge in Pixel.

Hinweis

Wenn der Name der Kurve die maximal erlaubte Länge übersteigt, wird er abgeschnitten und am Ende werden die Auslassungspunkte gesetzt.

© 2000-2025, MetaQuotes Ltd.

HistoryNameSize (Get-Methoden)

Liefert die Schriftgröße des Namens der Kurve.

```
int  HistoryNameSize()
```

Rückgabewert

Schriftgröße des Namens der Kurve.

HistoryNameSize (Set-Methode)

Setzt die Schriftgröße des Namens der Kurve.

```
void  HistoryNameSize (Set-Methode)(
   const int  size      // Schriftgröße des Namens
   )
```

Parameter

size

[in] Schriftgröße des Namens.

© 2000-2025, MetaQuotes Ltd.

HistorySymbolSize (Get-Methode)

Liefert die Größe verwendeter Symbole und Zeichen

```
int  HistorySymbolSize()
```

Rückgabewert

Größe verwendeter Symbole und Zeichen

HistorySymbolSize (Set-Methode)

Setzt die Größe verwendeter Symbole und Zeichen

```
void  HistorySymbolSize(
   const int  size      // Symbolgröße
   )
```

Parameter

size

[in] Größe verwendeter Symbole und Zeichen.

© 2000-2025, MetaQuotes Ltd.

TextAdd

Fügt einen Text auf den Chart hinzu.

Die Version für Operationen mit den X- und Y-Koordinaten

```
void  TextAdd(
   const int      x,               // X-Koordinate
   const int      y,               // Y-Koordinate
+   const string   text,             // Text
   const uint     clr,             // Farbe
   const uint     alignment=0      // Ausrichtung
   )
```

Die Version für CPoint

```
void  TextAdd(
   const CPoint   &point,          // Koordinate des Punktes
+   const string   text,             // Text
   const uint     clr,             // Farbe
   const uint     alignment=0      // Ausrichtung
   )
```

Parameter

x

[in] X-Koordinate.

y

[in] Y-Koordinate.

&point

[in] Koordinate des Punktes.

text

[in] Text.

clr

[in] Farbe.

alignment=0

[in] Ausrichtung.

© 2000-2025, MetaQuotes Ltd.

LineAdd

Fügt eine Linie auf den Chart hinzu.

Die Version für Operationen mit den X- und Y-Koordinaten

```
void  LineAdd(
   const int   x1,         // x1-Koordinate
   const int   y1,         // y1-Koordinate
   const int   x2,         // x2-Koordinate
   const int   y2,         // y2-Koordinate
   const uint  clr,        // Farbe
   const uint  style       // Stil
   )
```

Die Version für CPoint

```
void  LineAdd2(
   const CPoint  &point1,     // Koordinate des ersten Punktes
   const CPoint  &point2,     // Koordinate des zweiten Punktes
   const uint    clr,         // Farbe
   const uint    style        // Stil
   )
```

Parameter

x1

[in] X1-Koordinate.

y1

[in] Y1-Koordinate.

x2

[in] X2-Koordinate.

y2

[in] Y2-Koordinate.

&point1

[in] Koordinate des ersten Punktes.

&point2

[in] Koordinate des zweiten Punktes.

clr

[in] Farbe.

style

[in] Stil.

© 2000-2025, MetaQuotes Ltd.

CurveAdd

Erstellt und fügt eine neue Kurve auf den Chart hinzu.

Die Version für Operationen mit der Y-Koordinate (die Farbe wird automatisch gesetzt)

```
CCurve* CurveAdd(
   const double      &y[],          // Y-Koordinaten
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

Hinweis

Als X-Koordinaten dienen für diese Kurve die Indexe des Y-Arrays.

Die Version für Operationen mit den X und Y Koordinaten (die Farbe der Kurve wird automatisch gesetzt)

```
CCurve*  CurveAdd(
   const double      &x[],          // X-Koordinaten
   const double      &y[],          // Y-Koordinaten
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

Die Version für Operationen mit den Punkten CPoint2D (die Farbe der Kurve wird automatisch gesetzt)

```
CCurve*  CurveAdd(
   const CPoint2D    &points[],     // Koordinaten der Punkte
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

Die Version für Operationen mit dem Bezeichner auf die Funktion CurveFunction (die Farbe der Kurve wird automatisch gesetzt)

```
CCurve*  CurveAdd(
   CurveFunction     function,      // Bezeichner auf die Funktion
   const double      from,          // Anfangswert des Arguments
   const double      to,            // Endwert des Arguments
   const double      step,          // Inkrement des Arguments
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

© 2000-2025, MetaQuotes Ltd.

Die Version für Operationen mit Y-Koordinate (die Farbe der Kurve wird vom Nutzer definiert)

```
CCurve*  CurveAdd(
   const double      &y[],          // Y-Koordinaten
   const uint        clr,           // Farbe der Kurve
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

Hinweis

Als X-Koordinaten dienen für diese Kurve die Indexe des Y-Arrays.

Die Version für Operationen mit den X und Y Koordinaten (Farbe der Kurve wird vom Nutzer definiert)

```
CCurve*  CurveAdd(
   const double      &x[],          // X-Koordinaten
   const double      &y[],          // Y-Koordinaten
   const uint        clr,           // Farbe der Kurve
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

Die Version für Operationen mit den Punkten CPoint2D (die Farbe der Kurve wird vom Nutzer definiert)

```
CCurve*  CurveAdd(
   const CPoint2D    &points[],     // Koordinaten der Punkte
   const uint        clr,           // Farbe der Kurve
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
   const string      name=NULL      // Name der Kurve
   )
```

Die Version für Operationen mit dem Bezeichner auf die Funktion CurveFunction (die Farbe der Kurve wird vom Nutzer definiert)

```
CCurve*  CurveAdd(
   CurveFunction     function,      // Bezeichner auf die Funktion
   const double      from,          // Anfangswert des Arguments
   const double      to,            // Endwert des Arguments
   const double      step,          // Inkrement des Arguments
   const uint        clr,           // Farbe der Kurve
   ENUM_CURVE_TYPE   type,          // Typ der Kurve
```

© 2000-2025, MetaQuotes Ltd.

```
   const string      name=NULL      // Name der Kurve
   )
```

Parameter

&x[]

[in] X-Koordinate.

&y[]

[in] Y-Koordinate.

&points[]

[in] Koordinaten der Punkte.

function

[in] Bezeichner auf die Funktion.

from

[in] Anfangswert des Arguments.

to

[in] Endwert des Arguments.

step

[in] Inkrement des Arguments.

type

[in] Typ der Kurve.

name=NULL

[in] Name der Kurve.

clr

[in] Farbe der Kurve.

Rückgabewert

Bezeichner auf die erstellte Kurve.

© 2000-2025, MetaQuotes Ltd.

CurvePlot

Zeigt eine früher erstellte Kurve mit einem angegebenen Index an.

```
bool  CurvePlot(
   const int  index      // Index
   )
```

Parameter

index

[in] Index der Kurve

Rückgabewert

Wenn erfolgreich - true, wenn nicht - false.

© 2000-2025, MetaQuotes Ltd.

CurvePlotAll

Zeigt alle früher auf den Chart hinzugefügten Kurven.

```
bool  CurvePlotAll()
```

Rückgabewert

Wenn erfolgreich - true, wenn nicht - false.

© 2000-2025, MetaQuotes Ltd.

CurveGetByIndex

Erhält eine Kurve nach dem angegebenen Index.

```
CCurve*  CurveGetByIndex(
   const int  index      // Index der Kurve
   )
```

Parameter

index

[in] Index der Kurve.

Rückgabewert

Bezeichner auf die Kurve mit einem angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

CurveGetByName

Erhält die Kurve nach einem angegebenen Namen.

```
CCurve*  CurveGetByName(
   const string  name      // Name der Kurve
   )
```

Parameter

name

[in] Name der Kurve.

Rückgabewert

Bezeichner auf die erste gefundene Kurve mit dem angegebenen Namen.

© 2000-2025, MetaQuotes Ltd.

CurveRemoveByIndex

Löscht eine Kurve nach dem angegebenen Index.

```
bool  CurveRemoveByIndex(
   const int  index      // Index der Kurve
   )
```

Parameter

index

[in] Index der Kurve, die gelöscht werden muss.

Rückgabewert

true — wenn erfolgreich, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

CurveRemoveByName

Löscht eine Kurve nach dem angegebenen Namen.

```
bool  CurveRemoveByName(
   const string  name      // Name der Kurve
   )
```

Parameter

name

[in] Name der Kurve, die gelöscht werden muss.

Rückgabewert

true — wenn erfolgreich, andernfalls — false.

© 2000-2025, MetaQuotes Ltd.

CurvesTotal

Gibt die Anzahl der Kurven für angegebenen Chart.

```
int  CurvesTotal()
```

Rückgabewert

Anzahl der Kurven.

Hinweis

Es werden alle Kurven gezählt, die zu diesem Chart gehören, unabhängig von ihrem Stil und ihrer Sichtbarkeit.

© 2000-2025, MetaQuotes Ltd.

MarksToAxisAdd

Fügt Striche auf die angegebene Chart-Achse hinzu.

```
bool  MarksToAxisAdd(
   const double         &marks[],        // Koordinaten der "Striche"
   const int            mark_size,       // Größe der "Striche"
   ENUM_MARK_POSITION   position,        // Stelle der "Striche"
   const int            dimension=0      // Dimension
   )
```

Parameter

&marks[]

[in] Koordinaten der Striche

mark_size

[in] Größe der Striche

position

[in] Stelle der Striche

dimension=0

[in] 0 — auf die X-Achse hinzufügen,

1 — auf die Y-Achse hinzufügen

Rückgabewert

Wenn erfolgreich - true, wenn nicht - false.

© 2000-2025, MetaQuotes Ltd.

MajorMarkSize (Get-Methode)

Liefert die Größe der Striche auf der Skala auf den Koordinatenachsen.

```
int  MajorMarkSize()
```

MajorMarkSize (Set-Methode)

Setzt die Größe der Striche auf den Koordinatenachsen.

```
void  MajorMarkSize(
   const int  size      // Größe der "Striche"
   )
```

Parameter

size

[in] Größe der "Striche" in Pixel.

© 2000-2025, MetaQuotes Ltd.

FontSet

Setzt Parameter der aktuellen Schriftart.

```
bool  FontSet(
   const string  name,         // Name
   const int     size,         // Größe
   const uint    flags=0,      // Flags
   const uint    angle=0       // Winkel
   )
```

Parameter

name

[in] Name.

size

[in] Größe.

flags=0

[in] Flags.

angle=0

[in] Winkel.

Rückgabewert

true — wenn erfolgreich, sonst — false.

© 2000-2025, MetaQuotes Ltd.

FontGet

Erhält Parameter der aktuellen Schriftart.

```
void  FontGet(
   string  &name,      // Name
   int     &size,      // Größe
   uint    &flags,     // Flags
   uint    &angle      // Winkel
   )
```

Parameter

&name

[out] Name.

&size

[out] Größe.

&flags

[out] Flags.

&angle

[out] Winkel.

© 2000-2025, MetaQuotes Ltd.

Attach

Version für das Erhalten der grafischen Ressource aus dem OBJ_BITMAP_LABEL Objekt und seine Bindung an eine Instanz der CGraphic Klasse:

```
bool  Attach(
   const long     chart_id,      // Chart-ID
   const string   objname        // Name des grafischen Objekts
   )
```

Version für die Erstellung einer grafischen Ressource für das Objekt OBJ_BITMAP_LABEL und für seine Bindung an die Instanz der CGraphic Klasse:

```
bool  Attach(
   const long     chart_id,      // Chart-ID
   const string   objname,       // Name des grafischen Objekts
   const int      width,         // Bildbreite
   const int      height         // Bildhöhe
   )
```

Parameter

chart_id

[in] Charts-ID.

objname

[in] Bezeichnung (Name) des grafischen Objekts.

width

[in] Bildbreite in der Ressource.

height

[in] Bildhöhe in der Ressource.

Rückgabewert

true — wenn erfolgreich, false — wenn Bindung fehlgeschlagen ist.

© 2000-2025, MetaQuotes Ltd.

CalculateMaxMinValues

Berechnet die minimalen und maximalen Werte des Charts an beiden Achsen.

```
void  CalculateMaxMinValues()
```

© 2000-2025, MetaQuotes Ltd.

Height

Gibt die Höhe des Charts in Pixel zurück.

```
int  Height()
```

Rückgabewert

Charthöhe in Pixel.

© 2000-2025, MetaQuotes Ltd.

IndentDown (Get-Methode)

Gibt den Einzug des Charts von der unteren Grenze zurück.

```
int  IndentDown()
```

Rückgabewert

Einzug in Pixel.

IndentDown (Set-Methode)

Setzt den Einzug des Charts von der unteren Grenze.

```
void  IndentDown(
   const int  down      // Einzug
   )
```

Parameter

down

[in] Einzug in Pixel.

© 2000-2025, MetaQuotes Ltd.

IndentLeft (Get-Methode)

Gibt den Einzug des Charts von der linken Grenze zurück.

```
int  IndentLeft()
```

Rückgabewert

Einzug in Pixel.

IndentLeft (Set-Methode)

Setzt den Einzug des Charts von der linken Grenze.

```
void  IndentLeft(
   const int  left      // Einzug
   )
```

Parameter

left

[in] Einzug in Pixel.

© 2000-2025, MetaQuotes Ltd.

IndentRight (Get-Methode)

Gibt den Einzug des Charts von der rechten Grenze zurück.

```
int  IndentRight()
```

Rückgabewert

Einzug in Pixel.

IndentRight (Set-Methode)

Setzt den Einzug des Charts von der rechten Grenze.

```
void  IndentRight(
   const int  right      // Einzug
   )
```

Parameter

right

[in] Einzug in Pixel.

© 2000-2025, MetaQuotes Ltd.

IndentUp (Get-Methode)

Gibt den Einzug des Charts von der oberen Grenze zurück.

```
int  IndentUp()
```

Rückgabewert

Einzug in Pixel.

IndentUp (Set-Methode)

Setzt den Einzug des Charts von der oberen Grenze.

```
void  IndentUp(
   const int  up      // Einzug
   )
```

Parameter

up

[in] Einzug in Pixel

© 2000-2025, MetaQuotes Ltd.

Redraw

Zeichnet den Chart neu.

```
bool  Redraw(
   const bool  rescale=false      // Wert des Flags
   )
```

Parameter

rescale=false

[in] Das Flag gibt an, ob der Chart neu skaliert werden muss.

Rückgabewert

Wenn erfolgreich - true, wenn nicht - false.

© 2000-2025, MetaQuotes Ltd.

ResetParameters

Setzt alle Parameter zurück, die für Neuzeichnung des Charts.

```
void  ResetParameters()
```

© 2000-2025, MetaQuotes Ltd.

ScaleX

Skaliert den Wert nach der X-Achse.

```
virtual int  ScaleX(
   double  x      // Wert an der X-Achse
   )
```

Parameter

x

[in] Realer Wert an der X-Achse.

Rückgabewert

Wert in Pixel.

Hinweis

Skaliert den Wert in Pixel für Anzeige auf dem Chart.

© 2000-2025, MetaQuotes Ltd.

ScaleY

Skaliert den Wert nach der Y-Achse.

```
virtual int  ScaleY(
   double  y      // Wert an der Y-Achse
   )
```

Parameter

y

[in] Realer Wert an der Y-Achse.

Rückgabewert

Wert in Pixel.

Hinweis

Skaliert den Wert in Pixel für Anzeige auf dem Chart.

© 2000-2025, MetaQuotes Ltd.

SetDefaultParameters

Setzt Parameter des Charts als Standardparameter.

```
void  SetDefaultParameters()
```

© 2000-2025, MetaQuotes Ltd.

Width

Gibt die Chartbreite in Pixel zurück.

```
int  Width()
```

Rückgabewert

Chartbreite in Pixel.

© 2000-2025, MetaQuotes Ltd.

Technische Indikatoren und Zeitreihen

Dieser Abschnitt enthält die Details der Arbeit mit Klassen von technischen Indikatoren und Zeitreihen, und auch die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen von technischen Indikatoren und Zeitreihen, sparen Sie Zeit bei der Entwicklung von benutzerdefinierten Anwendungen (Skripte, Expert Advisors).

Die Standardbibliothek MQL5 (in Bezug auf technische Indikatoren und Zeitreihen) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Indicators.

Klasse/Gruppe	Beschreibung
Basisklassen	Gruppe von Basis- und Hilfsklassen
Zeitreihen	Klassengruppe "Zeitreihen"
Trendindikatoren	Klassengruppe "Trend"
Oszillatoren	Klassengruppe "Oszillatoren"
Volumenindikatoren	Klassengruppe "Volumes"
Bill Williams Indikatoren	Klassengruppe "Bill Williams "
Benutzerdefinierter Indikator	Klasse "Benutzerdefinierter Indikator"

© 2000-2025, MetaQuotes Ltd.

Eine Gruppe von Basis- und Hilfsklassen von technischen Indikatoren und Zeitreihen

Dieser Abschnitt enthält die Details der Arbeit mit der Gruppe von Basis- und Hilfsklassen von technischen Indikatoren und Zeitreihen, und auch die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klasse/Gruppe	Beschreibung
CSpreadBuffer	Klasse des Puffers von Spread-Geschichte
CTimeBuffer	Klasse des Puffers von Balkeneröffnungszeit
CTickVolumeBuffer	Klasse des Puffers von Tick-Volumen von Balken
CRealVolumeBuffer	Klasse des Puffers von realen Balken-Volumen der Balken
CDoubleBuffer	Die Basisklasse des double-Datenpuffers
COpenBuffer	Klasse des Puffers von Balkeneröffnungspreisen
CHighBuffer	Klasse des Puffers von Hoch-Preisen von Balken
CLowBuffer	Klasse des Puffers von Tief-Preisen von Balken
CCloseBuffer	Klasse des Puffers von Balkenschlusspreisen
CIndicatorBuffer	Klasse des Puffers von technischen Indikator
CSeries	Die Basisklasse für den Zugriff auf serielle Daten
CPriceSeries	Die Basisklasse für den Zugriff auf Preisdaten
CIndicator	Die Basisklasse des technischen Indikators
CIndicators	Eine Sammlung von technischen Indikatoren und Zeitreihen

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Klasse CSpreadBuffer

CSpreadBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Spread-Daten.

Beschreibung

Klasse CSpreadBuffer bietet den vereinfachten Zugriff auf die historischen Spread-Daten.

Deklaration

```
class CSpreadBuffer: public CArrayInt
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayInt
      CSpreadBuffer

Gruppen der Klassenmethode

Attribute	
Size	Setzt die Puffergröße
Einstellungen	
SetSymbolPeriod	Setzt das Arbeitssymbol und Periode
Datenzugriff	
At	Erhält ein Element des Puffers
Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayInt

Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

© 2000-2025, MetaQuotes Ltd.

Size

Setzt die Puffergröße.

```
void  Size(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

SetSymbolPeriod

Setzt das Arbeitssymbol und Periode.

```
void  SetSymbolPeriod(
   const string           symbol,      // Symbol
   const ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

Ein neues Arbeitssymbol.

period

[in] Das neue Arbeitssymbol (ein Wert der Enumeration ENUM_TIMEFRAMES).

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element des Puffers.

```
int  At(
   const int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Puffers am angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CTimeBuffer

CTimeBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Eröffnungszeitdaten von Balken.

Beschreibung

Klasse CTimeBuffer bietet den vereinfachten Zugriff auf die historischen Eröffnungszeitdaten von Balken.

Deklaration

```
class CTimeBuffer: public CArrayLong
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayLong
      CTimeBuffer

Gruppen der Klassenmethode

Attribute	
Size	Setzt die Puffergröße
Einstellungen	
SetSymbolPeriod	Setzt das Arbeitssymbol und Periode
Datenzugriff	
At	Erhält ein Element des Puffers
Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayLong

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

© 2000-2025, MetaQuotes Ltd.

Size

Setzt die Puffergröße.

```
void  Size(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

SetSymbolPeriod

Setzt das Arbeitssymbol und Periode.

```
void  SetSymbolPeriod(
   const string           symbol,      // Symbol
   const ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

Ein neues Arbeitssymbol.

period

[in] Das neue Arbeitssymbol (ein Wert der Enumeration ENUM_TIMEFRAMES).

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element des Puffers.

```
long  At(
   const int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Puffers am angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CTickVolumeBuffer

CTickVolumeBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Daten von Tick-Volumen von Balken.

Beschreibung

Klasse CTickVolumeBuffer bietet den vereinfachten Zugriff auf die historischen Daten von Tick-Volumen von Balken.

Deklaration

```
class CTickVolumeBuffer: public CArrayLong
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayLong
      CTickVolumeBuffer

Gruppen der Klassenmethode

Attribute	
Size	Setzt die Puffergröße
Einstellungen	
SetSymbolPeriod	Setzt das Arbeitssymbol und Periode
Datenzugriff	
At	Erhält ein Element des Puffers
Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayLong

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

© 2000-2025, MetaQuotes Ltd.

Size

Setzt die Puffergröße.

```
void  Size(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

SetSymbolPeriod

Setzt das Arbeitssymbol und Periode.

```
void  SetSymbolPeriod(
   const string           symbol,      // Symbol
   const ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

Ein neues Arbeitssymbol.

period

[in] Das neue Arbeitssymbol (ein Wert der Enumeration ENUM_TIMEFRAMES).

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element des Puffers.

```
long  At(
   const int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Puffers am angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CRealVolumeBuffer

CRealVolumeBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Daten von realen Balken-Volumen.

Beschreibung

Klasse CRealVolumeBuffer bietet den vereinfachten Zugriff auf die historischen Daten von realen Balken-Volumen.

Deklaration

```
class CRealVolumeBuffer: public CArrayLong
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayLong
      CRealVolumeBuffer

Gruppen der Klassenmethode

Attribute	
Size	Setzt die Puffergröße
Einstellungen	
SetSymbolPeriod	Setzt das Arbeitssymbol und Periode
Datenzugriff	
At	Erhält ein Element des Puffers
Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayLong

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

© 2000-2025, MetaQuotes Ltd.

Size

Setzt die Puffergröße.

```
void  Size(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

SetSymbolPeriod

Setzt das Arbeitssymbol und Periode.

```
void  SetSymbolPeriod(
   const string           symbol,      // Symbol
   const ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

Ein neues Arbeitssymbol.

period

[in] Das neue Arbeitssymbol (ein Wert der Enumeration ENUM_TIMEFRAMES).

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element des Puffers.

```
long  At(
   const int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Puffers am angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CDoubleBuffer

CDoubleBuffer ist eine Basisklasse für den vereinfachten Zugriff auf double-Pufferdaten.

Beschreibung

Klasse CDoubleBuffer bietet ihre abgeleitete Klasse die Möglichkeit des vereinfachten Zugriffs auf double-Pufferdaten.

Deklaration

```
class CDoubleBuffer: public CArrayDouble
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble
      CDoubleBuffer

Direkte Ableitungen

CCloseBuffer, CHighBuffer, CIndicatorBuffer, CLowBuffer, COpenBuffer

Gruppen der Klassenmethode

Attribute	
Size	Setzt die Puffergröße
Einstellungen	
SetSymbolPeriod	Setzt das Arbeitssymbol und Periode
Datenzugriff	
At	Erhält ein Element des Puffers
Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayDouble

Delta, Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

© 2000-2025, MetaQuotes Ltd.

Size

Setzt die Puffergröße.

```
void  Size(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

SetSymbolPeriod

Setzt das Arbeitssymbol und Periode.

```
void  SetSymbolPeriod(
   const string           symbol,      // Symbol
   const ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

Ein neues Arbeitssymbol.

period

[in] Das neue Arbeitssymbol (ein Wert der Enumeration ENUM_TIMEFRAMES).

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element des Puffers.

```
double  At(
   const int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Puffers am angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse COpenBuffer

COpenBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Eröffnungspreisdaten von Balken.

Beschreibung

Klasse COpenBuffer bietet den vereinfachten Zugriff auf die historischen Eröffnungspreisdaten von Balken.

Deklaration

```
class COpenBuffer: public CDoubleBuffer
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble
      CDoubleBuffer
        COpenBuffer

Gruppen der Klassenmethode

Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayDouble

Delta, Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

Methoden geerbt von der Klasse CDoubleBuffer

Size, At, SetSymbolPeriod

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CHighBuffer

CHighBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Hoch-Preisen von Balken.

Beschreibung

Klasse CHighBuffer bietet den vereinfachten Zugriff auf die historischen Hochpreisdaten von Balken.

Deklaration

```
class CHighBuffer: public CDoubleBuffer
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble
      CDoubleBuffer
        CHighBuffer

Gruppen der Klassenmethode

Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayDouble

Delta, Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

Methoden geerbt von der Klasse CDoubleBuffer

Size, At, SetSymbolPeriod

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CLowBuffer

CLowBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Tief-Preisen von Balken.

Beschreibung

Klasse CLowBuffer bietet den vereinfachten Zugriff auf die historischen Tiefpreisdaten von Balken.

Deklaration

```
class CLowBuffer: public CDoubleBuffer
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble
      CDoubleBuffer
        CLowBuffer

Gruppen der Klassenmethode

Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayDouble

Delta, Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

Methoden geerbt von der Klasse CDoubleBuffer

Size, At, SetSymbolPeriod

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CCloseBuffer

CCloseBuffer ist eine Klasse für vereinfachten Zugriff auf die historischen Schlusspreisdaten von Balken.

Beschreibung

Klasse CCloseBuffer bietet den vereinfachten Zugriff auf die historischen Schlusspreisdaten von Balken.

Deklaration

```
class CCloseBuffer: public CDoubleBuffer
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble
      CDoubleBuffer
        CCloseBuffer

Gruppen der Klassenmethode

Datenaktualisierung	
virtual Refresh	Aktualisiert den gesamten Puffer
virtual RefreshCurrent	Aktualisiert nur den aktuellen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayDouble

Delta, Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

Methoden geerbt von der Klasse CDoubleBuffer

Size, At, SetSymbolPeriod

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den gesamten Puffer.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
virtual bool  RefreshCurrent()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

CIndicatorBuffer Klasse

Die CIndicatorBuffer Klasse ist eine Klasse für einen vereinfachten Zugang zu Pufferdaten eines technischen Indikators.

Beschreibung

Die CIndicatorBuffer Klasse ermöglicht einen vereinfachten Zugang zu Pufferdaten eines technischen Indikators.

Deklaration

```
class CIndicatorBuffer: public CDoubleBuffer
```

Titel

```
#include <Indicators\Indicator.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayDouble
      CDoubleBuffer
        CIndicatorBuffer

Methoden der Klasse nach Gruppen

Attribute	
Offset	Erhält/setzt Buffer-Offset
Name	Erhält/setzt Buffer-Bezeichnung
Datenzugriff	
At	Erhält das Element des Buffers
Datenaktualisierung	
Refresh	Aktualisiert den ganzen Buffer
RefreshCurrent	Aktualisiert nur den laufenden Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayDouble

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Delta, Type, Save, Load, Reserve, Resize, Shutdown, Add, AddArray, AddArray, Insert, InsertArray, InsertArray, AssignArray, AssignArray, At, operator, Minimum, Maximum, Update, Shift, Delete, DeleteRange, CompareArray, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast, SearchLinear

Methoden geerbt von der Klasse CDoubleBuffer

Size, At, SetSymbolPeriod

© 2000-2025, MetaQuotes Ltd.

Offset

Erhält den Puffer-Offset.

```
int  Offset() const
```

Rückgabewert

Puffer-Offset.

Offset

Setzt den Puffer-Offset.

```
void  Offset(
   const int  offset      // Offset
   )
```

Parameter

offset

[in] Der neue Puffer-Offset.

© 2000-2025, MetaQuotes Ltd.

Name

Erhält den Puffernamen.

```
string  Name() const
```

Rückgabewert

Ouffername.

Name

Setzt den Puffernamen.

```
void  Name(
   const string  name      // Name
   )
```

Parameter

name

[in] Der neue Puffername.

© 2000-2025, MetaQuotes Ltd.

At

Erhält ein Element des Puffers.

```
double  At(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Puffers am angegebenen Index.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert den ganzen Puffer.

```
bool  Refresh(
   const int  handle,     // Indikator-Handle
   const int  num         // Puffernummer
   )
```

Parameter

handle

[in] Indikator-Handle.

num

[in] Nummer des Indikatorpuffers.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Aktualisiert das Nullelement des Puffers.

```
bool  RefreshCurrent(
   const int  handle,     // Indikator-Handle
   const int  num         // Puffernummer
   )
```

Parameter

handle

[in] Indikator-Handle.

num

[in] Nummer des Indikatorpuffers.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Puffer nicht aktualisiert werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CSeries

CSeries ist eine Basisklasse für die Klassen vom Zugriff auf Seriendaten in der Standardbibliothek MQL5.

Beschreibung

Klasse CSeries bietet allen ihren abgeleiteten Klassen (Klassen von Zeitreihen und Indikatoren) den vereinfachten Zugriff auf die allgemeinen Funktionen API MQL5 für die Arbeit mit Seriendaten.

Deklaration

```
class CSeries: public CArrayObj
```

Kopf

```
#include <Indicators\Series.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries

Direkte Ableitungen

CIndicator, CiRealVolume, CiSpread, CiTickVolume, CiTime, CPriceSeries

Gruppen der Klassenmethode

Attribute	
Name	Erhält den Namen der Zeitreihe oder des Indikators
BuffersTotal	Erhält die Anzahl der Puffer von Zeitreihe oder Indikator
Timeframe	Erhält das Timeframe-Flag von Zeitreihe oder Indikator
Symbol	Erhält den Namen des Arbeitssymbols der Zeitreihe oder des Indikators
Period	Erhält Arbeitstimeframe von Zeitreihe oder Indikator
RefreshCurrent	Setzt/löscht das Flag von Daten-Aktualisierung.
Datenzugriff	
virtual BufferResize	Setzt die Größe der Puffer von der Zeitreihe oder Indikator
Datenaktualisierung	
virtual Refresh	Aktualisiert Daten von Zeitreihe oder Indikator
PeriodDescription	Konvertiert ENUM_TIMEFRAMES in einen String

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Name

Erhält den Namen der Zeitreihe oder des Indikators.

```
string  Name() const
```

Rückgabewert

Name der Zeitreihe oder des Indikators.

© 2000-2025, MetaQuotes Ltd.

BuffersTotal

Erhält die Anzahl der Puffer von Zeitreihe oder Indikator.

```
int  BuffersTotal() const
```

Rückgabewert

Die Anzahl der Puffer von Zeitreihe oder Indikator.

Hinweis

Die Anzahl der Puffer der Zeitreihe ist immer 1.

© 2000-2025, MetaQuotes Ltd.

Timeframe

Erhält das Timeframe-Flag von Zeitreihe oder Indikator.

```
int  Timeframe() const
```

Rückgabewert

Das Timeframe-Flag von Zeitreihe oder Indikator.

Hinweis

Ist als Flags der Objektsichtbarkeit gebildet.

© 2000-2025, MetaQuotes Ltd.

Symbol

Erhält den Namen des Arbeitssymbols der Zeitreihe oder des Indikators.

```
string  Symbol() const
```

Rückgabewert

Der Name des Arbeitssymbols der Zeitreihe oder des Indikators.

© 2000-2025, MetaQuotes Ltd.

Period

Erhält Arbeitstimeframe von Zeitreihe oder Indikator.

```
ENUM_TIMEFRAMES  Period() const
```

Rückgabewert

Arbeitstimeframe der Zeitreihe oder des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

© 2000-2025, MetaQuotes Ltd.

RefreshCurrent

Setzt das Flag der Notwendigkeit die aktuelle Werte der Zeitreihe oder Indikator ständig zu aktualisieren.

```
string  RefreshCurrent(
   const bool  flag      // Wert
   )
```

Parameter

flag

[in] Ein neuer Flagwert.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

BufferSize

Erhält die Anzahl der verfügbaren Daten in Puffer von Zeitreihe oder Indikator.

```
int  BufferSize() const
```

Rückgabewert

Die Anzahl der verfügbaren Daten in Puffer von Zeitreihe oder Indikator.

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Größe der Puffer von der Zeitreihe oder Indikator.

```
virtual bool  BufferResize(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Alle Puffer eines Indikators oder Zeitreihe werden die gleiche Größe haben.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Daten von Zeitreihe oder Indikator.

```
virtual void  Refresh(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Flags von Timeframe-Aktualisierung

© 2000-2025, MetaQuotes Ltd.

PeriodDescription

Wandelt den Wert der Enumeration ENUM_TIMEFRAMES in eine Zeile.

```
string  PeriodDescription(
   const int  val=0      // Wert
   )
```

Parameter

val=0

[in] Der Wert für Umwandlung.

Rückgabewert

Ein String, der dem Wert der Enumeration ENUM_TIMEFRAMES entspricht.

Hinweis

Wenn kein Wert angegeben ist oder gleich Null ist, wird Arbeitstimeframe von Zeitreihe oder Indikator in eine Zeile umgewandelt.

© 2000-2025, MetaQuotes Ltd.

Klasse CPriceSeries

CPriceSeries ist eine Basisklasse für die Klassen vom Zugriff auf Preisdaten.

Beschreibung

Klasse CPreiceSeries bietet allen ihren abgeleiteten Klassen den vereinfachten Zugriff auf die allgemeinen Funktionen API MQL5 für die Arbeit mit Preisdaten.

Deklaration

```
class CPriceSeries: public CSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CPriceSeries

Direkte Ableitungen

CiClose, CiHigh, CiLow, CiOpen

Gruppen der Klassenmethode

Erstellung	
virtual BufferResize	Setzt die Größe des Serienpuffers
Datenzugriff	
virtual GetData	Erhält das angegebene Element des Serienpuffers
Datenaktualisierung	
virtual Refresh	Aktualisiert Seriendaten
Suche nach Extrema	
virtual MinIndex	Erhält den Index des Minimalwerts im bestimmten Bereich
virtual MinValue	Erhält den Minimalwert im angegebenen Bereich
virtual MaxIndex	Erhält den Index des Maximalwerts im bestimmten Bereich
virtual MaxValue	Erhält den Maximalwert im bestimmten Bereich

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Größe des Serienpuffers.

```
virtual bool  BufferResize(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

GetData

Erhält das angegebene Element des Serienpuffers.

```
double  GetData(
   const int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Serienpuffers oder EMPTY_VALUE.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Seriendaten.

```
virtual void  Refresh(
   const int  flags=OBJ_ALL_PERIODS      // Flags
   )
```

Parameter

flags=OBJ_ALL_PERIODS

[in] Flags von Timeframes.

© 2000-2025, MetaQuotes Ltd.

MinIndex

Erhält den Index des Minimalwerts im bestimmten Bereich.

```
virtual int  MinIndex(
   const int  start,     // Größe
   constint  count       // Anzahl
   ) const
```

Parameter

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

Rückgabewert

Der Index des minimalen Werts des Serienpuffers im angegebenen Bereich oder -1.

© 2000-2025, MetaQuotes Ltd.

MinValue

Erhält den Minimalwert im bestimmten Bereich.

```
virtual double  MinValue(
   const int    start,      // Größe
   const int    count,      // Anzahl
   int&         index       // Referenz
   ) const
```

Parameter

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

index

[out] Die Referenz an die Variable um den Wert des gefundenen Elements zu platzieren.

Rückgabewert

Der Minimalwert des Serienpuffers im bestimmten Bereich oder EMPTY_VALUE.

Hinweis

Der Index des gefundenen Elements wird an der Referenz index platziert.

© 2000-2025, MetaQuotes Ltd.

MaxIndex

Erhält den Index des Maximalwerts im bestimmten Bereich.

```
virtual int  MaxIndex(
   const int  start,      // Index
   const int  count       // Anzahl
   ) const
```

Parameter

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

Rückgabewert

Der Index des maximalen Werts des Serienpuffers im angegebenen Bereich oder -1.

© 2000-2025, MetaQuotes Ltd.

MaxValue

Erhält den Maximalwert im bestimmten Bereich.

```
virtual double  MaxValue(
   const int    start,      // Größe
   const int    count,      // Anzahl
   int&         index       // Referenz
   ) const
```

Parameter

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

index

[out] Die Referenz an die Variable um den Wert des gefundenen Elements zu platzieren.

Rückgabewert

Der Maximalwert des Serienpuffers im bestimmten Bereich oder EMPTY_VALUE.

Hinweis

Der Index des gefundenen Elements wird an der Referenz index platziert.

© 2000-2025, MetaQuotes Ltd.

CIndicator Klasse

Die CIndicator Klasse ist eine Basisklasse für Klassen technischer Indikatoren der MQL5-Standardbibliothek.

Beschreibung

Die CIndicator Klasse gewährleistet allen Nachfolgern einen vereinfachten Zugang zu allgemeinen Funktionen API MQL5 hinsichtlich der Arbeit mit technischen Indikatoren.

Deklaration

```
class CIndicator: public CSeries
```

Titel

```
#include <Indicators\Indicator.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator

Direkte Ableitungen

CiAC, CiAD, CiADX, CiADXWilder, CiAlligator, CiAMA, CiAO, CiATR, CiBands, CiBearsPower, CiBullsPower, CiBWMFI, CiCCI, CiChaikin, CiCustom, CiDEMA, CiDeMarker, CiEnvelopes, CiForce, CiFractals, CiFrAMA, CiGator, CiIchimoku, CiMA, CiMACD, CiMFI, CiMomentum, CiOBV, CiOsMA, CiRSI, CiRVI, CiSAR, CiStdDev, CiStochastic, CiTEMA, CiTriX, CiVIDyA, CiVolumes, CiWPR

Methoden der Klasse nach Gruppen

Attribute	
Handle	Erhält den Handle eines Indikators
Status	Erhält den Erstellungsstatus eines Indikators
FullRelease	Setzt das Flag der Befreiung eines Handles
Erstellen	
Create	Erstellt einen Indikator
BufferResize	Setzt die Puffer-Größe für einen Indikator
Datenzugriff	
GetData	Kopiert Daten aus dem Puffer eines Indikators
Datenaktualisierung	

© 2000-2025, MetaQuotes Ltd.

Attribute	
Refresh	Aktualisiert Daten eines Indikators
Extrema finden	
Minimum	Erhält den Index des kleinsten Wertes im angegebenen Bereich
MinValue	Erhält den kleinsten Wert im angegebenen Bereich
Maximum	Erhält den Index des höchsten Wertes im angegebenen Bereich
MaxValue	Erhält den höchsten Wert im angegebenen Bereich
Umwandlung von Aufzählungen	
MethodDescription	Wandelt ENUM_MA_METHOD in eine Zeile um
PriceDescription	Wandelt ENUM_APPLIED_PRICE in eine Zeile um
VolumeDescription	Wandelt ENUM_APPLIED_VOLUME in eine Zeile um
Mit Chart arbeiten	
AddToChart	Fügt einen Indikator
DeleteFromChart	Löscht den Indikator vom Chart

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Handle

Erhält den Handle eines Indikators.

```
int  Handle() const
```

Rückgabewert

Der Handle eines Indikators.

© 2000-2025, MetaQuotes Ltd.

Status

Erhält den Erstellungsstatus eines Indikators.

```
string  Status() const
```

Rückgabewert

Der Erstellungsstatus eines Indikators.

© 2000-2025, MetaQuotes Ltd.

FullRelease

Setzt das Flag der Befreiung eines Handles.

```
void  FullRelease(
   const bool  flag=true      // Flag
   )
```

Parameter

flag

[in] Der neue Wert des Flags von Handle-Befreiung.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator des angegebenen Typs mit angegebenen Parameter.

```
bool  Create(
   const string           symbol,          // Symbolname
   const ENUM_TIMEFRAMES  period,          // Periode
   const ENUM_INDICATOR   type,            // Typ
   const int              num_params,      // Anzahl der Parameter
   const MqlParam&        params[]         // Array von Parameter
   )
```

Parameter

symbol

[in] Der Symbolname.

period

[in] Timeframe des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

type

[in] Indikatortyp (ein Wert aus der Enumeration ENUM_INDICATOR).

num_params

[in] Anzahl der Parameter für die Erstellung des Indikators.

params

[in] Referenz auf ein Array von Parametern für die Erstellung des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Puffer-Größe für einen Indikator.

```
virtual bool  BufferResize(
   const int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffer-Größe.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Alle Puffer eines Indikators haben die gleiche Größe.

© 2000-2025, MetaQuotes Ltd.

BarsCalculated

Gibt die Anzahl der berechneten Daten für den Indikator zurück.

```
int  BarsCalculated() const;
```

Rückgabewert

Gibt die Anzahl der berechneten Daten im Indikator-Puffer oder -1 beim Fehler (Daten noch nicht berechnet) zurück.

© 2000-2025, MetaQuotes Ltd.

GetData

Erhält das angegebene Element des angegebenen Indikator-Puffers. Um mit aktuellen Daten zu arbeiten, muss Refresh() vor der Verwendung aufgerufen werden.

```
double  GetData(
   const int  buffer_num,     // Puffernummer
   const int  index           // Index
   ) const
```

Parameter

buffer_num

[in] Nummer des Indikator-Puffers.

index

[in] Der Index des Elements vom Indikator-Puffer.

Rückgabewert

Gibt beim Erfolg den Wert zurück oder EMPTY_VALUE wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Indikator-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   const int      start_pos,      // Position
   const int      count,          // Anzahl
   const int      buffer_num,     // Puffernummer
   double&        buffer[]        // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Indikator-Puffer.

count

[in] Die Anzahl der Elemente des Indikator-Puffers.

buffer_num

[in] Nummer des Indikator-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

Gibt Zahl der aus dem Puffer erhalten Indikatorwerte beim Erfolg zurück, ansonsten -1.

GetData

Empfängt Daten aus dem Indikator-Puffer bei dem Anfangszeit und die Anzahl.

© 2000-2025, MetaQuotes Ltd.

```
int  GetData(
   const datetime  start_time,      // Anfangszeit
   const int       count,           // Anzahl
   const int       buffer_num,      // Puffernummer
   double&         buffer[]         // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Indikator-Puffer.

count

[in] Die Anzahl der Elemente des Indikator-Puffers.

buffer_num

[in] Nummer des Indikator-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

Gibt Zahl der aus dem Puffer erhalten Indikatorwerte beim Erfolg zurück, ansonsten -1.

GetData

Empfängt Daten aus dem Indikator-Puffer bei dem Anfangszeit und Endzeit.

```
int  GetData(
   const datetime  start_time,      // Anfangszeit
   const datetime  stop_time,       // Endzeit
   const int       buffer_num,      // Puffernummer
   double&         buffer[]         // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Indikator-Puffer.

stop_time

[in] Der Zeit des Endelements vom Indikator-Puffer.

buffer_num

[in] Nummer des Indikator-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

Gibt Zahl der aus dem Puffer erhalten Indikatorwerte beim Erfolg zurück, ansonsten -1.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Daten eines Indikators. Es wird empfohlen, Refresh() vor der Verwendung von GetData() aufzurufen.

```
virtual void  Refresh(
   const int  flags=OBJ_ALL_PERIODS      // Flags
   )
```

Parameter

flags=OBJ_ALL_PERIODS

[in] Flags von Timeframe-Aktualisierung

© 2000-2025, MetaQuotes Ltd.

Minimum

Erhält den Index des kleinsten Wertes des angegebenen Puffers im angegebenen Bereich.

```
int  Minimum(
   const int  buffer_num,     // Puffernummer
   const int  start,          // Startindex
   const int  count           // Anzahl
   ) const
```

Parameter

buffer_num

[in] Nummer des Puffers um nach dem Wert zu suchen.

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

Rückgabewert

Den Index des minimalen Elements des angegebenen Puffers im angegebenen Bereich.

© 2000-2025, MetaQuotes Ltd.

MinValue

Erhält den kleinsten Wert des angegebenen Puffers im angegebenen Bereich.

```
double  MinValue(
   const int   buffer_num,     // Puffernummer
   const int   start,          // Startindex
   const int   count,          // Anzahl
   int&        index           // Referenz
   ) const
```

Parameter

buffer_num

[in] Nummer des Puffers um nach dem Wert zu suchen.

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

index

[out] Die Referenz an die Variable um den Wert des gefundenen Elements zu platzieren.

Rückgabewert

Der Wert des minimalen Elements des angegebenen Puffers im angegebenen Bereich.

Hinweis

Der Index des gefundenen Elements wird an der Referenz index platziert.

© 2000-2025, MetaQuotes Ltd.

Maximum

Erhält den Index des höchsten Wertes des angegebenen Puffers im angegebenen Bereich.

```
int  Maximum(
   const int  buffer_num,     // Puffernummer
   const int  start,          // Startindex
   const int  count           // Anzahl
   ) const
```

Parameter

buffer_num

[in] Nummer des Puffers um nach dem Wert zu suchen.

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

Rückgabewert

Der Index des maximalen Elements des angegebenen Puffers im angegebenen Bereich.

© 2000-2025, MetaQuotes Ltd.

MaxValue

Erhält den höchsten Wert des angegebenen Puffers im angegebenen Bereich.

```
double  MaxValue(
   const int    buffer_num,      // Puffernummer
   const int    start,           // Startindex
   const int    count,           // Anzahl
   int&         index            // Referenz
   ) const
```

Parameter

buffer_num

[in] Nummer des Puffers um nach dem Wert zu suchen.

start

[in] Startindex des Suchbereichs.

count

[in] Die Größe des Suchbereichs (Anzahl der Elementen).

index

[out] Die Referenz an die Variable um den Wert des gefundenen Elements zu platzieren.

Rückgabewert

Der Wert des maximalen Elements des angegebenen Puffers im angegebenen Bereich.

Hinweis

Der Index des gefundenen Elements wird an der Referenz index platziert.

© 2000-2025, MetaQuotes Ltd.

MethodDescription

Wandelt ENUM_MA_METHOD in eine Zeile um.

```
string  MethodDescription(
   const int  val      // Wert
   ) const
```

Parameter

val

[in] Der Wert für Umwandlung.

Rückgabewert

Eine Zeile, die dem Wert der Enumeration ENUM_MA_METHOD entspricht.

© 2000-2025, MetaQuotes Ltd.

PriceDescription

Wandelt den Wert der Enumeration ENUM_APPLIED_PRICE in eine Zeile um.

```
string  PriceDescription(
   const int  val      // Wert
   ) const
```

Parameter

val

[in] Der Wert für Umwandlung.

Rückgabewert

Eine Zeile, die dem Wert der Enumeration ENUM_APPLIED_PRICE entspricht.

© 2000-2025, MetaQuotes Ltd.

VolumeDescription

Wandelt den Wert der Enumeration ENUM_APPLIED_VOLUME in eine Zeile um.

```
string  VolumeDescription(
   const int  val      // Wert
   ) const
```

Parameter

val

[in] Der Wert für Konvertierung.

Rückgabewert

Ein String, der dem Wert der Enumeration ENUM_APPLIED_VOLUME entspricht.

© 2000-2025, MetaQuotes Ltd.

AddToChart

Fügt einen Indikator auf den Chart hinzu.

```
bool  AddToChart(
   const long  chart,      // Chart ID
   const int  subwin       // Index des Untenfensters
   )
```

Parameter

chart

[in] ID des Charts.

subwin

[in] Index des Unterfensters.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Indikator auf den Chart nicht hinzugefügt werden konnte.

© 2000-2025, MetaQuotes Ltd.

DeleteFromChart

Löscht einen Indikator vom Chart.

```
bool  DeleteFromChart(
   const long  chart,      // Chart ID
   const int  subwin       // Index des Untenfensters
   )
```

Parameter

chart

[in] ID des Charts.

subwin

[in] Index des Unterfensters.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Indikator aus dem Chart nicht gelöscht werden konnte.

© 2000-2025, MetaQuotes Ltd.

Klasse CIndicators

CIndicators ist eine Klasse für das Sammeln von Instanzen der Klassen von Zeitreihen und technischen Indikatoren.

Beschreibung

Klasse CIndicators bietet Etstellung von Instanzen der Klassen von technischen Indikatoren, ihre Speicherung und Verwaltung (Synchronisation, Speicherverwaltung und Handles).

Deklaration

```
class CIndicators: public CArrayObj
```

Kopf

```
#include <Indicators\Indicators.mqh>
```

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt den Indikator
Datenaktualisierung	
Refresh	Aktualisiert Indikatordaten

© 2000-2025, MetaQuotes Ltd.

Create

It creates the indicator with the specified parameters.

```
CIndicator*  Create(
   const string           symbol,      // Symbol name
   const ENUM_TIMEFRAMES  period,      // Period
   const ENUM_INDICATOR   type,        // Indicator's type
   const int              count,       // Number of parameters
   const MqlParam&        params       // Parameters array reference
   )
```

Parameters

symbol

[in] Symbol name.

period

[in] Period (ENUM_TIMEFRAMES enumeration).

type

[in] Indicator's type (ENUM_INDICATOR).

count

[in] Number of parameters for the indicator.

params

[in] Reference to the parameters array for the indicator.

Returned value

If successful, it returns the reference to the created indicator, and NULL if the indicator hasn't been created.

© 2000-2025, MetaQuotes Ltd.

Refresh

Updates data for all technical indicators in the collection.

```
int  Refresh()
```

Returned value

It returns the updated timeframe flags (formed as an object visibility flags).

© 2000-2025, MetaQuotes Ltd.

Klassen für Arbeit mit Zeitreihen

Dieser Abschnitt enthält die Beschreibungen der Klassen für die Arbeit mit Zeitreihen und auch die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klasse	Beschreibung
CiSpread	Klasse des Zugriffs auf Spread-Geschichtsdaten
CiTime	Klasse des Zugriffs auf Balkeneröffnungszeit
CiTickVolume	Klasse des Zugriffs auf Tick-Volumen von Balken
CiRealVolume	Klasse des Zugriffs auf realen Volumen von Balken
CiOpen	Klasse des Zugriffs auf Balkeneröffnungspreis
CiHigh	Klasse des Zugriffs auf Hoch-Preise von Balken
CiLow	Klasse des Zugriffs auf Tief-Preise von Balken
CiClose	Klasse des Zugriffs auf Balkenschlusspreis

© 2000-2025, MetaQuotes Ltd.

Klasse CiSpread

CiSpread ist eine Klasse für den Zugriff auf die Zeitreihen von Spread-Geschichte.

Beschreibung

Klasse CiSpread bietet den Zugriff auf die Zeitreihen von Spread-Geschichte.

Deklaration

```
class CiSpread: public CSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CiSpread

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
BufferResize	Setzt die Größe des Serienpuffers
Datenzugriff	
GetData	Erhält Seriendaten
Datenaktualisierung	
Refresh	Aktualisiert Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf die Spread-Geschichte.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Größe des Serienpuffers.

```
virtual void  BufferResize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

GetData

Erhält das angegebene Element des Serienpuffers.

```
int  GetData(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Serienpuffers oder 0.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int   start_pos,     // Position
   int   count,         // Anzahl
   int&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   int&      buffer          // Array
   ) const
```

Parameter

start_time

© 2000-2025, MetaQuotes Ltd.

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   datetime  stop_time,      // Endzeit
   int&      buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Seriendaten.

```
virtual void  Refresh(
   int  flags      // Flags
   )
```

Parameter

flags

[in] Flags von Timeframes.

© 2000-2025, MetaQuotes Ltd.

Klasse CiTime

CTime ist eine Klasse für den Zugriff auf die Reihen von Balkeneröffnungszeit.

Beschreibung

Klasse CTime bietet den Zugriff auf die Reihen von Balkeneröffnungszeit.

Deklaration

```
class CiTime: public CSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CiTime

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
BufferResize	Setzt die Größe des Serienpuffers
Datenzugriff	
GetData	Erhält Seriendaten
Datenaktualisierung	
Refresh	Aktualisiert Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf Balkeneröffnungszeit.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Größe des Serienpuffers.

```
virtual void  BufferResize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

GetData

Erhält das angegebene Element des Serienpuffers.

```
datetime  GetData(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Serienpuffers oder 0.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int    start_pos,     // Position
   int    count,         // Anzahl
   long&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   long&     buffer          // Array
   ) const
```

Parameter

start_time

© 2000-2025, MetaQuotes Ltd.

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   datetime  stop_time,      // Endzeit
   long&     buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Seriendaten.

```
virtual void  Refresh(
   int  flags      // Flags
   )
```

Parameter

flags

[in] Flags von Timeframes.

© 2000-2025, MetaQuotes Ltd.

Klasse CiTickVolume

CiTickVolume ist eine Klasse für den Zugriff auf die Zeitreihen von Tick-Volumen der Balken.

Beschreibung

Klasse CiTickVolume bietet den Zugriff auf die Zeitreihen von Tick-Volumen der Balken.

Deklaration

```
class CiTickVolume: public CSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CiTickVolume

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
BufferResize	Setzt die Größe des Serienpuffers
Datenzugriff	
GetData	Erhält Seriendaten
Datenaktualisierung	
Refresh	Aktualisiert Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf die Tick-Volumen der Balken.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Größe des Serienpuffers.

```
virtual void  BufferResize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

GetData

Erhält das angegebene Element des Serienpuffers.

```
long  GetData(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Serienpuffers oder 0.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int    start_pos,     // Position
   int    count,         // Anzahl
   long&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   long&     buffer          // Array
   ) const
```

Parameter

start_time

© 2000-2025, MetaQuotes Ltd.

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   datetime  stop_time,      // Endzeit
   long&     buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Seriendaten.

```
virtual void  Refresh(
   int  flags      // Flags
   )
```

Parameter

flags

[in] Flags von Timeframes.

© 2000-2025, MetaQuotes Ltd.

Klasse CiRealVolume

CiRealVolume ist eine Klasse für den Zugriff auf die Zeitreihen von realen Volumen der Balken.

Beschreibung

Klasse CiRealVolume bietet den Zugriff auf die Zeitreihen von realen Volumen der Balken.

Deklaration

```
class CiRealVolume: public CSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CiRealVolume

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
BufferResize	Setzt die Größe des Serienpuffers
Datenzugriff	
GetData	Erhält Seriendaten
Datenaktualisierung	
Refresh	Aktualisiert Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf die realen Volumen der Balken.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

BufferResize

Setzt die Größe des Serienpuffers.

```
virtual void  BufferResize(
   int  size      // Größe
   )
```

Parameter

size

[in] Die neue Puffergröße.

© 2000-2025, MetaQuotes Ltd.

GetData

Erhält das angegebene Element des Serienpuffers.

```
long  GetData(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Ein Element des Serienpuffers oder 0.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int    start_pos,     // Position
   int    count,         // Anzahl
   long&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   long&     buffer          // Array
   ) const
```

Parameter

start_time

© 2000-2025, MetaQuotes Ltd.

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   datetime  stop_time,      // Endzeit
   long&     buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Seriendaten.

```
virtual void  Refresh(
   int  flags      // Flags
   )
```

Parameter

flags

[in] Flags von Timeframes.

© 2000-2025, MetaQuotes Ltd.

Klasse CiOpen

CiOpen ist eine Klasse für den Zugriff auf die Reihen von Balkeneröffnungspreisen.

Beschreibung

Klasse CiOpen bietet den Zugriff auf die Reihen von Balkeneröffnungspreisen.

Deklaration

```
class CiOpen: public CPriceSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CPriceSeries
CiOpen

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
Datenzugriff	
GetData	Erhält Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CPriceSeries

BufferResize, MinIndex, MinValue, MaxIndex, MaxValue, GetData, Refresh

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf Balkeneröffnungspreise.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int      start_pos,     // Position
   int      count,         // Anzahl
   double&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   double&   buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

© 2000-2025, MetaQuotes Ltd.

```
int  GetData(
   datetime  start_time,      // Anfangszeit
   datetime  stop_time,       // Endzeit
   double&   buffer           // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Klasse CiHigh

CiHigh ist eine Klasse für den Zugriff auf die Reihen von Hochpreisen der Balken.

Beschreibung

Klasse bietet den Zugriff auf die Zeitreihen von Hochpreisen der Balken.

Deklaration

```
class CiHigh: public CPriceSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CPriceSeries
          CiHigh

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
Datenzugriff	
GetData	Erhält Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CPriceSeries

BufferResize, MinIndex, MinValue, MaxIndex, MaxValue, GetData, Refresh

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf Hochpreise der Balken.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int      start_pos,     // Position
   int      count,         // Anzahl
   double&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   double&   buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

© 2000-2025, MetaQuotes Ltd.

```
int  GetData(
   datetime  start_time,      // Anfangszeit
   datetime  stop_time,       // Endzeit
   double&   buffer           // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Klasse CiLow

CiLow ist eine Klasse für den Zugriff auf die Reihen von Tief-Preisen der Balken.

Beschreibung

Klasse CiLow bietet den Zugriff auf die Reihen von Tief-Preisen der Balken.

Deklaration

```
class CiLow: public CPriceSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CPriceSeries
CiLow

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
Datenzugriff	
GetData	Erhält Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CPriceSeries

BufferResize, MinIndex, MinValue, MaxIndex, MaxValue, GetData, Refresh

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf Tief-Preise der Balken.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int      start_pos,     // Position
   int      count,         // Anzahl
   double&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   double&   buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

© 2000-2025, MetaQuotes Ltd.

```
int  GetData(
   datetime  start_time,      // Anfangszeit
   datetime  stop_time,       // Endzeit
   double&   buffer           // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Klasse CiClose

CiClose ist eine Klasse für den Zugriff auf die Reihen von Balkenschlusspreisen.

Beschreibung

Klasse CiClose bietet den Zugriff auf die Reihen von Balkenschlusspreisen.

Deklaration

```
class CiClose: public CPriceSeries
```

Kopf

```
#include <Indicators\TimeSeries.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CPriceSeries
          CiClose

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt eine Zeitreihe
Datenzugriff	
GetData	Erhält Seriendaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Type, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CPriceSeries

BufferResize, MinIndex, MinValue, MaxIndex, MaxValue, GetData, Refresh

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt eine Zeitreihe mit angegebenen Parametern für den Zugriff auf Balkenschlusspreise.

```
bool  Create(
   string           symbol,     // Symbolname
   ENUM_TIMEFRAMES  period      // Periode
   )
```

Parameter

symbol

[in] Der Symbolname der Zeitreihe.

period

[in] Timeframe der Zeitreihe (ein Wert der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Zeitreihe nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei einer Startposition und die Anzahl.

```
int  GetData(
   int      start_pos,     // Position
   int      count,         // Anzahl
   double&  buffer         // Array
   ) const
```

Parameter

start_pos

[in] Die Startposition vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und die Anzahl.

```
int  GetData(
   datetime  start_time,     // Anfangszeit
   int       count,          // Anzahl
   double&   buffer          // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

count

[in] Die Anzahl der Elemente des Zeitreihe-Puffers.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

GetData

Empfängt Daten aus dem Zeitreihe-Puffer bei dem Anfangszeit und Endzeit.

© 2000-2025, MetaQuotes Ltd.

```
int  GetData(
   datetime  start_time,      // Anfangszeit
   datetime  stop_time,       // Endzeit
   double&   buffer           // Array
   ) const
```

Parameter

start_time

[in] Der Zeit des Anfangselements vom Zeitreihe-Puffer.

stop_time

[in] Der Zeit des Endelements vom Zeitreihe-Puffer.

buffer

[in] Die Referenz an ein Array um die Daten zu platzieren.

Rückgabewert

>=0 - falls erfolgreich, -1 - wenn Daten nicht erhalten werden konnten.

© 2000-2025, MetaQuotes Ltd.

Eine Gruppe von Basis- und Hilfsklassen von technischen Indikatoren und Zeitreihen

Dieser Abschnitt enthält die Details der Arbeit mit der Gruppe von Basis- und Hilfsklassen von technischen Indikatoren und Zeitreihen, und auch die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klasse/Gruppe	Beschreibung
CiADX	Indikator "Average Directional Index"
CiADXWilder	Indikator "Average Directional Index by Welles Wilder"
CiBands	Indikator "Bollinger Bands®"
CiEnvelopes	Indikator "Envelopes"
CiIchimoku	Indikator "Ichimoku Kinko Hyo"
CiMA	Indikator "Moving Average"
CiSAR	Indikator "Parabolic Stop And Reverse System"
CiStdDev	Indikator "Standard Deviation"
CiDEMA	Indikator "Double Exponential Moving Average"
CiTEMA	Indikator "Triple Exponential Moving Average"
CiFrAMA	Indikator "Fractal Adaptive Moving Average"
CiAMA	Indikator "Adaptive Moving Average"
CiVIDyA	Indikator "Variable Index Dynamic Average"

© 2000-2025, MetaQuotes Ltd.

Klasse CiADX

CiADX ist eine Klasse für Arbeit mit dem technischen Indikator "Average Directional Index".

Beschreibung

Klasse CiADX bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Average Directional Index".

Deklaration

```
class CiADX: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiADX

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten der Hauptlinie
Plus	Erhält Pufferdaten der Linie +DI
Minus	Erhält Pufferdaten der Linie -DI
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers der Hauptlinie am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Hauptlinie-Puffer.

Rückgabewert

Pufferelement der Hauptlinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Plus

Erhält das Element des Puffers der Linie +DI am angegebenen Index.

```
double  Plus(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie +DI.

Rückgabewert

Pufferelement der Linie +DI am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Minus

Erhält das Element des Puffers der Linie -DI am angegebenen Index.

```
double  Minus(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie -DI.

Rückgabewert

Pufferelement der Linie -DI am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für - IND_ADX).

© 2000-2025, MetaQuotes Ltd.

Klasse CiADXWilder

CiADXWilder ist eine Klasse für Arbeit mit dem technischen Indikator "Average Directional Index by Welles Wilder".

Beschreibung

Klasse CiADXWilder bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Average Directional Index by Welles Wilder".

Deklaration

```
class CiADXWilder: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiADXWilder

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten der Hauptlinie
Plus	Erhält Pufferdaten der Linie +DI
Minus	Erhält Pufferdaten der Linie -DI
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers der Hauptlinie am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Hauptlinie-Puffer.

Rückgabewert

Pufferelement der Hauptlinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Plus

Erhält das Element des Puffers der Linie +DI am angegebenen Index.

```
double  Plus(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie +DI.

Rückgabewert

Pufferelement der Linie +DI am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Minus

Erhält das Element des Puffers der Linie -DI am angegebenen Index.

```
double  Minus(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie -DI.

Rückgabewert

Pufferelement der Linie -DI am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiADXWilder - IND_ADXW).

© 2000-2025, MetaQuotes Ltd.

Klasse CiBands

CiBands ist eine Klasse für Arbeit mit dem technischen Indikator "Bollinger Bands®".

Beschreibung

Klasse CiBands bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Bollinger Bands".

Deklaration

```
class CiBands: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiBands

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
MaShift	Erhält den Versatz entlang der Preisachse
Deviation	Erhält den Wert der Abweichung
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Base	Erhält Pufferdaten der Hauptlinie
Upper	Erhält Pufferdaten der oberen Linie
Lower	Erhält Pufferdaten der unteren Linie
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaShift

Erhält den Versatz entlang der Preisachse.

```
int  MaShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Deviation

Erhält den Wert der Abweichung.

```
double  Deviation() const
```

Rückgabewert

Der Wert der Abweichung, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period,     // Mittelwertzeitraum
   int              ma_shift,      // Versatz
   double           deviation,     // Abweichung
   int              applied        // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_shift

[in] Indikator-Versatz entlang der Preisachse.

deviation

[in] Abweichung des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Base

Erhält das Element des Puffers der Hauptlinie am angegebenen Index

```
double  Base(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Hauptlinie-Puffer.

Rückgabewert

Pufferelement der Hauptlinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Upper

Erhält das Element des Puffers der oberen Linie am angegebenen Index.

```
double  Upper(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Puffer der oberen Linie.

Rückgabewert

Pufferelement der oberen Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Lower

Erhält das Element des Puffers der unteren Linie am angegebenen Index

```
double  Lower(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Puffer der unteren Linie.

Rückgabewert

Pufferelement der unteren Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiBands - IND_BANDS).

© 2000-2025, MetaQuotes Ltd.

Klasse CiEnvelopes

CiEnvelopes ist eine Klasse für Arbeit mit dem technischen Indikator "Envelopes".

Beschreibung

Klasse CiEnvelopes bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Envelopes".

Deklaration

```
class CiEnvelopes: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiEnvelopes

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
MaShift	Erhält den Versatz entlang der Preisachse
MaMethod	Erhält den Mittelungsverfahren
Deviation	Erhält den Wert der Abweichung
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Upper	Erhält Pufferdaten der oberen Linie
Lower	Erhält Pufferdaten der unteren Linie
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaShift

Erhält den Versatz entlang der Preisachse.

```
int  MaShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_MA_METHOD).

© 2000-2025, MetaQuotes Ltd.

Deviation

Erhält den Wert der Abweichung.

```
double  Deviation() const
```

Rückgabewert

Der Wert der Abweichung, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string            symbol,         // Symbol
   ENUM_TIMEFRAMES   period,         // Periode
   int               ma_period,      // Mittelwertzeitraum
   int               ma_shift,       // Versatz
   ENUM_MA_METHOD    ma_method,      // Mittelungsverfahren
   int               applied,        // Preistyp, Handle
   double            deviation       // Abweichung
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_shift

[in] Indikator-Versatz entlang der Preisachse.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

deviation

[in] Abweichung des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Upper

Erhält das Element des Puffers der oberen Linie am angegebenen Index.

```
double  Upper(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Puffer der oberen Linie.

Rückgabewert

Pufferelement der oberen Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Lower

Erhält das Element des Puffers der unteren Linie am angegebenen Index

```
double  Lower(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements vom Puffer der unteren Linie.

Rückgabewert

Pufferelement der unteren Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiEnvelopes - IND_ENVELOPES).

© 2000-2025, MetaQuotes Ltd.

Klasse CiIchimoku

CiIchimoku ist eine Klasse für Arbeit mit dem technischen Indikator "Ichimoku Kinko Hyo".

Beschreibung

Klasse CiIchimoku bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Ichimoku Kinko Hyo".

Deklaration

```
class CiIchimoku: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiIchimoku

Gruppen der Klassenmethode

Attribute	
TenkanSenPeriod	Erhält die Periode TenkanSen
KijunSenPeriod	Erhält die Periode KijunSen
SenkouSpanBPeriod	Erhält die Periode SenkouSpanB
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
TenkanSen	Erhält Pufferdaten der Linie TenkanSen
KijunSen	Erhält Pufferdaten der Linie KijunSen
SenkouSpanA	Erhält Pufferdaten der Linie SenkouSpanA
SenkouSpanB	Erhält Pufferdaten der Linie SenkouSpanB
ChinkouSpan	Erhält Pufferdaten der Linie ChikouSpan
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

TenkanSenPeriod

Erhält die Periode TenkanSen.

```
int  TenkanSenPeriod() const
```

Rückgabewert

Periode TenkanSen, die während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

KijunSenPeriod

Erhält die Periode KijunSen.

```
int  KijunSenPeriod() const
```

Rückgabewert

Periode KijunSen, die während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SenkouSpanBPeriod

Erhält die Periode SenkouSpanB.

```
int  SenkouSpanBPeriod() const
```

Rückgabewert

Periode SenkouSpanB, die während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,            // Symbol
   ENUM_TIMEFRAMES  period,            // Periode
   int              tenkan_sen,        // Periode von TenkanSen
   int              kijun_sen,         // Periode von KijunSen
   int              senkou_span_b      // Periode von SenkouSpanB
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

tenkan_sen

[in] Periode von TenkanSen des Indikators.

kijun_sen

[in] Periode von KijunSen des Indikators.

senkou_span_b

[in] Periode von SenkouSpanB des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

TenkanSen

Erhält das Element des Puffers der Linie TenkanSen am angegebenen Index.

```
double  TenkanSen(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie TenkanSen.

Rückgabewert

Pufferelement der Linie TenkanSen am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

KijunSen

Erhält das Element des Puffers der Linie KijunSen am angegebenen Index.

```
double  KijunSen(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie KijunSen.

Rückgabewert

Pufferelement der Linie KijunSen am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

SenkouSpanA

Erhält das Element des Puffers der Linie SenkouSpanA am angegebenen Index.

```
double  SenkouSpanA(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie SenkouSpanA.

Rückgabewert

Pufferelement der Linie SenkouSpanA am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

SenkouSpanB

Erhält das Element des Puffers der Linie SenkouSpanB am angegebenen Index.

```
double  SenkouSpanB(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie SenkouSpanB .

Rückgabewert

Pufferelement der Linie SenkouSpanB am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

ChinkouSpan

Erhält das Element des Puffers der Linie ChikouSpan am angegebenen Index.

```
double  ChinkouSpan(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der Linie ChikouSpan.

Rückgabewert

Pufferelement der Linie ChikouSpan am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für Cilchimoku - IND_ICHIMOKU).

© 2000-2025, MetaQuotes Ltd.

Klasse CiMA

CiMA ist eine Klasse für Arbeit mit dem technischen Indikator "Moving Average".

Beschreibung

Klasse CiAM bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Moving Average".

Deklaration

```
class CiMA: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiMA

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
MaShift	Erhält den Versatz entlang der Preisachse
MaMethod	Erhält den Mittelungsverfahren
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaShift

Erhält den Versatz entlang der Preisachse.

```
int  MaShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_MA_METHOD).

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           string,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period,     // Mittelwertzeitraum
   int              ma_shift,      // Versatz
   ENUM_MA_METHOD   ma_method,     // Mittelungsverfahren
   int              applied        // Preistyp, Handle
   )
```

Parameter

string

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_shift

[in] Indikator-Versatz entlang der Preisachse.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiMA - IND_MA).

© 2000-2025, MetaQuotes Ltd.

Klasse CiSAR

CiSAR ist eine Klasse für Arbeit mit dem technischen Indikator "Parabolic Stop And Reverse System".

Beschreibung

Klasse CiSAR bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Parabolic Stop And Reverse System".

Deklaration

```
class CiSAR: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiSAR

Gruppen der Klassenmethode

Attribute	
SarStep	Erhält den Schritt der Geschwindigkeitssteigerung
Maximum	Erhält den Faktor vom Folgen nach dem Preis
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

SarStep

Erhält den Schritt der Geschwindigkeitssteigerung.

```
double  SarStep() const
```

Rückgabewert

Schritt der Geschwindigkeitssteigerung, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Maximum

Erhält den Faktor vom Folgen nach dem Preis.

```
double  Maximum() const
```

Rückgabewert

Faktor vom Folgen nach dem Preis, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,      // Symbol
   ENUM_TIMEFRAMES  period,      // Periode
   double           step,        // Schritt
   double           maximum      // Faktor
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

step

[in] Schritt der Geschwindigkeitssteigerung.

maximum

[in] Faktor vom Folgen nach dem Preis.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiSAR - IND_SAR).

© 2000-2025, MetaQuotes Ltd.

Klasse CiStdDev

CiStdDev ist eine Klasse für Arbeit mit dem technischen Indikator "Standard Deviation".

Beschreibung

Klasse CiStdDev bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Standard Deviation".

Deklaration

```
class CiStdDev: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiStdDev

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
MaShift	Erhält den Versatz entlang der Preisachse
MaMethod	Erhält den Mittelungsverfahren
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaShift

Erhält den Versatz entlang der Preisachse.

```
int  MaShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_MA_METHOD).

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,         // Symbol
   ENUM_TIMEFRAMES  period,         // Periode
   int              ma_period,      // Mittelwertzeitraum
   int              ma_shift,       // Versatz
   ENUM_MA_METHOD   ma_method,      // Mittelungsverfahren
   int              applied         // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_shift

[in] Indikator-Versatz entlang der Preisachse.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiStdDev - IND_STDDEV).

© 2000-2025, MetaQuotes Ltd.

Klasse CiDEMA

CiDEMA ist eine Klasse für Arbeit mit dem technischen Indikator "Double Exponential Moving Average".

Beschreibung

Klasse CiDEMA bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Double Exponential Moving Average".

Deklaration

```
class CiDEMA: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiDEMA

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
IndShift	Erhält den Versatz entlang der Preisachse
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

IndShift

Erhält den Versatz entlang der Preisachse.

```
int  IndShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           string,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period,     // Mittelwertzeitraum
   int              ind_shift,     // Versatz
   int              applied        // Preistyp, Handle
   )
```

Parameter

string

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ind_shift

[in] Indikator-Versatz entlang der Preisachse.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiDEMA - IND_DEMA).

© 2000-2025, MetaQuotes Ltd.

Klasse CiTEMA

CiTEMA ist eine Klasse für Arbeit mit dem technischen Indikator "Triple Exponential Moving Average".

Beschreibung

Klasse CiTEMA bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Triple Exponential Moving Average".

Deklaration

```
class CiTEMA: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiTEMA

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
IndShift	Erhält den Versatz entlang der Preisachse
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

IndShift

Erhält den Versatz entlang der Preisachse.

```
int  IndShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,         // Symbol
   ENUM_TIMEFRAMES  period,         // Periode
   int              ma_period,      // Mittelwertzeitraum
   int              ma_shift,       // Versatz
   int              applied         // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_shift

[in] Indikator-Versatz entlang der Preisachse.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiTEMA - IND_TEMA).

© 2000-2025, MetaQuotes Ltd.

Klasse CiFrAMA

CiFrAMA ist eine Klasse für Arbeit mit dem technischen Indikator "Fractal Adaptive Moving Average".

Beschreibung

Klasse CiFrAMA bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Fractal Adaptive Moving Average".

Deklaration

```
class CiFrAMA: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiFrAMA

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
IndShift	Erhält den Versatz entlang der Preisachse
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

IndShift

Erhält den Versatz entlang der Preisachse.

```
int  IndShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,         // Symbol
   ENUM_TIMEFRAMES  period,         // Periode
   int              ma_period,      // Mittelwertzeitraum
   int              ma_shift,       // Versatz
   int              applied         // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_shift

[in] Indikator-Versatz entlang der Preisachse.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiFrAMA- IND_FRAMA).

© 2000-2025, MetaQuotes Ltd.

Klasse CiAMA

CiAMA ist eine Klasse für Arbeit mit dem technischen Indikator "Adaptive Moving Average".

Beschreibung

Klasse CiAMA bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Adaptive Moving Average".

Deklaration

```
class CiAMA: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiAMA

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
FastEmaPeriod	Erhält den Mittelungszeitraum des schnellen EMA
SlowEmaPeriod	Erhält den Mittelungszeitraum des langsamen EMA
IndShift	Erhält den Versatz entlang der Preisachse
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

FastEmaPeriod

Erhält den Mittelungszeitraum des schnellen EMA

```
int  FastEmaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des schnellen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SlowEmaPeriod

Erhält den Mittelungszeitraum des langsamen EMA

```
int  SlowEmaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des langsamen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

IndShift

Erhält den Versatz entlang der Preisachse.

```
int  IndShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           string,              // Symbol
   ENUM_TIMEFRAMES  period,              // Periode
   int              ma_period,           // Mittelwertzeitraum
   int              fast_ema_period,     // Periode des schnellen EMA
   int              slow_ema_period,     // Periode des langsamen EMA
   int              ind_shift,           // Versatz
   int              applied              // Preistyp, Handle
   )
```

Parameter

string

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

fast_ema_period

[in] Mittelwertzeitraum des schnellen EMA des Indikators.

slow_ema_period

[in] Mittelwertzeitraum des langsamen EMA des Indikators.

ind_shift

[in] Indikator-Versatz entlang der Preisachse.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiAMA - IND_AMA).

© 2000-2025, MetaQuotes Ltd.

Klasse CiVIDyA

CiVIDyA ist eine Klasse für Arbeit mit dem technischen Indikator "Variable Index Dynamic Average".

Beschreibung

Klasse CiVIDyA bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Variable Index Dynamic Average".

Deklaration

```
class CiVIDyA: public CIndicator
```

Kopf

```
#include <Indicators\Trend.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiVIDyA

Gruppen der Klassenmethode

Attribute	
CmoPeriod	Erhält den Zeitraum von Momentum
EmaPeriod	Erhält den Glättungszeitraum
IndShift	Erhält den Versatz entlang der Preisachse
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

CmoPeriod

Schließt den auf Klasseninstanz gebundenen Chart.

```
int  CmoPeriod() const
```

© 2000-2025, MetaQuotes Ltd.

EmaPeriod

Erhält den Namen des Chartsymbols.

```
int  EmaPeriod() const
```

Rückgabewert

Symbolname des Charts, der an eine Klasseninstanz gebunden ist. Wenn es keinen gebundenen Chart gibt, wird "" zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

IndShift

Erhält den Versatz entlang der Preisachse.

```
int  IndShift() const
```

Rückgabewert

Der Versatz entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,          // Symbol
   ENUM_TIMEFRAMES  period,          // Periode
   int              cmo_period,      // Periode von Momentum
   int              ema_period,      // Glättungszeitraum
   int              ind_shift,       // Versatz
   int              applied          // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

cmo_period

[in] Momentum-Periode des Indikators.

ema_period

[in] Glättungsperiode des Indikators.

ind_shift

[in] Indikator-Versatz entlang der Preisachse.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiVIDyA - IND_VIDYA).

© 2000-2025, MetaQuotes Ltd.

Klassengruppe "Oszillatoren"

Dieser Abschnitt enthält die Details der Arbeit mit der Klassengruppe "Oszillatoren" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klasse/Gruppe	Beschreibung
CiATR	Oszillator "Average True Range"
CiBearsPower	Oszillator "Bears Power"
CiBullsPower	Oszillator "Bulls Power"
CiCCI	Oszillator "Commodity Channel Index"
CiChaikin	Oszillator "Chaikin Oscillator"
CiDeMarker	Oszillator "DeMarker"
CiForce	Oszillator "Force Index"
CiMACD	Oszillator "Moving Averages Convergence-Divergence"
CiMomentum	Oszillator "Momentum"
CiOsMA	Oszillator "Moving Average of Oscillator (MACD histogram)"
CiRSI	Oszillator "Relative Strength Index"
CiRVI	Oszillator "Relative Vigor Index"
CiStochastic	Oszillator "Stochastic Oscillator"
CiWPR	Oszillator "Williams' Percent Range"
CiTriX	Oszillator "Triple Exponential Moving Averages Oscillator"

© 2000-2025, MetaQuotes Ltd.

Klasse CiATR

CiATR ist eine Klasse für Arbeit mit dem technischen Indikator "Average True Range".

Beschreibung

Klasse CiATR bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Average True Range".

Deklaration

```
class CiATR: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiATR

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiATR - IND_ATR).

© 2000-2025, MetaQuotes Ltd.

Klasse CiBearsPower

CiBearsPower ist eine Klasse für Arbeit mit dem technischen Indikator "Bears Power".

Beschreibung

Klasse CiBearsPower bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Bears Power".

Deklaration

```
class CiBearsPower: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiBearsPower

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Typ des Indikators (für CiBearsPower - IND_BEARS).

© 2000-2025, MetaQuotes Ltd.

Klasse CiBullsPower

CiBullsPower ist eine Klasse für Arbeit mit dem technischen Indikator "Bulls Power".

Beschreibung

Klasse CiBullsPower bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Bulls Power".

Deklaration

```
class CiBullsPower: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiBullsPower

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für - IND_BULLS).

© 2000-2025, MetaQuotes Ltd.

Klasse CiCCI

CiCCI ist eine Klasse für Arbeit mit dem technischen Indikator "Commodity Channel Index".

Beschreibung

Klasse CiCCI bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Commodity Channel Index".

Deklaration

```
class CiCCI: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiCCI

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,         // Symbol
   ENUM_TIMEFRAMES  period,         // Periode
   int              ma_period,      // Mittelwertzeitraum
   int              applied         // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiCCI - IND_CCI).

© 2000-2025, MetaQuotes Ltd.

Klasse CiChaikin

CiChaikin ist eine Klasse für Arbeit mit dem technischen Indikator "Chaikin Oscillator".

Beschreibung

Klasse CiChaikin bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Chaikin Oscillator".

Deklaration

```
class CiChaikin: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiChaikin

Gruppen der Klassenmethode

Attribute	
FastMaPeriod	Erhält den Mittelungszeitraum der schnellen MA
SlowMaPeriod	Erhält den Mittelungszeitraum der langsamen MA
MaMethod	Erhält den Mittelungszeitraum
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

FastMaPeriod

Erhält den Mittelungszeitraum des schnellen EMA

```
int  FastMaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des schnellen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SlowMaPeriod

Erhält den Mittelungszeitraum des langsamen EMA

```
int  SlowMaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des langsamen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_MA_METHOD).

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string               symbol,              // Symbol
   ENUM_TIMEFRAMES      period,              // Periode
   int                  fast_ma_period,      // Periode des schnellen EMA
   int                  slow_ma_period,      // Periode des langsamen EMA
   ENUM_MA_METHOD       ma_method,           // Mittelungsverfahren
   ENUM_APPLIED_VOLUME  applied              // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

fast_ma_period

[in] Mittelwertzeitraum des schnellen EMA des Indikators.

slow_ma_period

[in] Mittelwertzeitraum des langsamen EMA des Indikators.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Ein Verwendungsobjekt (Volumentyp) des Indikators (ein Wert der Enumeration ENUM_APPLIED_VOLUME)..

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiChaikin - IND_CHAIKIN).

© 2000-2025, MetaQuotes Ltd.

Klasse CiDeMarker

CiDeMarker ist eine Klasse für Arbeit mit dem technischen Indikator "DeMarker".

Beschreibung

Klasse CiDeMarker bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "DeMarker".

Deklaration

```
class CiDeMarker: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiDeMarker

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Typ des Indikators (für CiDeMarker - IND_DEMARKER).

© 2000-2025, MetaQuotes Ltd.

Klasse CiForce

CiForce ist eine Klasse für Arbeit mit dem technischen Indikator "Force Index".

Beschreibung

Klasse CiForce bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Force Index".

Deklaration

```
class CiForce: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiForce

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
MaMethod	Erhält den Mittelungsverfahren
Applied	Erhält das Verwendungsobjekt (Volumentyp)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_MA_METHOD).

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string                symbol,         // Symbol
   ENUM_TIMEFRAMES       period,         // Periode
   int                   ma_period,      // Mittelwertzeitraum
   ENUM_MA_METHOD        ma_method,      // Mittelungsverfahren
   ENUM_APPLIED_VOLUME   applied         // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Ein Verwendungsobjekt (Volumentyp) des Indikators (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiForce - IND_FORCE).

© 2000-2025, MetaQuotes Ltd.

Klasse CiMACD

CiMACD ist eine Klasse für Arbeit mit dem technischen Indikator "Moving Averages Convergence-Divergence".

Beschreibung

Klasse CiMACD bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Moving Averages Convergence-Divergence".

Deklaration

```
class CiMACD: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiMACD

Gruppen der Klassenmethode

Attribute	
FastEmaPeriod	Erhält den Mittelungszeitraum des schnellen EMA
SlowEmaPeriod	Erhält den Mittelungszeitraum des langsamen EMA
SignalPeriod	Erhält den Mittelungszeitraum der Signallinie
Applied	Erhält das Verwendungsobjekt (Volumentyp)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten der Hauptlinie
Signal	Erhält Pufferdaten der Signallinie
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

FastEmaPeriod

Erhält den Mittelungszeitraum des schnellen EMA

```
int  FastEmaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des schnellen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SlowEmaPeriod

Erhält den Mittelungszeitraum des langsamen EMA

```
int  SlowEmaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des langsamen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SignalPeriod

Erhält den Mittelungszeitraum der Signallinie.

```
int  SignalPeriod() const
```

Rückgabewert

Der Mittelungszeitraum der Signallinie, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,              // Symbol
   ENUM_TIMEFRAMES  period,              // Periode
   int              fast_ema_period,     // Periode des schnellen EMA
   int              slow_ema_period,     // Periode des langsamen EMA
   int              signal_period,       // Signalperiode
   int              applied              // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

fast_ema_period

[in] Mittelwertzeitraum des schnellen EMA des Indikators.

slow_ema_period

[in] Mittelwertzeitraum des langsamen EMA des Indikators.

signal_period

[in] Mittelwertzeitraum der Signallinie des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers der Hauptlinie am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement der Hauptlinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Signal

Erhält das Element des Puffers der Signallinie am angegebenen Index

```
double  Signal(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement der Signallinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiMACD - IND_MACD).

© 2000-2025, MetaQuotes Ltd.

Klasse CiMomentum

CiMomentum ist eine Klasse für Arbeit mit dem technischen Indikator "Momentum".

Beschreibung

Klasse CiMomentum bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Momentum".

Deklaration

```
class CiMomentum: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiMomentum

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Applied	Erhält das Verwendungsobjekt (Volumentyp)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,         // Symbol
   ENUM_TIMEFRAMES  period,         // Periode
   int              ma_period,      // Mittelwertzeitraum
   int              applied         // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiMomentum - IND_MOMENTUM).

© 2000-2025, MetaQuotes Ltd.

Klasse CiOsMA

CiOsMA ist eine Klasse für Arbeit mit dem technischen Indikator "Moving Average of Oscillator (MACD histogram)".

Beschreibung

Klasse CiOsMA bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Moving Average of Oscillator (MACD histogram)".

Deklaration

```
class CiOsMA: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiOsMA

Gruppen der Klassenmethode

Attribute	
FastEmaPeriod	Erhält den Mittelungszeitraum des schnellen EMA
SlowEmaPeriod	Erhält den Mittelungszeitraum des langsamen EMA
SignalPeriod	Erhält den Mittelungszeitraum der Signallinie
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

FastEmaPeriod

Erhält den Mittelungszeitraum des schnellen EMA

```
int  FastEmaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des schnellen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SlowEmaPeriod

Erhält den Mittelungszeitraum des langsamen EMA

```
int  SlowEmaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum des langsamen EMA, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

SignalPeriod

Erhält den Mittelungszeitraum der Signallinie.

```
int  SignalPeriod() const
```

Rückgabewert

Der Mittelungszeitraum der Signallinie, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,              // Symbol
   ENUM_TIMEFRAMES  period,              // Periode
   int              fast_ema_period,     // Periode des schnellen EMA
   int              slow_ema_period,     // Periode des langsamen EMA
   int              signal_period,       // Signalperiode
   int              applied              // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

fast_ema_period

[in] Mittelwertzeitraum des schnellen EMA des Indikators.

slow_ema_period

[in] Mittelwertzeitraum des langsamen EMA des Indikators.

signal_period

[in] Mittelwertzeitraum der Signallinie des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiOsMA - IND_OSMA).

© 2000-2025, MetaQuotes Ltd.

Klasse CiRSI

CiRSI ist eine Klasse für Arbeit mit dem technischen Indikator "Relative Strength Index".

Beschreibung

Klasse CiRSI bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Relative Strength Index".

Deklaration

```
class CiRSI: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiRSI

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period,     // Mittelwertzeitraum
   int              applied        // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiRSI - IND_RSI).

© 2000-2025, MetaQuotes Ltd.

Klasse CiRVI

CiRVI ist eine Klasse für Arbeit mit dem technischen Indikator "Relative Vigor Index".

Beschreibung

Klasse CiRVI bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Relative Vigor Index".

Deklaration

```
class CiRVI: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiRVI

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten der Hauptlinie
Signal	Erhält Pufferdaten der Signallinie
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,        // Symbol
   ENUM_TIMEFRAMES  period,        // Periode
   int              ma_period      // Mittelwertzeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers der Hauptlinie am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement der Hauptlinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Signal

Erhält das Element des Puffers der Signallinie am angegebenen Index

```
double  Signal(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement der Signallinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiRVI - IND_RVI).

© 2000-2025, MetaQuotes Ltd.

Klasse CiStochastic

CiStochastic ist eine Klasse für Arbeit mit dem technischen Indikator "Stochastic Oscillator".

Beschreibung

Klasse CiStochastic bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Stochastic Oscillator".

Deklaration

```
class CiStochastic: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiStochastic

Gruppen der Klassenmethode

Attribute	
Kperiod	Erhält den Mittelungszeitraum %K
Dperiod	Erhält den Mittelungszeitraum %D
Slowing	Ruft die Zeitraum der Verlangsamung
MaMethod	Erhält den Mittelungsverfahren
PriceField	Erhält das Verwendungsobjekt (Tief/Hoch oder Schluss/Schluss)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten der Hauptlinie
Signal	Erhält Pufferdaten der Signallinie
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Kperiod

Erhält den Mittelungszeitraum %K.

```
int  Kperiod() const
```

Rückgabewert

Der Mittelungszeitraum %K, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Dperiod

Erhält den Mittelungszeitraum %D.

```
int  Dperiod() const
```

Rückgabewert

Der Mittelungszeitraum %D, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Slowing

Ruft die Zeitraum der Verlangsamung.

```
int  Slowing() const
```

Rückgabewert

Der Zeitraum der Verlangsamung, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_MA_METHOD).

© 2000-2025, MetaQuotes Ltd.

PriceField

Erhält das Verwendungsobjekt (Tief/Hoch oder Schluss/Schluss).

```
ENUM_STO_PRICE  PriceField() const
```

Rückgabewert

Ein Verwendungsobjekt (Tief/Hoch oder Schluss/Schluss) das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_STO_PRICE).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,          // Symbol
   ENUM_TIMEFRAMES  period,          // Periode
   int              Kperiod,         // Periode %K
   int              Dperiod,         // Periode %D
   int              slowing,         // Verlangsamung-Periode
   ENUM_MA_METHOD   ma_method,       // Mittelungsverfahren
   ENUM_STO_PRICE   price_field      // Verwendung
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

Kperiod

[in] Mittelwertzeitraum %K des Indikators.

Dperiod

[in] Mittelwertzeitraum %D des Indikators.

slowing

[in] Periode von Verlangsamung des Indikators.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

price_field

[in] Ein Verwendungsobjekt (Tief/Hoch oder Schluss/Schluss) des Indikators (ein Wert der Enumeration ENUM_STO_VOLUME)..

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers der Hauptlinie am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement der Hauptlinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Signal

Erhält das Element des Puffers der Signallinie am angegebenen Index

```
double  Signal(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement der Signallinie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiStochastic - IND_STOCHASTIC).

© 2000-2025, MetaQuotes Ltd.

Klasse CiTriX

CiTriX ist eine Klasse für Arbeit mit dem technischen Indikator "Triple Exponential Moving Averages Oscillator".

Beschreibung

Klasse CiTriX bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Triple Exponential Moving Averages Oscillator".

Deklaration

```
class CiTriX: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiTriX

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,         // Symbol
   ENUM_TIMEFRAMES  period,         // Periode
   int              ma_period,      // Mittelwertzeitraum
   int              applied         // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiTriX - IND_TRIX).

© 2000-2025, MetaQuotes Ltd.

Klasse CiWPR

CiWPR ist eine Klasse für Arbeit mit dem technischen Indikator "Williams' Percent Range".

Beschreibung

Klasse CiWPR bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Williams' Percent Range".

Deklaration

```
class CiWPR: public CIndicator
```

Kopf

```
#include <Indicators\Oscilators.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiWPR

Gruppen der Klassenmethode

Attribute	
CalcPeriod	Erhält den Berechnungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

CalcPeriod

Erhält den Berechnungszeitraum.

```
int  CalcPeriod() const
```

Rückgabewert

Der Berechnungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,          // Symbol
   ENUM_TIMEFRAMES  period,          // Periode
   int              calc_period      // Berechnungszeitraum
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

calc_period

[in] Berechnungszeitraum des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiWPR - IND_WPR).

© 2000-2025, MetaQuotes Ltd.

Eine Gruppe von technischen Indikatoren "Volumen"

Dieser Abschnitt enthält die Details der Arbeit mit einer Gruppe von Klassen von technischen Indikatoren "Volumen" und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klasse/Gruppe	Beschreibung
CiAD	Indikator "Accumulation/Distribution"
CiMFI	Indikator "Money Flow Index"
CiOBV	Indikator "On Balance Volume"
CiVolumes	Indikator "Volumes"

© 2000-2025, MetaQuotes Ltd.

Klasse CiAD

CiAD ist eine Klasse für Arbeit mit dem technischen Indikator "Accumulation/Distribution".

Beschreibung

Klasse CiAD bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Accumulation/Distribution".

Deklaration

```
class CiAD: public CIndicator
```

Kopf

```
#include <Indicators\Volumes.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiAD

Gruppen der Klassenmethode

Attribute	
Applied	Erhält den Berechnungszeitraum
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string                symbol,      // Symbol
   ENUM_TIMEFRAMES       period,      // Periode
   ENUM_APPLIED_VOLUME   applied      // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

applied

[in] Verwendungsobjekt (Volumentyp) des Indikators (ENUM_APPLIED_VOLUME).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiAD - IND_AD).

© 2000-2025, MetaQuotes Ltd.

Klasse CiMFI

CiMFI ist eine Klasse für Arbeit mit dem technischen Indikator "Money Flow Index".

Beschreibung

Klasse CiMFI bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Money Flow Index".

Deklaration

```
class CiMFI: public CIndicator
```

Kopf

```
#include <Indicators\Volumes.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiMFI

Gruppen der Klassenmethode

Attribute	
MaPeriod	Erhält den Mittelungszeitraum
Applied	Erhält das Verwendungsobjekt (Volumentyp)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

MaPeriod

Erhält den Mittelungszeitraum.

```
int  MaPeriod() const
```

Rückgabewert

Der Mittelungszeitraum, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string                symbol,         // Symbol
   ENUM_TIMEFRAMES       period,         // Periode
   int                   ma_period,      // Mittelwertzeitraum
   ENUM_APPLIED_VOLUME   applied         // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

ma_period

[in] Mittelwertzeitraum des Indikators.

applied

[in] Ein Verwendungsobjekt (Volumentyp) des Indikators (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiMFI - IND_MFI).

© 2000-2025, MetaQuotes Ltd.

Klasse CiMFI

CiOBV ist eine Klasse für Arbeit mit dem technischen Indikator "On Balance Volume".

Beschreibung

Klasse CiOBV bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "On Balance Volume".

Deklaration

```
class CiOBV: public CIndicator
```

Kopf

```
#include <Indicators\Volumes.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiOBV

Gruppen der Klassenmethode

Attribute	
Applied	Erhält das Verwendungsobjekt (Volumentyp)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string               symbol,      // Symbol
   ENUM_TIMEFRAMES      period,      // Periode
   ENUM_APPLIED_VOLUME  applied      // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

applied

[in] Ein Verwendungsobjekt (Volumentyp) des Indikators (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiOBV - IND_OBV).

© 2000-2025, MetaQuotes Ltd.

Klasse CiVolumes

CiVolumes ist eine Klasse für Arbeit mit dem technischen Indikator "Volumes".

Beschreibung

Klasse CiVolumes bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Volumes".

Deklaration

```
class CiVolumes: public CIndicator
```

Kopf

```
#include <Indicators\Volumes.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiVolumes

Gruppen der Klassenmethode

Attribute	
Applied	Erhält das Verwendungsobjekt (Volumentyp)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string               symbol,      // Symbol
   ENUM_TIMEFRAMES      period,      // Periode
   ENUM_APPLIED_VOLUME  applied      // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

applied

[in] Ein Verwendungsobjekt (Volumentyp) des Indikators (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiVolumes - IND_VOLUMES).

© 2000-2025, MetaQuotes Ltd.

Eine Gruppe von technischen Indikatoren von Bill Williams

Dieser Abschnitt enthält die Details der Arbeit mit einer Gruppe von Klassen von Bill Williams technischen Indikatoren und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Klasse/Gruppe	Beschreibung
CiAC	Indikator "Accelerator Oscillator"
CiAlligator	Indikator "Alligator"
CiAO	Indikator "Awesome Oscillator"
CiFractals	Indikator "Fractals"
CiGator	Indikator "Gator Oscillator"
CiBWMFI	Indikator "Market Facilitation Index"

© 2000-2025, MetaQuotes Ltd.

Klasse CiAC

CiAC ist eine Klasse für Arbeit mit dem technischen Indikator "Accelerator Oscillator".

Beschreibung

Klasse CiAC bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Accelerator Oscillator".

Deklaration

```
class CiAC: public CIndicator
```

Kopf

```
#include <Indicators\BillWilliams.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiAC

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string               symbol,      // Symbol
   ENUM_TIMEFRAMES      period       // Periode
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiAC - IND_AC).

© 2000-2025, MetaQuotes Ltd.

Klasse CiAlligator

CiAlligator ist eine Klasse für Arbeit mit dem technischen Indikator "Alligator".

Beschreibung

Klasse CiAlligator bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Alligator".

Deklaration

```
class CiAlligator: public CIndicator
```

Kopf

```
#include <Indicators\BillWilliams.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiAlligator

Gruppen der Klassenmethode

Attribute	
JawPeriod	Erhält den Mittelungszeitraum von "Jaws"
JawShift	Erhält den Versatz von "Jaws" entlang der Preisachse
TeethPeriod	Erhält den Mittelungszeitraum von "Teeth"
TeethShift	Erhält den Versatz von "Teeth" entlang der Preisachse
LipsPeriod	Erhält den Mittelungszeitraum von "Lips"
LipsShift	Erhält den Versatz von "Lips" entlang der Preisachse
MaMethod	Erhält den Mittelungsverfahren
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Jaw	Erhält Pufferdaten der Linie "Jaws"
Teeth	Erhält Pufferdaten der Linie "Teeth"

© 2000-2025, MetaQuotes Ltd.

Attribute	
Lips	Erhält Pufferdaten der Linie "Lips"
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

JawPeriod

Erhält den Mittelungszeitraum von "Jaws".

```
int  JawPeriod() const
```

Rückgabewert

Der Mittelungszeitraum von "Jaws", der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

JawShift

Erhält den Versatz von "Jaws" entlang der Preisachse.

```
int  JawShift() const
```

Rückgabewert

Der Versatz von "Jaws" entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

TeethPeriod

Erhält den Mittelungszeitraum von "Teeth".

```
int  TeethPeriod() const
```

Rückgabewert

Der Mittelungszeitraum von "Teeth", der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

TeethShift

Erhält den Versatz von "Teeth" entlang der Preisachse.

```
int  TeethShift() const
```

Rückgabewert

Der Versatz von "Teeth" entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

LipsPeriod

Erhält den Mittelungszeitraum von "Lips".

```
int  LipsPeriod() const
```

Rückgabewert

Der Mittelungszeitraum von "Lips", der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

LipsShift

Erhält den Versatz von "Lips" entlang der Preisachse.

```
int  LipsShift() const
```

Rückgabewert

Der Versatz von "Lips" entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,           // Symbol
   ENUM_TIMEFRAMES  period,           // Periode
   int              jaw_period,       // Periode von "Jaws"
   int              jaw_shift,        // Versatz von "Jaws"
   int              teeth_period,     // Periode von "Teeth"
   int              teeth_shift,      // Versatz von "Teeth"
   int              lips_period,      // Periode von "Lips"
   int              lips_shift,       // Versatz von "Lips"
   ENUM_MA_METHOD   ma_method,        // Mittelungsverfahren
   int              applied           // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

jaw_period

[in] Mittelungszeitraum von "Jaws".

jaw_shift

[in] Versatz von "Jaws" entlang der Preisachse.

teeth_period

[in] Mittelungszeitraum von "Teeth".

teeth_shift

[in] Versatz von "Teeth" entlang der Preisachse.

lips_period

[in] Mittelungszeitraum von "Lips".

lips_shift

[in] Versatz von "Lips" entlang der Preisachse.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Jaw

Erhält das Element des Puffers der "Jaws"-Linie am angegebenen Index

```
double  Jaw(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der "Jaws"-Linie.

Rückgabewert

Pufferelement der "Jaws"-Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Teeth

Erhält das Element des Puffers der "Teeth"-Linie am angegebenen Index

```
double  Teeth(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der "Teeth"-Linie.

Rückgabewert

Pufferelement der "Teeth"-Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Lips

Erhält das Element des Puffers der "Lips"-Linie am angegebenen Index

```
double  Lips(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements der "Lips"-Linie.

Rückgabewert

Pufferelement der "Lips"-Linie am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiAlligator - IND_ALLIGATOR).

© 2000-2025, MetaQuotes Ltd.

Klasse CiAO

CiAO ist eine Klasse für Arbeit mit dem technischen Indikator "Awesome Oscillator".

Beschreibung

Klasse CiAO bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Awesome Oscillator".

Deklaration

```
class CiAO: public CIndicator
```

Kopf

```
#include <Indicators\BillWilliams.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiAO

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,      // Symbol
   ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiAO - IND_AO).

© 2000-2025, MetaQuotes Ltd.

Klasse CiFractals

CiFractals ist eine Klasse für Arbeit mit dem technischen Indikator "iFractals".

Beschreibung

Klasse CiFractals bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "iFractals".

Deklaration

```
class CiFractals: public CIndicator
```

Kopf

```
#include <Indicators\BillWilliams.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiFractals

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Upper	Erhält Daten des oberen Puffers.
Lower	Erhält Daten des unteren Puffers.
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,      // Symbol
   ENUM_TIMEFRAMES  period       // Periode
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Upper

Erhält das Element des oberen Puffers am angegebenen Index

```
double  Upper(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements des oberen Puffers.

Rückgabewert

Element des oberen Puffers am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Lower

Erhält das Element des unteren Puffers am angegebenen Index

```
double  Lower(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements des unteren Puffers.

Rückgabewert

Element des unteren Puffers am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiFractals - IND_FRACTALS).

© 2000-2025, MetaQuotes Ltd.

Klasse CiGator

CiGator ist eine Klasse für Arbeit mit dem technischen Indikator "Gator Oscillator".

Beschreibung

Klasse CiGator bietet die Erstellung, Konfiguration und Datenzugriff für den Indikator "Gator Oscillator".

Deklaration

```
class CiGator: public CIndicator
```

Kopf

```
#include <Indicators\BillWilliams.mqh>
```

Vererbungshierarchie

CObject
CArray
CArrayObj
CSeries
CIndicator
CiGator

Gruppen der Klassenmethode

Attribute	
JawPeriod	Erhält den Mittelungszeitraum von "Jaws"
JawShift	Erhält den Versatz von "Jaws" entlang der Preisachse
TeethPeriod	Erhält den Mittelungszeitraum von "Teeth"
TeethShift	Erhält den Versatz von "Teeth" entlang der Preisachse
LipsPeriod	Erhält den Mittelungszeitraum von "Lips"
LipsShift	Erhält den Versatz von "Lips" entlang der Preisachse
MaMethod	Erhält den Mittelungsverfahren
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Upper	Erhält Daten des oberen Puffers.
Lower	Erhält Daten des unteren Puffers.

© 2000-2025, MetaQuotes Ltd.

Attribute	
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

JawPeriod

Erhält den Mittelungszeitraum von "Jaws".

```
int  JawPeriod() const
```

Rückgabewert

Der Mittelungszeitraum von "Jaws", der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

JawShift

Erhält den Versatz von "Jaws" entlang der Preisachse.

```
int  JawShift() const
```

Rückgabewert

Der Versatz von "Jaws" entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

TeethPeriod

Erhält den Mittelungszeitraum von "Teeth".

```
int  TeethPeriod() const
```

Rückgabewert

Der Mittelungszeitraum von "Teeth", der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

TeethShift

Erhält den Versatz von "Teeth" entlang der Preisachse.

```
int  TeethShift() const
```

Rückgabewert

Der Versatz von "Teeth" entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

LipsPeriod

Erhält den Mittelungszeitraum von "Lips".

```
int  LipsPeriod() const
```

Rückgabewert

Der Mittelungszeitraum von "Lips", der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

LipsShift

Erhält den Versatz von "Lips" entlang der Preisachse.

```
int  LipsShift() const
```

Rückgabewert

Der Versatz von "Lips" entlang der Preisachse, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

MaMethod

Erhält den Mittelungsverfahren.

```
ENUM_MA_METHOD  MaMethod() const
```

Rückgabewert

Die Mittelungsmethode, der während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Preistyp, Handle).

```
int  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Preistyp, Handle), das während der Erstellung des Indikators angegeben war.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string           symbol,           // Symbol
   ENUM_TIMEFRAMES  period,           // Periode
   int              jaw_period,       // Periode von "Jaws"
   int              jaw_shift,        // Versatz von "Jaws"
   int              teeth_period,     // Periode von "Teeth"
   int              teeth_shift,      // Versatz von "Teeth"
   int              lips_period,      // Periode von "Lips"
   int              lips_shift,       // Versatz von "Lips"
   ENUM_MA_METHOD   ma_method,        // Mittelungsverfahren
   int              applied           // Preistyp, Handle
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

jaw_period

[in] Mittelungszeitraum von "Jaws".

jaw_shift

[in] Versatz von "Jaws" entlang der Preisachse.

teeth_period

[in] Mittelungszeitraum von "Teeth".

teeth_shift

[in] Versatz von "Teeth" entlang der Preisachse.

lips_period

[in] Mittelungszeitraum von "Lips".

lips_shift

[in] Versatz von "Lips" entlang der Preisachse.

ma_method

[in] Mittelungsverfahren des Indikators (ein Wert der Enumeration ENUM_MA_METHOD).

applied

[in] Verwendungsobjekt (Preistyp, Handle) des Indikators.

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Upper

Erhält das Element des oberen Puffers am angegebenen Index

```
double  Upper(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements des oberen Puffers.

Rückgabewert

Element des oberen Puffers am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Lower

Erhält das Element des unteren Puffers am angegebenen Index

```
double  Lower(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Elements des unteren Puffers.

Rückgabewert

Element des unteren Puffers am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Typ des Indikators (für CiGator - IND_GATOR).

© 2000-2025, MetaQuotes Ltd.

Klasse CiBWMFI

CiBWMFI ist eine Klasse für Arbeit mit dem technischen Indikator "Market Facilitation Index by Bill Williams".

Beschreibung

Klasse CiBWMFI bietet die Erstellung, Konfiguration und Datenzugriff für des Indikators "Market Facilitation Index by Bill Williams".

Deklaration

```
class CiBWMFI: public CIndicator
```

Kopf

```
#include <Indicators\BillWilliams.mqh>
```

Vererbungshierarchie

CObject
  CArray
    CArrayObj
      CSeries
        CIndicator
          CiBWMFI

Gruppen der Klassenmethode

Attribute	
Applied	Erhält das Verwendungsobjekt (Preistyp, Handle)
Erstellung	
Create	Erstellt den Indikator
Datenzugriff	
Main	Erhält Pufferdaten
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

Methoden geerbt von der Klasse CArray

Step, Step, Total, Available, Max, IsSorted, SortMode, Clear, Sort

Methoden geerbt von der Klasse CArrayObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Compare

FreeMode, FreeMode, Save, Load, CreateElement, Reserve, Resize, Shutdown, Add, AddArray, Insert, InsertArray, AssignArray, At, Update, Shift, Detach, Delete, DeleteRange, Clear, CompareArray, InsertSort, Search, SearchGreat, SearchLess, SearchGreatOrEqual, SearchLessOrEqual, SearchFirst, SearchLast

Methoden geerbt von der Klasse CSeries

Name, BuffersTotal, BufferSize, Timeframe, Symbol, Period, PeriodDescription, RefreshCurrent

Methoden geerbt von der Klasse CIndicator

Handle, Status, FullRelease, Redrawer, Create, BufferResize, BarsCalculated, GetData, GetData, GetData, GetData, Minimum, MinValue, Maximum, MaxValue, Refresh, AddToChart, DeleteFromChart, MethodDescription, PriceDescription, VolumeDescription

© 2000-2025, MetaQuotes Ltd.

Applied

Erhält das Verwendungsobjekt (Volumentyp).

```
ENUM_APPLIED_VOLUME  Applied() const
```

Rückgabewert

Ein Verwendungsobjekt (Volumentyp), das während der Erstellung des Indikators angegeben war (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Indikator mit angegebenen Parameter. Um die Indikatorwerten zu aktualisieren und zu erhalten, verwenden Sie Refresh() und GetData().

```
bool  Create(
   string               symbol,      // Symbol
   ENUM_TIMEFRAMES      period,      // Periode
   ENUM_APPLIED_VOLUME  applied      // Volumentyp
   )
```

Parameter

symbol

[in] Symbol des Indikators.

period

[in] Die Arbeitsperiode des Indikators (ein Wert aus der Enumeration ENUM_TIMEFRAMES).

applied

[in] Ein Verwendungsobjekt (Volumentyp) des Indikators (ein Wert der Enumeration ENUM_APPLIED_VOLUME).

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn der Indikator nicht erstellt werden konnte.

© 2000-2025, MetaQuotes Ltd.

Main

Erhält das Element des Puffers am angegebenen Index

```
double  Main(
   int  index      // Index
   ) const
```

Parameter

index

[in] Der Index des Pufferelements.

Rückgabewert

Pufferelement am angegebenen Index oder EMPTY_VALUE, wenn es keine gültigen Daten gibt.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiBWMFI - IND_BWMFI).

© 2000-2025, MetaQuotes Ltd.

Klasse CiCustom

CiCustom ist eine Klasse für Arbeit mit dem benutzerdefinierten technischen Indikator.

Beschreibung

Klasse CiCustom bietet die Erstellung, Konfiguration und Datenzugriff für einen benutzerdefinierten Indikator.

Deklaration

```
class CiCustom: public CIndicator
```

Kopf

```
#include <Indicators\Custom.mqh>
```

Gruppen der Klassenmethode

Attribute	
NumBuffers	Setzt die Größe der Indikatorpuffer
NumParams	Erhält die Anzahl den bei der Erstellung des Indikators verwendeten Parametern
ParamType	Erhält den Typ des angegebenen Parameters
ParamLong	Erhält den Wert des angegebenen Integer-Parameters
ParamDouble	Erhält den Wert des angegebenen Double-Parameters
ParamString	Erhält den Wert des angegebenen String-Parameters
Eingabe/Ausgabe	
virtual Type	Virtuelle Identifizierungsmethode

© 2000-2025, MetaQuotes Ltd.

NumBuffers

Setzt die Größe der Indikatorpuffer.

```
bool  NumBuffers()
```

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Anzahl der Puffer nicht gesetzt werden konnte.

© 2000-2025, MetaQuotes Ltd.

NumParams

Erhält die Anzahl den bei der Erstellung des Indikators verwendeten Parametern.

```
int  NumParams() const
```

Rückgabewert

Die Anzahl den bei der Erstellung des Indikators verwendeten Parametern.

© 2000-2025, MetaQuotes Ltd.

ParamType

Erhält den Typ des angegebenen Parameters.

```
ENUM_DATATYPE  ParamType(
   int  index      // Nummer
   ) const
```

Parameter

index

[in] Nummer des Parameters.

Rückgabewert

Erhält den Typ des bei der Erstellung des Indikators verwendeten Parameters.

Hinweis

Wenn der Parameter-Nummer ist falsch, wird WRONG_VALUE zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

ParamLong

Erhält den Wert des angegebenen Integer-Parameters.

```
long  ParamLong(
   int  index      // Nummer
   ) const
```

Parameter

index

[in] Nummer des Parameters.

Rückgabewert

Den Wert des angegebenen Integer-Parameters, der bei der Erstellung des Indikators verwendet war.

Hinweis

Wenn der Parameter-Nummer ist falsch oder die Parametertyp ist kein Integer-Wert, wird 0 zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

ParamDouble

Erhält den Wert des angegebenen Double-Parameters.

```
double  ParamDouble(
   int  index      // Nummer
   ) const
```

Parameter

index

[in] Nummer des Parameters.

Rückgabewert

Den Wert des angegebenen Double-Parameters, der bei der Erstellung des Indikators verwendet war.

Hinweis

Wenn der Parameter-Nummer ist falsch oder die Parametertyp ist kein Double-Wert, wird EMPTY_VALUE zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

ParamString

Erhält den Wert des angegebenen String-Parameters.

```
string  ParamString(
   int  index      // Nummer
   ) const
```

Parameter

index

[in] Nummer des Parameters.

Rückgabewert

Den Wert des angegebenen String-Parameters, der bei der Erstellung des Indikators verwendet war.

Hinweis

Wenn der Parameter-Nummer ist falsch oder die Parametertyp ist kein String-Wert, wird ein leerer String zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

Type

Virtuelle Identifizierungsmethode.

```
virtual int  Type() const
```

Rückgabewert

Indikatortyp (für CiCustom - IND_CUSTOM).

© 2000-2025, MetaQuotes Ltd.

Handelsklassen

Dieser Abschnitt enthält die technischen Details der Arbeit mit Handelsklassen und die Beschreibungen der Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Handelsklassen sparen Sie Zeit bei der Entwicklung von benutzerdefinierten Anwendungen (Expert Advisors).

Die Standardbibliothek MQL5 (in Bezug auf Handelsklassen) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Trade.

Klasse/Gruppe	Beschreibung
CAccountInfo	Klasse für die Arbeit mit den Eigenschaften des Handelskontos
CSymbolInfo	Klasse für die Arbeit mit den Eigenschaften des Handelsinstruments
COrderInfo	Klasse für die Arbeit mit den Eigenschaften der Pending-Ordern
CHistoryOrderInfo	Klasse für die Arbeit mit den Eigenschaften der "historischen" Ordern
CPositionInfo	Klasse für die Arbeit mit den Eigenschaften der offenen Position
CDealInfo	Klasse für die Arbeit mit den Eigenschaften des "historischen" Deals
CTrade	Klasse für den Handel
CTerminalInfo	Klasse für den Zugriff auf die Parameter der Terminal-Umgebung

© 2000-2025, MetaQuotes Ltd.

Klasse CAccountInfo

Klasse CAccountInfo ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften des aktuellen offenen Handelskontos.

Beschreibung

Klasse CAccountInfo bietet den vereinfachten Zugriff auf die Eigenschaften des aktuellen offenen Handelskontos.

Deklaration

```
class CAccountInfo : public CObject
```

Kopf

```
#include <Trade\AccountInfo.mqh>
```

Vererbungshierarchie

CObject

CAccountInfo

Gruppen der Klassenmethode

Zugriff auf die ganzzahligen Eigenschaften	
Login	Erhält die Kontonummer
TradeMode	Erhält den Handelsmodus
TradeModeDescription	Erhält den Handelsmodus als einen String
Leverage	Erhält die Größe des gewährten Hebels
StopoutMode	Erhält den Modus der Angabe von minimaler Margin
StopoutModeDescription	Erhält den Modus der Angabe von minimaler Margin als einen String
TradeAllowed	Erhält das Flag der Handelserlaubnis
TradeExpert	Erhält das Flag der Erlaubnis zum automatisierten Handel
LimitOrders	Erhält die maximale Anzahl der aktiven Pending-Ordern
MarginMode	Erhält den Berechnungsmodus der Marge
MarginModeDescription	Erhält den Berechnungsmodus der Marge als einen String
Zugriff auf double-Eigenschaften	
Balance	Erhält den Kontostand

© 2000-2025, MetaQuotes Ltd.

Zugriff auf die ganzzahligen Eigenschaften	
Credit	Erhält die Größe des gewährten Kredits
Profit	Erhält den Wert des aktuellen Profits auf dem Konto
Equity	Erhält den Wert des Eigenkapitals auf dem Konto
Margin	Erhält die Größe der Margin
FreeMargin	Erhält die Größe der freie Margin
MarginLevel	Erhält den Wert von Margin-Level
MarginCall	Erhält den Wert von Margin-Call
MarginStopOut	Erhält den Wert von Margin-Stop-Out
Zugriff auf die Texteigenschaften	
Name	Erhält den Kundennamen
Server	Erhält den Namen des Handel-Servers
Currency	Erhält den Namen der Deposit-Währung
Company	Es erhält den Namen des Unternehmens, welches das Konto bedient
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält den Wert der angegebenen Integer-Eigenschaft des Kontos
InfoDouble	Erhält den Wert der angegebenen double-Eigenschaft des Kontos
InfoString	Erhält den Wert der angegebenen Texteigenschaft des Kontos
Zusätzliche Methoden	
OrderProfitCheck	Berechnet Gewinn auf der Grundlage der übergebenen Parameter
MarginCheck	Erhält die für den Handel erforderliche Margin-Größe
FreeMarginCheck	Erhält die Margin-Größe, die nach der Transaktion bleibt
MaxLotCheck	Erhält die maximal möglichen Handelsvolumen

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Login

Erhält die Kontonummer.

```
long  Login() const
```

Rückgabewert

Die Kontonummer.

© 2000-2025, MetaQuotes Ltd.

TradeMode

Erhält den Handelsmodus.

```
ENUM_ACCOUNT_TRADE_MODE  TradeMode() const
```

Rückgabewert

Handelsmodus aus der Enumeration ENUM_ACCOUNT_TRADE_MODE.

© 2000-2025, MetaQuotes Ltd.

TradeModeDescription

Erhält den Handelsmodus als einen String.

```
string  TradeModeDescription() const
```

Rückgabewert

Handelsmodus als String.

© 2000-2025, MetaQuotes Ltd.

Leverage

Erhält die Größe des gewährten Hebels.

```
long  Leverage() const
```

Rückgabewert

Sie Größe des gewährten Hebels.

© 2000-2025, MetaQuotes Ltd.

StopoutMode

Erhält den Modus der Angabe von minimaler Margin.

```
ENUM_ACCOUNT_STOPOUT_MODE  StopoutMode() const
```

Rückgabewert

Modus der Angabe von minimaler Margin aus der Ausyahlung ENUM_ACCOUNT_STOPOUT_MODE.

© 2000-2025, MetaQuotes Ltd.

StopoutModeDescription

Erhält den Modus der Angabe von minimaler Margin als einen String.

```
string  StopoutModeDescription() const
```

Rückgabewert

Der Modus der Angabe von minimaler Margin als String.

© 2000-2025, MetaQuotes Ltd.

MarginMode

Erhält den Berechnungsmodus der Marge.

```
ENUM_ACCOUNT_MARGIN_MODE  MarginMode() const
```

Rückgabewert

Der Berechnungsmodus der Marge aus der Aufzählung ENUM_ACCOUNT_MARGIN_MODE.

© 2000-2025, MetaQuotes Ltd.

MarginModeDescription

Erhält den Berechnungsmodus der Marge als String.

```
string  MarginModeDescription() const
```

Rückgabewert

Der Berechnungsmodus der Marge als String.

© 2000-2025, MetaQuotes Ltd.

TradeAllowed

Erhält das Flag der Handelserlaubnis.

```
bool  TradeAllowed() const
```

Rückgabewert

Das Flag der Handelserlaubnis.

© 2000-2025, MetaQuotes Ltd.

TradeExpert

Erhält das Flag der Erlaubnis zum automatisierten Handel.

```
bool  TradeExpert() const
```

Rückgabewert

Das Flag der Erlaubnis zum automatisierten Handel.

© 2000-2025, MetaQuotes Ltd.

LimitOrders

Erhält die maximale Anzahl der aktiven Pending-Ordern.

```
int  LimitOrders() const
```

Rückgabewert

Die maximale Anzahl der aktiven Pending-Ordern.

Hinweis

0 - es gibt keine Beschränkungen.

© 2000-2025, MetaQuotes Ltd.

Balance

Erhält den Kontostand.

```
double  Balance() const
```

Rückgabewert

Kontostand in Deposit-Währung.

© 2000-2025, MetaQuotes Ltd.

Credit

Erhält die Größe des gewährten Kredits.

```
double  Credit() const
```

Rückgabewert

Wert des gewährten Kredits in Deposit-Währung.

© 2000-2025, MetaQuotes Ltd.

Profit

Erhält den Wert des aktuellen Profits auf dem Konto.

```
double  Profit() const
```

Rückgabewert

Der Wert des aktuellen Profits des Kontos in Kontowährung.

© 2000-2025, MetaQuotes Ltd.

Equity

Erhält den Wert des Eigenkapitals auf dem Konto.

```
double  Equity() const
```

Rückgabewert

Der Wert des Eigenkapitals auf dem Konto in Deposit-Währung.

© 2000-2025, MetaQuotes Ltd.

Margin

Erhält die Größe der Margin.

```
double  Margin() const
```

Rückgabewert

Die Größe der Margin in Deposit-Währung.

© 2000-2025, MetaQuotes Ltd.

FreeMargin

Erhält die Größe der freie Margin.

```
double  FreeMargin() const
```

Rückgabewert

Die Größe der freie Margin in Deposit-Währung.

© 2000-2025, MetaQuotes Ltd.

MarginLevel

Erhält den Wert von Margin-Level.

```
double  MarginLevel() const
```

Rückgabewert

Der Wert von Margin-Level.

© 2000-2025, MetaQuotes Ltd.

MarginCall

Erhält den Wert von Margin-Call.

```
double  MarginCall() const
```

Rückgabewert

Der Wert von Margin-Call.

© 2000-2025, MetaQuotes Ltd.

MarginStopOut

Erhält den Wert von Margin-Stop-Out.

```
double  MarginStopOut() const
```

Rückgabewert

Der Wert von Margin-Stop-Out.

© 2000-2025, MetaQuotes Ltd.

Name

Erhält den Kundennamen.

```
string  Name() const
```

Rückgabewert

Kundennamen.

© 2000-2025, MetaQuotes Ltd.

Server

Erhält den Namen des Handel-Servers.

```
string  Server() const
```

Rückgabewert

Der Name des Handel-Servers.

© 2000-2025, MetaQuotes Ltd.

Currency

Erhält den Namen der Deposit-Währung.

```
string  Currency() const
```

Rückgabewert

Der Name der Deposit-Währung.

© 2000-2025, MetaQuotes Ltd.

Company

Es erhält den Namen des Unternehmens, welches das Konto bedient.

```
string  Company() const
```

Rückgabewert

Der Name des Unternehmens, welches das Konto bedient.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Erhält den Wert der angegebenen Integer-Eigenschaft des Kontos.

```
long  InfoInteger(
   ENUM_ACCOUNT_INFO_INTEGER  prop_id     // Identifikator der Eigenschaft
   ) const
```

Parameter

prop_id

[in] Identifikator der Eigenschaft. Der Wert kann ein aus ENUM_ACCOUNT_INFO_INTEGER sein.

Rückgabewert

Ein Wert vom Typ long.

© 2000-2025, MetaQuotes Ltd.

InfoDouble

Erhält den Wert der angegebenen double-Eigenschaft des Kontos.

```
double  InfoDouble(
   ENUM_ACCOUNT_INFO_DOUBLE  prop_id     // Identifikator der Eigenschaft
   ) const
```

Parameter

prop_id

[in] Identifikator der Eigenschaft. Der Wert kann ein aus ENUM_ACCOUNT_INFO_DOUBLE sein.

Rückgabewert

Ein Wert vom Typ double.

© 2000-2025, MetaQuotes Ltd.

InfoString

Erhält den Wert der angegebenen Texteigenschaft des Kontos.

```
string  InfoString(
   ENUM_ACCOUNT_INFO_STRING  prop_id     // Identifikator der Eigenschaft
   ) const
```

Parameter

prop_id

[in] Identifikator der Eigenschaft. Der Wert kann ein aus ENUM_ACCOUNT_INFO_STRING sein.

Rückgabewert

Ein Wert vom Typ string.

© 2000-2025, MetaQuotes Ltd.

OrderProfitCheck

Berechnet Gewinn für das aktuellen Konto auf der Grundlage der übergebenen Parameter. Erlaubt die Ergebnisse einer Handelsoperation vorläufig zu bewerten. Der Wert wird in der Kontowährung zurückgegeben.

```
double  OrderProfitCheck(
   const string        symbol,              // Symbol
   ENUM_ORDER_TYPE     trade_operation,     // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYP
   double              volume,              // Volumen
   double              price_open,          // Positioneröffnungspreis
   double              price_close          // Positionschlusspreis
   ) const
```

Parameter

symbol

[in] Das Symbol, für das Sie eine Transaktion führen möchten.

trade_operation

[in] Typ der Handelsoperation aus der Enumeration ENUM_ORDER_TYPE.

volume

[in] Volumen der Handelsoperation.

price_open

[in] Eröffnungspreis.

price_close

[in] Schlusspreis.

Rückgabewert

Falls erfolgreich, gibt den Gewinn-Wert zurück, oder EMPTY_VALUE im Falle eines Fehlers.

© 2000-2025, MetaQuotes Ltd.

MarginCheck

Erhält die für den Handel erforderliche Margin-Größe.

```
double  MarginCheck(
   const string        symbol,              // Symbol
   ENUM_ORDER_TYPE     trade_operation,     // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYP
   double              volume,              // Volumen
   double              price                // Preis
   ) const
```

Parameter

symbol

[in] Das Symbol, für das Sie eine Transaktion führen möchten.

trade_operation

[in] Typ der Handelsoperation aus der Enumeration ENUM_ORDER_TYPE.

volume

[in] Volumen der Handelsoperation.

price

[in] Preis der Handelsoperation.

Rückgabewert

Die für den Handel erforderliche Margin-Größe.

© 2000-2025, MetaQuotes Ltd.

FreeMarginCheck

Erhält die Margin-Größe, die nach der Transaktion bleibt.

```
double  FreeMarginCheck(
   const string        symbol,              // Symbol
   ENUM_ORDER_TYPE     trade_operation,     // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYP
   double              volume,              // Volumen
   double              price                // Preis
   ) const
```

Parameter

symbol

[in] Das Symbol, für das Sie eine Transaktion führen möchten.

trade_operation

[in] Typ der Handelsoperation aus der Enumeration ENUM_ORDER_TYPE.

volume

[in] Volumen der Handelsoperation.

price

[in] Preis der Handelsoperation.

Rückgabewert

Die Margin-Größe, die nach der Transaktion bleibt.

© 2000-2025, MetaQuotes Ltd.

MaxLotCheck

Erhält die maximal möglichen Handelsvolumen.

```
double  MaxLotCheck(
   const string         symbol,              // Symbol
   ENUM_ORDER_TYPE      trade_operation,     // Ordertyp (ORDER_TYPE_BUY oder ORDER_TYP
   double               price                // Preis
   double               percent=100          // Anteil der freie Margin (der Standardwe
   ) const
```

Parameter

symbol

[in] Das Symbol, für das Sie eine Transaktion führen möchten.

trade_operation

[in] Typ der Handelsoperation aus der Enumeration ENUM_ORDER_TYPE.

price

[in] Preis der Handelsoperation.

percent=100

[in] Der Anteil der freie Margin (in Prozent), den angeblich für eine Handelsoperation verwenden werden wird.

Rückgabewert

Die maximal möglichen Handelsvolumen.

© 2000-2025, MetaQuotes Ltd.

Klasse CSymbolInfo

CSymbolInfo ist eine Klasse für den vereinfachten Zugriff auf die Symboleigenschaften.

Beschreibung

Die Klasse CSymbolInfo bietet Zugriff auf Symboleigenschaften.

Deklaration

```
class CSymbolInfo : public CObject
```

Kopf

```
#include <Trade\SymbolInfo.mqh>
```

Vererbungshierarchie

CObject

CSymbolInfo

Gruppen der Klassenmethode

Verwaltung	
Refresh	Aktualisiert Symboldaten
RefreshRates	Aktualisiert Symbolpreise
Eigenschaften	
Name	Erhält/setzt den Namen des Finanzinstruments
Select	Erhält/setzt das Flag von "Symbol in Market Watch"
IsSynchronized	Prüft Symbolsynchronisation mit dem Server
Volumes	
Volume	Erhält das Volumen des letzten Deals
VolumeHigh	Erhält das maximale Volumen pro Tag
VolumeLow	Erhält das minimale Volumen pro Tag
Verschiedenes	
Time	Erhält die Zeit des letzten Preises
Spread	Erhält die Spreadgröße in Punkten.
SpreadFloat	Erhält das Zeichen von variablen Spread
TicksBookDepth	Erhält die Tiefe der Tick-Historie
Ebenen	

© 2000-2025, MetaQuotes Ltd.

Verwaltung	
StopsLevel	Erhält die minimale Distanz für Ordern in Punkten.
FreezeLevel	Erhält die Distanz, wo Handelsoperationen eingefroren werden, in Punkten.
Angebotspreise	
Bid	Erhält den aktuellen Bid-Preis
BidHigh	Erhält den maximalen Bid-Preis pro Tag.
BidLow	Erhält den minimalen Bid-Preis pro Tag.
Briefpreise	
Ask	Erhält den aktuellen Ask-Preis
AskHigh	Erhält den maximalen Ask-Preis pro Tag.
AskLow	Erhält den minimalen Ask-Preis pro Tag.
Preise	
Last	Erhält den aktuellen Last-Preis
LastHigh	Erhält den maximalen Last-Preis pro Tag.
LastLow	Erhält den minimalen Last-Preis pro Tag.
Handelsregime	
TradeCalcMode	Erhält die Methode für die Berechnung des Auftragswertes
TradeCalcModeDescription	Erhält die Methode für die Berechnung des Auftragswertes als String
TradeMode	Erhält Orderausführungstyp
TradeModeDescription	Erhält Orderausführungstyp als String
TradeExecution	Erhält den Modus von Handelsabschluss
TradeExecutionDescription	Erhält den Modus von Handelsabschluss als String
Swaps	
SwapMode	Erhält das Modell zur Swap-Berechnung
SwapModeDescription	Erhält das Modell zur Swap-Berechnung als String
SwapRollover3days	Erhält den Wochentag, an dem triple-Swap zugerechnet wird
SwapRollover3daysDescription	Erhält den Wochentag, an dem triple-Swap zugerechnet wird, als String
Margin und Flags	
MarginInitial	Erhält den Wert der Anfängliche Margin

© 2000-2025, MetaQuotes Ltd.

Verwaltung	
MarginMaintenance	Erhält den Wert der Haltemargin
MarginLong	Erhält den Faktor für Margin-Berechnung für Long-Positionen
MarginShort	Erhält den Faktor für Margin-Berechnung für Short-Positionen
MarginLimit	Erhält den Faktor für Margin-Berechnung für Limit-Ordern
MarginStop	Erhält den Faktor für Margin-Berechnung für Stop-Ordern
MarginStopLimit	Erhält den Faktor für Margin-Berechnung für Stop-Limit-Ordern
TradeTimeFlags	Erhält die Flags von erlaubten Modi des Order-Ablaufs
TradeFillFlags	Erhält die Flags von erlaubten Modi der Order-Füllung
Quantisierung	
Digits	Erhält die Anzahl der Dezimalstellen
Point	Erhält den Wert eines Punktes
TickValue	Erhält den Wert der Mindestpreisänderung
TickValueProfit	Erhält den berechneten Wert von Tick für eine profitable Position
TickValueLoss	Erhält den berechneten Wert von Tick für eine Verlustposition
TickSize	Erhält die Mindestpreisänderung
Kontraktgröße	
ContractSize	Erhält die Größe des Handelskontrakts
LotsMin	Erhält das minimale Volumen für eine Transaktion.
LotsMax	Erhält das maximale Volumen für eine Transaktion.
LotsStep	Erhält den minimalen Schritt der Volumenänderung für eine Transaktion.
LotsLimit	Erhält das maximal erlaubte Gesamtvolumen der offenen Position und Pending-Ordern für Symbol.
Swap-Größe	
SwapLong	Erhält die Größe der Swap der Long-Position
SwapShort	Erhält die Größe der Swap der Short-Position
Texteigenschaften	
CurrencyBase	Erhält den Namen der Basiswährung des Symbols
CurrencyProfit	Erhält den Namen der Profit-Währung
CurrencyMargin	Erhält den Namen der Margin-Währung
Bank	Erhält den Namen der Quelle des aktuellen Preises

© 2000-2025, MetaQuotes Ltd.

Verwaltung	
Description	Erhält die String-Beschreibung des Symbols
Path	Erhält den Pfad im Symbol-Baum
Symboleigenschaften	
SessionDeals	Erhält die Anzahl der Trades in der aktuellen Sitzung
SessionBuyOrders	Erhält die Gesamtzahl der Kauf-Ordern im Moment
SessionSellOrders	Erhält die Gesamtzahl der Verkauf-Ordern im Moment
SessionTurnover	Erhält den Gesamtumsatz in der aktuellen Sitzung
SessionInterest	Erhält das Gesamtvolumen der offenen Positionen
SessionBuyOrdersVolume	Erhält das Gesamtvolumen der Kauf-Ordern im Moment
SessionSellOrdersVolume	Erhält das Gesamtvolumen der Verkauf-Ordern im Moment
SessionOpen	Erhält den Eröffnungspreis der aktuellen Sitzuntg.
SessionClose	Erhält den Schlusspreis der aktuellen Sitzuntg.
SessionAW	Erhält den gewichteten Durchschnittspreis der aktuellen Sitzung
SessionPriceSettlement	Erhält den Abrechnungspreis für die aktuelle Sitzung
SessionPriceLimitMin	Erhält den minimal zulässigen Preiswert für die aktuelle Sitzung
SessionPriceLimitMax	Erhält den maximal zulässigen Preiswert für die aktuelle Sitzung
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält den Wert der angegebenen Integer-Eigenschaft
InfoDouble	Erhält den Wert der angegebenen double-Eigenschaft
InfoString	Erhält den Wert der angegebenen Texteigenschaft
Servicefunktionen	
NormalizePrice	Normalisiert den Preis unter Berücksichtigung der Symboleigenschaften

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert Symboldaten.

```
void  Refresh()
```

Rückgabewert

Nichts.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

RefreshRates

Aktualisiert Symbolpreise.

```
bool  RefreshRates()
```

Rückgabewert

Gibt bei Erfolg true zurück, und false wenn die Preise nicht aktualisiert werden konnten.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Name

Erhält den Namen des Finanzinstruments.

```
string  Name() const
```

Rückgabewert

Der Name des Finanzinstruments.

Name

Setzt den Namen des Finanzinstruments um mit ihm weiter zu arbeiten.

```
bool  Name(string name)
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Select

Erhält das Flag "Symbol in Market Watch".

```
bool  Select() const
```

Rückgabewert

Das Flag von "Symbol in Market Watch".

Select

Setzt das Flag "Symbol in Market Watch".

```
bool  Select()
```

Rückgabewert

Gibt bei Erfolg true zurück, false wenn ein Flag nicht geändert werden konnte.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

IsSynchronized

Prüft Symbolsynchronisation mit dem Server.

```
bool  IsSynchronized() const
```

Rückgabewert

Gibt true zurück, wenn das Symbol mit den Server synchronisiert ist, ansonsten false.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Volume

Erhält das Volumen des letzten Deals.

```
long  Volume() const
```

Rückgabewert

Das Volumen des letzten Deals.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

VolumeHigh

Erhält das maximale Volumen pro Tag.

```
long  VolumeHigh() const
```

Rückgabewert

Das maximale Volumen pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

VolumeLow

Erhält das minimale Volumen pro Tag.

```
long  VolumeLow() const
```

Rückgabewert

Das minimale Volumen pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Time

Erhält die Zeit des letzten Preises.

```
datetime  Time() const
```

Rückgabewert

Die Zeit des letzten Preises.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Spread

Erhält die Spreadgröße in Punkten.

```
int  Spread() const
```

Rückgabewert

Die Spreadgröße in Punkten.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SpreadFloat

Erhält das Zeichen von variablen Spread.

```
bool  SpreadFloat() const
```

Rückgabewert

Das Zeichen von variablen Spread.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TicksBookDepth

Erhält die Tiefe der Tick-Historie.

```
int  TicksBookDepth() const
```

Rückgabewert

Die Tiefe der Tick-Historie.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StopsLevel

Erhält die minimale Distanz für Ordern in Punkten.

```
int  StopsLevel() const
```

Rückgabewert

Die minimale Distanz für Ordern in Punkten.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

FreezeLevel

Erhält die Distanz, wo Handelsoperationen eingefroren werden, in Punkten.

```
int  FreezeLevel() const
```

Rückgabewert

Die Distanz, wo Handelsoperationen eingefroren werden, in Punkten.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Bid

Erhält den aktuellen Bid-Preis.

```
double  Bid() const
```

Rückgabewert

Der aktuelle Bid-Preis.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

BidHigh

Erhält den maximalen Bid-Preis pro Tag.

```
double  BidHigh() const
```

Rückgabewert

Der maximale Bid-Preis pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

BidLow

Erhält den minimalen Bid-Preis pro Tag.

```
double  BidLow() const
```

Rückgabewert

Der minimale Bid-Preis pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Ask

Erhält den aktuellen Ask-Preis.

```
double  Ask() const
```

Rückgabewert

Der aktuelle Ask-Preis.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

AskHigh

Erhält den maximalen Ask-Preis pro Tag.

```
double  AskHigh() const
```

Rückgabewert

Der maximale Ask-Preis pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

AskLow

Erhält den minimalen Ask-Preis pro Tag.

```
double  AskLow() const
```

Rückgabewert

Der minimale Ask-Preis pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Last

Erhält den aktuellen Last-Preis.

```
double  Last() const
```

Rückgabewert

Der aktuelle Last-Preis.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

LastHigh

Erhält den maximalen Last-Preis pro Tag.

```
double  LastHigh() const
```

Rückgabewert

Der maximale Last-Preis pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

LastLow

Erhält den minimalen Last-Preis pro Tag.

```
double  LastLow() const
```

Rückgabewert

Der minimale Last-Preis pro Tag.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeCalcMode

Erhält die Methode für die Berechnung des Auftragswertes.

```
ENUM_SYMBOL_CALC_MODE  TradeCalcMode() const
```

Rückgabewert

Die Methode für die Berechnung des Auftragswertes aus der Enumeration ENUM_SYMBOL_CALC_MODE.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeCalcModeDescription

Erhält die Methode für die Berechnung des Auftragswertes als String.

```
string  TradeCalcModeDescription() const
```

Rückgabewert

Die Methode für die Berechnung des Auftragswertes als String.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeMode

Erhält Orderausführungstyp.

```
ENUM_SYMBOL_TRADE_MODE  TradeMode() const
```

Rückgabewert

Orderausführungstyp aus der Enumeration ENUM_SYMBOL_TRADE_MODE.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeModeDescription

Erhält Orderausführungstyp als String.

```
string  TradeModeDescription() const
```

Rückgabewert

Orderausführungstyp als String.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeExecution

Erhält den Modus von Handelsabschluss.

```
ENUM_SYMBOL_TRADE_EXECUTION  TradeExecution() const
```

Rückgabewert

Der Modus von Handelsabschluss aus der Enumeration ENUM_SYMBOL_TRADE_EXECUTION.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeExecutionDescription

Erhält den Modus von Handelsabschluss als String.

```
string  TradeExecutionDescription() const
```

Rückgabewert

Der Modus von Handelsabschluss als String.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SwapMode

Erhält das Modell zur Swap-Berechnung.

```
ENUM_SYMBOL_SWAP_MODE  SwapMode() const
```

Rückgabewert

Modell zur Swap-Berechnung aus der Enumeration ENUM_SYMBOL_SWAP_MODE.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SwapModeDescription

Erhält das Modell zur Swap-Berechnung als String.

```
string  SwapModeDescription() const
```

Rückgabewert

Das Modell zur Swap-Berechnung als String.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SwapRollover3days

Erhält den Wochentag, an dem triple-Swap zugerechnet wird.

```
ENUM_DAY_OF_WEEK  SwapRollover3days() const
```

Rückgabewert

Der Wochentag, an dem triple-Swap zugerechnet wird, aus der Enumeration ENUM_DAY_OF_WEEK.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SwapRollover3daysDescription

Erhält den Wochentag, an dem triple-Swap zugerechnet wird, als String.

```
string  SwapRollover3daysDescription() const
```

Rückgabewert

Der Wochentag, an dem triple-Swap zugerechnet wird, als String.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginInitial

Erhält den Wert der Anfängliche Margin.

```
double  MarginInitial()
```

Rückgabewert

Der Wert der Anfängliche Margin.

Hinweis

Die Margin-Größe wird in Margin-Währung des Symbols pro einem Lot angegeben. Es wird verwendet, um die Mittel des Client bei der Markteintritt zu überprüfen.

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginMaintenance

Erhält den Wert der Haltemargin.

```
double  MarginMaintenance()
```

Rückgabewert

Der Wert der Haltemargin.

Hinweis

Die Margin-Größe wird in Margin-Währung des Symbols pro einem Lot angegeben. Es wird verwendet, um die Mittel des Client bei der Änderung des Kontozustands zu überprüfen. Wenn die Haltemargin 0 ist, verwendet es die anfängliche Margin.

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginLong

Erhält den Faktor für Margin-Berechnung für Long-Positionen.

```
double  MarginLong() const
```

Rückgabewert

Der Faktor für Margin-Berechnung für Long-Positionen.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginShort

Erhält den Faktor für Margin-Berechnung für Short-Positionen.

```
double  MarginShort() const
```

Rückgabewert

Der Faktor für Margin-Berechnung für Short-Positionen.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginLimit

Erhält den Faktor für Margin-Berechnung für Limit-Ordern.

```
double  MarginLimit() const
```

Rückgabewert

Der Faktor für Margin-Berechnung für Limit-Ordern.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginStop

Erhält den Faktor für Margin-Berechnung für Stop-Ordern.

```
double  MarginStop() const
```

Rückgabewert

Der Faktor für Margin-Berechnung für Stop-Ordern.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

MarginStopLimit

Erhält den Faktor für Margin-Berechnung für Stop-Limit-Ordern.

```
double  MarginStopLimit() const
```

Rückgabewert

Der Faktor für Margin-Berechnung für Stop-Limit-Ordern.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeTimeFlags

Erhält die Flags von erlaubten Modi von Orderablauf.

```
int  TradeTimeFlags() const
```

Rückgabewert

Flags von erlaubten Modi von Orderablauf.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TradeFillFlags

Erhält die Flags von erlaubten Modi der Order-Füllung.

```
int  TradeFillFlags() const
```

Rückgabewert

Die Flags von erlaubten Modi der Order-Füllung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Digits

Erhält die Anzahl der Dezimalstellen.

```
int  Digits() const
```

Rückgabewert

Die Anzahl der Dezimalstellen.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Point

Erhält den Wert eines Punktes.

```
double  Point() const
```

Rückgabewert

Der Wert eines Punktes.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TickValue

Erhält den Wert der Mindestpreisänderung.

```
double  TickValue() const
```

Rückgabewert

Der Wert der Mindestpreisänderung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TickValueProfit

Erhält den berechneten Wert von Tick für eine profitable Position.

```
double  TickValueProfit() const
```

Rückgabewert

Der berechnete Wert von Tick für eine profitable Position.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TickValueLoss

Erhält den berechneten Wert von Tick für eine Verlustposition.

```
double  TickValueLoss() const
```

Rückgabewert

Der berechnete Wert von Tick für eine Verlustposition.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TickSize

Erhält die Mindestpreisänderung.

```
double  TickSize() const
```

Rückgabewert

Die Mindestpreisänderung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

ContractSize

Erhält die Größe des Handelskontrakts.

```
double  ContractSize() const
```

Rückgabewert

Die Größe des Handelskontrakts.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

LotsMin

Erhält das minimale Volumen für eine Transaktion.

```
double  LotsMin() const
```

Rückgabewert

Das minimale Volumen für eine Transaktion.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

LotsMax

Erhält das maximale Volumen für eine Transaktion.

```
double  LotsMax() const
```

Rückgabewert

Das maximale Volumen für eine Transaktion.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

LotsStep

Erhält den minimalen Schritt der Volumenänderung für eine Transaktion.

```
double  LotsStep() const
```

Rückgabewert

Der minimale Schritt der Volumenänderung für eine Transaktion.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

LotsLimit

Erhält das maximal erlaubte Gesamtvolumen der offenen Position und Pending-Ordern (unabhängig von der Richtung) für Symbol.

```
double  LotsLimit() const
```

Rückgabewert

Das maximal erlaubte Gesamtvolumen der offenen Position und Pending-Ordern (unabhängig von der Richtung) für Symbol.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SwapLong

Erhält die Größe der Swap der Long-Position.

```
double  SwapLong() const
```

Rückgabewert

Die Größe der Swap der Long-Position.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SwapShort

Erhält die Größe der Swap der Short-Position.

```
double  SwapShort() const
```

Rückgabewert

Die Größe der Swap der Short-Position.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

CurrencyBase

Erhält den Namen der Basiswährung des Symbols.

```
string  CurrencyBase() const
```

Rückgabewert

Der Name der Basiswährung des Symbols.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

CurrencyProfit

Erhält den Namen der Profit-Währung.

```
string  CurrencyProfit() const
```

Rückgabewert

Der Name der Profit-Währung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

CurrencyMargin

Erhält den Namen der Margin-Währung.

```
string  CurrencyMargin() const
```

Rückgabewert

Der Name der Margin-Währung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Bank

Erhält den Namen der Quelle des aktuellen Preises.

```
string  Bank() const
```

Rückgabewert

Der Name der Quelle des aktuellen Preises.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Description

Erhält die String-Beschreibung des Symbols.

```
string  Description() const
```

Rückgabewert

Die String-Beschreibung des Symbols.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Path

Erhält den Pfad im Symbol-Baum.

```
string  Path() const
```

Rückgabewert

Der Pfad im Symbol-Baum.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionDeals

Erhält die Anzahl der Trades in der aktuellen Sitzung.

```
long  SessionDeals() const
```

Rückgabewert

Die Anzahl der Trades in der aktuellen Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionBuyOrders

Erhält die Gesamtzahl der Kauf-Ordern im Moment.

```
long  SessionBuyOrders() const
```

Rückgabewert

Die Gesamtzahl der Kauf-Ordern im Moment.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionSellOrders

Erhält die Gesamtzahl der Verkauf-Ordern im Moment.

```
long  SessionSellOrders() const
```

Rückgabewert

Die Gesamtzahl der Verkauf-Ordern im Moment.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionTurnover

Erhält den Gesamtumsatz in der aktuellen Sitzung.

```
double  SessionTurnover() const
```

Rückgabewert

Der Gesamtumsatz in der aktuellen Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionInterest

Erhält das Gesamtvolumen der offenen Positionen.

```
double  SessionInterest() const
```

Rückgabewert

Das Gesamtvolumen der offenen Positionen.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionBuyOrdersVolume

Erhält das Gesamtvolumen der Kauf-Ordern im Moment.

```
double  SessionBuyOrdersVolume() const
```

Rückgabewert

Das Gesamtvolumen der Kauf-Ordern im Moment.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionSellOrdersVolume

Erhält das Gesamtvolumen der Verkauf-Ordern im Moment.

```
double  SessionSellOrdersVolume() const
```

Rückgabewert

Das Gesamtvolumen der Verkauf-Ordern im Moment.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionOpen

Erhält den Eröffnungspreis der aktuellen Sitzuntg.

```
double  SessionOpen() const
```

Rückgabewert

Eröffnungspreis der aktuellen Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionClose

Erhält den Schlusspreis der aktuellen Sitzuntg.

```
double  SessionClose() const
```

Rückgabewert

Der Schlusspreis der aktuellen Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionAW

Erhält den gewichteten Durchschnittspreis der aktuellen Sitzung.

```
double  SessionAW() const
```

Rückgabewert

Der Wert des gewichteten Durchschnittspreises der aktuellen Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionPriceSettlement

Erhält den Abrechnungspreis für die aktuelle Sitzung.

```
double  SessionPriceSettlement() const
```

Rückgabewert

Der Wert des Abrechnungspreises für die aktuelle Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionPriceLimitMin

Erhält den minimal zulässigen Preiswert für die aktuelle Sitzung.

```
double  SessionPriceLimitMin() const
```

Rückgabewert

Der minimal zulässige Preiswert während der Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

SessionPriceLimitMax

Erhält den maximal zulässigen Preiswert während der Sitzung.

```
double  SessionPriceLimitMax() const
```

Rückgabewert

Der maximal zulässige Preiswert während der Sitzung.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Erhält den Wert der angegebenen Integer-Eigenschaft.

```
bool  InfoInteger(
   ENUM_SYMBOL_INFO_INTEGER  prop_id,     // Identifikator der Eigenschaft
   long&                     var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer Integer-Eigenschaft aus der Enumeration ENUM_SYMBOL_INFO_INTEGER.

var

[out] Referenz auf die Variable vom Typ long um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoDouble

Erhält den Wert der angegebenen double-Eigenschaft.

```
bool  InfoDouble(
   ENUM_SYMBOL_INFO_DOUBLE  prop_id,     // Identifikator der Eigenschaft
   double&                  var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft aus der Enumeration ENUM_SYMBOL_INFO_DOUBLE.

var

[out] Referenz auf die Variable vom Typ double um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoString

Erhält den Wert der angegebenen Texteigenschaft.

```
bool  InfoString(
   ENUM_SYMBOL_INFO_STRING  prop_id,      // Identifikator der Eigenschaft
   string&                  var           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Identifikator der Texteigenschaft.

var

[out] Referenz auf die Variable vom Typ string um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

NormalizePrice

Normalisiert den Preis unter Berücksichtigung der Symboleigenschaften.

```
double  NormalizePrice(
   double  price   // Preis
   ) const
```

Parameter

price

[in] Preis.

Rückgabewert

Die normalisierte Preis.

Hinweis

Das Symbol soll vorher mit der Methode Name ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Klasse COrderInfo

COrderInfo ist eine Klasse für den vereinfachten Zugriff auf die Eigenschaften der Pending-Order.

Beschreibung

Klasse COrderInfo bietet den Zugriff auf die Eigenschaften der Pending-Order.

Deklaration

```
class COrderInfo : public CObject
```

Kopf

```
#include <Trade\OrderInfo.mqh>
```

Vererbungshierarchie

CObject

COrderInfo

Gruppen der Klassenmethode

Zugriff auf die ganzzahligen Eigenschaften	
Ticket	Erhält Ticket einer vorausgewählten Order.
TimeSetup	Erhält die Zeit der Order-Aufgabe
TimeSetupMsc	Erhält die Zeit der Order-Aufgabe in Millisekunden seit 01.01.1970
OrderType	Erhält den Typ der Order
TypeDescription	Erhält den Typ der Order als String
State	Erhält den Status der Order
StateDescription	Erhält den Status der Order als String
TimeExpiration	Erhält die Ablaufzeit der Order
TimeDone	Erhält die Zeit der Ausführung oder Löschung einer Order
TimeDoneMsc	Erhält die Zeit der Ausführung oder Abbestellung einer Order in Millisekunden seit 01.01.1970
TypeFilling	Erhält Orderausführungstyp je nach Rest
TypeFillingDescription	Erhält Orderausführungstyp je nach Rest als String
TypeTime	Erhält den Typ der Order je nach Ablaufzeit
TypeTimeDescription	Erhält den Typ der Order je nach Ablaufzeit als String

© 2000-2025, MetaQuotes Ltd.

Zugriff auf die ganzzahligen Eigenschaften	
Magic	Erhält die Identifikator des Expert Advisors, der den Order aufgegeben hat
PositionId	Erhält die ID der Position
Zugriff auf double-Eigenschaften	
VolumeInitial	Erhält die Anfangsgröße der Order
VolumeCurrent	Erhält das ungefüllte Volumen der Order
PriceOpen	Erhält den Preis der Order
StopLoss	Erhält Stop Loss der Order
TakeProfit	Erhält Take Profit der Order
PriceCurrent	Erhält den aktuellen Preis des Symbols der Order
PriceStopLimit	Erhält den Aufgabepreis der Limit-Order
Zugriff auf die Texteigenschaften	
Symbol	Erhält den Namen des Symbols der Order
Comment	Erhält den Kommentar der Order
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält den Wert der angegebenen Integer-Eigenschaft
InfoDouble	Erhält den Wert der angegebenen double-Eigenschaft
InfoString	Erhält den Wert der angegebenen Texteigenschaft
Zustand	
StoreState	Speichert die Parameter der Order
CheckState	Vergleicht die aktuellen Einstellungen mit gespeicherten
Auswahl	
Select	Wählt eine Order aus nach Ticket, um ihr weiter zuzugreifen.
SelectByIndex	Wählt eine Order aus nach dem Index, um ihr weiter zuzugreifen.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Ticket

Erhält Ticket einer Order.

```
ulong  Ticket() const
```

Rückgabewert

Ticket der Order beim Erfolg oder ULONG_MAX im Falle eines Fehlers.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeSetup

Erhält die Zeit der Order-Aufgabe.

```
datetime  TimeSetup() const
```

Rückgabewert

Die Zeit der Order-Aufgabe.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeSetupMsc

Erhält die Zeit der Order-Aufgabe in Millisekunden seit 01.01.1970.

```
ulong  TimeSetupMsc() const
```

Rückgabewert

Die Zeit der Order-Aufgabe in Millisekunden seit 01.01.1970.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

OrderType

Erhält den Typ der Order.

```
ENUM_ORDER_TYPE  OrderType()
```

Rückgabewert

Typ der Order aus der Enumeration ENUM_ORDER_TYPE.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeDescription

Erhält den Typ der Order als String.

```
string  TypeDescription() const
```

Rückgabewert

Typ der Order als String.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

State

Erhält den Status der Order.

```
ENUM_ORDER_STATE  State() const
```

Rückgabewert

Status der Order aus der Enumeration ENUM_ORDER_STATE.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StateDescription

Erhält den Status der Order als String.

```
string  StateDescription() const
```

Rückgabewert

Der Status einer Order als String..

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeExpiration

Erhält die Ablaufzeit der Order.

```
datetime  TimeExpiration() const
```

Rückgabewert

Die Ablaufzeit der Order, die während Order-Aufgabe.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeDone

Erhält die Zeit der Ausführung oder Löschung einer Order.

```
datetime  TimeDone() const
```

Rückgabewert

Die Zeit der Ausführung oder Löschung einer Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeDoneMsc

Erhält die Zeit der Ausführung oder Abbestellung einer Order in Millisekunden seit 01.01.1970.

```
ulong  TimeDoneMsc() const
```

Rückgabewert

Die Zeit der Ausführung oder Abbestellung einer Order in Millisekunden seit 01.01.1970.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeFilling

Erhält Orderausführungstyp je nach Rest.

```
ENUM_ORDER_TYPE_FILLING  TypeFilling() const
```

Rückgabewert

Typ der Order-Ausführung je nach Rest aus der Enumeration ENUM_ORDER_TYPE_FILLING.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeFillingDescription

Erhält Orderausführungstyp je nach Rest als String.

```
string  TypeFillingDescription() const
```

Rückgabewert

Orderausführungstyp je nach Rest als String.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeTime

Erhält den Typ der Order je nach Ablaufzeit.

```
ENUM_ORDER_TYPE_TIME  TypeTime() const
```

Rückgabewert

Typ der Order je nach Ablaufzeit aus der Enumeration ENUM_ORDER_TYPE_TIME.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeTimeDescription

Erhält den Typ der Order je nach Ablaufzeit als String.

```
string  TypeTimeDescription() const
```

Rückgabewert

Typ der Order je nach Ablaufzeit als String.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Magic

Erhält die Identifikator des Expert Advisors, der den Order aufgegeben hat.

```
long  Magic() const
```

Rückgabewert

Die Identifikator des Expert Advisors, der den Order aufgegeben hat.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PositionId

Erhält die ID der Position.

```
long  PositionId() const
```

Rückgabewert

Die Identifikator der Position, an der die Order beteiligt.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

VolumeInitial

Erhält die Anfangsgröße der Order.

```
double  VolumeInitial() const
```

Rückgabewert

Die Anfangsgröße der Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

VolumeCurrent

Erhält das ungefüllte Volumen der Order.

```
double  VolumeCurrent() const
```

Rückgabewert

Ungefüllte Volumen der Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceOpen

Erhält den Preis der Order.

```
double  PriceOpen() const
```

Rückgabewert

Preis der Order-Platzierung

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StopLoss

Erhält den Preis von Stop Loss einer Order.

```
double  StopLoss() const
```

Rückgabewert

Der Preis von Stop Loss einer Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TakeProfit

Erhält den Preis von Take Profit einer Order.

```
double  TakeProfit() const
```

Rückgabewert

Der Preis von Take Profit einer Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceCurrent

Erhält den aktuellen Preis des Symbols der Order.

```
double  PriceCurrent() const
```

Rückgabewert

Der aktuelle Preis des Symbols der Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceStopLimit

Erhält den Aufgabepreis der Pending-Order.

```
double  PriceStopLimit() const
```

Rückgabewert

Der Aufgabepreis der Pending-Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Symbol

Erhält den Namen des Symbols der Order.

```
string  Symbol() const
```

Rückgabewert

Der Name des Symbols der Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Comment

Erhält den Kommentar der Order.

```
string  Comment() const
```

Rückgabewert

Kommentar zur Order.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Erhält den Wert der angegebenen Integer-Eigenschaft.

```
bool  InfoInteger(
   ENUM_ORDER_PROPERTY_INTEGER  prop_id,     // Identifikator der Eigenschaft
   long&                        var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer Integer-Eigenschaft aus der Enumeration ENUM_ORDER_PROPERTY_INTEGER.

var

[out] Referenz auf die Variable vom Typ long um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoDouble

Erhält den Wert der angegebenen double-Eigenschaft.

```
bool  InfoDouble(
   ENUM_ORDER_PROPERTY_DOUBLE  prop_id,     // Identifikator der Eigenschaft
   double&                     var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft aus der Enumeration ENUM_ORDER_PROPERTY_DOUBLE.

var

[out] Referenz auf die Variable vom Typ double um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoString

Erhält den Wert der angegebenen Texteigenschaft.

```
bool  InfoString(
   ENUM_ORDER_PROPERTY_STRING  prop_id,     // Identifikator der Eigenschaft
   string&                     var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer String-Eigenschaft aus der Enumeration ENUM_ORDER_PROPERTY_STRING.

var

[out] Referenz auf die Variable vom Typ string um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Eine Order muss vorher mit der Zugriffsmethode Select (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StoreState

Speichert die Parameter der Order.

```
void  StoreState()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CheckState

Vergleicht die aktuellen Einstellungen mit gespeicherten.

```
bool  CheckState()
```

Rückgabewert

Gibt true zurück, wenn die Parameter der Order seit dem letzten Aufruf der Methode StoreState() geändert haben, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Select

Wählt eine Order aus nach Ticket, um ihr weiter zuzugreifen.

```
bool  Select(
   ulong  ticket   // Order-Ticket
   )
```

Rückgabewert

Gibt bei Erfolg true zurück, oder false wenn keine Order ausgewählt ist.

© 2000-2025, MetaQuotes Ltd.

SelectByIndex

Wählt eine Order nach den Index aus um mit ihr weiter zu arbeiten.

```
bool  SelectByIndex(
   int  index      // Index der Order
   )
```

Parameter

index

[in] Der Index der Order.

Rückgabewert

Gibt bei Erfolg true zurück, oder false wenn keine Order ausgewählt ist.

© 2000-2025, MetaQuotes Ltd.

Klasse CHistoryOrderInfo

CHistoryOrderInfo ist eine Klasse für den vereinfachten Zugriff auf die Eigenschaften der historischen Order.

Beschreibung

Klasse CHistoryOrderInfo bietet den Zugriff auf die Eigenschaften der historischen Order.

Deklaration

```
class CHistoryOrderInfo : public CObject
```

Kopf

```
#include <Trade\HistoryOrderInfo.mqh>
```

Vererbungshierarchie

CObject

CHistoryOrderInfo

Gruppen der Klassenmethode

Zugriff auf die ganzzahligen Eigenschaften	
TimeSetup	Erhält die Zeit der Order-Aufgabe
TimeSetupMsc	Erhält die Zeit der Order-Aufgabe in Millisekunden seit 01.01.1970
OrderType	Erhält den Typ der Order
TypeDescription	Erhält den Typ der Order als String
State	Erhält den Status der Order
StateDescription	Erhält den Status der Order als String
TimeExpiration	Erhält die Ablaufzeit der Order
TimeDone	Erhält die Zeit der Ausführung oder Löschung einer Order
TimeDoneMsc	Erhält die Zeit der Ausführung oder Abbestellung einer Order in Millisekunden seit 01.01.1970
TypeFilling	Erhält Orderausführungstyp je nach Rest
TypeFillingDescription	Erhält Orderausführungstyp je nach Rest als String
TypeTime	Erhält den Typ der Order je nach Ablaufzeit
TypeTimeDescription	Erhält den Typ der Order je nach Ablaufzeit als String

© 2000-2025, MetaQuotes Ltd.

Zugriff auf die ganzzahligen Eigenschaften	
Magic	Erhält die Identifikator des Expert Advisors, der den Order aufgegeben hat
PositionId	Erhält die ID der Position
Zugriff auf double-Eigenschaften	
VolumeInitial	Erhält die Anfangsgröße der Order
VolumeCurrent	Erhält das ungefüllte Volumen der Order
PriceOpen	Erhält den Preis der Order
StopLoss	Erhält Stop Loss der Order
TakeProfit	Erhält Take Profit der Order
PriceCurrent	Erhält den aktuellen Preis des Symbols der Order
PriceStopLimit	Erhält den Aufgabepreis der Limit-Order
Zugriff auf die Texteigenschaften	
Symbol	Erhält das Symbol der Order
Comment	Erhält den Kommentar der Order
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält den Wert der angegebenen Integer-Eigenschaft
InfoDouble	Erhält den Wert der angegebenen double-Eigenschaft
InfoString	Erhält den Wert der angegebenen Texteigenschaft
Auswahl	
Ticket	Erhält Ticket/wählt eine Order aus
SelectByIndex	Wählt eine historische Order nach den Index aus

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

TimeSetup

Erhält die Zeit der Order-Aufgabe.

```
datetime  TimeSetup() const
```

Rückgabewert

Die Zeit der Order-Aufgabe.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeSetupMsc

Erhält die Zeit der Order-Aufgabe in Millisekunden seit 01.01.1970.

```
ulong  TimeSetupMsc() const
```

Rückgabewert

Die Zeit der Order-Aufgabe in Millisekunden seit 01.01.1970.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

OrderType

Erhält den Typ der Order.

```
ENUM_ORDER_TYPE  OrderType() const
```

Rückgabewert

Typ der Order aus der Enumeration ENUM_ORDER_TYPE.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeDescription

Erhält den Typ der Order als String.

```
string  TypeDescription() const
```

Rückgabewert

Typ der Order als String.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

State

Erhält den Status der Order.

```
ENUM_ORDER_STATE  State() const
```

Rückgabewert

Status der Order aus der Enumeration ENUM_ORDER_STATE.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StateDescription

Erhält den Status der Order als String.

```
string  StateDescription() const
```

Rückgabewert

Der Status einer Order als String..

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeExpiration

Erhält die Ablaufzeit der Order.

```
datetime  TimeExpiration() const
```

Rückgabewert

Die Ablaufzeit der Order, die während Order-Aufgabe.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeDone

Erhält die Zeit der Ausführung oder Löschung einer Order.

```
datetime  TimeDone() const
```

Rückgabewert

Die Zeit der Ausführung oder Löschung einer Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeDoneMsc

Erhält die Zeit der Ausführung oder Abbestellung einer Order in Millisekunden seit 01.01.1970.

```
ulong  TimeDoneMsc() const
```

Rückgabewert

Die Zeit der Ausführung oder Abbestellung einer Order in Millisekunden seit 01.01.1970.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeFilling

Erhält Orderausführungstyp je nach Rest.

```
ENUM_ORDER_TYPE_FILLING  TypeFilling() const
```

Rückgabewert

Typ der Order-Ausführung je nach Rest aus der Enumeration ENUM_ORDER_TYPE_FILLING.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeFillingDescription

Erhält Orderausführungstyp je nach Rest als String.

```
string  TypeFillingDescription() const
```

Rückgabewert

Orderausführungstyp je nach Rest als String.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeTime

Erhält den Typ der Order je nach Ablaufzeit.

```
ENUM_ORDER_TYPE_TIME  TypeTime() const
```

Rückgabewert

Typ der Order je nach Ablaufzeit aus der Enumeration ENUM_ORDER_TYPE_TIME.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeTimeDescription

Erhält den Typ der Order je nach Ablaufzeit als String.

```
string  TypeTimeDescription() const
```

Rückgabewert

Typ der Order je nach Ablaufzeit als String.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Magic

Erhält die Identifikator des Expert Advisors, der den Order aufgegeben hat.

```
long  Magic() const
```

Rückgabewert

Die Identifikator des Expert Advisors, der den Order aufgegeben hat.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PositionId

Erhält die ID der Position.

```
long  PositionId() const
```

Rückgabewert

Die Identifikator der Position, an der die Order beteiligte.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

VolumeInitial

Erhält die Anfangsgröße der Order.

```
double  VolumeInitial() const
```

Rückgabewert

Die Anfangsgröße der Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

VolumeCurrent

Erhält das ungefüllte Volumen der Order.

```
double  VolumeCurrent() const
```

Rückgabewert

Ungefüllte Volumen der Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceOpen

Erhält den Preis der Order.

```
double  PriceOpen() const
```

Rückgabewert

Preis der Order-Platzierung

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StopLoss

Erhält den Preis von Stop Loss einer Order.

```
double  StopLoss() const
```

Rückgabewert

Der Preis von Stop Loss einer Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TakeProfit

Erhält den Preis von Take Profit einer Order.

```
double  TakeProfit() const
```

Rückgabewert

Der Preis von Take Profit einer Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceCurrent

Erhält den aktuellen Preis des Symbols der Order.

```
double  PriceCurrent() const
```

Rückgabewert

Der aktuelle Preis des Symbols der Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceStopLimit

Erhält den Aufgabepreis der Pending-Order.

```
double  PriceStopLimit() const
```

Rückgabewert

Der Aufgabepreis der Pending-Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Symbol

Erhält den Namen des Symbols der Order.

```
string  Symbol() const
```

Rückgabewert

Der Name des Symbols der Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Comment

Erhält den Kommentar der Order.

```
string  Comment() const
```

Rückgabewert

Kommentar zur Order.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Erhält den Wert der angegebenen Integer-Eigenschaft.

```
bool  InfoInteger(
   ENUM_ORDER_PROPERTY_INTEGER  prop_id,     // Identifikator der Eigenschaft
   long&                        var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer Integer-Eigenschaft aus der Enumeration ENUM_ORDER_PROPERTY_INTEGER.

var

[out] Referenz auf die Variable vom Typ long um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoDouble

Erhält den Wert der angegebenen double-Eigenschaft.

```
bool  InfoDouble(
   ENUM_ORDER_PROPERTY_DOUBLE  prop_id,     // Identifikator der Eigenschaft
   double&                     var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft aus der Enumeration ENUM_ORDER_PROPERTY_DOUBLE.

var

[out] Referenz auf die Variable vom Typ double um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoString

Erhält den Wert der angegebenen Texteigenschaft.

```
bool  InfoString(
   ENUM_ORDER_PROPERTY_STRING  prop_id,     // Identifikator der Eigenschaft
   string&                     var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer String-Eigenschaft aus der Enumeration ENUM_ORDER_PROPERTY_STRING.

var

[out] Referenz auf die Variable vom Typ string um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Eine historische Order muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Ticket (Get-Methode)

Erhält Ticket einer Order.

```
ulong  Ticket() const
```

Rückgabewert

Order-Ticket.

Ticket (Set-Methode)

Wählt eine Order aus, um mit ihr weiter zu arbeiten.

```
void  Ticket(
   ulong  ticket      // Ticket
   )
```

Parameter

ticket

[in] Ticket der Order.

© 2000-2025, MetaQuotes Ltd.

SelectByIndex

Wählt eine historische Order nach den Index aus um mit ihr weiter zu arbeiten.

```
bool  SelectByIndex(
   int  index      // Index der Order
   )
```

Parameter

index

[in] Der Index der historischen Order.

Rückgabewert

Gibt bei Erfolg true zurück, oder false wenn keine Order ausgewählt ist.

© 2000-2025, MetaQuotes Ltd.

Klasse CPositionInfo

CPositionInfo ist eine Klasse für vereinfachten Zugriff auf die Eigenschaften der offenen Marktposition.

Beschreibung

CPositionInfo den Zugriff auf die Eigenschaften der offenen Marktposition.

Deklaration

```
class CPositionInfo : public CObject
```

Kopf

```
#include <Trade\PositionInfo.mqh>
```

Vererbungshierarchie

CObject

CPositionInfo

Gruppen der Klassenmethode

Zugriff auf die ganzzahligen Eigenschaften	
Time	Erhält die Eröffnungszeit einer Position
TimeMsc	Erhält Eröffnungszeit der Position in Millisekunden seit 01.01.1970
TimeUpdate	Erhält die Zeit der Positionsänderung in Sekunden seit 01.01.1970
TimeUpdateMsc	Erhält die Zeit der Positionsänderung in Millisekunden seit 01.01.1970
PositionType	Erhält den Typ der Position
TypeDescription	Erhält den Typ der Position als String
Magic	Erhält die Identifikator des Expert Advisors, der die Position eröffnet hat
Identifier	Erhält die ID der Position
Zugriff auf double-Eigenschaften	
Volume	Erhält das Volumen der Position
PriceOpen	Erhält den Eröffnungspreis einer Position
StopLoss	Erhält den Preis von Stop Loss einer Position
TakeProfit	Erhält den Preis von Take Profit einer Position

© 2000-2025, MetaQuotes Ltd.

Zugriff auf die ganzzahligen Eigenschaften	
PriceCurrent	Erhält den aktuellen Preis des Symbols der Position
Commission	Erhält die Größe der Kommission der Position
Swap	Erhält die Größe der Swap der Position
Profit	Erhält den Wert des aktuellen Profits der Position
Zugriff auf die Texteigenschaften	
Symbol	Erhält den Namen des Symbols der Position
Comment	Erhält den Kommentar der Position
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält den Wert der angegebenen Integer-Eigenschaft
InfoDouble	Erhält den Wert der angegebenen double-Eigenschaft
InfoString	Erhält den Wert der angegebenen Texteigenschaft
Auswahl	
Select	Wählt die Position aus
SelectByIndex	Wählt eine Position nach den Index aus
SelectByMagic	Wählt eine Position nach dem Symbolnamen und der Magic Number für die weitere Arbeit aus
SelectByTicket	Wählt eine Position nach dem Ticket
Zustand	
StoreState	Speichert die Parameter der Position
CheckState	Vergleicht die aktuellen Einstellungen mit gespeicherten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Time

Erhält die Eröffnungszeit einer Position.

```
datetime  Time() const
```

Rückgabewert

Die Eröffnungszeit einer Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeMsc

Erhält Eröffnungszeit der Position in Millisekunden seit 01.01.1970.

```
ulong  TimeMsc() const
```

Rückgabewert

Eröffnungszeit der Position in Millisekunden seit 01.01.1970.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeUpdate

Erhält die Zeit der Positionsänderung in Sekunden seit 01.01.1970.

```
datetime  TimeUpdate() const
```

Rückgabewert

Die Zeit der Positionsänderung in Sekunden seit 01.01.1970.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeUpdateMsc

Erhält die Zeit der Positionsänderung in Millisekunden seit 01.01.1970.

```
ulong  TimeUpdateMsc() const
```

Rückgabewert

Die Zeit der Positionsänderung in Millisekunden seit 01.01.1970.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PositionType

Erhält den Typ der Position.

```
ENUM_POSITION_TYPE  PositionType() const
```

Rückgabewert

Typ der Position aus der Enumeration ENUM_POSITION_TYPE.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeDescription

Erhält den Typ der Position als String.

```
string  TypeDescription() const
```

Rückgabewert

Der Typ der Position als String.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Magic

Erhält die Identifikator des Expert Advisors, der die Position eröffnet hat.

```
long  Magic() const
```

Rückgabewert

Die Identifikator des Expert Advisors, der die Position eröffnet hat.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Identifier

Erhält die ID der Position.

```
long  Identifier() const
```

Rückgabewert

Die ID der Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Volume

Erhält das Volumen der Position.

```
double  Volume() const
```

Rückgabewert

Das Volumen der Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceOpen

Erhält den Eröffnungspreis einer Position.

```
double  PriceOpen() const
```

Rückgabewert

Der Eröffnungspreis einer Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

StopLoss

Erhält den Preis von Stop Loss einer Position.

```
double  StopLoss() const
```

Rückgabewert

Preis von Stop Loss der Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TakeProfit

Erhält den Preis von Take Profit einer Position.

```
double  TakeProfit() const
```

Rückgabewert

Der Preis von Take Profit einer Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PriceCurrent

Erhält den aktuellen Preis des Symbols der Position.

```
double  PriceCurrent() const
```

Rückgabewert

Der aktuelle Preis des Symbols der Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Commission

Erhält die Größe der Kommission der Position.

```
double  Commission() const
```

Rückgabewert

Die Größe der Kommission für die Position in Kontowährung.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Swap

Erhält die Größe der Swap der Position.

```
double  Swap() const
```

Rückgabewert

Die Größe von Swap der Position in Kontowährung.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Profit

Erhält den Wert des aktuellen Profits der Position.

```
double  Profit() const
```

Rückgabewert

Der aktuelle Profit für die Position in Kontowährung.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Symbol

Erhält den Namen des Symbols der Position.

```
string  Symbol() const
```

Rückgabewert

Der Name des Symbols der Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Comment

Erhält den Kommentar der Position.

```
string  Comment() const
```

Rückgabewert

Kommentar zur Position.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Erhält den Wert der angegebenen Integer-Eigenschaft.

```
bool  InfoInteger(
   ENUM_POSITION_PROPERTY_INTEGER  prop_id,     // Identifikator der Eigenschaft
   long&                           var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer Integer-Eigenschaft aus der Enumeration ENUM_POSITION_PROPERTY_INTEGER.

var

[out] Referenz auf die Variable vom Typ long um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoDouble

Erhält den Wert der angegebenen double-Eigenschaft.

```
bool  InfoDouble(
   ENUM_POSITION_PROPERTY_DOUBLE  prop_id,      // Identifikator der Eigenschaft
   double&                        var           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft aus der Enumeration ENUM_POSITION_PROPERTY_DOUBLE.

var

[out] Referenz auf die Variable vom Typ double um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoString

Erhält den Wert der angegebenen Texteigenschaft.

```
bool  InfoString(
   ENUM_POSITION_PROPERTY_STRING  prop_id,      // Identifikator der Eigenschaft
   string&                        var           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer String-Eigenschaft aus der Enumeration ENUM_POSITION_PROPERTY_STRING.

var

[out] Referenz auf die Variable vom Typ string um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Die Position muss vorher mit der Zugriffsmethode Select (nach Symbol) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Select

Wählt eine Position aus, um mit ihr weiter zu arbeiten.

```
bool  Select(
   const string  symbol      // Symbol
   )
```

Parameter

symbol

[in] Das Symbol um Position auszuwählen.

© 2000-2025, MetaQuotes Ltd.

SelectByIndex

Wählt eine Position nach den Index aus um mit ihr weiter zu arbeiten.

```
bool  SelectByIndex(
   int  index      // Index der Position
   );
```

Parameter

index

[in] Index der Position.

Rückgabewert

Gibt bei Erfolg true zurück, oder false wenn keine Position ausgewählt ist.

© 2000-2025, MetaQuotes Ltd.

SelectByMagic

Wählt eine Position nach dem Symbolnamen und der Magic Number für die weitere Arbeit aus.

```
bool  SelectByMagic(
   const string  symbol, // Symbolname
   const ulong   magic   // Magic Number
   );
```

Parameter

symbol

[in] Symbolname.

magic

[in] Magic Number der Position.

Rückgabewert

true wenn erfolgreich, false wenn keine Position ausgewählt wurde.

© 2000-2025, MetaQuotes Ltd.

SelectByTicket

Wählt eine Position nach dem Ticket für die weitere Arbeit aus.

```
bool  SelectByTicket(
   ulong  ticket      // das Ticket der Position
   )
```

Parameter

ticket

[in] Ticket der Position.

Rückgabewert

true wenn erfolgreich, false wenn keine Position ausgewählt wurde.

© 2000-2025, MetaQuotes Ltd.

StoreState

Speichert die Parameter der Position.

```
void  StoreState()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CheckState

Vergleicht die aktuellen Einstellungen mit gespeicherten.

```
bool  CheckState()
```

Rückgabewert

Gibt true zurück, wenn die Parameter der Position seit dem letzten Aufruf der Methode StoreState() geändert haben, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Klasse CDealInfo

CDealInfo ist eine Klasse für den vereinfachten Zugriff auf die Deal-Eigenschaften.

Beschreibung

Die Klasse CDealInfo bietet Zugriff auf Deal-Eigenschaften.

Deklaration

```
class CDealInfo : public CObject
```

Kopf

```
#include <Trade\DealInfo.mqh>
```

Vererbungshierarchie

CObject

CDealInfo

Gruppen der Klassenmethode

Zugriff auf die ganzzahligen Eigenschaften	
Order	Erhält die Order, auf die eine Transaktion erfüllt wurde
Time	Erhält die Deal-Zeit
TimeMsc	Erhält die Zeit der Deal-Erfüllung in Millisekunden seit 01.01.1970
DealType	Erhält den Deal-Typ
TypeDescription	Erhält den Deal-Typ als String
Entry	Erhält die Richtung des Deals
EntryDescription	Erhält die Richtung des Deals als String
Magic	Erhält die Identifikator des Expert Advisors, der den Deal gemacht hat
PositionId	Erhält die Identifikator der Position, an der der Deal beteiligte
Zugriff auf double-Eigenschaften	
Volume	Erhält das Volumen des Deals
Price	Erhält den Deal-Preis
Commision	Erhält die Größe der Kommission beim Deal
Swap	Erhält die Größe der Swap bei der Positionschließung
Profit	Erhält das Finanzergebnis der Transaktion

© 2000-2025, MetaQuotes Ltd.

Zugriff auf die ganzzahligen Eigenschaften	
Zugriff auf die Texteigenschaften	
Symbol	Erhält den Namen des Symbols der Transaktion
Comment	Erhält einen Kommentar zum Deal
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält den Wert der angegebenen Integer-Eigenschaft
InfoDouble	Erhält den Wert der angegebenen double-Eigenschaft
InfoString	Erhält den Wert der angegebenen Texteigenschaft
Auswahl	
Ticket	Erhält Ticket/wählt einen Deal aus
SelectByIndex	Wählt einen Deal nach den Index aus

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Order

Erhält die Order, auf die eine Transaktion erfüllt wurde.

```
long  Order() const
```

Rückgabewert

Die Order, auf die eine Transaktion erfüllt wurde.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Time

Erhält die Zeit der Transaktion.

```
datetime  Time() const
```

Rückgabewert

Die Zeit der Transaktion.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TimeMsc

Erhält die Zeit der Deal-Erfüllung in Millisekunden seit 01.01.1970.

```
ulong  TimeMsc() const
```

Rückgabewert

Die Zeit der Deal-Erfüllung in Millisekunden seit 01.01.1970.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

DealType

Erhält den Deal-Typ.

```
ENUM_DEAL_TYPE  DealType() const
```

Rückgabewert

Typ der Transaktion aus der Enumeration ENUM_DEAL_TYPE.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

TypeDescription

Erhält den Deal-Typ als String.

```
string  TypeDescription() const
```

Rückgabewert

Deal-Typ als String.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Entry

Vergleicht die Richtung des Deals

```
ENUM_DEAL_ENTRY  Entry() const
```

Rückgabewert

Die Richtung der Transaktion (ein Wert der Enumeration ENUM_DEAL_ENTRY).

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

EntryDescription

Erhält die Richtung des Deals als String.

```
string  EntryDescription() const
```

Rückgabewert

Die Richtung des Deals als String.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Magic

Erhält die Identifikator des Expert Advisors, der den Deal gemacht hat.

```
long  Magic() const
```

Rückgabewert

Die Identifikator des Expert Advisors, der den Deal gemacht hat.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

PositionId

Erhält die Identifikator der Position, an der der Deal beteiligte.

```
long  PositionId() const
```

Rückgabewert

Die Identifikator der Position, an der der Deal beteiligte.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Volume

Erhält das Volumen des Deals.

```
double  Volume() const
```

Rückgabewert

Das Volumen des Deals.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Price

Erhält den Deal-Preis.

```
double  Price() const
```

Rückgabewert

Der Preis der Transaktion.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Commission

Erhält die Größe der Kommission beim Deal.

```
double  Commission() const
```

Rückgabewert

Die Größe der Kommission beim Deal.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Swap

Erhält die Größe der Swap bei der Positionschließung.

```
double  Swap() const
```

Rückgabewert

Die Größe der Swap bei der Positionschließung.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Profit

Erhält das Finanzergebnis der Transaktion.

```
double  Profit() const
```

Rückgabewert

Das Finanzergebnis der Transaktion in Kontowährung.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Symbol

Erhält den Namen des Symbols der Transaktion.

```
string  Symbol() const
```

Rückgabewert

Der Name des Symbols der Transaktion.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Comment

Erhält einen Kommentar zum Deal.

```
string  Comment() const
```

Rückgabewert

Kommentar zum Deal.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Erhält den Wert der angegebenen Integer-Eigenschaft.

```
bool  InfoInteger(
   ENUM_DEAL_PROPERTY_INTEGER  prop_id,     // Identifikator der Eigenschaft
   long&                       var          // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer Integer-Eigenschaft aus der Enumeration ENUM_DEAL_PROPERTY_INTEGER.

var

[out] Referenz auf die Variable vom Typ long um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoDouble

Erhält den Wert der angegebenen double-Eigenschaft.

```
bool  InfoDouble(
   ENUM_DEAL_PROPERTY_DOUBLE  prop_id,      // Identifikator der Eigenschaft
   double&                    var           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer double-Eigenschaft aus der Enumeration ENUM_DEAL_PROPERTY_DOUBLE.

var

[in] Referenz auf die Variable vom Typ double um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

InfoString

Erhält den Wert der angegebenen Texteigenschaft.

```
bool  InfoString(
   ENUM_DEAL_PROPERTY_STRING  prop_id,      // Identifikator der Eigenschaft
   string&                    var           // Referenz an die Variable
   ) const
```

Parameter

prop_id

[in] Die Identifikator einer String-Eigenschaft aus der Enumeration ENUM_DEAL_PROPERTY_STRING.

var

[out] Referenz auf die Variable vom Typ string um das Ergebnis zu halten.

Rückgabewert

Gibt bei Erfolg true zurück, false wenn der Wert der Eigenschaft nicht ausgelesen werden konnte.

Hinweis

Der Deal muss vorher mit der Zugriffsmethode Ticket (nach Ticket) oder SelectByIndex (nach Index) ausgewählt werden.

© 2000-2025, MetaQuotes Ltd.

Ticket (Get-Methode)

Erhält Ticket des Deals.

```
ulong  Ticket() const
```

Rückgabewert

Ticket des Deals.

Ticket (Set-Methode)

Wählt einen Deal aus, um mit ihm weiter zu arbeiten.

```
void  Ticket(
   ulong  ticket      // Ticket
   )
```

Parameter

ticket

[in] Ticket der Transaktion.

© 2000-2025, MetaQuotes Ltd.

SelectByIndex

Wählt einen Deal nach den Index aus um mit ihm weiter zu arbeiten.

```
bool  SelectByIndex(
   int  index      // Index des Deals
   )
```

Parameter

index

[in] Index des Deals.

Rückgabewert

Gibt bei Erfolg true zurück, oder false wenn kein Deal ausgewählt ist.

© 2000-2025, MetaQuotes Ltd.

Klasse CTrade

CTrade ist eine Klasse für den vereinfachten Zugriff auf die Handelsfunktionen.

Beschreibung

Klasse CTrade bietet den vereinfachten Zugriff auf die Handelsfunktionen.

Deklaration

```
class CTrade : public CObject
```

Kopf

```
#include <Trade\Trade.mqh>
```

Vererbungshierarchie

CObject

CTrade

Direkte Ableitungen

CExpertTrade

Gruppen der Klassenmethode

Einstellung	
LogLevel	Setzt die Protokollierungsstufe von Nachrichten
SetExpertMagicNumber	Setzt die Identifikator von Expert Advisor
SetDeviationInPoints	Setzt die zulässige Abweichung
SetTypeFilling	Setzt den Orderausführungstyp
SetTypeFillingBySymbol	Setzt den Typ derAusführung einer Order entsprechend den Einstellungen des angegebenen Symbols
SetAsyncMode	Setzt den asynchronen Handelsmodus
SetMarginMode	Setzt den Modus der Berechnung der Marge entsprechend den Einstellungen des aktuellen Kontos
Order-Operationen	
OrderOpen	Liegt eine Pending-Order mit angegebenen Parameter an
OrderModify	Ändert die Eigenschaften einer Pending-Order
OrderDelete	Löscht eine Pending-Order
Operationen mit Positionen	
PositionOpen	Eröffnet eine Position mit den angegebenen Parametern

© 2000-2025, MetaQuotes Ltd.

Einstellung	
PositionModify	Ändert die Parameter der Position
PositionClose	Schließt eine Position
PositionClosePartial	Schließt einen Teil der Position nach dem angegebenen Symbol oder Ticket
PositionCloseBy	Schließt die Position mit dem angegebenen Ticket zur Gegenposition
Zusätzliche Methoden	
Buy	Eröffnet eine Long-Position mit den angegebenen Parametern
Sell	Eröffnet eine Short-Position mit den angegebenen Parametern
BuyLimit	Stellt eine Pending-Kaufsorder aus zum Preis, der besser als Marktpreis ist
BuyStop	Stellt eine Pending-Kaufsorder aus zum Preis, der schlechter als Marktpreis ist
SellLimit	Stellt eine Pending-Verkaufsorder aus zum Preis, der besser als Marktpreis ist
SellStop	Stellt eine Pending-Verkaufsorder aus zum Preis, der schlechter als Marktpreis ist
Zugriff auf die Parameter der letzten Anfrage	
Request	Erhält eine Kopie der Struktur der letzten Anfrage
RequestAction	Erhält den Typ der Transaktion
RequestActionDescription	Erhält den Typ der Transaktion als String
RequestMagic	Erhält die ID des Expert Advisors
RequestOrder	Erhält Ticket einer Order
RequestSymbol	Erhält den Namen des Handelsinstruments, für das die Order ausgestellt ist
RequestVolume	Erhält das angefragten Volumen des Deals in Lots
RequestPrice	Erhält den Preis zu dem die Order ausgeführt werden soll
RequestStopLimit	Erhält den Preis, zu dem eine Pending-Stop-Limit-Order ausgestellt wird
RequestSL	Erhält den Preis, zu dem Stop Loss aktiviert wird
RequestTP	Erhält den Preis, zu dem Take Profit aktiviert wird
RequestDeviation	Erhält die maximal zulässige Abweichung von der Preisvorstellung
RequestType	Erhält den Typ der Order

© 2000-2025, MetaQuotes Ltd.

Einstellung	
RequestTypeDescription	Erhält den Typ der Order als String
RequestTypeFilling	Erhält den Orderausführungstyp
RequestTypeFillingDescription	Erhält den Orderausführungstyp als String
RequestTypeTime	Erhält den Typ der Order je nach Ablaufzeit
RequestTypeTimeDescription	Erhält den Typ der Order je nach Ablaufzeit als String
RequestExpiration	Erhält die Ablaufzeit der Pending-Order
RequestComment	Erhält den Kommentar zur Order
RequestPosition	Erhält das Ticket der Position
RequestPositionBy	Erhält das Ticket der Gegenposition
Zugriff auf die Prüfergebnisse der letzten Anfrage	
CheckResult	Erhält eine Kopie der Struktur der Prüfergebnisse der letzten Anfrage
CheckResultRetcode	Gibt den Wert des retcode-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultRetcodeDescription	Gibt die String-Beschreibung des retcode-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultBalance	Gibt den Wert des balance-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultEquity	Gibt den Wert des balance-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultProfit	Erhält den Wert des schwebenden Profits, der nach der Transaktion erwartet ist
CheckResultMargin	Gibt den Wert des margin-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultMarginFree	Gibt den Wert des margin_free-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultMarginLevel	Gibt den Wert des margin_level-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
CheckResultComment	Gibt den Wert des comment-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war
Zugriff auf die Ausführungsergebnisse der letzten Anfrage	

© 2000-2025, MetaQuotes Ltd.

Einstellung	
Result	Erhält eine Kopie der Struktur der Ausführungsergebnisse der letzten Anfrage
ResultRetcode	Erhält den Code der Abfrageausführungsergebnisse
ResultRetcodeDescription	Erhält den Code der Abfrageausführungsergebnisse als String
ResultDeal	Erhält Ticket des Deals
ResultOrder	Erhält Ticket einer Order
ResultVolume	Erhält das Volumen des Deals oder der Order
ResultPrice	Erhält den durch Broker bestätigten Preis
ResultBid	Erhält den aktuellen Bid-Preis (Requote)
ResultAsk	Erhält den aktuellen Ask-Preis (Requote)
ResultComment	Erhält den Kommentar des Brokers
Hilfsmethoden	
PrintRequest	Schreibt die Parameter der letzten Anfrage ins Journal
PrintResult	Schreibt die Ausführungsergebnisse der letzten Anfrage ins Journal
FormatRequest	Schreibt die Parameter der letzten Anfrage in String
FormatRequestResult	Schreibt die Ausführungsergebnisse der letzten Anfrage in String

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

LogLevel

Setzt die Protokollierungsstufe von Nachrichten.

```
void  LogLevel(
   ENUM_LOG_LEVELS  log_level      // Stufe
   )
```

Parameter

log_level

[in] Eine neue Protokollirungsstufe.

Rückgabewert

Nichts.

Hinweis

LOG_LEVEL_NO und weniger deaktiviert Schreiben von allen Nachrichten (wird automatisch beim Optimieren verwendet). LOG_LEVEL_ERRORS aktiviert Schreiben nur von Fehlermeldungen (Standardwert). LOG_LEVEL_ALL und mehr aktiviert Schreiben von allen Nachrichten (wird automatisch beim Testen verwendet).

ENUM_LOG_LEVELS

Identifikator	Beschreibung	Wert
LOG_LEVEL_NO	Meldungausgabe ist deaktiviert	0
LOG_LEVEL_ERRORS	Nur Fehlermeldungen werden geschrieben	1
LOG_LEVEL_ALL	Alle Meldungen werden geschrieben	2

© 2000-2025, MetaQuotes Ltd.

SetExpertMagicNumber

Setzt die Identifikator von Expert Advisor

```
void  SetExpertMagicNumber(
   ulong  magic      // Identifikator
   )
```

Parameter

magic

[in] Die neue Expert Advisor ID

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

SetDeviationInPoints

Setzt die zulässige Abweichung.

```
void  SetDeviationInPoints(
   ulong  deviation      // Slippage
   )
```

Parameter

deviation

[in] Die zulässige Abweichung in Punkten.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

SetTypeFilling

Setzt den Orderausführungstyp.

```
void  SetTypeFilling(
   ENUM_ORDER_TYPE_FILLING  filling      // Orderausführungstyp
   )
```

Parameter

filling

[in] Orderausführungstyp aus der Enumeration ENUM_ORDER_TYPE_FILLING.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

SetTypeFillingBySymbol

Setzt den Typ derAusführung einer Order entsprechend den Einstellungen des angegebenen Symbols.

```
bool  SetTypeFillingBySymbol(
   const string   symbol      // Name des Symbols
   )
```

Parameter

symbol

[in] Name des Symbols, in welchem SYMBOL_FILLING_MODE zulässige Politiken der Ausführung von Orders beinhaltet.

Rückgabewert

true - erfolgreiche Ausführung, false - die Ausführungspolitik konnte nicht definiert werden.

Hinweis

+Wenn für das Symbol gleichzeitig zwei Politiken der Ausführung SYMBOL_FILLING_FOK und SYMBOL_FILLING_IOC erlaubt sind, wird der Order der Wert ORDER_FILLING_FOK zugewiesen.

© 2000-2025, MetaQuotes Ltd.

SetAsyncMode

Setzt den asynchronen Handelsmodus.

```
void  SetAsyncMode(
   bool  mode      // asynchroner Handelsmodus
   )
```

Parameter

mode

[in] Status der Verwendung vom asynchronen Handelsmodus

Rückgabewert

Nichts.

Hinweis

Asynchroner Handelsmodus erlaubt Handelsoperationen ohne nach Antwort aus dem Handelsserver zu warten auszuführen (siehe OrderSendAsync).

© 2000-2025, MetaQuotes Ltd.

SetMarginMode

Setzte den Berechnungsmodus der Marge in Übereinstimmung mit den Einstellungen des aktuellen Kontos.

```
void  SetMarginMode()
```

Rückgabewert

Nein.

Hinweis

Der Berechnungsmodus der Marge wird in ENUM_ACCOUNT_MARGIN_MODE angegeben.

© 2000-2025, MetaQuotes Ltd.

OrderOpen

Liegt eine Pending-Order mit angegebenen Parameter an.

```
bool  OrderOpen(
   const string          symbol,          // Symbol
   ENUM_ORDER_TYPE       order_type,      // Ordertyp
   double                volume,          // Volumen der Order
   double                limit_price,     // Preis von StopLomit
   double                price,           // Ausführungspreis
   double                sl,              // Preis von stop loss
   double                tp,              // Preis von take profit
   ENUM_ORDER_TYPE_TIME  type_time,       // Ablauftyp
   datetime              expiration,      // Ablauf
   const string          comment=""       // Kommentar
   )
```

Parameter

symbol

[in] Name des Handelssymbols.

order_type

[in] Typ der Handelsoperation für die Order aus der Enumeration ENUM_ORDER_TYPE.

volume

[in] Das angefragte Volumen der Order.

limit_price

[in] Der Preis, zu dem eine Pending-Stop-Limit-Order ausgestellt wird.

price

[in] Preis, zu dem die Order ausgeführt werden soll.

sl

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp

[in] Der Preis, zu dem Take Profit aktiviert wird.

type_time

[in] Typ der Order nach dem Ablauf aus der Enumeration ENUM_ORDER_TYPE_TIME.

expiration

[in] Die Ablaufzeit der Pending-Order.

comment=""

[in] Kommentar zur Order.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Hinweis

Der erfolgreiche Abschluss der Methode OrderOpen(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultOrder() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

OrderModify

Ändert die Eigenschaften einer Pending-Order.

```
bool  OrderModify(
   ulong                ticket,         // Ticket der Order
   double               price,          // Ausführungspreis
   double               sl,             // Preis von stop loss
   double               tp,             // Preis von take profit
   ENUM_ORDER_TYPE_TIME type_time,      // Ablauftyp
   datetime             expiration,     // Ablauf
   double               stoplimit       // Preis von Limit-Order
   )
```

Parameter

ticket

[in] Ticket der Order.

price

[in] Der neue Preis, zu dem die Order ausgeführt werden soll (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

sl

[in] Der neue Preis, zu dem Stop Loss aktiviert werden soll (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

tp

[in] Der neue Preis, zu dem Take Profit aktiviert werden soll (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

type_time

[in] Der neue Ordertyp nach dem Ablauf aus der Enumeration ENUM_ORDER_TYPE_TIME (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

expiration

[in] Die neue Ablaufzeit der Pending-Order (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

stoplimit

[in] Der neue Preis zu dem die Limit-Order ausgestellt wird, nachdem der Preis den Wert *price erreicht*. Wird nur für StopLimit-Ordern angegeben.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode OrderModify(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode().

© 2000-2025, MetaQuotes Ltd.

OrderDelete

Löscht eine Pending-Order.

```
bool  OrderDelete(
   ulong  ticket      // Ticket der Order
   )
```

Parameter

ticket

[in] Ticket der Order.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode OrderDelete(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode().

© 2000-2025, MetaQuotes Ltd.

PositionOpen

Eröffnet eine Position mit den angegebenen Parametern für das angegebenen Symbol.

```
bool  PositionOpen(
   const string      symbol,          // Symbol
   ENUM_ORDER_TYPE   order_type,      // Typ der Handelsoperation für Positioneröffnung
   double            volume,          // Volumen der Position
   double            price,           // Ausführungspreis
   double            sl,              // Preis von Stop Loss
   double            tp,              // Preis von Take Profit
   const string      comment=""       // Kommentar
   )
```

Parameter

symbol

[in] Name des Handelsinstruments, für das eine Position eröffnet wird.

order_type

[in] Typ der Handelsoperation für die Position aus der Enumeration ENUM_ORDER_TYPE.

volume

[in] Das angefragte Volumen der Position.

price

[in] Der Preis zu welchem die Position eröffnet werden soll.

sl

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp

[in] Der Preis, zu dem Take Profit aktiviert wird.

comment=""

[in] Kommentar zur Position.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode PositionOpen(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultDeal() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

PositionModify

Ändert die Parameter der Position für das angegebenen Symbol.

```
bool  PositionModify(
   const string  symbol,      // Symbol
   double        sl,          // Preis von Stop Loss
   double        tp           // Preis von Take Profit
   )
```

Modifies position parameters by the specified ticket.

```
bool  PositionModify(
   const ulong   ticket,      // position ticket
   double        sl,          // Stop Loss price
   double        tp           // Take Profit price
   )
```

Parameter

symbol

[in] Name des Handelsinstruments, für das eine Position geändert werden soll.

ticket

[in] das Ticket der zu modifizierenden Position.

sl

[in] Der neue Preis, zu dem Stop Loss aktiviert werden soll (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

tp

[in] Der neue Preis, zu dem Take Profit aktiviert werden soll (oder vorheriger Wert, wenn die Änderung ist nicht gebraucht).

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Ein erfolgreiches Beenden der Methode PositionModify(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis der Ausführung von Trade Request (Return Code des Handelsservers) durch den Aufruf der Methode ResultRetcode() zu prüfen.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehreren Trades darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen pro Symbol vorhanden sein. In diesem Fall ändert PositionModify die Position mit dem kleinsten Ticket.

© 2000-2025, MetaQuotes Ltd.

PositionClose

Schließt eine Position für das angegebenen Symbol.

```
bool  PositionClose(
   const string  symbol,                  // Symbol
   ulong         deviation=ULONG_MAX      // Abweichung
   )
```

Closes a position with the specified ticket.

```
bool  PositionClose(
   const ulong   ticket,                  // Position ticket
   ulong         deviation=ULONG_MAX      // Deviation
   )
```

Parameter

symbol

[in] Name des Handelsinstruments, für das eine Position geschlossen werden soll.

ticket

[in] das Ticket der zu schließenden Position.

deviation=ULONG_MAX

[in] Die maximale Abweichung vom aktuellen Preis (in Punkten).

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Ein erfolgreiches Beenden der Methode PositionClose(...) bedeutet nicht immer ein erfolgreiches Ausführen der Transaktion. Es ist notwendig, das Ergebnis der Ausführung von Trade Request (Return Code des Handelsservers) durch den Aufruf der Methode ResultRetcode() zu prüfen.

Im Netting Mode (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehreren Trades darstellt. Man darf Positionen und aktuelle Pending Orders, die auch im Tab Handel in der Werkzeugleiste angezeigt werden, nicht verwechseln.

Bei einer unabhängigen Verrechnung von Positionen (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen pro Symbol vorhanden sein. In diesem Fall schließt PositionClose die Position mit dem kleinsten Ticket.

© 2000-2025, MetaQuotes Ltd.

PositionClosePartial

Schließt einen Teil der Position mit dem angegebenen Symbol im Hedging-Modus.

```
bool  PositionClosePartial(
   const string  symbol,                  // Symbol
   const double  volume,                  // Volumen
   ulong         deviation=ULONG_MAX      // Abweichung
   )
```

Schießt einen Teil der Position mit dem angegebenen Ticket im Hedging-Modus.

```
bool  PositionClosePartial(
   const ulong   ticket,                  // Ticket der Position
   const double  volume,                  // Volumen
   ulong         deviation=ULONG_MAX      // Abweichung
   )
```

Parameter

symbol

[in] Bezeichnung des Symbols, auf welchem ein Teil der Position geschlossen werden soll. Wenn für die Schließung eines Teils der Position ihr Symbol (nicht das Ticket) angegeben wurde, wird die erste nach dem Symbol gefundene Position ausgewählt, die die angegebene MagicNumber hat (Identifier des Expert Advisors). Aus diesem Grund ist es in einigen Fällen besser, PositionClosePartial() mit der Angabe des Tickets der Position zu verwenden.

volume

[in] Volumen, um welches die Position reduziert werden muss. Wenn dieser Wert größer als das Volumen des zu schließenden Teils ist, wird die Position komplett geschlossen. Es wird keine neue Position in die Gegenrichtung eröffnet.

ticket

[in] Ticket der zu schließenden Position.

deviation=ULONG_MAX

[in] Maximale Abweichung vom aktuellen Preis (in Punkten).

Rückgabewert

true - wenn die Basisprüfung von Strukturen erfolgreich ist, andernfalls false.

Hinweis

Wenn die Methode PositionClosePartial(...) erfolgreich beendet wurde, bedeutet dies nicht immer, dass ein Trade erfolgreich ausgeführt wurde. Man muss das Ergebnis der Ausführung einer Handelsanfrage (Rückgabewert des Handelsservers) durch den Aufruf der ResultRetcode() Methode überprüfen.

Im Netting-Modus (ACCOUNT_MARGIN_MODE_RETAIL_NETTING und ACCOUNT_MARGIN_MODE_EXCHANGE) kann nur eine Position pro Symbol vorhanden sein, die das Ergebnis eines oder mehrerer Abschlüsse darstellt. Man darf Positionen und aktuelle Pending Orders nicht verwechseln, die auch im Reiter "Handel" auf der Werkzeugleiste angezeigt werden.

© 2000-2025, MetaQuotes Ltd.

Im Hedging-Modus (ACCOUNT_MARGIN_MODE_RETAIL_HEDGING) können gleichzeitig mehrere Positionen auf einem Symbol eröffnet werden. In diesem Fall schließt PositionClose die Position mit dem kleinsten Ticket.

© 2000-2025, MetaQuotes Ltd.

PositionCloseBy

Schließt die Position mit dem angegebenen Ticket zur Gegenposition.

```
bool  PositionCloseBy(
   const ulong   ticket,        // das Ticket der Position
   const ulong   ticket_by      // das Ticket der Gegenposition
   )
```

Parameter

ticket

[in] das Ticket der zu schließenden Position.

ticket_by

[in] Das Ticket der Gegenposition, die zum Schließen verwendet wird.

Rückgabewert

true - wenn Strukturen erfolgreich überprüft wurden, anderenfalls - false.

Hinweis

Ein erfolgreiches Beenden der Methode PositionCloseBy(...) bedeutet nicht immer eine erfolgreiche Ausführung des Trades. Es ist notwendig, das Ergebnis der Ausführung von Trade Request (Return Code des Handelsservers) durch den Aufruf der Methode ResultRetcode() zu prüfen.

© 2000-2025, MetaQuotes Ltd.

Buy

Eröffnet eine Long-Position mit den angegebenen Parametern.

```
bool  Buy(
   double         volume,          // Volumen der Position
   const string   symbol=NULL,     // Symbol
   double         price=0.0,       // Ausführungspreis
   double         sl=0.0,          // Preis von Stop Loss
   double         tp=0.0,          // Preis von Take Profit
   const string   comment=""       // Kommentar
   )
```

Parameter

volume

[in] Das angefragte Volumen der Position.

symbol=NULL

[in] Name des Handelsinstruments, für das eine Position eröffnet wird. Wenn kein Symbol angegeben ist, dann wird die Position für das aktuellen Symbol geöffnet werden.

price=0.0

[in] Der Preis zu welchem die Position eröffnet werden soll. Wenn kein Preis angegeben ist, dann wird die Position zum aktuellen Ask-Preis geöffnet werden.

sl=0.0

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp=0.0

[in] Der Preis, zu dem Take Profit aktiviert wird.

comment=""

[in] Kommentar zur Position.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode Buy(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultDeal() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

Sell

Eröffnet eine Short-Position mit den angegebenen Parametern.

```
bool  Sell(
   double         volume,           // Volumen der Position
   const string   symbol=NULL,      // Symbol
   double         price=0.0,        // Ausführungspreis
   double         sl=0.0,           // Preis von Stop Loss
   double         tp=0.0,           // Preis von Take Profit
   const string   comment=""        // Kommentar
   )
```

Parameter

volume

[in] Das angefragte Volumen der Position.

symbol=NULL

[in] Name des Handelsinstruments, für das eine Position eröffnet wird. Wenn kein Symbol angegeben ist, dann wird die Position für das aktuellen Symbol geöffnet werden.

price=0.0

[in] Der Preis zu welchem die Position eröffnet werden soll. Wenn kein Preis angegeben ist, dann wird die Position zum aktuellen Bid-Preis geöffnet werden.

sl=0.0

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp=0.0

[in] Der Preis, zu dem Take Profit aktiviert wird.

comment=""

[in] Kommentar zur Position.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode Sell(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultDeal() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

BuyLimit

Stellt eine Pending-Kaufsorder aus zum Preis, der besser als Marktpreis ist.

```
bool  BuyLimit(
   double                volume,                    // Volumen der Position
   double                price,                     // Ausführungspreis
   const string          symbol=NULL,               // Symbol
   double                sl=0.0,                    // Preis von Stop Loss
   double                tp=0.0,                    // Preis von Take Profit
   ENUM_ORDER_TYPE_TIME  type_time=ORDER_TIME_GTC,  // Ablauftyp
   datetime              expiration=0,              // Ablaufzeit
   const string          comment=""                 // Kommentar
   )
```

Parameter

volume

[in] Das angefragte Volumen der Order.

price

[in] Der Preis zu welchem die Order ausgestellt werden soll.

symbol=NULL

[in] Name des Handelsinstruments, für das eine Order ausgestellt werden soll . Wenn kein Symbol angegeben ist, dann wird die Order für das aktuellen Symbol ausgestellt werden.

sl=0.0

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp=0.0

[in] Der Preis, zu dem Take Profit aktiviert wird.

type_time=ORDER_TIME_GTC

[in] Typ der Orderablauf aus der Enumeration ENUM_ORDER_TYPE_TIME.

expiration=0

[in] Ablaufzeit der Order (nur für type_time=ORDER_TIME_SPECIFIED).

comment=""

[in] Kommentar zur Order.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode BuyLimit(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultOrder() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

BuyStop

Stellt eine Pending-Kaufsorder aus zum Preis, der schlechter als Marktpreis ist.

```
bool  BuyStop(
   double                volume,                    // Volumen der Position
   double                price,                     // Ausführungspreis
   const string          symbol=NULL,               // Symbol
   double                sl=0.0,                    // Preis von Stop Loss
   double                tp=0.0,                    // Preis von Take Profit
   ENUM_ORDER_TYPE_TIME  type_time=ORDER_TIME_GTC,  // Ablauftyp
   datetime              expiration=0,              // Ablaufzeit
   const string          comment=""                 // Kommentar
   )
```

Parameter

volume

[in] Das angefragte Volumen der Order.

price

[in] Der Preis zu welchem die Order ausgestellt werden soll.

symbol=NULL

[in] Name des Handelsinstruments, für das eine Order ausgestellt werden soll . Wenn kein Symbol angegeben ist, dann wird die Order für das aktuellen Symbol ausgestellt werden.

sl=0.0

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp=0.0

[in] Der Preis, zu dem Take Profit aktiviert wird.

type_time=ORDER_TIME_GTC

[in] Typ der Orderablauf aus der Enumeration ENUM_ORDER_TYPE_TIME.

expiration=0

[in] Ablaufzeit der Order (nur für type_time=ORDER_TIME_SPECIFIED).

comment=""

[in] Kommentar zur Order.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode BuyStop(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultOrder() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

SellLimit

Stellt eine Pending-Verkaufsorder aus zum Preis, der besser als Marktpreis ist.

```
bool  SellLimit(
   double                volume,                    // Volumen der Position
   double                price,                     // Ausführungspreis
   const string          symbol=NULL,               // Symbol
   double                sl=0.0,                    // Preis von Stop Loss
   double                tp=0.0,                    // Preis von Take Profit
   ENUM_ORDER_TYPE_TIME  type_time=ORDER_TIME_GTC,  // Ablauftyp
   datetime              expiration=0,              // Ablaufzeit
   const string          comment=""                 // Kommentar
   )
```

Parameter

volume

[in] Das angefragte Volumen der Order.

price

[in] Der Preis zu welchem die Order ausgestellt werden soll.

symbol=NULL

[in] Name des Handelsinstruments, für das eine Order ausgestellt werden soll . Wenn kein Symbol angegeben ist, dann wird die Order für das aktuellen Symbol ausgestellt werden.

sl=0.0

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp=0.0

[in] Der Preis, zu dem Take Profit aktiviert wird.

type_time=ORDER_TIME_GTC

[in] Typ der Orderablauf aus der Enumeration ENUM_ORDER_TYPE_TIME.

expiration=0

[in] Ablaufzeit der Order (nur für type_time=ORDER_TIME_SPECIFIED).

comment=""

[in] Kommentar zur Order.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode SellLimit(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultOrder() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

SellStop

Stellt eine Pending-Verkaufsorder aus zum Preis, der schlechter als Marktpreis ist.

```
bool  SellStop(
   double                volume,                    // Volumen der Position
   double                price,                     // Ausführungspreis
   const string          symbol=NULL,               // Symbol
   double                sl=0.0,                    // Preis von Stop Loss
   double                tp=0.0,                    // Preis von Take Profit
   ENUM_ORDER_TYPE_TIME  type_time=ORDER_TIME_GTC,  // Ablauftyp
   datetime              expiration=0,              // Ablaufzeit
   const string          comment=""                 // Kommentar
   )
```

Parameter

volume

[in] Das angefragte Volumen der Order.

price

[in] Der Preis zu welchem die Order ausgestellt werden soll.

symbol=NULL

[in] Name des Handelsinstruments, für das eine Order ausgestellt werden soll . Wenn kein Symbol angegeben ist, dann wird die Order für das aktuellen Symbol ausgestellt werden.

sl=0.0

[in] Der Preis, zu dem Stop Loss aktiviert wird.

tp=0.0

[in] Der Preis, zu dem Take Profit aktiviert wird.

type_time=ORDER_TIME_GTC

[in] Typ der Orderablauf aus der Enumeration ENUM_ORDER_TYPE_TIME.

expiration=0

[in] Ablaufzeit der Order (nur für type_time=ORDER_TIME_SPECIFIED).

comment=""

[in] Kommentar zur Order.

Rückgabewert

Gibt bei bei der erfolgreichen Struktur-Prüfung true zurück, ansonsten false.

Hinweis

Der erfolgreiche Abschluss der Methode SellStop(...) bedeutet nicht immer eine erfolgreiche Ausführung der Transaktion. Es ist notwendig, das Ergebnis einer Handelsanfrage zu prüfen (Rückgabecode des Handelsservers) durch den Aufruf von ResultRetcode(), und auch den durch ResultOrder() zurückgegebenen Wert.

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Request

Erhält eine Kopie der Struktur der letzten Anfrage.

```
void  Request(
   MqlTradeRequest&  request      // Referenz
   ) const
```

Parameter

request

[out] Referenz auf die Struktur für Kopieren.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

RequestAction

Erhält den Typ der Transaktion.

```
ENUM_TRADE_REQUEST_ACTIONS  RequestAction() const
```

Rückgabewert

Typ der Handelsoperation in der letzten Abfrage aus der Enumeration ENUM_TRADE_REQUEST_ACTIONS.

© 2000-2025, MetaQuotes Ltd.

RequestActionDescription

Erhält den Typ der Transaktion als String.

```
string  RequestActionDescription()) const
```

Rückgabewert

Typ der Handelsoperation in der letzten Abfrage als String.

© 2000-2025, MetaQuotes Ltd.

RequestMagic

Erhält die ID des Expert Advisors.

```
ulong  RequestMagic() const
```

Rückgabewert

ID des Expert Advisors, die in der letzten Abfrage verwendet war.

© 2000-2025, MetaQuotes Ltd.

RequestOrder

Erhält Ticket einer Order.

```
ulong  RequestOrder() const
```

Rückgabewert

Ticket der Order, der in der letzten Abfrage verwendet war.

© 2000-2025, MetaQuotes Ltd.

RequestSymbol

Erhält den Namen des Handelsinstruments.

```
string  RequestSymbol() const
```

Rückgabewert

Der Name des Handelsinstruments, für das die Order ausgestellt ist, der in der letzte Abfrage verwendet war.

© 2000-2025, MetaQuotes Ltd.

RequestVolume

Erhält das angefragten Volumen des Deals in Lots.

```
double  RequestVolume() const
```

Rückgabewert

Abgefragtes Deal-Volumen in Lots, das in der letzten Abfrage verwendet war.

© 2000-2025, MetaQuotes Ltd.

RequestPrice

Erhält den Preis zu dem die Order ausgeführt werden soll.

```
double  RequestPrice() const
```

Rückgabewert

Preis aus der letzten Abfrage, zu dem die Order ausgeführt werden soll.

© 2000-2025, MetaQuotes Ltd.

RequestStopLimit

Erhält den Preis, zu dem eine Pending-Stop-Limit-Order ausgestellt wird.

```
double  RequestStopLimit() const
```

Rückgabewert

Preis aus der letzten Abfrage, zu dem eine Stop Limit Pending-Order ausgestellt werden soll.

© 2000-2025, MetaQuotes Ltd.

RequestSL

Erhält den Preis, zu dem Stop Loss aktiviert wird.

```
double  RequestSL() const
```

Rückgabewert

Preis aus der letzten Abfrage, zu dem Stop Loss aktiviert werden soll.

© 2000-2025, MetaQuotes Ltd.

RequestTP

Erhält den Preis, zu dem Take Profit aktiviert wird.

```
double  RequestTP() const
```

Rückgabewert

Preis aus der letzten Abfrage, zu dem Take Profit aktiviert werden soll.

© 2000-2025, MetaQuotes Ltd.

RequestDeviation

Erhält die maximal zulässige Abweichung von der Preisvorstellung.

```
ulong  RequestDeviation() const
```

Rückgabewert

Die maximal zulässige Abweichung von der Preisvorstellung, die in der letzten Abfrage verwendet war

© 2000-2025, MetaQuotes Ltd.

RequestType

Erhält den Typ der Order.

```
ENUM_ORDER_TYPE  RequestType() const
```

Rückgabewert

Typ der Order in der letzten Abfrage aus der Enumeration ENUM_ORDER_TYPE.

© 2000-2025, MetaQuotes Ltd.

RequestTypeTimeDescription

Erhält den Typ der Order nach Ablauf als String.

```
string  RequestTypeTimeDescription() const
```

Rückgabewert

Typ der Order nach Ablauf in der letzten Abfrage als String.

© 2000-2025, MetaQuotes Ltd.

RequestTypeFilling

Erhält den Orderausführungstyp.

```
ENUM_ORDER_TYPE_FILLING  RequestTypeFilling() const
```

Rückgabewert

Orderausführungstyp (aus der Enumeration ENUM_ORDER_TYPE_FILLING), der in der letzten Abfrage verwendet war.

© 2000-2025, MetaQuotes Ltd.

RequestTypeFillingDescription

Erhält den Orderausführungstyp als String.

```
string  RequestTypeFillingDescription() const
```

Rückgabewert

Orderausführungstyp in der letzten Abfrage als String.

© 2000-2025, MetaQuotes Ltd.

RequestTypeTime

Erhält den Typ der Order nach Ablauf.

```
ENUM_ORDER_TYPE_TIME  RequestTypeTime() const
```

Rückgabewert

Typ der Order nach Ablauf (aus der Enumeration ENUM_ORDER_TYPE_TIME), der in der letzten Abfrage verwendet war.

© 2000-2025, MetaQuotes Ltd.

RequestTypeTimeDescription

Erhält den Typ der Order nach Ablauf als String.

```
string  RequestTypeTimeDescription() const
```

Rückgabewert

Typ der Order nach Ablauf in der letzten Abfrage als String.

© 2000-2025, MetaQuotes Ltd.

RequestExpiration

Erhält die Ablaufzeit der Pending-Order.

```
datetime  RequestExpiration() const
```

Rückgabewert

Ablaufzeit der Pending-Order in der letzten Abfrage.

© 2000-2025, MetaQuotes Ltd.

RequestComment

Erhält den Kommentar zur Order.

```
string  RequestComment() const
```

Rückgabewert

Kommentar an die Order in der letzten Abfrage.

© 2000-2025, MetaQuotes Ltd.

RequestPosition

Erhält das Ticket der Position.

```
ulong  RequestPosition() const
```

Rückgabewert

Das Ticket der Position, die beim letzten Abruf verwendet wurde.

© 2000-2025, MetaQuotes Ltd.

RequestPositionBy

Erhält das Ticket der Gegenposition.

```
ulong  RequestPositionBy() const
```

Rückgabewert

Das Ticket der Gegenposition , die beim letzten Abruf verwendet wurde.

© 2000-2025, MetaQuotes Ltd.

Result

Erhält eine Kopie der Struktur der Ausführungsergebnisse der letzten Anfrage.

```
void  Result(
   MqlTradeResult&  result      // Referenz
   ) const
```

Parameter

result

[out] Referenz auf die Struktur vom Typ MqlTradeResult für Kopieren.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

ResultRetcode

Erhält den Code der Abfrageausführungsergebnisse.

```
uint  ResultRetcode() const
```

Rückgabewert

Ergebniscode der letzten Abfrage.

© 2000-2025, MetaQuotes Ltd.

ResultRetcodeDescription

Erhält den Code der Abfrageausführungsergebnisse als String.

```
string  ResultRetcodeDescription() const
```

Rückgabewert

Gibt den String mit der Beschreibung des Ergebnisses der letzten Abfrage zurück.

© 2000-2025, MetaQuotes Ltd.

ResultDeal

Erhält Ticket des Deals.

```
ulong  ResultDeal() const
```

Rückgabewert

Ticket des Deals, wenn er ausgeführt ist.

© 2000-2025, MetaQuotes Ltd.

ResultOrder

Erhält Ticket einer Order.

```
ulong  ResultOrder() const
```

Rückgabewert

Ticket der Order, wenn sie ausgeführt ist.

© 2000-2025, MetaQuotes Ltd.

ResultVolume

Erhält das Volumen des Deals oder der Order.

```
double  ResultVolume() const
```

Rückgabewert

Das durch Broker bestätigte Volumen des Deals oder der Order.

© 2000-2025, MetaQuotes Ltd.

ResultPrice

Erhält den durch Broker bestätigten Preis.

```
double  ResultPrice() const
```

Rückgabewert

Der durch Broker bestätigte Preis.

© 2000-2025, MetaQuotes Ltd.

ResultBid

Erhält den aktuellen Bid-Preis (Requote).

```
double  ResultBid() const
```

Rückgabewert

Der aktuelle Bid-Marktpreis (Preis von Requote).

© 2000-2025, MetaQuotes Ltd.

ResultAsk

Erhält den aktuellen Ask-Preis (Requote).

```
double  ResultAsk() const
```

Rückgabewert

Der aktuelle Ask-Marktpreis (Preis von Requote).

© 2000-2025, MetaQuotes Ltd.

ResultComment

Erhält den Kommentar des Brokers.

```
string  ResultComment() const
```

Rückgabewert

Kommentar des Brokers zum Deal.

© 2000-2025, MetaQuotes Ltd.

CheckResult

Erhält eine Kopie der Struktur der Prüfergebnisse der letzten Anfrage.

```
void  CheckResult(
   MqlTradeCheckResult&  check_result      // Referenz
   ) const
```

Parameter

check_result

[out] Referenz auf die Struktur vom Typ MqlTradeCheckResult für Kopieren.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CheckResultRetcode

Gibt den Wert des retcode-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
uint  CheckResultRetcode() const
```

Rückgabewert

Den Wert des retcode-Felds (Fehlercode) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultRetcodeDescription

Gibt die String-Beschreibung des retcode-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
string  ResultRetcodeDescription() const
```

Rückgabewert

Die String-Beschreibung des retcode-Felds (Fehlercode) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultBalance

Gibt den Wert des balance-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
double  CheckResultBalance() const
```

Rückgabewert

Der Wert des balance-Felds (Kontostand, der nach der Transaktion erwartet ist) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultEquity

Gibt den Wert des balance-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
double  CheckResultEquity() const
```

Rückgabewert

Der Wert des equity-Felds (Erwartete Equity nach der Transaktion) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultProfit

Erhält den Wert des schwebenden Profits.

```
double  CheckResultProfit() const
```

Rückgabewert

dER Wert des schwebenden Profits, der nach der Transaktion erwartet ist.

© 2000-2025, MetaQuotes Ltd.

CheckResultMargin

Gibt den Wert des margin-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
double  CheckResultMargin() const
```

Rückgabewert

Der Wert des margin-Felds (Margin, die für die Transaktion erforderlich ist) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultMarginFree

Gibt den Wert des margin_free-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
double  CheckResultMarginFree() const
```

Rückgabewert

Der Wert des margin_free-Felds (Erwartete Equity, das nach der Ausführung erforderlicher Transaktion bleiben wird) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultMarginLevel

Gibt den Wert des margin_level-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
double  CheckResultMarginLevel() const
```

Rückgabewert

Der Wert des margin_level-Felds (Margin-Ebene, die nach der Ausführung erforderlichen Transaktion bleibt) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

CheckResultComment

Gibt den Wert des comment-Felds der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

```
string  CheckResultComment() const
```

Rückgabewert

Der Wert des comment-Felds (Kommentar zum Antwortcode, Fehlerbeschreibung) der MqlTradeCheckResult-Struktur zurück, die während der Abfrageüberprüfung gefüllt war.

© 2000-2025, MetaQuotes Ltd.

PrintRequest

Schreibt die Parameter der letzten Anfrage ins Journal.

```
void  PrintRequest() const
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

PrintResult

Schreibt die Ausführungsergebnisse der letzten Anfrage ins Journal.

```
void  PrintResult() const
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

FormatRequest

Schreibt die Parameter der letzten Anfrage in String.

```
string  FormatRequest(
   string&                 str,          // String
   const MqlTradeRequest&  request       // Abfrage
   ) const
```

Parameter

str

[in] Die Referenz an String um die Ergebnisse zu platzieren.

request

[in] Referenz auf die Struktur vom Typ MqlTradeRequest, die Informationen über die letzte Anfrage enthält.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

FormatRequestResult

Schreibt die Ausführungsergebnisse der letzten Anfrage in String.

```
string  FormatRequestResult(
   string&                 str,          // String
   const MqlTradeRequest&  request,      // Abfrage
   const MqlTradeResult&   result        // Ergebnis
   ) const
```

Parameter

str

[in] Die Referenz an String um die Ergebnisse zu platzieren.

request

[in] Referenz auf die Struktur vom Typ MqlTradeRequest, die Informationen über die letzte Anfrage enthält.

result

[in] Referenz auf die Struktur vom Typ MqlTradeResult, die Informationen über die Ausführung der letzten Anfrage enthält.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Klasse CTerminalInfo

CTerminalInfo ist eine Klasse für den vereinfachten Zugriff auf die Eigenschaften der Umgebung eines MQL5-Programms.

Beschreibung

CTerminalInfo bietet den vereinfachten Zugriff auf die Eigenschaften der Umgebung eines MQL5-Programms.

Deklaration

```
class CTerminalInfo : public CObject
```

Kopf

```
#include <Trade\TerminalInfo.mqh>
```

Vererbungshierarchie

CObject

CTerminalInfo

Gruppen der Klassenmethode

Methoden der Zugriff auf die Terminal-Parameter vom Typ integer	
Build	Erhält die Buildnummer der Terminals
IsConnected	Erhält Informationen über die Verfügbarkeit der Verbindung an den Handelsserver
IsDLLsAllowed	Erhält Informationen über die Erlaubnis DLL zu verwenden
IsTradeAllowed	Erhält Informationen über die Handelserlaubnis
IsEmailEnabled	Erhält Informationen über die Erlaubnis E-Mails mit dem SMTP-Server und in Terminal-Einstellungen eingegebenem Login zu senden
IsFtpEnabled	Erhält Informationen über die Erlaubnis Berichte über FTP für in Terminal-Einstellungen eingegebenen FTP-Server und Login zu senden
MaxBars	Erhält die maximale Anzahl der Balken auf dem Chart
CodePage	Erhält Informationen über die Nummer der Codepage der im Terminal eingestellten Sprache
CPUCores	Erhält Informationen über die Anzahl der im System installierten Prozessorkerne

© 2000-2025, MetaQuotes Ltd.

Methoden der Zugriff auf die Terminal-Parameter vom Typ integer	
MemoryPhysical	Erhält Informationen über die Größe des physischen Speichers im System (in MB)
MemoryTotal	Erhält Informationen über die Größe des Speichers, das dem Prozess von Terminal/Agent verfügbar ist (in MB)
MemoryAvailable	Erhält Informationen über die Größe des freien Speichers, das dem Prozess von Terminal/Agent verfügbar ist (in MB)
MemoryUsed	Erhält Informationen über die Größe des Speichers, das von Terminal/Agent besetzt ist (in MB)
IsX64	Erhält Informationen über den Typ des Client-Terminals/Agenten
OpenCLSupport	Erhält Informationen über die vom Grafiken unterstützten OpenCL-Version
DiskSpace	Erhält Informationen über die Größe des freien Festplattenspeichers
Methoden der Zugriff auf die Terminal-Parameter vom Typ string	
Language	Erhält Informationen über die Sprache des Client-Terminals
Name	Erhält Informationen über den Namen des Terminals
Company	Erhält Informationen über den Namen des Unternehmens
Path	Erhält Informationen über den Ordner, in dem das Terminal gestartet ist
DataPath	Erhält Informationen über den Ordner, in dem Terminaldaten gespeichert sind
CommonDataPath	Erhält Informationen über den Datenordner aller auf dem Computer installierten Terminals
Zugriff auf Funktionen API MQL5	
InfoInteger	Erhält Informationen über den integer-Parameter des Terminals
InfoString	Erhält Informationen über den string-Parameter des Terminals

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Build

Erhält die Buildnummer der Terminals.

```
int  CBuild() const
```

Rückgabewert

Die Buildnummer der Terminals.

Hinweis

Die Buildnummer des Terminals wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_BUILD) bestimmt.

© 2000-2025, MetaQuotes Ltd.

IsConnected

Erhält Informationen über die Verfügbarkeit der Verbindung an den Handelsserver.

```
bool  IsConnected() const
```

Rückgabewert

true - wenn das Terminal an den Server verbunden ist, ansonsten false.

Hinweis

Die Verbindung an den Handelsserver wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_CONNECTED) bestimmt.

© 2000-2025, MetaQuotes Ltd.

IsDLLsAllowed

Erhält Informationen über die Erlaubnis DLL zu verwenden.

```
bool  IsDLLsAllowed() const
```

Rückgabewert

true - wenn DLL erlaubt sind, ansonsten false.

Hinweis

DLL-Erlaubnis wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_DLLS_ALLOWED) bestimmt.

© 2000-2025, MetaQuotes Ltd.

IsTradeAllowed

Erhält Informationen über die Handelserlaubnis.

```
bool  IsTradeAllowed() const
```

Rückgabewert

true - wenn der Handel erlaubt ist, ansonsten false.

Hinweis

Handelserlaubnis wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_TRADE_ALLOWED) bestimmt.

© 2000-2025, MetaQuotes Ltd.

IsEmailEnabled

Erhält Informationen über die Erlaubnis E-Mails mit dem SMTP-Server und in Terminal-Einstellungen eingegebenem Login zu senden.

```
bool  IsEmailEnabled() const
```

Rückgabewert

true - wenn Senden von E-Mails erlaubt ist, ansonsten false.

Hinweis

E-Mail-Erlaubnis wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_EMAIL_ENABLED) bestimmt.

© 2000-2025, MetaQuotes Ltd.

IsFtpEnabled

Erhält Informationen über die Erlaubnis Berichte über FTP für in Terminal-Einstellungen eingegebenen FTP-Server und Login zu senden.

```
bool  IsFtpEnabled() const
```

Rückgabewert

true - wenn Senden von Berichte über FTP erlaubt ist, ansonsten false.

Hinweis

Erlaubnis vom Bericht-Senden wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_FTP_ENABLED) bestimmt.

© 2000-2025, MetaQuotes Ltd.

MaxBars

Erhält die in Terminal-Einstellungen angegebenen maximale Anzahl der Balken auf dem Chart.

```
int  MaxBars() const
```

Rückgabewert

Die maximale Anzahl der Balken auf dem Chart

Hinweis

Die maximale Anzahl der Balken auf dem Chart wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_MAXBARS) bestimmt.

© 2000-2025, MetaQuotes Ltd.

CodePage

Erhält Informationen über die Nummer der Codepage der im Terminal eingestellten Sprache.

```
int  CodePage() const
```

Rückgabewert

Die Nummer der Codepage der im Terminal eingestellten Sprache.

Hinweis

Die Nummer der Codepage der im Terminal eingestellten Sprache wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_CODEPAGE) bestimmt.

© 2000-2025, MetaQuotes Ltd.

CPUCores

Erhält Informationen über die Anzahl der im System installierten Prozessorkerne.

```
int  CPUCores() const
```

Rückgabewert

Die Anzahl der Prozessorkerne.

Hinweis

Die Anzahl der Prozessorkerne wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_CPU_CORES) bestimmt.

© 2000-2025, MetaQuotes Ltd.

MemoryPhysical

Erhält Informationen über die Größe des physischen Speichers im System (in MB).

```
int  MemoryPhysical() const
```

Rückgabewert

Die Größe des physischen Speichers.

Hinweis

Die Größe des physischen Speichers wird durch die Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_MEMORY_PHYSICAL) bestimmt.

© 2000-2025, MetaQuotes Ltd.

MemoryTotal

Erhält Informationen über die Größe des Speichers, das dem Prozess von Terminal/Agent verfügbar ist (in MB).

```
int  MemoryTotal() const
```

Rückgabewert

Die Größe des dem Prozess des Terminals (Agenten) verfügbaren Speichers.

Hinweis

Die Größe des Speichers, das dem Prozess des Terminals (Agenten) verfügbar ist, wird durch die Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_MEMORY_TOTAL) bestimmt.

© 2000-2025, MetaQuotes Ltd.

MemoryAvailable

Erhält Informationen über die Größe des freien Speichers, das dem Prozess von Terminal/Agent verfügbar ist (in MB).

```
int  MemoryTotal() const
```

Rückgabewert

Die Größe des dem Terminal verfügbaren freien Speichers.

Hinweis

Die Größe des freien Speichers, das dem Terminal verfügbar ist, wird durch die Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_MEMORY_TOTAL) bestimmt.

© 2000-2025, MetaQuotes Ltd.

MemoryUsed

Erhält Informationen über die Größe des Speichers, das von Terminal/Agent besetzt ist (in MB).

```
int  MemoryUsed() const
```

Rückgabewert

Die Größe des Speichers, das von Terminal/Agent besetzt ist.

Hinweis

Die Größe des Speichers, das vom Terminal (Agent) besetzt ist, wird durch die Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_MEMORY_USED) bestimmt.

© 2000-2025, MetaQuotes Ltd.

IsX64

Erhält Informationen über den Typ des Client-Terminals/Agenten.

```
bool  IsX64() const
```

Rückgabewert

true - wenn 64-Bit-Version des Terminals verwendet wird, ansonsten false.

Hinweis

Terminaltyp wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_X64) bestimmt.

© 2000-2025, MetaQuotes Ltd.

OpenCLSupport

Erhält Informationen über die vom Grafiken unterstützten OpenCL-Version.

```
int  OpenCLSupport() const
```

Rückgabewert

OpenCL-Version wie 0x00010002 = "1.2". Ein Wert von 0 bedeutet, dass OpenCL nicht unterstützt ist.

Hinweis

OpenCL-Version wird mit der Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_OPENCL_SUPPORT) bestimmt.

© 2000-2025, MetaQuotes Ltd.

DiskSpace

Erhält Informationen über die Größe des freien Festplattenspeichers für den Ordner MQL5\Files von Terminal/Agent (in MB).

```
int  MDiskSpace() const
```

Rückgabewert

Die Größe des freien Festplattenspeichers.

Hinweis

Die Größe des freien Festplattenspeichers wird durch die Funktion TerminalInfoInteger() (Eigenschaft TERMINAL_DISK_SPACE) bestimmt.

© 2000-2025, MetaQuotes Ltd.

Language

Erhält Informationen über die Sprache des Client-Terminals.

```
string  Language() const
```

Rückgabewert

Die Sprache des Client-Terminals.

Hinweis

Terminalsprache wird mit der Funktion TerminalInfoString() (Eigenschaft TERMINAL_LANGUAGE) bestimmt.

© 2000-2025, MetaQuotes Ltd.

Name

Erhält Informationen über den Namen des Client-Terminals.

```
string  Name() const
```

Rückgabewert

Der Name des Client-Terminals.

Hinweis

Der Name des Terminals wird mit der Funktion TerminalInfoString() (Eigenschaft TERMINAL_NAME) bestimmt.

© 2000-2025, MetaQuotes Ltd.

Company

Erhält Informationen über den Namen des Unternehmens.

```
string  Company() const
```

Rückgabewert

Der Name des Unternehmens.

Hinweis

Der Name des Unternehmens wird mit der Funktion TerminalInfoString() (Eigenschaft TERMINAL_COMPANY) bestimmt.

© 2000-2025, MetaQuotes Ltd.

Path

Erhält Informationen über den Ordner, in dem das Terminal gestartet ist.

```
string  Path() const
```

Rückgabewert

Ordner des Client-Terminals.

Hinweis

Der Ordner des Client-Terminals wird durch die Funktion TerminalInfoString() (Eigenschaft TERMINAL_PATH) bestimmt.

© 2000-2025, MetaQuotes Ltd.

DataPath

Erhält Informationen über den Ordner, in dem Terminaldaten gespeichert sind.

```
string  DataPath() const
```

Rückgabewert

Datenordner des Client-Terminals.

Hinweis

Der Datenordner des Client-Terminals wird durch die Funktion TerminalInfoString() (Eigenschaft TERMINAL_DATA_PATH) bestimmt.

© 2000-2025, MetaQuotes Ltd.

CommonDataPath

Erhält Informationen über den Datenordner aller auf dem Computer installierten Terminals.

```
string  CommonDataPath() const
```

Rückgabewert

Der gemeinsamer Ordner aller installierten Terminals.

Hinweis

Der gemeinsamer Ordner aller installierten Terminals wird mit der Funktion TerminalInfoString() (Eigenschaft COMMON_DATA_PATH) bestimmt.

© 2000-2025, MetaQuotes Ltd.

InfoInteger

Gibt den Wert der entsprechenden integer-Eigenschaft der Umgebung des MQL5-Programms zurück.

```
int  InfoInteger(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einen der Werte der Enumeration ENUM_TERMINAL_INFO_INTEGER sein.

Rückgabewert

Wert vom Typ int.

Hinweis

Der Wert der Eigenschaft wird mit der Funktion TerminalInfoInteger()bestimmt.

© 2000-2025, MetaQuotes Ltd.

InfoString

Gibt den Wert der entsprechenden Eigenschaft der Umgebung des MQL5-Programms zurück. Die Eigenschaft muss ein string-Wert sein.

```
string  InfoString(
   int  property_id      // Identifikator der Eigenschaft
   );
```

Parameter

property_id

[in] Identifikator der Eigenschaft. Kann einen der Werte der Enumeration ENUM_TERMINAL_INFO_STRING sein.

Rückgabewert

Wert vom Typ string.

Hinweis

Der Wert der Eigenschaft wird mit der Funktion TerminalInfoString()bestimmt.

© 2000-2025, MetaQuotes Ltd.

Eine Reihe von Klassen zum Erstellen und Testen von Handelsstrategien

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen für die Erstellung und Testen von Handelsstrategien und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen für die Erstellung und Testen von Handelsstrategien sparen Sie Zeit bei der Entwicklung (und vor allem beim Testen) von Handelsstrategien.

Die Standardbibliothek MQL5 (in Bezug auf die Klassen für die Erstellung und Testen von Handelsstrategien) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Expert.

Basisklassen von Expert Advisors	Beschreibung
CExpertBase	Die Basisklasse für die Klassen von Handelsstrategien
CExpert	Die Basisklasse der Handelsstrategie
CExpertSignal	Die Basisklasse für die Erstellung Handelsstrategiengeneratoren
CExpertTrailing	Die Basisklasse für die Steuerung von offenen Positionen
CExpertMoney	Die Basisklasse für die Implementierung von Algorithmen für Risiko- und Geldmanagement.

Klassen von Handelssignalen	Beschreibung
CSignalAC	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Accelerator Oszillator.
CSignalAMA	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Adaptive Moving Average.
CSignalAO	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Awesome Oszillator.
CSignalBearsPower	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Bears Power.
CSignalBullsPower	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Bulls Power.
CSignalCCI	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Commodity Channel Index.
CSignalDeM	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators DeMarker.
CSignalDEMA	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Double Exponential Moving Average.
CSignalEnvelopes	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Envelopes.

© 2000-2025, MetaQuotes Ltd.

Klassen von Handelssignalen	Beschreibung
CSignalFrAMA	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Fractal Adaptive Moving Average.
CSignalITF	Das Modul von Zeit-Filterung der Signale.
CSignalMACD	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators MACD.
CSignalMA	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Moving Average.
CSignalSAR	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Parabolic SAR.
CSignalRSI	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Relative Strength Index.
CSignalRVI	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Relative Vigor Index.
CSignalStoch	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Stochastic.
CSignalTRIX	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Triple Exponential Average.
CSignalTEMA	Das Modul von Signalen auf Basis der Marktmodelle des Indikators Tripple Exponential Moving Average.
CSignalWPR	Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Williams Percent Range.

Klassen für die Steuerung von offenen Positionen	Beschreibung
CTrailingFixedPips	Die Klasse implementiert einen Algorithmus für Steuerung offenen Positionen in einer festgelegten "Distanz"
CTrailingMA	Die Klasse implementiert einen Algorithmus für Steuerung offenen Positionen aufgrund den Werten vom gleitenden Durchschnitt
CTrailingNone	Stub-Klasse von Algorithmen für Steuerung offenen Positionen
CTrailingPSAR	Die Klasse implementiert einen Algorithmus für Steuerung offenen Positionen aufgrund den Werten des Indikators Parabolic SAR

Klasse für Risiko- und Geldmanagement.	Beschreibung
CMoneyFixedLot	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Lot, das beim Einrichten angegeben war
CMoneyFixedMargin	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Marginniveau, das beim Einrichten

© 2000-2025, MetaQuotes Ltd.

Klasse für Risiko- und Geldmanagement.	Beschreibung
	angegeben war
CMoneyFixedRisk	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Risikowert, der beim Einrichten angegeben war
CMoneyNone	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem minimalen Lot
CMoneySizeOptimized	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit dem Volumen, das durch früheren Transaktionen definiert wird

© 2000-2025, MetaQuotes Ltd.

Basisklassen für die Erstellung von Expert Advisors

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen für die Erstellung und Testen von Handelsstrategien und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen für die Erstellung und Testen von Handelsstrategien sparen Sie Zeit bei der Entwicklung (und vor allem beim Testen) von Handelsstrategien.

Die Standardbibliothek MQL5 (in Bezug auf die Klassen für die Erstellung und Testen von Handelsstrategien) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Expert.

Klasse	Beschreibung
CExpertBase	Die Basisklasse für die Klassen von Handelsstrategien
CExpert	Die Basisklasse für die Implementierung von Handelsstrategien
CExpertSignal	Die Basisklasse für die Erstellung Handelsstrategiengeneratoren
CExpertTrailing	Die Basisklasse für die Implementierung von Algorithmen für Positionensteuerung
CExpertMoney	Die Basisklasse für die Implementierung von Algorithmen für Risiko- und Geldmanagement.

© 2000-2025, MetaQuotes Ltd.

Klasse CExpertBase

CExpertBase ist eine Basisklasse für die CExpert-Klasse und alle Hilfsklassen von Handelsstrategien.

Beschreibung

Klasse CExpert bietet eine Satz von Daten und Methoden, die allgemein für alle Komponenten eines Expert Advisor sind.

Deklaration

```
class CExpertBase : public CObject
```

Kopf

```
#include <Expert\ExpertBase.mqh>
```

Vererbungshierarchie

CObject

CExpertBase

Direkte Ableitungen

CExpert, CExpertMoney, CExpertSignal, CExpertTrailing

Gruppen der Klassenmethode

Öffentliche Methoden:

Initialisierung	
virtual Init	Initialisiert ein Objekt
virtual ValidationSettings	Überprüft die Einstellungen
Einstellung	
Symbol	Setzt den Arbeitssymbol eines Objekts
Period	Setzt die Arbeitsrahmen eines Objekts
Magic	Setzt die Identifikator (Magic) von Expert Advisor
Methoden für die Erstellung von Indikatoren und Zeitreihen	
virtual SetPriceSeries	Setzt Zeiger auf die externe Preisobjekte-Zeitreihen.
virtual SetOtherSeries	Setzt Zeiger auf die externe Nicht-Preis-Objekte-Zeitreihen
virtual InitIndicators	Initialisiert die notwendigen Indikatoren und Zeitreihen
Zugriff auf geschützte Daten	

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
InitPhase	Erhält die aktuelle Phase der Objektinitialisierung
TrendType	Setzt die Identifikator des Trends
UsedSeries	Erhält eine Liste der verwendeten Zeitreihen
EveryTick	Setzt ein Flag von Analyse des aktuellen Balkens
Zugriff auf Zeitrahmen	
Open	Erhält den Wert eines Elements von der Open-Zeitreihe am angegebenen Index
High	Erhält den Wert eines Elements von der High-Zeitreihe am angegebenen Index
Low	Erhält den Wert eines Elements von der Low-Zeitreihe am angegebenen Index
Close	Erhält den Wert eines Elements von der Close-Zeitreihe am angegebenen Index
Spread	Erhält den Wert eines Elements von der Spread-Zeitreihe am angegebenen Index
Time	Erhält den Wert eines Elements von der Time-Zeitreihe am angegebenen Index
TickVolume	Erhält den Wert eines Elements von der TickVolume-Zeitreihe am angegebenen Index
RealVolume	Erhält den Wert eines Elements von der RealVolume-Zeitreihe am angegebenen Index

Geschützte Methoden:

Initialisierung von Zeitreihen	
InitOpen	Initialisiert die Open-Zeitreihe
InitHigh	Initialisiert die High-Zeitreihe
InitLow	Initialisiert die Low-Zeitreihe
InitClose	Initialisiert die Close-Zeitreihe
InitSpread	Initialisiert die Spread-Zeitreihe
InitTime	Initialisiert die Time-Zeitreihe
InitTickVolume	Initialisiert die TickVolume-Zeitreihe
InitRealVolume	Initialisiert die RealVolume-Zeitreihe
Dienstmethoden	

© 2000-2025, MetaQuotes Ltd.

Initialisierung von Zeitreihen	
virtual PriceLevelUnit	Erhält die Einheit von Preisniveaus
virtual StartIndex	Erhält den Index des ersten analysierten Balkens
virtual CompareMagic	Vergleicht den Wert der Identifikator (magic) mit dem angegebenen Wert

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

InitPhase

Erhält die aktuelle Phase der Objektinitialisierung.

```
ENUM_INIT_PHASE  InitPhase()
```

Rückgabewert

Die aktuelle Phase der Objektinitialisierung.

Hinweis

Objektinitialisierung besteht aus mehreren Phasen (Schritte). Wir nennen sie bedingt:

1. Die Phase der Anfangsinitialisierung.

   - Anfang - nach dem Ende der Arbeit des Konstruktors
   - Ende - nach erfolgreicher Durchführung der Methode Init(...)
   - erlaubt - Aufruf der Methode Init(...)
   - verboten - Aufruf aller anderen Initialisierungsmethoden und Methode ValidationSettings(...)

2. Die Phase von Einstellungen. In dieser Phase müssen alle Parameter des Objekts, die für Erstellung von Indikatoren verwendet werden,festgelegt werden.

   - Anfang - nach erfolgreicher Durchführung der Methode Init(...)
   - Ende - nach erfolgreicher Durchführung der Methode ValidationSettings(...)
   - erlaubt - Aufrufe der Methoden Symbol(...) и Period(...)
   - verboten - Aufrufe von Methoden Init(...), SetPriceSeries(...), SetOtherSeries(...) und InitIndicators(...)

3. Die Phase von Parameter-Prüfung.

   - Anfang - nach erfolgreicher Durchführung der Methode ValidationSettings()
   - Ende - nach erfolgreicher Durchführung der Methode InitIndicators(...)
   - erlaubt - Aufrufe der Methoden Symbol(...), Period(...) und InitIndicators(...)
   - verboten - Aufruf anderen Initialisierungsmethoden

4. Die Phase der Initialisierungsende.

   - Anfang - nach erfolgreicher Durchführung der Methode InitIndicators(...)
   - verboten - Aufrufen von Initialisierungsmethoden

© 2000-2025, MetaQuotes Ltd.

TrendType

Setzt die Identifikator des Trends.

```
void  TrendType(
   M_TYPE_TREND    value         // Wert
   )
```

Parameter

value

[in] Ein neuer Wert der Trend-Identifikator.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

UsedSeries

Erhält eine Liste der verwendeten Zeitreihen.

```
int  UsedSeries()
```

Rückgabewert

Eine Liste der verwendeten Zeitreihen in einer Bitkette.

Hinweis

Wenn das Bit gesetzt ist, wird die entsprechende Zeitreihe verwendet, wenn es rückgesetzt ist - nicht verwendet.

Entsprechen Bit-Zeitreihe:

Bit 0 - Zeitreihe Open,
Bit 1 - Zeitreihe High,
Bit 2 - Zeitreihe Low,
Bit 3 - Zeitreihe Close,
Bit 4 - Zeitreihe Spread,
Bit 5 - Zeitreihe Time,
Bit 6 - Zeitreihe TickVolume,
Bit 7 - Zeitreihe RealVolume.

© 2000-2025, MetaQuotes Ltd.

EveryTick

Setzt ein Flag von Analyse des aktuellen Balkens.

```
void  EveryTick(
   bool     value          // Flag
   )
```

Parameter

value

[in] Ein neuer Flagwert.

Rückgabewert

Nichts.

Hinweis

Wenn das Flag gesetzt ist, wird jede Preisveränderung (Tick) des Arbeitssymbols verarbeitet.

Wenn das Flag gelöscht ist, wird die Verarbeitung nur auf die Bildung eines neuen Balkens für das Arbeitssymbol auf Arbeitszeitrahmen durchgeführt.

© 2000-2025, MetaQuotes Ltd.

Open

Erhält den Wert eines Elements von der Open-Zeitreihe am angegebenen Index.

```
double  Open(
   int     ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der Open-Zeitreihe am angegebenen Index beim Erfolg, sonst EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

High

Erhält den Wert eines Elements von der High-Zeitreihe am angegebenen Index.

```
double  High(
   int     ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der High-Zeitreihe am angegebenen Index beim Erfolg, sonst EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

Low

Erhält den Wert eines Elements von der Low-Zeitreihe am angegebenen Index.

```
double  Low(
   int     ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der Low-Zeitreihe am angegebenen Index beim Erfolg, sonst EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

Close

Erhält den Wert eines Elements von der Close-Zeitreihe am angegebenen Index.

```
double  Close(
   int    ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der Close-Zeitreihe am angegebenen Index beim Erfolg, sonst EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

Spread

Erhält den Wert eines Elements von der Spread-Zeitreihe am angegebenen Index.

```
double  Spread(
   int     ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der Spread-Zeitreihe am angegebenen Index beim Erfolg, sonst EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

Time

Erhält den Wert eines Elements von der Time-Zeitreihe am angegebenen Index.

```
datetime  Time(
   int    ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der Time-Zeitreihe am angegebenen Index beim Erfolg, anderenfalls EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

TickVolume

Erhält den Wert eines Elements von der TickVolume-Zeitreihe am angegebenen Index.

```
long  TickVolume(
   int    ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der TickVolume-Zeitreihe am angegebenen Index beim Erfolg, anderenfalls EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

RealVolume

Erhält den Wert eines Elements von der RealVolume-Zeitreihe am angegebenen Index.

```
long  RealVolume(
   int    ind          // Index
   )
```

Parameter

ind

[in] Der Index des Elements der Zeitreihe.

Rückgabewert

Der Wert eines Elements von der RealVolume-Zeitreihe am angegebenen Index beim Erfolg, sonst EMPTY_VALUE.

Hinweis

Wert EMPTY_VALUE kann in zwei Fälle zurückgegeben werden:

1. Die Zeitreihe wird nicht verwendet (das entsprechende Bit der Verwendung ist nicht festgelegt).
2. Der Element-Index hat über die vorbereiteten Zeitreihendaten gegangen.

© 2000-2025, MetaQuotes Ltd.

Init

Initialisiert ein Objekt.

```
bool  Init(
   CSymbolInfo       symbol,    //
   ENUM_TIMEFRAMES   period,    //
   double            point      //
   )
```

Parameter

symbol

[in] Ein Zeiger auf das Objekt CSymbolInfo für den Zugriff auf Informationen über das Arbeitssymbol.

period

[in] Zeitrahmen aus der Enumeration ENUM_TIMEFRAMES.

point

[in] "Gewicht" vom Punkt mit 2/4 Zeichen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Symbol

Setzt den Arbeitssymbol eines Objekts.

```
bool  Symbol(
   string    name     // Symbol
   )
```

Parameter

name

[in] Name des Arbeitssymbols.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Setzen vom Arbeitssymbol ist erforderlich, wenn das Objekt ein Symbol, das anders als das bei der ersten Initialisierung angegebenen Symbol ist, verwendet.

© 2000-2025, MetaQuotes Ltd.

Period

Setzt die Arbeitsrahmen eines Objekts.

```
bool  Period(
   ENUM_TIMEFRAMES  value     // Zeitrahmen
   )
```

Parameter

value

[in] Arbeitszeitrahmen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Setzen von der Arbeitszeitrahmen ist erforderlich, wenn das Objekt eine Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Zeitrahmen ist, verwendet.

© 2000-2025, MetaQuotes Ltd.

Magic

Setzt die Identifikator (Magic) von Expert Advisor.

```
void  Magic(
   ulong  value     // ID
   )
```

Parameter

value

[in] Identifikator vom Expert Advisor.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

SetPriceSeries

Setzt Zeiger auf die externe Preisobjekte-Zeitreihen.

```
virtual bool  SetPriceSeries(
   CiOpen*    open,         // Zeiger
   CiHigh*    high,         // Zeiger
   CiLow*     low,          // Zeiger
   CiClose*   close         // Zeiger
   )
```

Parameter

open

[in] Ein Zeiger für den Zugriff auf die Zeitreihe Open.

high

[in] Ein Zeiger für den Zugriff auf die Zeitreihe High.

low

[in] Ein Zeiger für den Zugriff auf die Zeitreihe Low.

close

[in] Ein Zeiger für den Zugriff auf die Zeitreihe Close.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Setzten von Zeigern auf Preis-Objekte-Zeitreihen ist erforderlich, wenn das Objekt ein Symbol und Zeitrahmen (die bei der Anfangsinitialisierung gesetzt wurden) und Preis-Zeitreihen, die für die weitere Arbeit erforderlich sind, verwendet.

© 2000-2025, MetaQuotes Ltd.

SetOtherSeries

Setzt Zeiger auf die externe Nicht-Preis-Objekte-Zeitreihen.

```
virtual bool  SetOtherSeries(
   CiSpread*       spread,        // Zeiger
   CiTime*         time,          // Zeiger
   CiTickVolume*   tick_volume,   // Zeiger
   CiRealVolume*   real_volume    // Zeiger
   )
```

Parameter

spread

[in] Ein Zeiger für den Zugriff auf die Zeitreihe Spread.

time

[in] Ein Zeiger für den Zugriff auf die Zeitreihe Time.

tick_volume

[in] Ein Zeiger für den Zugriff auf die Zeitreihe TickVolume.

real_volume

[in] Ein Zeiger für den Zugriff auf die Zeitreihe RealVolume.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Setzten von Zeigern auf Nicht-Preis-Objekte-Zeitreihen ist erforderlich, wenn das Objekt ein Symbol und Zeitrahmen (die bei der Anfangsinitialisierung gesetzt wurden) und Nicht-Preis-Zeitreihen, die für die weitere Arbeit erforderlich sind, verwendet.

© 2000-2025, MetaQuotes Ltd.

InitIndicators

Initialisiert die notwendigen Indikatoren und Zeitreihen.

```
virtual bool  InitIndicators(
   CIndicators*  indicators=NULL    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die notwendige Zeitreihen werden initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet.

© 2000-2025, MetaQuotes Ltd.

InitOpen

Initialisiert die Open-Zeitreihe.

```
bool  InitOpen(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Open-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitHigh

Initialisiert die High-Zeitreihe.

```
bool  InitHigh(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die High-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitLow

Initialisiert die Low-Zeitreihe.

```
bool  InitLow(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Low-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitClose

Initialisiert die Close-Zeitreihe.

```
bool  InitClose(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Close-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitSpread

Initialisiert die Spread-Zeitreihe.

```
bool  InitSpread(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Spread-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitTime

Initialisiert die Time-Zeitreihe.

```
bool  InitTime(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die Time-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitTickVolume

Initialisiert die TickVolume-Zeitreihe.

```
bool  InitTickVolume(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die TickVolume-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

InitRealVolume

Initialisiert die RealVolume-Zeitreihe.

```
bool  InitRealVolume(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die RealVolume-Zeitreihen wird initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet und die Zeitreihen für die weitere Arbeit erforderlich ist.

© 2000-2025, MetaQuotes Ltd.

PriceLevelUnit

Erhält die Einheit von Preisniveaus.

```
virtual double  PriceLevelUnit()
```

Rückgabewert

Die Einheit von Preisniveaus.

Hinweis

In der Basisklasse wird "Gewicht" vom Punkt mit 2/4 Zeichen zurückgegeben.

© 2000-2025, MetaQuotes Ltd.

StartIndex

Erhält den Index des ersten analysierten Balkens.

```
virtual int  StartIndex()
```

Rückgabewert

Der Index des ersten analysierten Balkens.

Hinweis

Wenn das Flag der Analyse des aktuellen Balkens festgesetzt ist, wird 0 zurückgegeben (die Analyse wird mit dem aktuellen Balken anfangen). Wenn das Flag der Analyse des aktuellen Balkens gelöscht ist, wird 1 zurückgegeben (die Analyse wird mit dem letzten gebildeten Balken anfangen).

© 2000-2025, MetaQuotes Ltd.

CompareMagic

Vergleicht den Wert der Identifikator (magic) mit dem angegebenen Wert.

```
virtual bool  CompareMagic(
   ulong  magic    // Identifikator
   )
```

Parameter

magic

[in] ID wert für Vergleich.

Rückgabewert

Gibt true zurück, wenn die ID mit dem angegebenen Wert gleich sind, andernfalls false.

© 2000-2025, MetaQuotes Ltd.

Klasse CExpert

Klasse CExpert ist die Basisklasse für die Implementierung von Handelsstrategien.

Sie enthält die grundlegenden "Handelsfähigkeiten". Die Klasse hat integrierte Mechanismen für die Arbeit mit Zeitreihen und Indikatoren und enthält virtuelle Methoden, die die Handelsstrategie bestimmen.

Damit Ihr Expert Advisor in einer anderen Weise arbeitet, müssen Sie:

1. Algorithmen der Strategien bestimmen;

2. Ihre eigene von CExpert geerbte Klasse erstellen;

3. virtuelle Methoden der Basisklasse in Ihrer Klasse überschreiben und Ihre Algorithmen hinzufügen.

Beschreibung

Klasse CExpert bietet eine Satz von virtuellen Methoden für die Implementierung von Handelsstrategien.

Deklaration

```
class CExpert : public CExpertBase
```

Kopf

```
#include <Expert\Expert.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpert

Gruppen der Klassenmethode

Initialisierung	
Init	Initialisiert eine Instanz der Klasse
virtual InitSignal	Initialisiert ein Objekt von Handelssignalen
virtual InitTrailing	Initialisiert ein Trailing-Objekt
virtual InitMoney	Initialisiert ein Objekt von Geldmanagement
virtual InitTrade	Initialisiert ein Handelsobjekt
virtual ValidationSettings	Überprüft die Einstellungen
virtual InitIndicators	Initialisiert die notwendigen Indikatoren und Zeitreihen
virtual InitParameters	Initialisiert die Parameter von Expert Advisor

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
virtual Deinit	Deinitialisiert eine Instanz der Klasse
virtual DeinitSignal	Deinitialisiert ein Objekt von Handelssignalen
virtual DeinitTrailing	Deinitialisiert ein Trailing-Objekt
virtual DeinitMoney	Deinitialisiert ein Objekt von Geldmanagement
virtual DeinitTrade	Deinitialisiert ein Handelsobjekt
virtual DeinitIndicators	Deinitialisiert Indikatoren und Zeitreihen
Einstellung	
Magic	Setzt die Identifikator von Expert Advisor
MaxOrders	Erhält/setzt die maximale Anzahl der Ordern
OnTickProcess	Setzt ein Flag von "OnTick"-Ereignisverarbeitung
OnTradeProcess	Setzt ein Flag von "OnTrade"-Ereignisverarbeitung
OnTimerProcess	Setzt ein Flag von "OnTimer"-Ereignisverarbeitung
OnChartEventProcess	Setzt ein Flag von "OnChartEvent"-Ereignisverarbeitung
OnBookEventProcess	Setzt ein Flag von "OnBookEvent"-Ereignisverarbeitung
Event Handlers	
OnTick	Handler vom OnTick-Ereignis
OnTrade	Handler vom OnTrade-Ereignis
OnTimer	Handler vom OnTime-Ereignis
OnChartEvent	Handler vom OnChartEvent-Ereignis
OnBookEvent	Handler vom OnBookEvent-Ereignis
Aktualisierung	
Refresh	Aktualisiert alle notwendigen Daten
Die Grundmethode der Behandlung	
Processing	Implementiert den Basisalgorithmus
Methoden der Markteingang	
CheckOpen	Prüft die Notwendigkeit und die Möglichkeit in den Markt zu gehen
CheckOpenLong	Prüft die Notwendigkeit und die Möglichkeit eine Long-Position zu eröffnen
CheckOpenShort	Prüft die Notwendigkeit und die Möglichkeit eine Short-Position zu eröffnen

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
OpenLong	Führt Operationen für Eröffnung einer Long-Position
OpenShort	Führt Operationen für Eröffnung einer Short-Position
Methoden der Ausgang aus dem Markt	
CheckClose	Prüft die Notwendigkeit aus dem Markt zu gehen
CheckCloseLong	Prüft die Notwendigkeit eine Long-Position zu schließen
CheckCloseShort	Prüft die Notwendigkeit eine Short-Position zu schließen
CloseAll	Komplett geht aus dem Markt (schließt alle Positionen)
Close	Geht aus dem Markt
CloseLong	Schließt die Long-Position
CloseShort	Schließt die Short-Position
Methoden von Position-Umkehr	
CheckReverse	Prüft die Notwendigkeit eine Position umzukehren
CheckReverseLong	Prüft die Notwendigkeit eine Long-Position umzukehren
CheckReverseShort	Prüft die Notwendigkeit eine Short-Position umzukehren
ReverseLong	Kehrt eine Long-Position um
ReverseShort	Kehrt eine Short-Position um
Methoden von Positionen/Ordern-Pflege	
CheckTrailingStop	Prüft die Notwendigkeit Position-Eigenschaften zu ändern
CheckTrailingStopLong	Prüft die Notwendigkeit Long-Position-Eigenschaften zu ändern
CheckTrailingStopShort	Prüft die Notwendigkeit Short-Position-Eigenschaften zu ändern
TrailingStopLong	Ändert die Eigenschaften einer Long-Position
TrailingStopShort	Ändert die Eigenschaften einer Short-Position
CheckTrailingOrderLong	Prüft die Notwendigkeit Eigenschaften einer Kauf-Order zu ändern
CheckTrailingOrderShort	Prüft die Notwendigkeit Eigenschaften einer Verkauf-Order zu ändern
TrailingOrderLong	Ändert die Eigenschaften einer Kauf-Order
TrailingOrderShort	Ändert die Eigenschaften einer Verkauf-Order
Methoden von Order-Löschung	

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
CheckDeleteOrderLong	Prüft die Notwendigkeit Kauf-Ordern zu löschen
CheckDeleteOrderShort	Prüft die Notwendigkeit Verkauf-Ordern zu löschen
DeleteOrders	Löscht alle Ordern
DeleteOrder	Löscht eine Order
DeleteOrderLong	Löscht eine Kauf-Order
DeleteOrderShort	Löscht eine Verkauf-Order
Methoden zur Bestimmung der Volumen	
LotOpenLong	Findet Volumen einer Kauftransaktion heraus
LotOpenShort	Findet Volumen einer Verkauftransaktion heraus
LotReverse	Findet Volumen einer Transaktion der Position-Umkehr heraus
Methoden der Arbeit mit Handelsgeschichte	
PrepareHistoryDate	Setzt das Startdatum einer kontrollierten Handelsgeschichte
HistoryPoint	Erstellt einen Kontrollpunkt in der Handelsgeschichte
CheckTradeState	Behandelt die Änderungen in der Handelsgeschichte
Methoden der Arbeit mit den Flags der Handelsereignisse	
WaitEvent	Setzt ein Flag der Erwartung auf ein Handelsereignis
NoWaitEvent	Löscht ein Flag der Erwartung auf ein Handelsereignis
Methoden der Bearbeitung von Handelsereignissen	
TradeEventPositionStopTake	Handler vom Ereignis "Stop Loss/Take Profit aktiviert"
TradeEventOrderTriggered	Handler des Ereignisses "Pending-Order aktiviert"
TradeEventPositionOpened	Handler des Ereignisses "Positionseröffnung"
TradeEventPositionVolumeChanged	Handler des Ereignisses "Position erhöhen/reduzieren"
TradeEventPositionModified	Handler des Ereignisses "Änderung der Parameter einer Position"
TradeEventPositionClosed	Handler des Ereignisses "Positionsschluss"
TradeEventOrderPlaced	Handler des Ereignisses "Pending-Order platziert"
TradeEventOrderModified	Handler des Ereignisses "Pending-Order geändert"
TradeEventOrderDeleted	Handler des Ereignisses "Pending-Order gelöscht"

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
TradeEventNotIdentified	Handler eines nicht identifizierten Ereignisses
Dienstmethoden	
TimeframeAdd	Fügt einen Timeframe für die Überwachung hinzu
TimeframesFlags	Bildet Flags von Timeframes
SelectPosition	Wählt eine Position für die weitere Arbeit aus.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries

© 2000-2025, MetaQuotes Ltd.

Init

Initialisiert eine Instanz der Klasse.

```
bool  Init(
   string              symbol,         // Symbol
   ENUM_TIMEFRAMES     period,         // Periode
   bool                every_tick,     // Flag
   ulong               magic           // Expert Advisor ID
   )
```

Parameter

symbol

[in] Symbol des Expert Advisors.

period

[in] Die Arbeitsperiode von Expert Advisor aus der Enumeration ENUM_TIMEFRAMES.

every_tick

[in] Flag.

magic

[in] Expert Advisor ID.

Rückgabewert

Nichts.

Hinweis

Wenn der Parameter every_tick auf true gesetzt ist, wird die Methode Processing() an jedem Tick des Handelssymbols aufgerufen. Andernfalls wird die Processing()-Methode nur bei der Formung vom neuen Bar an der Arbeitsperiode des Arbeitssymbols von Expert Advisor angerufen.

© 2000-2025, MetaQuotes Ltd.

Magic

Setzt die Identifikator (Magic) von Expert Advisor.

```
void  Magic(
   ulong  value     // ein neuer Wert
   )
```

Parameter

value

[in] Ein neuer Wert der Expert Advisor ID.

Rückgabewert

Nichts.

Hinweis

Wenn Sie die ID von Expert Advisor ändern, wird dieselbe ID allen Hilfsobjekte zugewiesen.

Implementierung

```
//+------------------------------------------------------------------+
//| Sets magic number for object and its dependent objects           |
//| INPUT:  value - new value of magic number.                       |
//| OUTPUT: no.                                                      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
void CExpert::Magic(ulong value)
  {
   if(m_trade!=NULL)     m_trade.SetExpertMagicNumber(value);
   if(m_signal!=NULL)    m_signal.Magic(value);
   if(m_money!=NULL)     m_money.Magic(value);
   if(m_trailing!=NULL) m_trailing.Magic(value);
//---
   CExpertBase::Magic(value);
  }
```

© 2000-2025, MetaQuotes Ltd.

InitSignal

Initialisiert ein Objekt von Handelssignalen.

```
virtual bool  InitSignal(
   CExpertSignal*     signal=NULL,        // Zeiger
   )
```

Parameter

signal

[in] Zeiger auf das CExpertSignal-Objekt oder sein Kind.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Wenn der Parameter 'signal' gleich NULL ist, wird das Handelssignale-Objekt durch die Basisklasse CExpertSignal (welche immer nichts tut) initialisiert werden.

© 2000-2025, MetaQuotes Ltd.

InitTrailing

Initialisiert ein Trailing-Objekt.

```
virtual bool  InitTrailing(
   CExpertTrailing*     trailing=NULL,        // Zeiger
   )
```

Parameter

trailing

[in] Zeiger auf das CExpertTrailing-Objekt oder sein Kind.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Wenn der Parameter 'trailing' gleich NULL ist, wird das Trailing-Objekt durch die Basisklasse ExpertTrailing (welche immer nichts tut) initialisiert werden.

© 2000-2025, MetaQuotes Ltd.

InitMoney

Initialisiert ein Objekt von Geldmanagement.

```
virtual bool  InitMoney(
   CExpertMoney*     money=NULL,         // Zeiger
   )
```

Parameter

money

[in] Zeiger auf das CExpertMoney-Objekt oder sein Kind.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Wenn der Parameter 'money' gleich NULL ist, wird das Geldmanagement-Objekt durch die Basisklasse CExpertMoney (welche immer Minimallot bietet) initialisiert werden.

© 2000-2025, MetaQuotes Ltd.

InitTrade

Initialisiert ein Handelsobjekt.

```
virtual bool  InitTrade(
   ulong            magic,       // Identifikator
   CExpertTrade*    trade=NULL   // Zeiger
   )
```

Parameter

magic

[in] ID von Expert Advisor, die für Handelsordern verwendet wird.

trade

[in] Ein Zeiger auf das Handelsobjekt.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Init

Deinitialisiert eine Instanz der Klasse.

```
virtual void Deinit()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnTickProcess

Setzt das Flag von OnTick-Ereignisverarbeitung.

```
void  OnTickOProcess(
   bool     value        // Flag
   )
```

Parameter

value

[in] Flag der Verarbeitung von OnTick.

Rückgabewert

Nichts.

Hinweis

Wenn der Flag-Wert auf true ist, dann wird OnTick verarbeitet werden, sonst wird OnTick nicht verarbeitet werden. Der Standardwert des Flags ist true.

© 2000-2025, MetaQuotes Ltd.

OnTradeProcess

Setzt das Flag von OnTrade-Ereignisverarbeitung.

```
void  OnTradeProcess(
   bool     value        // Flag
   )
```

Parameter

value

[in] Flag der Verarbeitung von OnTrade.

Rückgabewert

Nichts.

Hinweis

Wenn der Flag-Wert auf true ist, dann wird OnTrade-Ereignis verarbeitet werden, sonst wird OnTrade nicht verarbeitet werden. Der Standardwert des Flags ist false.

© 2000-2025, MetaQuotes Ltd.

OnTimerProcess

Setzt das Flag von OnTimer-Ereignisverarbeitung.

```
void  OnTimerProcess(
   bool     value        // Flag
   )
```

Parameter

value

[in] Flag der Verarbeitung von OnTimer.

Rückgabewert

Nichts.

Hinweis

Wenn der Flag-Wert auf true ist, dann wird OnTimer verarbeitet werden, sonst wird OnTimer nicht verarbeitet werden. Der Standardwert des Flags ist false.

© 2000-2025, MetaQuotes Ltd.

OnChartEventProcess

Setzt das Flag von OnChartEvent-Ereignisverarbeitung.

```
void  OnChartEventProcess(
   bool     value        // Flag
   )
```

Parameter

value

[in] Flag der Verarbeitung von OnChartEvent.

Rückgabewert

Nichts.

Hinweis

Wenn der Flag-Wert auf true ist, dann wird OnChartEvent verarbeitet werden, sonst wird OnChartEvent nicht verarbeitet werden. Der Standardwert des Flags ist false.

© 2000-2025, MetaQuotes Ltd.

OnBookEventProcess

Setzt das Flag von OnBookEvent-Ereignisverarbeitung.

```
void  OnChartEventProcess(
   bool     value        // Flag
   )
```

Parameter

value

[in] Flag der Verarbeitung von OnBookEvent.

Rückgabewert

Nichts.

Hinweis

Wenn der Flag-Wert auf true ist, dann wird OnBookEvent verarbeitet werden, sonst wird OnBookEvent nicht verarbeitet werden. Der Standardwert des Flags ist false.

© 2000-2025, MetaQuotes Ltd.

MaxOrders (Get Method)

Erhält die maximale Anzahl der Ordern.

```
int  MaxOrders()
```

Rückgabewert

Die maximal erlaubte Anzahl der Ordern.

MaxOrders (Set Method)

Setzt die maximale Anzahl der Ordern.

```
void  MaxOrders(
   int     max_orders        // Anzahl der Ordern
   )
```

Parameter

max_orders

[in] Der neue Wert der maximal erlaubte Anzahl von Ordern.

Rückgabewert

Nichts.

Hinweis

Die standardmäßige maximale Anzahl der Ordern ist 1.

© 2000-2025, MetaQuotes Ltd.

Signal

Gibt einen Zeiger auf ein Objekt von Handelssignalen zurück.

```
CExpertSignal*  Signal() const
```

Rückgabewert

Ein Zeiger auf ein Objekt von Handelssignalen zurück.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Die Korrektheit der Einstellungen aller Hilfsobjekte von Expert Advisor wird geprüft.

© 2000-2025, MetaQuotes Ltd.

InitIndicators

Initialisiert die notwendigen Indikatoren und Zeitreihen.

```
virtual bool  InitIndicators(
   CIndicators*  indicators=NULL    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die notwendige Zeitreihen werden initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet.

Initialisiert alle Indikatoren und Zeitreihen aller Hilfsobjekten von Expert Advisor.

© 2000-2025, MetaQuotes Ltd.

OnTick

Handler vom OnTick-Ereignis.

```
virtual void  OnTick()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnTrade

Handler vom OnTrade-Ereignis.

```
virtual void  OnTrade()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnTimer

Handler vom OnTimer-Ereignis.

```
virtual void  OnTimer()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnChartEvent

Handler vom OnChartEvent-Ereignis.

```
virtual void  OnChartEvent(
   const int     id,       // Ereignis-ID
   const long&   lparam,   // Parameter des Ereignisses vom Typ long
   const double  dparam,   // Parameter des Ereignisses vom Typ double
   const string  sparam    // Parameter des Ereignisses vom Typ string
   )
```

Parameter

id

[in] Ereignis-ID.

lparam

[in] Parameter des Ereignisses vom Typ long.

dparam

[in] Parameter des Ereignisses vom Typ double.

sparam

[in] Parameter des Ereignisses vom Typ string.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnBookEvent

Handler vom OnBookEvent-Ereignis.

```
virtual void  OnBookEvent(
   const string&     symbol       // Symbol
)
```

Parameter

symbol

[in] Das Symbol des OnBookEvent-Ereignisses.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

InitParameters

Initialisiert die Parameter von Expert Advisor.

```
virtual bool  InitParameters()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

In der Klasse CExpert tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

DeinitTrade

Deinitialisiert ein Handelsobjekt.

```
virtual void  DeinitTrade()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

DeinitSignal

Deinitialisiert ein Objekt von Handelssignalen.

```
virtual void  DeinitSignal()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

DeinitTrailing

Deinitialisiert das Trailing Objekt.

```
virtual void  DeinitTrailing()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

DeinitMoney

Deinitialisiert ein Objekt von Geldmanagement.

```
virtual void  DeinitMoney()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

DeinitIndicators

Deinitialisiert Indikatoren und Zeitreihen.

```
virtual void  DeinitIndicators()
```

Rückgabewert

Nichts.

Hinweis

Deinitialisiert alle Indikatoren und Zeitreihen aller Hilfsobjekten von Expert Advisor.

© 2000-2025, MetaQuotes Ltd.

Refresh

Aktualisiert alle notwendigen Daten.

```
virtual bool  Refresh()
```

Rückgabewert

Gibt true zurück, wenn die weitere Tick-Behandlung verbraucht ist, ansonsten false.

Hinweis

Prüft die Notwendigkeit und die Möglichkeit eine Position umzukehren. Wenn die Tick-Verarbeitung nicht gebraucht (oder unmöglich) ist, wird false zurückgegeben. Wenn eine Behandlung notwendig ist, aktualisiert die Preise und Daten aller Indikatoren und Zeitreihen und gibt true zurück.

Implementierung

```
//+------------------------------------------------------------------+
//| Refreshing data for processing                                   |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if successful, false otherwise.                     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::Refresh()
  {
   MqlDateTime time;
//--- refresh rates
   if(!m_symbol.RefreshRates()) return(false);
//--- check need processing
   TimeToStruct(m_symbol.Time(),time);
   if(m_period_flags!=WRONG_VALUE && m_period_flags!=0)
      if((m_period_flags & TimeframesFlags(time))==0) return(false);
   m_last_tick_time=time;
//--- refresh indicators
   m_indicators.Refresh();
//--- ok
   return(true);
  }
```

© 2000-2025, MetaQuotes Ltd.

Processing

Implementiert den Basisalgorithmus.

```
virtual bool  Processing()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Die Methode der Basisklasse führt die folgende Aktionen aus:

1. Prüft ob irgendwelche Position für das Arbeitssymbol geöffnet ist. Wenn es kein Position gibt, werden die Schritte 2, 3 und 4 ausgelassen.

2. Prüft die Notwendigkeit eine Position umzukehren (Aufruf der CheckReverse()-Methode). Wenn die Position "Reverse" ist, dann verlassen.

3. Prüft die Notwendigkeit eine Position zu schließen (Aufruf der CheckClose()-Methode). Wenn die Position geschlossen ist, wird Schritt 4 ausgelassen.

4. Prüft die Notwendigkeit eine Position zu ändern (Aufruf der CheckTrailingStop()-Methode). Wenn die Position geändert war, dann verlassen.

5. Prüft ob er irgendwelche Pending-Ordern für das Arbeitssymbol geöffnet ist. Wenn es keine Ordern gibt, dann gehen wir weiter auf den Schritt .9.

6. Prüft die Notwendigkeit eine Order zu löschen (Aufruf der CheckDeleteOrderLong()-Methode für Kaufsordern und CheckDeleteOrderShort() für Verkaufsordern). Wird eine Order gelöscht, dann gehen wir weiter auf den Schritt 9.

7. Prüft die Notwendigkeit eine Order zu ändern (Aufruf der CheckTrailingOrderLong()-Methode für Kaufsordern und CheckTrailingOrderShort() für Verkaufsordern). Wenn die Order geändert war, dann verlassen.

8. Verlassen.

9. Prüft die Notwendigkeit eine Position zu eröffnen (Aufruf der CheckOpen()-Methode).

Um Ihren eigenen Algorithmus zu implementieren, überschreiben Sie Processing()-Methode in der abgeleiteten Klasse.

Implementierung:

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//| Main function                                                    |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if any trade operation processed, false otherwise.  |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::Processing()
  {
//--- check if open positions
   if(m_position.Select(m_symbol.Name()))
     {
      //--- open position is available
      //--- check the possibility of reverse the position
      if(CheckReverse()) return(true);
      //--- check the possibility of closing the position/delete pending orders
      if(!CheckClose())
        {
         //--- check the possibility of modifying the position
         if(CheckTrailingStop()) return(true);
         //--- return without operations
         return(false);
        }
     }
//--- check if placed pending orders
   int total=OrdersTotal();
   if(total!=0)
     {
      for(int i=total-1;i>=0;i--)
        {
         m_order.SelectByIndex(i);
         if(m_order.Symbol()!=m_symbol.Name()) continue;
         if(m_order.OrderType()==ORDER_TYPE_BUY_LIMIT || m_order.OrderType()==ORDER_TY
           {
            //--- check the ability to delete a pending order to buy
            if(CheckDeleteOrderLong()) return(true);
            //--- check the possibility of modifying a pending order to buy
            if(CheckTrailingOrderLong()) return(true);
           }
         else
           {
            //--- check the ability to delete a pending order to sell
            if(CheckDeleteOrderShort()) return(true);
            //--- check the possibility of modifying a pending order to sell
            if(CheckTrailingOrderShort()) return(true);
           }
         //--- return without operations
         return(false);
        }
     }
//--- check the possibility of opening a position/setting pending order
   if(CheckOpen()) return(true);
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

SelectPosition

Wählt eine Position für die weitere Arbeit aus.

```
void  SelectPosition()
```

Rückgabewert

Nicht vorhanden.

Umsetzung

```
//+------------------------------------------------------------------+
//| Position select                                                  |
//| INPUT:  no.                                                      |
//| OUTPUT: no.                                                      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::SelectPosition(void)
  {
   bool res=false;
//---
   if(m_margin_mode==ACCOUNT_MARGIN_MODE_RETAIL_HEDGING)
      res=m_position.SelectByMagic(m_symbol.Name(),m_magic);
   else
      res=m_position.Select(m_symbol.Name());
//---
   return(res);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckOpen

Prüft die Notwendigkeit und die Möglichkeit in den Markt zu gehen.

```
virtual bool  CheckOpen()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit in den Markt durch die Eröffnung einer Long-Position zu gehen (Aufruf der CheckOpenLong()-Methode), dann durch Eröffnung einer Short-Position (Aufruf der CheckOpenShort()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for position open or limit/stop order set                  |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckOpen()
  {
   if(CheckOpenLong())  return(true);
   if(CheckOpenShort()) return(true);
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Prüft die Notwendigkeit und die Möglichkeit in den Markt durch die Eröffnung einer Long-Position zu gehen.

```
virtual bool  CheckOpenLong()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit in den Markt durch die Eröffnung einer Long-Position zu gehen (Aufruf der CheckOpenLong()-Methode eines Handelssignale-Objekts) und wenn die Bedienung erfüllt ist, tritt in den Markt mit den durch Handelssignale-Objekt eingestellten Parametern (Aufruf der OpenLong()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for long position open or limit/stop order set             |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckOpenLong()
  {
   double   price=EMPTY_VALUE;
   double   sl=0.0;
   double   tp=0.0;
   datetime expiration=TimeCurrent();
//--- check signal for long enter operations
   if(m_signal.CheckOpenLong(price,sl,tp,expiration))
     {
      if(!m_trade.SetOrderExpiration(expiration))
        {
         m_expiration=expiration;
        }
      return(OpenLong(price,sl,tp));
     }
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Prüft die Notwendigkeit und die Möglichkeit in den Markt durch die Eröffnung einer Short-Position zu gehen.

```
virtual bool  CheckOpenShort()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Short-Position zu eröffnen (Aufruf der CheckOpenShort()-Methode eines Handelssignale-Objekts) und wenn die Bedienung erfüllt ist, tritt in den Markt mit den durch Handelssignale-Objekt eingestellten Parametern (Aufruf der OpenShort()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for short position open or limit/stop order set            |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckOpenShort()
  {
   double   price=EMPTY_VALUE;
   double   sl=0.0;
   double   tp=0.0;
   datetime expiration=TimeCurrent();
//--- check signal for short enter operations
   if(m_signal.CheckOpenShort(price,sl,tp,expiration))
     {
      if(!m_trade.SetOrderExpiration(expiration))
        {
         m_expiration=expiration;
        }
      return(OpenShort(price,sl,tp));
     }
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

OpenLong

Führt Operationen für Eröffnung einer Long-Position.

```
virtual bool  OpenLong(
   double    price,    // Preis
   double    sl,       // Stop Loss
   double    tp        // Take Profit
   )
```

Parameter

price

[in] Markteintrittspreis.

sl

[in] Preis von Stop Loss.

tp

[in] Preis von Take Profit.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Die Lotgröße ist durch den Aufruf von LotOpenLong()-Methode definiert. Wenn ein Lot nicht gleich 0.0 ist, eröffnet die Position (Aufruf der Buy(...)-Methode vom Handelsobjekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Long position open or limit/stop order set                       |
//| INPUT:  price - price,                                           |
//|         sl    - stop loss,                                       |
//|         tp    - take profit.                                     |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::OpenLong(double price,double sl,double tp)
  {
   if(price==EMPTY_VALUE) return(false);
//--- get lot for open
   double lot=LotOpenLong(price,sl);
//--- check lot for open
   if(lot==0.0) return(false);
//---
   return(m_trade.Buy(lot,price,sl,tp));
  }
```

© 2000-2025, MetaQuotes Ltd.

OpenShort

Führt Operationen für Eröffnung einer Short-Position.

```
virtual bool  OpenShort(
   double    price,    // Preis
   double    sl,       // Stop Loss
   double    tp        // Take Profit
   )
```

Parameter

price

[in] Markteintrittspreis.

sl

[in] Preis von Stop Loss.

tp

[in] Preis von Take Profit.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Die Lotgröße ist durch den Aufruf von LotOpenShort()-Methode definiert. Wenn ein Lot nicht gleich 0.0 ist, eröffnet die Position (Aufruf der Sell(...)-Methode vom Handelsobjekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Short position open or limit/stop order set                      |
//| INPUT:  price - price,                                           |
//|         sl    - stop loss,                                       |
//|         tp    - take profit.                                     |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::OpenShort(double price,double sl,double tp)
  {
   if(price==EMPTY_VALUE) return(false);
//--- get lot for open
   double lot=LotOpenShort(price,sl);
//--- check lot for open
   if(lot==0.0) return(false);
//---
   return(m_trade.Sell(lot,price,sl,tp));
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckReverse

Prüft die Notwendigkeit und die Möglichkeit eine Position umzukehren.

```
virtual bool  CheckReverse()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Long-Position umzukehren (Aufruf der CheckReverseLong()-Methode), dann für eine Short-Position (Aufruf der CheckReverseShort()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for position reverse                                       |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckReverse()
  {
   if(m_position.PositionType()==POSITION_TYPE_BUY)
     {
      //--- check the possibility of reverse the long position
      if(CheckReverseLong())  return(true);
     }
   else
      //--- check the possibility of reverse the short position
      if(CheckReverseShort()) return(true);
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckReverseLong

Prüft die Notwendigkeit und die Möglichkeit eine Long-Position umzukehren.

```
virtual bool  CheckReverseLong()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Long-Position umzukehren (Aufruf der CheckReverseLong()-Methode eines Handelssignale-Objekts) und wenn die Bedienung erfüllt ist, kehrt die Long-Position mit den durch Handelssignale-Objekt eingestellten Parametern um (Aufruf der ReverseLong()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for long position reverse                                  |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckReverseLong()
  {
   double   price=EMPTY_VALUE;
   double   sl=0.0;
   double   tp=0.0;
   datetime expiration=TimeCurrent();
//--- check signal for long reverse operations
   if(m_signal.CheckReverseLong(price,sl,tp,expiration)) return(ReverseLong(price,sl,t
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckReverseShort

Prüft die Notwendigkeit und die Möglichkeit eine Short-Position umzukehren.

```
virtual bool  CheckReverseLong()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Short-Position umzukehren (Aufruf der CheckReverseShort()-Methode eines Handelssignale-Objekts) und wenn die Bedienung erfüllt ist, kehrt die Short-Position mit den durch Handelssignale-Objekt eingestellten Parametern um (Aufruf der ReverseShort()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for short position reverse                                 |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckReverseShort()
  {
   double   price=EMPTY_VALUE;
   double   sl=0.0;
   double   tp=0.0;
   datetime expiration=TimeCurrent();
//--- check signal for short reverse operations
   if(m_signal.CheckReverseShort(price,sl,tp,expiration)) return(ReverseShort(price,sl
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

ReverseLong

Führt Operationen um eine Long-Position umzukehren

```
virtual bool  ReverseLong(
   double    price,    // Preis
   double    sl,       // Stop Loss
   double    tp        // Take Profit
   )
```

Parameter

price

[in] Markteintrittspreis.

sl

[in] Preis von Stop Loss.

tp

[in] Preis von Take Profit.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Die Lotgröße für Umkehr eine Long-Position wird durch den Aufruf von LotReverse()-Methode definiert. Wenn ein Lot nicht gleich 0.0 ist, kehrt die Long-Position um (Aufruf der Sell(...)-Methode vom Handelsobjekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Long position reverse                                            |
//| INPUT:  price - price,                                           |
//|         sl    - stop loss,                                       |
//|         tp    - take profit.                                     |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::ReverseLong(double price,double sl,double tp)
  {
   if(price==EMPTY_VALUE) return(false);
//--- get lot for reverse
   double lot=LotReverse(sl);
//--- check lot
   if(lot==0.0) return(false);
//---
   return(m_trade.Sell(lot,price,sl,tp));
  }
```

© 2000-2025, MetaQuotes Ltd.

ReverseShort

Führt Operationen um eine Short-Position umzukehren

```
virtual bool  ReverseShort(
   double    price,    // Preis
   double    sl,       // Stop Loss
   double    tp        // Take Profit
   )
```

Parameter

price

[in] Markteintrittspreis.

sl

[in] Preis von Stop Loss.

tp

[in] Preis von Take Profit.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Die Lotgröße für Umkehr eine Short-Position wird durch den Aufruf von LotReverse()-Methode definiert. Wenn ein Lot nicht gleich 0.0 ist, kehrt die Short-Position um (Aufruf der Buy(...)-Methode vom Handelsobjekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Short position reverse                                           |
//| INPUT:  price - price,                                           |
//|         sl    - stop loss,                                       |
//|         tp    - take profit.                                     |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::ReverseShort(double price,double sl,double tp)
  {
   if(price==EMPTY_VALUE) return(false);
//--- get lot for reverse
   double lot=LotReverse(sl);
//--- check lot
   if(lot==0.0) return(false);
//---
   return(m_trade.Buy(lot,price,sl,tp));
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckClose

Prüft die Notwendigkeit aus dem Markt zu gehen.

```
virtual bool  CheckClose()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

1. Prüft die Notwendigkeit aus dem Markt durch Software-Stop-Out (Aufruf der CheckClose()-Methode des Kapitalmanagement-Objekts) zu gehen. Wenn die Bedingung erfüllt ist, schließen wir die Position, löschen wir alle Ordern (Aufruf der CloseAll ()-Methode) und verlassen.
2. Prüft die Notwendigkeit eine Long- oder Short-Position zu schließen (Aufruf von CheckCloseLong() oder CheckCloseShort()) und wenn die Position geschlossen ist, löscht alle Ordern (Aufruf vonDeleteOrders()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for position close or limit/stop order delete              |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckClose()
  {
   double lot;
//--- position must be selected before call
   if((lot=m_money.CheckClose(GetPointer(m_position)))!=0.0)
      return(CloseAll(lot));
//--- check for position type
   if(m_position.PositionType()==POSITION_TYPE_BUY)
     {
      //--- check the possibility of closing the long position / delete pending orders
      if(CheckCloseLong())
        {
         DeleteOrders();
         return(true);
        }
     }
   else
     {
      //--- check the possibility of closing the short position / delete pending order
      if(CheckCloseShort())
        {
         DeleteOrders();
         return(true);
        }
     }
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckCloseLong

Prüft die Notwendigkeit eine Long-Position zu schließen.

```
virtual bool  CheckCloseLong()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Long-Position zu schließen (Aufruf der CheckCloseLong()-Methode eines Handelssignale-Objekts) und wenn die Bedienung erfüllt ist, schließt sie die Position (Aufruf der CloseLong()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for long position close or limit/stop order delete         |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckCloseLong()
  {
   double price=EMPTY_VALUE;
//--- check for long close operations
   if(m_signal.CheckCloseLong(price))
      return(CloseLong(price));
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckCloseShort

Prüft die Notwendigkeit eine Short-Position zu schließen.

```
virtual bool  CheckCloseShort()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Short-Position zu schließen (Aufruf der CheckCloseShort()-Methode eines Handelssignale-Objekts) und wenn die Bedienung erfüllt ist, schließt sie die Position (Aufruf der CloseShort()-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for short position close or limit/stop order delete        |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckCloseShort()
  {
   double price=EMPTY_VALUE;
//--- check for short close operations
   if(m_signal.CheckCloseShort(price))
      return(CloseShort(price));
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CloseAll

Geht aus dem Markt ganz oder teilweise.

```
virtual bool  CloseAll(
   double    lot    // Lot
   )
```

Parameter

lot

[in] Die Lotgröße, auf die Sie die Position reduzieren wollen.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Im Netting-Modus werden Positionen mithilfe der Methoden CExpertTrade::Buy und CExpertTrade::Sell geschlossen. Im Hedging-Modus - mithilfe der Methode CTrade::PositionClose, welche auch für Konten mit dem Netting-Modus verwendet werden kann. Für das Löschen aller Pending Orders wir die Methode DeleteOrders() verwendet.

Implementierung

```
//+------------------------------------------------------------------+
//| Position close and orders delete                                 |
//| INPUT:  lot - volume for close.                                  |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CloseAll(double lot)
  {
   bool result;
//--- check for close operations
   if(m_position.PositionType()==POSITION_TYPE_BUY) result=m_trade.Sell(lot,0,0,0);
   else                                             result=m_trade.Buy(lot,0,0,0);
   result|=DeleteOrders();
//---
   return(result);
  }
```

© 2000-2025, MetaQuotes Ltd.

Close

Geht aus dem Markt.

```
virtual bool  Close()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Schließt eine Position (Aufruf der Methode von Handelsobjekt PositionClose()).

Implementierung

```
//+------------------------------------------------------------------+
//| Position close                                                   |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::Close()
  {
   return(m_trade.PositionClose(m_symbol.Name()));
  }
```

© 2000-2025, MetaQuotes Ltd.

CloseLong

Schließt eine Long-Position.

```
virtual bool  CloseLong(
   double    price    // Preis
   )
```

Parameter

price

[in] Markteintrittspreis.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Schließt eine Long-Position (Aufruf der Methode von Handelsobjekt Sell(...)).

Implementierung

```
//+------------------------------------------------------------------+
//| Long position close                                              |
//| INPUT:  price - price for close.                                 |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CloseLong(double price)
  {
   if(price==EMPTY_VALUE) return(false);
//---
   return(m_trade.Sell(m_position.Volume(),price,0,0));
  }
```

© 2000-2025, MetaQuotes Ltd.

CloseShort

Schließt eine Short-Position.

```
virtual bool  CloseShort(
   double    price    // Preis
   )
```

Parameter

price

[in] Markteintrittspreis.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Schließt eine Short-Position (Aufruf der Methode von Handelsobjekt Buy(...)).

Implementierung

```
//+------------------------------------------------------------------+
//| Short position close                                             |
//| INPUT:  price - price for close.                                 |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CloseShort(double price)
  {
   if(price==EMPTY_VALUE) return(false);
//---
   return(m_trade.Buy(m_position.Volume(),price,0,0));
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStop

Prüft die Notwendigkeit Position-Eigenschaften zu ändern.

```
virtual bool  CheckTrailingStop()
```

Rückgabewert

Gibt true zurück wenn einige Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit eine Position zu ändern (Aufruf der CheckTrailingStopLong()-Methode für eine Long-Position und CheckTrailingStopShort() für eine Short-Position).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for trailing stop/profit position                          |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckTrailingStop()
  {
//--- position must be selected before call
   if(m_position.PositionType()==POSITION_TYPE_BUY)
     {
      //--- check the possibility of modifying the long position
      if(CheckTrailingStopLong()) return(true);
     }
   else
     {
      //--- check the possibility of modifying the short position
      if(CheckTrailingStopShort()) return(true);
     }
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopLong

Prüft die Notwendigkeit Long-Position-Eigenschaften zu ändern.

```
virtual bool  CheckTrailingStopLong()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit die Parameter einer Long-Position zu ändern (Aufruf der CheckTrailingStopLong()-Methode des Trailing-Objekts). Wenn die Bedienung erfüllt ist, ändert sie die Long-Position-Parameter entsprechend den Einstellungen des Trailing-Objekts (Aufruf der Methode TrailingStopLong()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for trailing stop/profit long position                     |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckTrailingStopLong()
  {
   double sl=EMPTY_VALUE;
   double tp=EMPTY_VALUE;
//--- check for long trailing stop operations
   if(m_trailing.CheckTrailingStopLong(GetPointer(m_position),sl,tp))
     {
      if(sl==EMPTY_VALUE) sl=m_position.StopLoss();
      if(tp==EMPTY_VALUE) tp=m_position.TakeProfit();
      //--- long trailing stop operations
      return(TrailingStopLong(sl,tp));
     }
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopShort

Prüft die Notwendigkeit Short-Position-Eigenschaften zu ändern.

```
virtual bool  CheckTrailingStopShort()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit die Parameter einer Short-Position zu ändern (Aufruf der CheckTrailingStopShort()-Methode des Trailing-Objekts). Wenn die Bedienung erfüllt ist, ändert sie die Short-Position-Parameter entsprechend den Einstellungen des Trailing-Objekts (Aufruf der Methode TrailingStopShort()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for trailing stop/profit short position                    |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckTrailingStopShort()
  {
   double sl=EMPTY_VALUE;
   double tp=EMPTY_VALUE;
//--- check for short trailing stop operations
   if(m_trailing.CheckTrailingStopShort(GetPointer(m_position),sl,tp))
     {
      if(sl==EMPTY_VALUE) sl=m_position.StopLoss();
      if(tp==EMPTY_VALUE) tp=m_position.TakeProfit();
      //--- short trailing stop operations
      return(TrailingStopShort(sl,tp));
     }
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

TrailingStopLong

Ändert die Eigenschaften einer Long-Position.

```
virtual bool  TrailingStopLong(
   double    sl,    // Preis von Stop Loss
   double    tp,    // Preis von Take Profit
   )
```

Parameter

sl

[in] Preis von Stop Loss.

tp

[in] Preis von Take Profit.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Ändert die Position (Aufruf der Methode PositionModify() eines Handelsobjekts).

Implementierung

```
//+------------------------------------------------------------------+
//| Trailing stop/profit long position                               |
//| INPUT:  sl - new stop loss,                                      |
//|         tp - new take profit.                                    |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::TrailingStopLong(double sl,double tp)
  {
   return(m_trade.PositionModify(m_symbol.Name(),sl,tp));
  }
```

© 2000-2025, MetaQuotes Ltd.

TrailingStopShort

Ändert die Eigenschaften einer Short-Position.

```
virtual bool  TrailingStopLong(
   double    sl,    // Preis von Stop Loss
   double    tp,    // Preis von Take Profit
   )
```

Parameter

sl

[in] Preis von Stop Loss.

tp

[in] Preis von Take Profit.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Ändert die Position (Aufruf der Methode PositionModify() eines Handelsobjekts).

Implementierung

```
//+------------------------------------------------------------------+
//| Trailing stop/profit short position                              |
//| INPUT:  sl - new stop loss,                                      |
//|         tp - new take profit.                                    |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::TrailingStopShort(double sl,double tp)
  {
   return(m_trade.PositionModify(m_symbol.Name(),sl,tp));
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckTrailingOrderLong

Prüft die Notwendigkeit Eigenschaften einer Kauf-Order zu ändern.

```
virtual bool  CheckTrailingOrderLong()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit Parameter einer Kauf-Order zu ändern (Aufruf der CheckTrailingOrderLong()-Methode des Handelssignale-Objekts). Wenn die Bedienung erfüllt ist, ändert sie die Order-Parameter (Aufruf der Methode TrailingOrderLong()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for trailing long limit/stop order                         |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckTrailingOrderLong()
  {
   double price;
//--- check the possibility of modifying the long order
   if(m_signal.CheckTrailingOrderLong(GetPointer(m_order),price))
      return(TrailingOrderLong(m_order.PriceOpen()-price));
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckTrailingOrderShort

Prüft die Notwendigkeit Eigenschaften einer Verkauf-Order zu ändern.

```
virtual bool  CheckTrailingOrderShort()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Prüft die Notwendigkeit Parameter einer Verkauf-Order zu ändern (Aufruf der CheckTrailingOrderShort()-Methode des Handelssignale-Objekts). Wenn die Bedienung erfüllt ist, ändert sie die Order-Parameter (Aufruf der Methode TrailingOrderShort()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for trailing short limit/stop order                        |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckTrailingOrderShort()
  {
   double price;
//--- check the possibility of modifying the short order
   if(m_signal.CheckTrailingOrderShort(GetPointer(m_order),price))
      return(TrailingOrderShort(m_order.PriceOpen()-price));
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

TrailingOrderLong

Ändert die Eigenschaften einer Kauf-Order.

```
virtual bool  TrailingOrderLong(
   double    delta    // Verschiebung
   )
```

Parameter

delta

[in] Preisänderung.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Ändert Order-Parameter (Aufruf der OrderModify(...)-Methode des Handelsobjekts).

Implementierung

```
//+------------------------------------------------------------------+
//| Trailing long limit/stop order                                   |
//| INPUT:  delta - price change.                                    |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::TrailingOrderLong(double delta)
  {
   ulong  ticket=m_order.Ticket();
   double price =m_order.PriceOpen()-delta;
   double sl    =m_order.StopLoss()-delta;
   double tp    =m_order.TakeProfit()-delta;
//--- modifying the long order
   return(m_trade.OrderModify(ticket,price,sl,tp,m_order.TypeTime(),m_order.TimeExpira
  }
```

© 2000-2025, MetaQuotes Ltd.

TrailingOrderShort

Ändert die Eigenschaften einer Verkauf-Order.

```
virtual bool  TrailingOrderShort(
   double    delta    // Verschiebung
   )
```

Parameter

delta

[in] Preisänderung.

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Ändert Order-Parameter (Aufruf der OrderModify(...)-Methode des Handelsobjekts).

Implementierung

```
//+------------------------------------------------------------------+
//| Trailing short limit/stop order                                  |
//| INPUT:  delta - price change.                                    |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::TrailingOrderShort(double delta)
  {
   ulong  ticket=m_order.Ticket();
   double price =m_order.PriceOpen()-delta;
   double sl    =m_order.StopLoss()-delta;
   double tp    =m_order.TakeProfit()-delta;
//--- modifying the short order
   return(m_trade.OrderModify(ticket,price,sl,tp,m_order.TypeTime(),m_order.TimeExpira
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckDeleteOrderLong

Prüft die Notwendigkeit eine Kauf-Order zu löschen.

```
virtual bool  CheckDeleteOrderLong()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

1. Prüft die Ablaufzeit der Order.
2. Prüft die Notwendigkeit eine Order zu löschen (Aufruf der CheckCloseLong()-Methode des Handelssignale-Objekts). Wenn eine der Bedienungen erfüllt ist, löscht sie die Order (Aufruf der Methode DeleteOrderLong()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for delete long limit/stop order                           |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckDeleteOrderLong()
  {
   double price;
//--- check the possibility of deleting the long order
   if(m_expiration!=0 && TimeCurrent()>m_expiration)
     {
      m_expiration=0;
      return(DeleteOrderLong());
     }
   if(m_signal.CheckCloseLong(price))
      return(DeleteOrderLong());
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

CheckDeleteOrderShort

Prüft die Notwendigkeit eine Verkauf-Order zu löschen.

```
virtual bool  CheckDeleteOrderShort()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

1. Prüft die Ablaufzeit der Order.
2. Prüft die Notwendigkeit eine Order zu löschen (Aufruf der CheckCloseShort()-Methode des Handelssignale-Objekts). Wenn eine der Bedienungen erfüllt ist, löscht sie die Order (Aufruf der Methode DeleteOrderShort()).

Implementierung

```
//+------------------------------------------------------------------+
//| Check for delete short limit/stop order                          |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation processed, false otherwise.      |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::CheckDeleteOrderShort()
  {
   double price;
//--- check the possibility of deleting the short order
   if(m_expiration!=0 && TimeCurrent()>m_expiration)
     {
      m_expiration=0;
      return(DeleteOrderShort());
     }
   if(m_signal.CheckCloseShort(price))
      return(DeleteOrderShort());
//--- return without operations
   return(false);
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteOrders

Löscht alle Ordern.

```
virtual bool  DeleteOrders()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Löscht alle ordern (Aufruf in der Schleife der DeleteOrder-Methode).

Implementierung

```
//+------------------------------------------------------------------+
//| Delete all limit/stop orders                                     |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::DeleteOrders()
  {
   bool result=false;
   int  total=OrdersTotal();
//---
   for(int i=total-1;i>=0;i--)
     {
      if(m_order.Select(OrderGetTicket(i)))
        {
         if(m_order.Symbol()!=m_symbol.Name()) continue;
         result|=DeleteOrder();
        }
     }
//---
   return(result);
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteOrder

Löscht eine Order.

```
virtual bool  DeleteOrder()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Löscht eine Order (Aufruf der Methode von Handelsobjekt OrderDelete()).

Implementierung

```
//+------------------------------------------------------------------+
//| Delete limit/stop order                                          |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::DeleteOrder()
  {
   return(m_trade.OrderDelete(m_order.Ticket()));
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteOrderLong

Löscht eine Kauf-Order.

```
virtual bool  DeleteOrderLong()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Löscht eine Kauf-Order (Aufruf der Handelsobjekt-Methode OrderDelete()).

Implementierung

```
//+------------------------------------------------------------------+
//| Delete long limit/stop order                                     |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::DeleteOrderLong()
  {
   return(m_trade.OrderDelete(m_order.Ticket()));
  }
```

© 2000-2025, MetaQuotes Ltd.

DeleteOrderShort

Löscht eine Verkauf-Order.

```
virtual bool  DeleteOrderShort()
```

Rückgabewert

Gibt true zurück wenn eine Operation ausgeführt ist, ansonsten gibt false zurück.

Hinweis

Löscht eine Verkauf-Order (Aufruf der Handelsobjekt-Methode OrderDelete()).

Implementierung

```
//+------------------------------------------------------------------+
//| Delete short limit/stop order                                    |
//| INPUT:  no.                                                      |
//| OUTPUT: true-if trade operation successful, false otherwise.     |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
bool CExpert::DeleteOrderShort()
  {
   return(m_trade.OrderDelete(m_order.Ticket()));
  }
```

© 2000-2025, MetaQuotes Ltd.

LotOpenLong

Findet Volumen einer Kauftransaktion heraus.

```
double  LotOpenLong(
   double    price,    // Preis
   double    sl        // Preis von Stop Loss
   )
```

Parameter

price

[in] Markteintrittspreis.

sl

[in] Preis von Stop Loss.

Rückgabewert

Volumen (in Lots) einer Kauftransaktion.

Hinweis

Findet Volumen einer Kauftransaktion heraus (Aufruf der CheckOpenLong(...)-Methode vom Kapitalmanagement-Objekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Method of getting the lot for open long position.                |
//| INPUT:  price - price,                                           |
//|         sl    - stop loss.                                       |
//| OUTPUT: lot for open.                                            |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
double CExpert::LotOpenLong(double price,double sl)
  {
   return(m_money.CheckOpenLong(price,sl));
  }
```

© 2000-2025, MetaQuotes Ltd.

LotOpenShort

Findet Volumen einer Verkauftransaktion heraus.

```
double  LotOpenShort(
   double    price,   // Preis
   double    sl       // Preis von Stop Loss
   )
```

Parameter

price

[in] Markteintrittspreis.

sl

[in] Preis von Stop Loss.

Rückgabewert

Volumen (in Lots) einer Verkauftransaktion.

Hinweis

Findet Volumen einer Verkauftransaktion heraus (Aufruf der CheckOpenShort(...)-Methode vom Kapitalmanagement-Objekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Method of getting the lot for open short position.               |
//| INPUT:  price - price,                                           |
//|         sl    - stop loss.                                       |
//| OUTPUT: lot for open.                                            |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
double CExpert::LotOpenShort(double price,double sl)
  {
   return(m_money.CheckOpenShort(price,sl));
  }
```

© 2000-2025, MetaQuotes Ltd.

LotReverse

Findet Volumen einer Transaktion der Position-Umkehr heraus.

```
double  LotReverse(
   double    sl       // Preis von Stop Loss
   )
```

Parameter

sl

[in] Preis von Stop Loss.

Rückgabewert

Volumen einer Transaktion der Position-Umkehr.

Hinweis

Findet Volumen einer Position-Umkehr-Operation heraus (Aufruf der CheckReverse(...)-Methode vom Kapitalmanagement-Objekt).

Implementierung

```
//+------------------------------------------------------------------+
//| Method of getting the lot for reverse position.                  |
//| INPUT:  sl - stop loss.                                          |
//| OUTPUT: lot for open.                                            |
//| REMARK: no.                                                      |
//+------------------------------------------------------------------+
double CExpert::LotReverse(double sl)
  {
   return(m_money.CheckReverse(GetPointer(m_position),sl));
  }
```

© 2000-2025, MetaQuotes Ltd.

PrepareHistoryDate

Setzt das Startdatum einer kontrollierten Handelsgeschichte.

```
void  PrepareHistoryDate()
```

Hinweis

Standardmäßig wird der Starttermin der kontrollierten Geschichte auf den Anfang des Monats (aber nicht weniger als ein Tag) festgesetzt.

© 2000-2025, MetaQuotes Ltd.

HistoryPoint

Erstellt einen Kontrollpunkt in der Handelsgeschichte.

```
void  HistoryPoint(
   bool   from_check_trade=false   // Flag
   )
```

Parameter

from_check_trade=false

[in] Flag um Rekursion zu vermeiden.

Hinweis

Speichert die Anzahl der Positionen, Ordern, Trades und Ordern in der Geschichte.

© 2000-2025, MetaQuotes Ltd.

CheckTradeState

Behandelt die Änderungen in der Handelsgeschichte.

```
bool  CheckTradeState()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Prüft die Anzahl der Positionen, Ordern, Trades und Ordern in der Geschichte durch den Vergleich mit zuvor in der History ()-Methode gespeicherten. Im Falle einer Änderung ruft einen entsprechenden virtuellen Prozessor auf.

© 2000-2025, MetaQuotes Ltd.

WaitEvent

Setzt ein Flag der Erwartung auf ein Handelsereignis.

```
void  WaitEvent(
   ENUM_TRADE_EVENTS     event          // Flag
   )
```

Parameter

event

[in] Flag der Erwartung auf ein Handelsereignis (aus der Enumeration ENUM_TRADE_EVENTS), der gesetzt werden soll.

Rückgabewert

Nichts.

Flags von Ereignisse

```
//--- flags of expected events
enum ENUM_TRADE_EVENTS
  {
   TRADE_EVENT_NO_EVENT               =0,         // no expected events
   TRADE_EVENT_POSITION_OPEN          =0x1,       // flag of expecting the "opening of
   TRADE_EVENT_POSITION_VOLUME_CHANGE=0x2,        // flag of expecting of the "modifica
   TRADE_EVENT_POSITION_MODIFY        =0x4,       // flag of expecting of the "modifica
   TRADE_EVENT_POSITION_CLOSE         =0x8,       // flag of expecting of the "closing
   TRADE_EVENT_POSITION_STOP_TAKE     =0x10,      // flag of expecting of the "triggeri
   TRADE_EVENT_ORDER_PLACE            =0x20,      // flag of expecting of the "placing
   TRADE_EVENT_ORDER_MODIFY           =0x40,      // flag of expecting of the "modifica
   TRADE_EVENT_ORDER_DELETE           =0x80,      // flag of expecting of the "deletion
   TRADE_EVENT_ORDER_TRIGGER          =0x100      // flag of expecting of the "triggeri
  };
```

© 2000-2025, MetaQuotes Ltd.

NoWaitEvent

Löscht das Flag der Erwartung auf ein Handelsereignis.

```
void  NoWaitEvent(
   ENUM_TRADE_EVENTS     event          // Flag
   )
```

Parameter

event

[in] Flag der Erwartung auf ein Handelsereignis (aus der Enumeration ENUM_TRADE_EVENTS), der gelöscht werden soll.

Rückgabewert

Nichts.

Flags von Ereignisse

```
//--- flags of expected events
enum ENUM_TRADE_EVENTS
  {
   TRADE_EVENT_NO_EVENT               =0,         // no expected events
   TRADE_EVENT_POSITION_OPEN          =0x1,       // flag of expecting the "opening of
   TRADE_EVENT_POSITION_VOLUME_CHANGE=0x2,        // flag of expecting of the "modifica
   TRADE_EVENT_POSITION_MODIFY        =0x4,       // flag of expecting of the "modifica
   TRADE_EVENT_POSITION_CLOSE         =0x8,       // flag of expecting of the "closing
   TRADE_EVENT_POSITION_STOP_TAKE     =0x10,      // flag of expecting of the "triggeri
   TRADE_EVENT_ORDER_PLACE            =0x20,      // flag of expecting of the "placing
   TRADE_EVENT_ORDER_MODIFY           =0x40,      // flag of expecting of the "modifica
   TRADE_EVENT_ORDER_DELETE           =0x80,      // flag of expecting of the "deletion
   TRADE_EVENT_ORDER_TRIGGER          =0x100      // flag of expecting of the "triggeri
  };
```

© 2000-2025, MetaQuotes Ltd.

TradeEventPositionStopTake

Handler vom Ereignis "Stop Loss/Take Profit aktiviert".

```
virtual bool  TradeEventPositionStopTake()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventOrderTriggered

Handler des Ereignisses "Pending-Order aktiviert".

```
virtual bool  TradeEventOrderTriggered()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventPositionOpened

Handler des Ereignisses "Positionseröffnung".

```
virtual bool  TradeEventPositionOpened()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventPositionVolumeChanged

Handler des Ereignisses "Position erhöhen/reduzieren".

```
virtual bool  TradeEventPositionVolumeChanged()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventPositionModified

Handler des Ereignisses "Änderung der Parameter einer Position".

```
virtual bool  TradeEventPositionModified()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventPositionClosed

Handler des Ereignisses "Positionsschluss".

```
virtual bool  TradeEventPositionClosed()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventOrderPlaced

Handler des Ereignisses "Pending-Order platziert".

```
virtual bool  TradeEventOrderPlaced()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventOrderModified

Handler des Ereignisses "Pending-Order geändert".

```
virtual bool  TradeEventOrderModified()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventOrderDeleted

Handler des Ereignisses "Pending-Order gelöscht".

```
virtual bool  TradeEventOrderDeleted()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

TradeEventNotIdentified

Handler eines nicht identifizierten Ereignisses.

```
virtual bool  TradeEventNotIdentified()
```

Rückgabewert

Methode der CExpert-Klasse tut nichts und gibt immer true zurück.

Hinweis

Es sei darauf hingewiesen, dass die Handelsereignisse nicht immer einzeln sind, am meisten kommen sie als "Packs". In diesem Fall ist die eindeutige Identifikation der Handelsereignisse schwierig.

© 2000-2025, MetaQuotes Ltd.

TimeframeAdd

Fügt einen Timeframe für die Überwachung hinzu.

```
void  TimeframeAdd(
   ENUM_TIMEFRAMES    period         // Periode
   )
```

Parameter

period

[in] Periode (aus der Enumeration ENUM_TIMEFRAMES), die überwacht werden soll.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

TimeframesFlags

Bildet Flags von Timeframes.

```
int  TimeframesFlags(
   MqlDateTime&    time          // Referenz
   )
```

Parameter

time

[in] Referenz auf die Struktur vom Typ MqlDateTime, die die neue Zeit enthält.

Rückgabewert

Flags von Zeitrahmen, für die "ein neuer Bar" erhalten war.

© 2000-2025, MetaQuotes Ltd.

Klasse CExpertSignal

CExpertSignal ist die Basisklasse für die Erstellung von Handelssignale-Generatoren, darum bietet sie Schnittstelle und tut nichts (außer Methoden CheckReverseLong() und CheckReverseShort()).

Damit der Generator von Handelssignale "generieren startet", müssen Sie:

1. Algorithmen von Handelssignale-Erzeugung bestimmen;
2. Ihre eigene von CExpert geerbte Generator-Klasse erstellen;
3. virtuelle Methoden der Basisklasse in Ihrer Klasse überschreiben und Ihre Algorithmen hinzufügen.

Als Beispiel betrachten wir eine beliebige mqh-Datei aus dem Ordner Expert\Signal\.

Beschreibung

Klasse CExpertSignal ist eine Basis für die Implementierung von Algorithmen für Handelssignale-Generierung.

Deklaration

```
class CExpertSignal : public CExpertBase
```

Kopf

```
#include <Expert\ExpertSignal.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertSignal

Direkte Ableitungen

CSignalAC, CSignalAMA, CSignalAO, CSignalBearsPower, CSignalBullsPower, CSignalCCI, CSignalDeM, CSignalDEMA, CSignalEnvelopes, CSignalFrAMA, CSignalRSI, CSignalRVI, CSignalSAR, CSignalStoch, CSignalTEMA, CSignalTriX, CSignalWPR

Gruppen der Klassenmethode

Initialisierung	
virtual InitIndicators	Initialisiert die notwendigen Indikatoren und Zeitreihen
virtual ValidationSettings	Überprüft die Objekt-Einstellungen
virtual AddFilter	Fügt einen Filter in ein kombiniertes Signal hinzu
Zugriff auf geschützte Daten	
BasePrice	Setzt die Basispreisniveau

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
UsedSeries	Erhält die Flags der verwendeten Zeitreihen
Einstellung	
Weight	Setzt den Wert des Parameters "Weight"
PatternsUsage	Setzt den Wert des Parameters "PatternsUsage"
General	Setzt den Wert des Parameters "General"
Ignore	Setzt den Wert des Parameters "Ignore"
Invert	Setzt den Wert des Parameters "Invert"
ThresholdOpen	Setzt den Wert des Parameters "ThresholdOpen"
ThresholdClose	Setzt den Wert des Parameters "ThresholdClose"
PriceLevel	Setzt den Wert des Parameters "PriceLevel"
StopLevel	Setzt den Wert des Parameters "StopLevel"
TakeLevel	Setzt den Wert des Parameters "TakeLevel"
Expiration	Setzt den Wert des Parameters "Expiration"
Magic	Setzt den Wert des Parameters "Magic"
Methoden von Prüfung der Notwendigkeit eine Position zu eröffnen/umkehren/schließen	
virtual CheckOpenLong	Findet heraus, ob eine Long-Position eröffnet werden soll
virtual CheckCloseLong	Findet heraus, ob eine Long-Position geschlossen werden soll
virtual CheckOpenShort	Findet heraus, ob eine Short-Position eröffnet werden soll
virtual CheckCloseShort	Findet heraus, ob eine Short-Position geschlossen werden soll
virtual CheckReverseLong	Findet heraus, ob eine Long-Position umgekehrt werden soll
virtual CheckReverseShort	Findet heraus, ob eine Short-Position umgekehrt werden soll
Methoden für Einstellen von Parametern	
virtual OpenLongParams	Setzt die Eröffnungsparameter für eine Long-Position
virtual OpenShortParams	Setzt die Eröffnungsparameter für eine Short-Position
virtual CloseLongParams	Setzt die Schlussparameter für eine Long-Position
virtual CloseShortParams	Setzt die Schlussparameter für eine Short-Position
Methoden von Prüfung der Notwendigkeit Pending-	

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
Order zu steuern	
virtual CheckTrailingOrderLong	Findet heraus, ob Eigenschaften einer Pending-Kauf-Order geändert werden soll
virtual CheckTrailingOrderShort	Findet heraus, ob Eigenschaften einer Pending-Verkauf-Order geändert werden soll
Die Methoden der Prüfung der Bildung von Marktmodellen	
virtual LongCondition	Gibt einen Ergebnis der Überprüfung, ob Bedingungen für den Kauf aufgetreten waren, zurück
virtual ShortCondition	Gibt einen Ergebnis der Überprüfung, ob Bedingungen für den Verkauf aufgetreten waren, zurück
virtual Direction	Gibt den Wert der "gewichteten" Richtung der Kursbewegung zurück

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries

© 2000-2025, MetaQuotes Ltd.

BasePrice

Setzt die Basispreisniveau.

```
void  BasePrice(
   double    value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

UsedSeries

Erhält die Flags der verwendeten Zeitreihen.

```
int  UsedSeries()
```

Rückgabewert

Flags von verwendeten Zeitreihen (sofern kein anderes Symbol oder andere Zeitrahmen festgelegt sind), andernfalls 0.

© 2000-2025, MetaQuotes Ltd.

Weight

Setzt den Wert des Parameters "Weight".

```
void  Weight(
   double    value         // Wert
   )
```

Parameter

value

[in] Parameter "Weight".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

PatternUsage

Setzt den Wert des Parameters "PatternsUsage".

```
void  PatternUsage(
   double    value         // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "PatternsUsage".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

General

Setzt den Wert des Parameters "General".

```
void  General(
   int     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "General".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Ignore

Setzt den Wert des Parameters "Ignore".

```
void  Ignore(
   long     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "Ignore".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Invert

Setzt den Wert des Parameters "Invert".

```
void  Invert(
   long     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "Invert".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

ThresholdOpen

Setzt den Wert des Parameters "ThresholdOpen".

```
void  ThresholdOpen(
   long     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "ThresholdOpen".

Rückgabewert

Nichts.

Hinweis

Parameter "ThresholdOpen" kann Werte von 0 bis 100 annehmen. Sie wird verwendet, um herauszufinden, ob eine Position auf den Ergebnissen der "Abstimmung" eröffnet werden soll.

© 2000-2025, MetaQuotes Ltd.

ThresholdClose

Setzt den Wert des Parameters "ThresholdClose".

```
void  ThresholdOpen(
   long     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "ThresholdClose".

Rückgabewert

Nichts.

Hinweis

Parameter "ThresholdClose" kann Werte von 0 bis 100 annehmen. Sie wird verwendet, um herauszufinden, ob eine Position auf den Ergebnissen der "Abstimmung" geschlossen werden soll.

© 2000-2025, MetaQuotes Ltd.

PriceLevel

Setzt den Wert des Parameters "PriceLevel".

```
void  PriceLevel(
   double    value         // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "PriceLevel".

Rückgabewert

Nichts.

Hinweis

Parameter "PriceLevel" wird in Einheiten von Preisniveaus definiert. Der Wert der Einheit von Preisniveaus wird durch die Methode PriceLevelUnit() zurückgegeben. Verwendet, um die Eröffnungsniveau relativ zu dem Basispreis festzulegen.

© 2000-2025, MetaQuotes Ltd.

StopLevel

Setzt den Wert des Parameters "StopLevel".

```
void  StopLevel(
   double    value         // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "StopLevel".

Rückgabewert

Nichts.

Hinweis

Parameter "StopLevel" wird in Einheiten von Preisniveaus definiert. Der Wert der Einheit von Preisniveaus wird durch die Methode PriceLevelUnit() zurückgegeben. Verwendet, um Stop Loss relativ zu dem Eröffnungspreis festzulegen.

© 2000-2025, MetaQuotes Ltd.

TakeLevel

Setzt den Wert des Parameters "TakeLevel".

```
void  TakeLevel(
   double    value         // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "TakeLevel".

Rückgabewert

Nichts.

Hinweis

Parameter "TakeLevel" wird in Einheiten von Preisniveaus definiert. Der Wert der Einheit von Preisniveaus wird durch die Methode PriceLevelUnit() zurückgegeben. Verwendet, um Take Profit relativ zu dem Eröffnungspreis festzulegen.

© 2000-2025, MetaQuotes Ltd.

Expiration

Setzt den Wert des Parameters "Expiration".

```
void  Expiration(
   int     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "Expiration".

Rückgabewert

Nichts.

Hinweis

Parameter "Expiration" wird in Balken angegeben. Es wird bei der Berechnung der Ablaufzeit von einer Pending-Order verwendet (wenn Eröffnung nicht zum aktuellen Marktpreis ist geplant).

© 2000-2025, MetaQuotes Ltd.

Magic

Setzt den Wert des Parameters "Magic".

```
void  Magic(
   int     value          // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "Magic" (ID von Expert Advisor).

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Objekt-Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt true zurück, wenn die Objekt-Einstellungen korrekt sind, anderenfalls gibt false zurück.

© 2000-2025, MetaQuotes Ltd.

InitIndicators

Initialisiert die notwendigen Indikatoren und Zeitreihen.

```
virtual bool  InitIndicators(
   CIndicators*  indicators    // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf ein Objekt der Sammlung von Indikatoren und Zeitreihen.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Die notwendige Zeitreihen werden initialisiert, wenn das Objekt ein Symbol oder Zeitrahmen, die anders als die bei der ersten Initialisierung angegebenen Symbol und Zeitrahmen sind, verwendet.

© 2000-2025, MetaQuotes Ltd.

AddFilter

Fügt einen Filter in ein kombiniertes Signal hinzu.

```
virtual bool  AddFilter(
   CExpertSignal*  filter    // Zeiger
   )
```

Parameter

filter

[in] Ein Zeiger auf das Filter-Objekt.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Findet heraus, ob eine Long-Position eröffnet werden soll.

```
virtual bool  CheckOpenLong(
   double&    price,          // Referenz
   double&    sl,             // Referenz
   double&    tp,             // Referenz
   datetime&  expiration      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Eröffnungspreises zu platzieren.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

expiration

[in][out] Die Referenz an die Variable um die Durchlaufzeit der Order (falls erforderlich) zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Findet heraus, ob eine Short-Position eröffnet werden soll.

```
virtual bool  CheckOpenShort(
   double&    price,          // Referenz
   double&    sl,             // Referenz
   double&    tp,             // Referenz
   datetime&  expiration      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Eröffnungspreises zu platzieren.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

expiration

[in][out] Die Referenz an die Variable um die Durchlaufzeit der Order (falls erforderlich) zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

OpenLongParams

Setzt die Eröffnungsparameter für eine Long-Position.

```
virtual bool  OpenLongParams(
   double&    price,          // Referenz
   double&    sl,             // Referenz
   double&    tp,             // Referenz
   datetime&  expiration      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Eröffnungspreises zu platzieren.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

expiration

[in][out] Die Referenz an die Variable um die Durchlaufzeit der Order (falls erforderlich) zu platzieren.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OpenShortParams

Setzt die Eröffnungsparameter für eine Short-Position.

```
virtual bool  OpenShortParams(
   double&    price,          // Referenz
   double&    sl,             // Referenz
   double&    tp,             // Referenz
   datetime&  expiration      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Eröffnungspreises zu platzieren.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

expiration

[in][out] Die Referenz an die Variable um die Durchlaufzeit der Order (falls erforderlich) zu platzieren.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CheckCloseLong

Findet heraus, ob eine Long-Position geschlossen werden soll.

```
virtual bool  CheckCloseLong(
   double&  price      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Schlusspreises zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

CheckCloseShort

Findet heraus, ob eine Short-Position geschlossen werden soll.

```
virtual bool  CheckCloseShort(
   double&  price      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Schlusspreises zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

CloseLongParams

Setzt die Schlussparameter für eine Long-Position.

```
virtual bool  CloseLongParams(
   double&    price          // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Schlusspreises zu platzieren.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CloseShortParams

Setzt die Schlussparameter für eine Short-Position.

```
virtual bool  CloseShortParams(
   double&    price          // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Schlusspreises zu platzieren.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CheckReverseLong

Findet heraus, ob eine Long-Position umgekehrt werden soll.

```
virtual bool  CheckReverseLong(
   double&    price,          // Referenz
   double&    sl,             // Referenz
   double&    tp,             // Referenz
   datetime&  expiration      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Umkehrpreises zu platzieren.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

expiration

[in][out] Die Referenz an die Variable um die Durchlaufzeit der Order (falls erforderlich) zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

CheckReverseShort

Findet heraus, ob eine Short-Position umgekehrt werden soll.

```
virtual bool  CheckReverseShort(
   double&    price,          // Referenz
   double&    sl,             // Referenz
   double&    tp,             // Referenz
   datetime&  expiration      // Referenz
   )
```

Parameter

price

[in][out] Die Referenz an die Variable um den Wert des Umkehrpreises zu platzieren.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

expiration

[in][out] Die Referenz an die Variable um die Durchlaufzeit der Order (falls erforderlich) zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingOrderLong

Findet heraus, ob Parameter einer Pending-Kauf-Order geändert werden soll.

```
virtual bool  CheckTrailingOrderLong(
   COrderInfo*     order,          // Zeiger
   double&         price           // Referenz
   )
```

Parameter

order

[in] Ein Zeiger auf das Objekt COrderInfo.

price

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingOrderShort

Findet heraus, ob Parameter einer Pending-Verkauf-Order geändert werden soll.

```
virtual bool  CheckTrailingOrderShort(
   COrderInfo*     order,          // Zeiger
   double&         price           // Referenz
   )
```

Parameter

order

[in] Ein Zeiger auf das Objekt COrderInfo.

price

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

LongCondition

Gibt einen Ergebnis der Überprüfung, ob Bedingungen für den Kauf aufgetreten waren, zurück.

```
virtual int  LongCondition()
```

Rückgabewert

Wenn eine Bedingung für den Kauf ist aufgetreten, wird eine Zahl von 1 bis 100 (je höher der Wert, desto stärker das Signal) zurückgegeben, wenn es kein Signal gibt, wird 0 zurückgegeben.

Hinweis

Die Basisklasse hat keine Implementierung von Algorithmus zur Bestimmung der Kauf-Bedingungen, darum gibt die Basisklassenmethode immer 0 zurück.

© 2000-2025, MetaQuotes Ltd.

ShortCondition

Gibt einen Ergebnis der Überprüfung, ob Bedingungen für den Verkauf aufgetreten waren, zurück.

```
virtual int  ShortCondition()
```

Rückgabewert

Wenn eine Bedingung für den Verkauf ist aufgetreten, wird eine Zahl von 1 bis 100 (je höher der Wert, desto stärker das Signal) zurückgegeben, wenn es kein Signal gibt, wird 0 zurückgegeben.

Hinweis

Die Basisklasse hat keine Implementierung von Algorithmus zur Bestimmung der Verkauf-Bedingungen, darum gibt die Basisklassenmethode immer 0 zurück.

© 2000-2025, MetaQuotes Ltd.

Direction

Gibt den Wert der "gewichteten" Richtung der Kursbewegung zurück.

```
virtual double  Direction()
```

Rückgabewert

Im Falle der möglichen Preisbewegung nach oben, ist der Wert >0, im Falle einer möglichen Preisbewegung nach unten, ist der Wert <0. Je größer der absolute Wert, desto "stärker" das Signal.

Hinweis

Wenn es integrierte Filter gibt, werden ihre Ergebnisse bei der Bestimmung der allgemeinen Richtung berücksichtigt.

© 2000-2025, MetaQuotes Ltd.

Klasse CExpertTrailing

CExpertTrailing ist eine Basisklasse für die Implementierung von Algorithmen für Positionensteuerung, darum bietet sie Schnittstelle und tut nichts.

Damit "Trailing" in einer anderen Weise arbeitet, müssen Sie:

1. Algorithmen von Steuerung offener Positionen bestimmen;
2. Ihre eigene von CExpertTrailing geerbte Klasse erstellen;
3. virtuelle Methoden der Basisklasse in Ihrer Klasse überschreiben und Ihre Algorithmen hinzufügen.

Als Beispiel betrachten wir eine beliebige mqh-Datei aus dem Ordner Expert\Trailing.

Beschreibung

Klasse CExpertTrailing ist eine Basis für die Implementierung von Algorithmen für Steuerung offener Positionen.

Deklaration

```
class CExpertTrailing : public CExpertBase
```

Kopf

```
#include <Expert\ExpertTrailing.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertTrailing

Direkte Ableitungen

CTrailingFixedPips, CTrailingMA, CTrailingNone, CTrailingPSAR

Gruppen der Klassenmethode

Methoden von Prüfung der Notwendigkeit Pending-Order zu steuern	
virtual CheckTrailingStopLong	Findet hinzu, ob Long-Position-Eigenschaften geändert werden sollen
virtual CheckTrailingStopShort	Findet hinzu, ob Short-Position-Eigenschaften geändert werden sollen

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, ValidationSettings, SetPriceSeries, SetOtherSeries, InitIndicators

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopLong

Findet hinzu, ob Long-Position-Eigenschaften geändert werden sollen.

```
virtual bool  CheckTrailingStopLong(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz für Stop Loss
   double&         tp            // Referenz für Take Profit
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Die Basisklassenmethode gibt immer false zurück.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopShort

Findet hinzu, ob Short-Position-Eigenschaften geändert werden sollen.

```
virtual bool  CheckTrailingStopShort(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz für Stop Loss
   double&         tp            // Referenz für Take Profit
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Die Basisklassenmethode gibt immer false zurück.

© 2000-2025, MetaQuotes Ltd.

Klasse CExpertMoney

CExpertMoney ist eine Basisklasse für die Implementierung von Algorithmen für Risiko- und Geldmanagement.

Beschreibung

Klasse CExpertMoney ist eine Basis für die Implementierung von Algorithmen für Risiko- und Geldmanagement.

Deklaration

```
class CExpertMoney : public CObject
```

Kopf

```
#include <Expert\ExpertMoney.mqh>
```

Vererbungshierarchie

CObject

CExpertBase

CExpertMoney

Direkte Ableitungen

CMoneyFixedLot, CMoneyFixedMargin, CMoneyFixedRisk, CMoneyNone, CMoneySizeOptimized

Gruppen der Klassenmethode

Zugriff auf geschützte Daten	
Percent	Setzt den Wert des Parameters "Prozentsatz für Risiko"
Initialisierung	
virtual ValidationSettings	Überprüft die Einstellungen
Methoden von Prüfung der Notwendigkeit eine Position zu eröffnen/umkehren/schließen	
virtual CheckOpenLong	Bestimmt das Volumen für Eröffnung einer Long-Position
virtual CheckOpenShort	Bestimmt das Volumen für Eröffnung einer Short-Position
virtual CheckReverse	Bestimmt das Volumen für Position-Umkehr
virtual CheckClose	Findet heraus, ob eine Position geschlossen werden soll

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

© 2000-2025, MetaQuotes Ltd.

Percent

Setzt den Prozentsatz des Risikos.

```
void  Percent(
   double  percent      // Risiko-Prozent
   )
```

Parameter

percent

[in] Prozentsatz des Risikos.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Die Basisklassenmethode gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Bestimmt das Volumen für Eröffnung einer Long-Position.

```
virtual double  CheckOpenLong(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Eröffnungspreis einer Long-Position

sl

[in] Preis von Stop Loss für die Long-Position

Rückgabewert

Volumen für Eröffnung einer Long-Position.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Bestimmt das Volumen für Eröffnung einer Short-Position.

```
virtual double  CheckOpenShort(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Eröffnungspreis einer Short-Position

sl

[in] Preis von Stop Loss für die Short-Position

Rückgabewert

Volumen für Eröffnung einer Short-Position.

© 2000-2025, MetaQuotes Ltd.

CheckReverse

Bestimmt das Volumen für Position-Umkehr.

```
virtual double  CheckReverse(
   CPositionInfo*  position,     // Zeiger
   double          sl            // Preis von Stop Loss
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in] Preis von Stop Loss für die Position

Rückgabewert

Volumen für Position-Umkehr.

© 2000-2025, MetaQuotes Ltd.

CheckClose

Findet heraus, ob eine Position geschlossen werden soll.

```
virtual double  CheckClose()
```

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

© 2000-2025, MetaQuotes Ltd.

Die Module der Handelssignale

Der Standardlieferumfang von Client Terminal enthält eine Reihe von Modulen der Handelssignale für MQL5 Wizard. Bei der Erstellung von Expert Advisors in MQL5 Wizard MQL5, können Sie eine Kombination von Modulen der Handelssignale (bis zu 64) verwenden. Die endgültige Entscheidung über Transaktion wird auf der Basis der Gesamtanalyse der Signale aller enthaltenen Modulen gefällt. Eine detaillierte Beschreibung des Mechanismus der Entscheidungsfindung finden Sie unten.

Der Standardlieferumfang enthält die folgende Module der Handelssignale:

- Signale des Indikators Accelerator Oscillator
- Signale von Adaptive Moving Average
- Signale des Indikators Awesome Oscillator
- Signale des Oszillators Bears Power
- Signale des Oszillators Bulls Power
- Signale des Oszillators Commodity Channel Index
- Signale des Oszillators DeMarker
- Signale des Indikators Exponential Moving Average
- Signale des Indikators Envelopes
- Signale des Indikators Fractal Adaptive Moving Average
- Signale von Intraday-Zeitfilter
- Signale des Oszillators MACD
- Signale des Indikators Moving Average
- Signale des Indikators Parabolic SAR
- Signale des Oszillators Relative Strength Index
- Signale des Oszillators Relative Vigor Index
- Signale des Oszillators Stochastic
- Signale des Oszillators Triple Exponential Average
- Signale des Indikators Triple Exponential Moving Average
- Signale des Oszillators Williams Percent Range

Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule

Der Mechanismus der Handelsentscheidungen kann als die folgenden grundlegenden Bestimmungen dargestellt werden:

- Jeder Signalmodul verfügt über einen eigenen Satz von Marktmodellen (eine Kombination aus Preise und Indikatorwerte).
- Jedes Modell hat einen Einflusswert von 1 bis 100. Je höher der Wert, desto stärker das Modell.
- Jedes Modell erzeugt eine Prognose der Preisbewegung in eine bestimmte Richtung.
- Die Preisprognose der Signalmodule ist das Ergebnis der Suche nach bestimmten Modellen und wird als eine Zahl im Bereich von -100 bis +100 zurückgegeben. Das Vorzeichen bestimmt die Richtung

© 2000-2025, MetaQuotes Ltd.

der Prognosebewegung (negative — der Preis wird fallen, die positive — der Preis wird steigen). Der Absolutwert entspricht der Stärke des gefunden besten Modell.

- Prognose aus jedem Modul wird zur Abstimmung mit dem Gewichtungsfaktor von 0 bis 1,0, der in seinem "Weight"-Parameter angegeben ist, gesendet.
- Das Ergebnis der Abstimmung ist eine Zahl zwischen -100 und +100, wobei das Vorzeichen die Richtung des Prognosebewegung ist und der Absolutwert charakterisiert die Stärke des Signals. Er wird als eine gewichtete arithmetische Mittel der Prognosen aller Signalmodule berechnet. Dieser Gesamtwert wird in der EA für Handelsentscheidungen verwendet.

Die Einstellungen jedes generierten Expert Advisor enthalten zwei Parameter — die Schwelle für die Entscheidung über Öffnen oder Schließen einer Position (ThresholdOpen und ThresholdClose), die einen Wert von 0 bis 100 haben kann. Wenn die Stärke des resultierenden Signals (Absolutwert) übergeht den Schwellwert, die Entscheidung ist getroffen in einer entsprechend Richtung zu handeln.

Beispiele

Sei es ein Expert Advisor mit Schwellwerten ThresholdOpen=20 und ThresholdClose=90. Für die Entscheidung über Handelsoperationen werden Signalmodule MA mit Gewicht 0,4 und Stochastic mit Gewicht 0.8 verwendet. Betrachten wir zwei Varianten:

Variante 1.

Preis durch den aufsteigenden Indikator MA nach oben gegangen. Dies entspricht einem Marktmodell aus dem Modul MA, die Preisentwicklung erwartet. Seine Einfluss ist 100. Zur gleichen Zeit, Stochastic Oszillator hat nach unten umgekehrt und eine Divergenz mit dem Preis gebildet. Dies ist ein Marktmodell aus dem Modul Stochastic, die Preissenkung erwartet. Seine Einfluss ist 80.

Wir berechnen das Ergebnis der Schlussabstimmung. Gewichtete Prognose aus dem Modul MA ist als 0,4 * 100 = 40 berechnet. Gewichtete Prognose aus dem Modul Stochastic ist als 0,8 * (-80) = -64 berechnet. Die Schlussprognose ist als der arithmetische Mittel dieser zwei Prognosen berechnet: (40 - 64)/2 = -12. Dies ist ein Verkaufssignal mit relativer Stärke 12. Der Schwellenwert von 20 wurde nicht erreicht. Dementsprechend wird der Handel nicht durchgeführt.

Variante 2.

Preis durch den aufsteigenden Indikator MA nach unten gegangen. Dies entspricht einem Marktmodell aus dem Modul MA, die Preisentwicklung erwartet. Seine Einfluss ist 10. Zur gleichen Zeit, Stochastic Oszillator hat nach unten umgekehrt und eine Divergenz mit dem Preis gebildet. Dies ist ein Marktmodell aus dem Modul Stochastic, die Preissenkung erwartet. Seine Einfluss ist 80.

Wir berechnen das Ergebnis der Schlussabstimmung. Gewichtete Prognose aus dem Modul MA ist als 0,4 * 10 = 4 berechnet. Gewichtete Prognose aus dem Modul Stochastic ist als 0,8 * (-80) = -64 berechnet. Die Schlussprognose ist als der arithmetische Mittel dieser zwei Prognosen berechnet: (4 - 64)/2 = -30. Dies ist ein Verkaufssignal mit relativer Stärke 30. Der Schwellenwert von 20 wurde erreicht. So ist das Ergebnis ein Signal zum Eröffnung einer Short-Position.

© 2000-2025, MetaQuotes Ltd.



a) Die Divergenz der Preis und Oszillator Stochastic (Varianten 1 und 2).

b) Der Preis hat durch den Indikator MA nach oben gegangen (Variante 1).

c) Der Preis hat durch den Indikator MA nach unten gegangen (Variante 2).

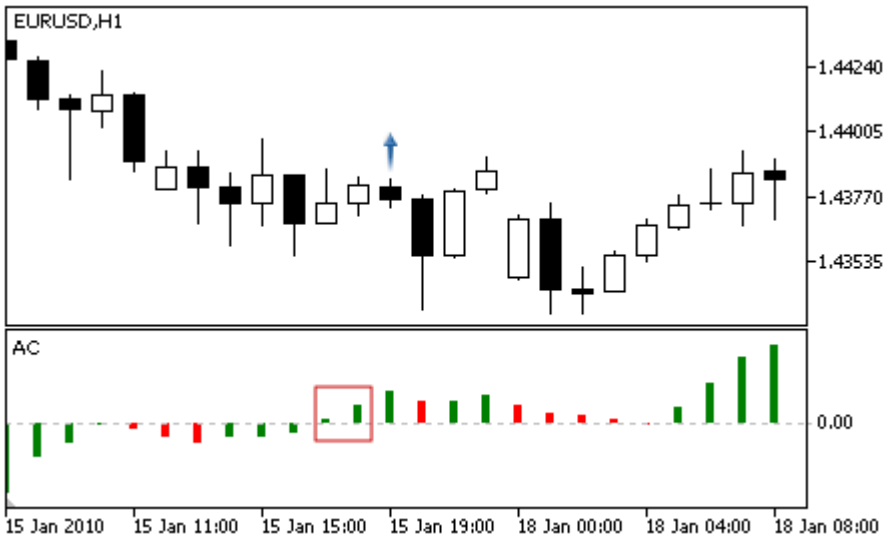
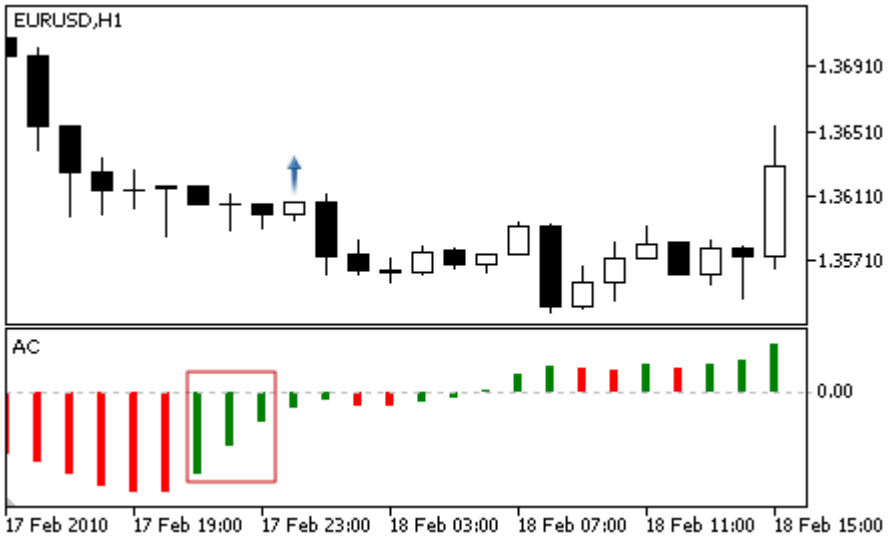
© 2000-2025, MetaQuotes Ltd.

Signale von Accelerator Oscillator

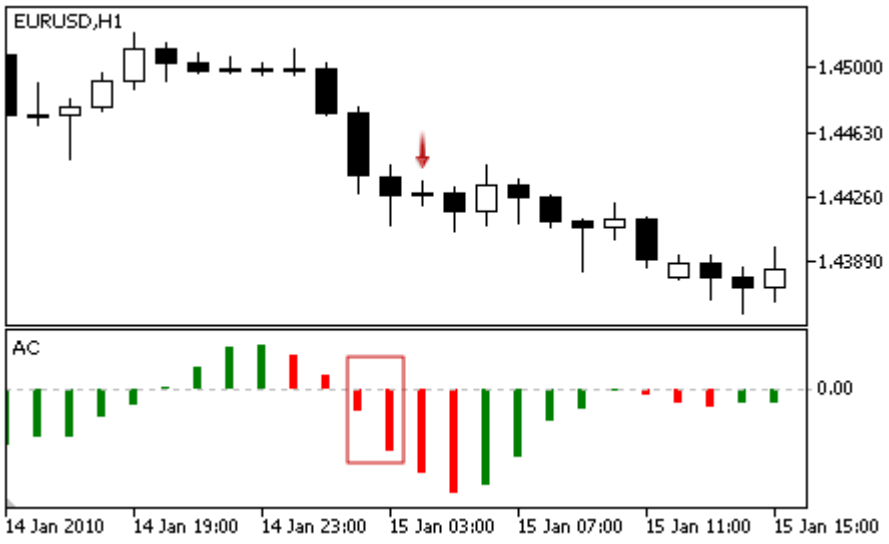
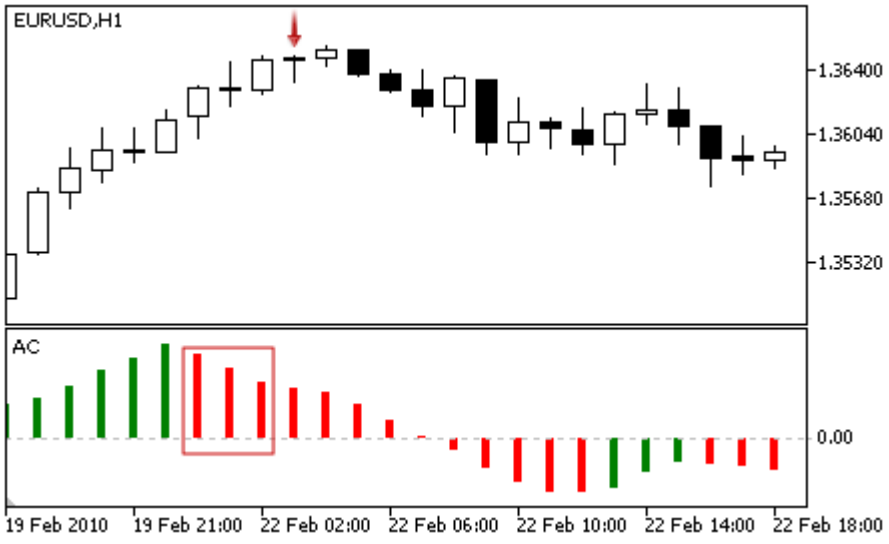
Dieses Signalmodul basiert auf Marktmodelle des Indikators Accelerator Oscillator. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• Der Indikatorwert größer als 0 und wächst am analysierten und vorherigen Balken.  • Der Indikatorwert unter 0 und wächst am analysierten und zwei vorherigen Balken. 
Zum Verkauf	• Der Indikatorwert unter 0 und fällt am analysierten und zwei vorherigen Balken.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • Der Indikatorwert größer als 0 und fällt am analysierten und zwei vorherigen Balken. 
Nicht gegen den Kauf	Der Indikatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Indikatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

© 2000-2025, MetaQuotes Ltd.

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.

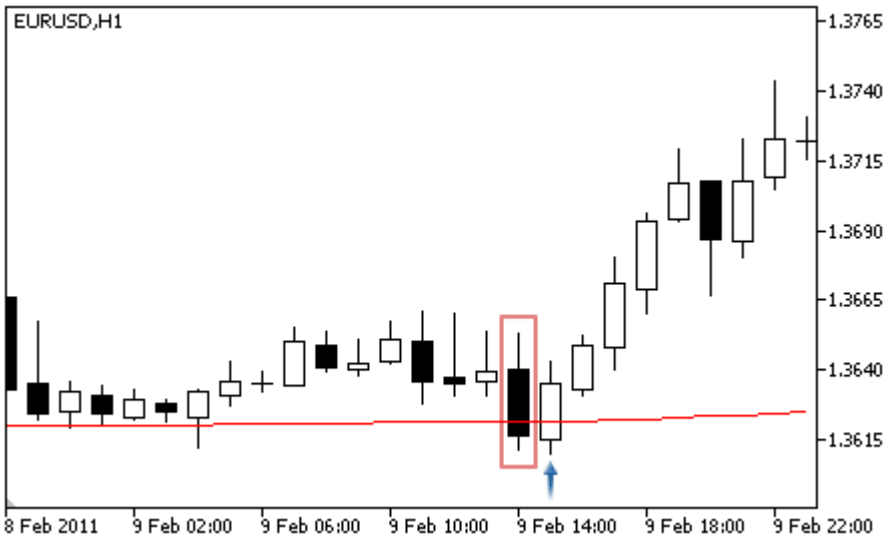
© 2000-2025, MetaQuotes Ltd.

Signale von Adaptive Moving Average

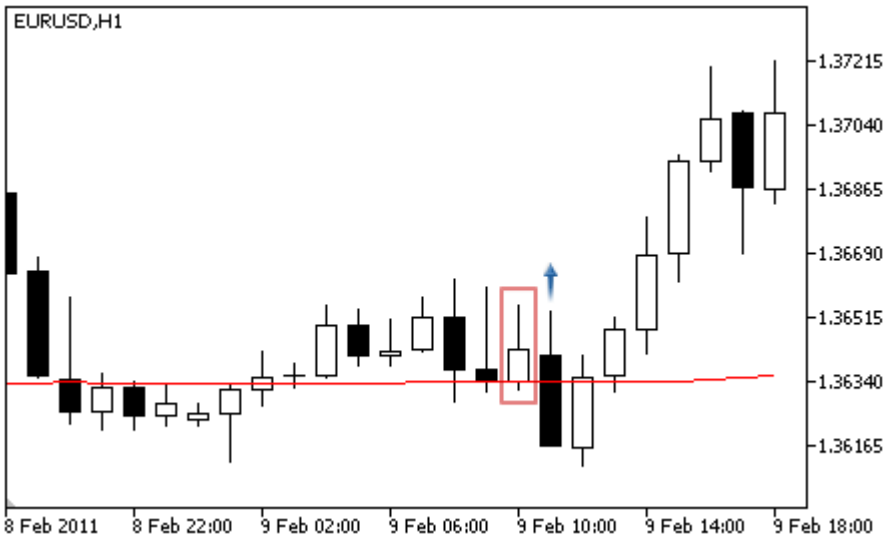
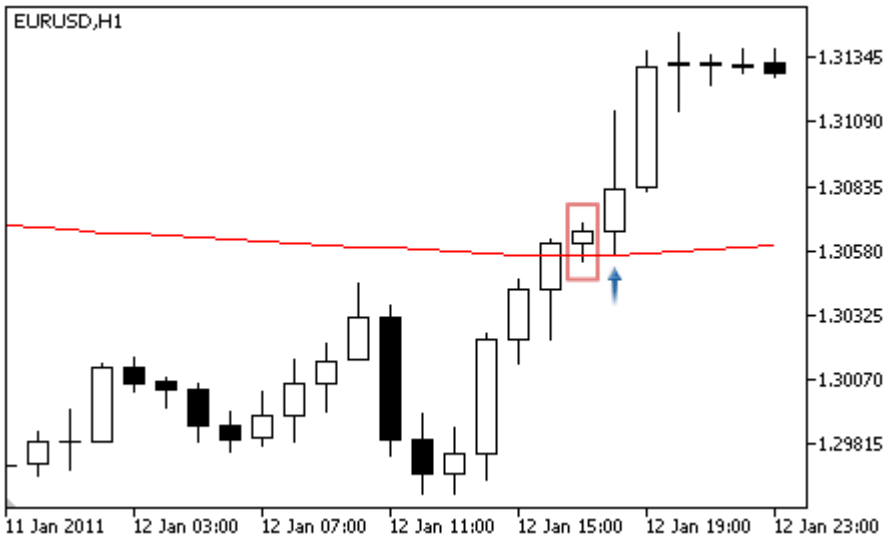
Dieses Signalmodul basiert auf Marktmodelle des Indikators Adaptive Moving Average. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

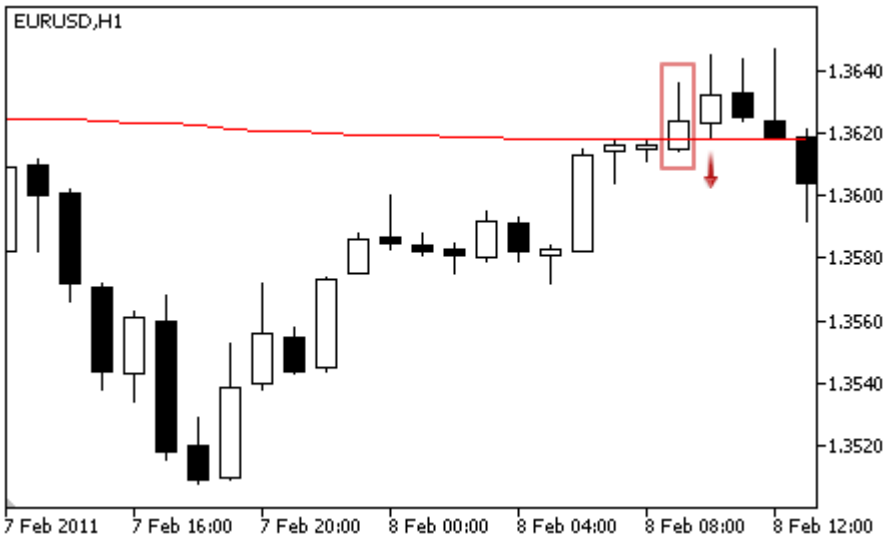
Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator wächst (schwaches Signal auf dem Rückzug aus der Indikatorlinie).  • **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator wächst (starkes Signal auf).

© 2000-2025, MetaQuotes Ltd.

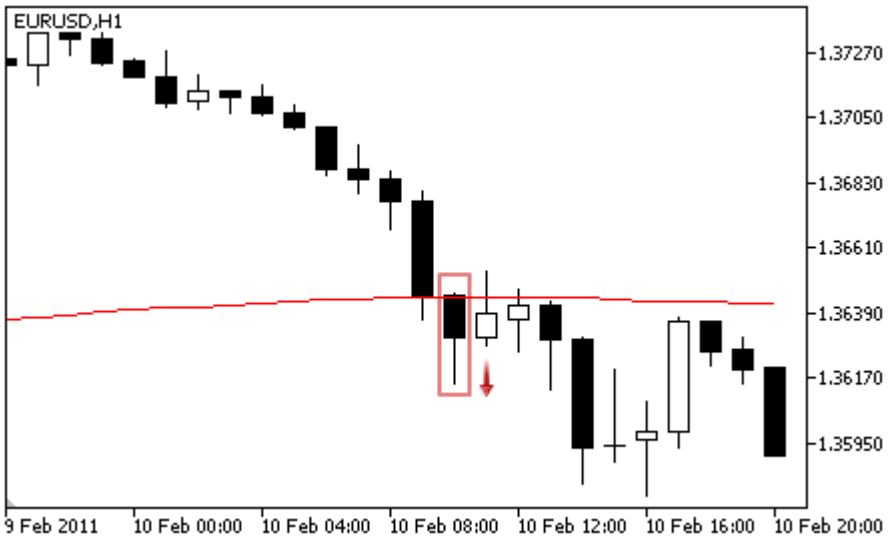
Signaltyp	Beschreibung der Bedingungen
	 • **Geformter Durchstich.** Der Preis hat den Indikator mit dem unteren Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind über der Indikatorlinie, der Low-Preis ist darunter), aber der Indikator wächst (Signal auf dem Rückzug aus der Indikatorlinie). 
Zum Verkauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator fällt (schwaches Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

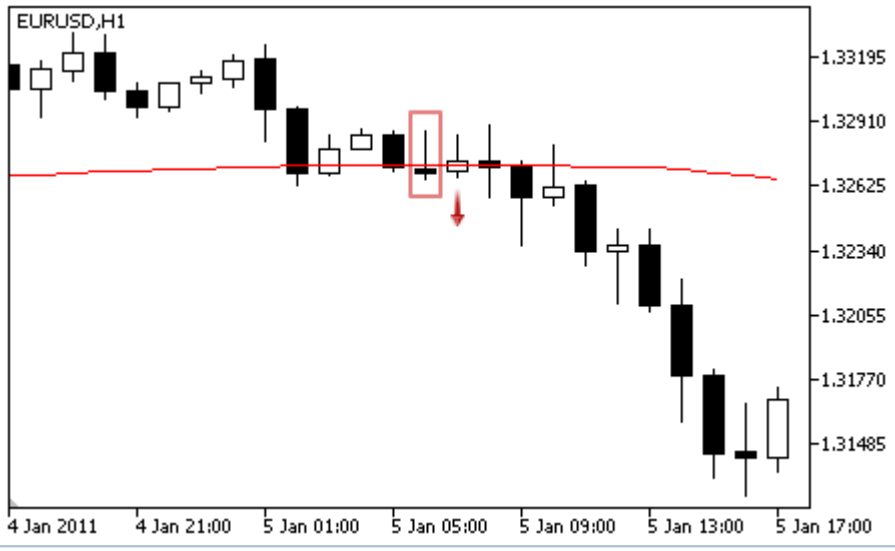


- **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator fällt (starkes Signal auf).



- **Geformter Durchstich.** Der Preis hat den Indikator mit dem oberen Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind unter der Indikatorlinie, der High-Preis ist darüber), aber der Indikator fällt (Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	
Nicht gegen den Kauf	Der Preis ist über dem Indikator.
Nicht gegen den Verkauf	Der Preis ist unter dem Indikator.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodMA	Mittelwertzeitraum des Indikators.
Shift	Verschiebung des Indikators entlang der Zeitachse (in Balken).
Method	Mittelungsverfahren.
Applied	Preistyp für die Berechnung von Indikatorwerten.

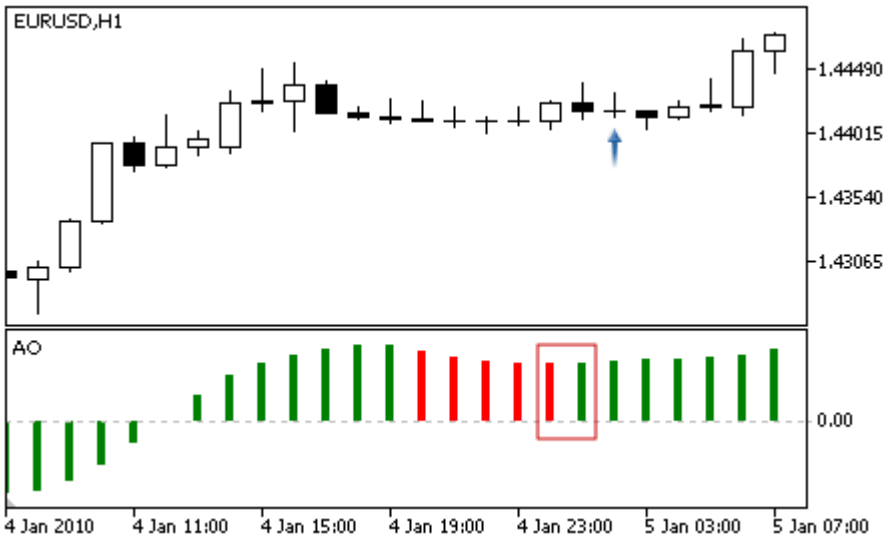
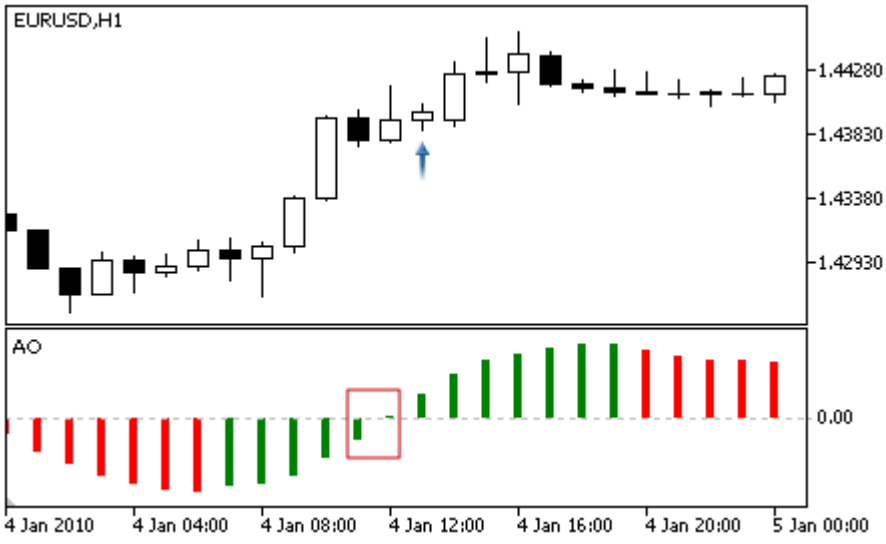
© 2000-2025, MetaQuotes Ltd.

Signale von Awesome Oscillator

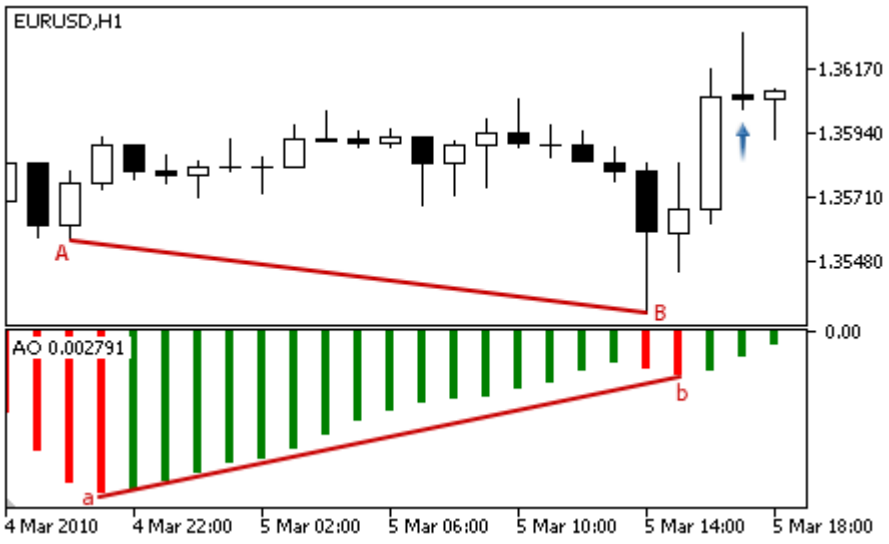
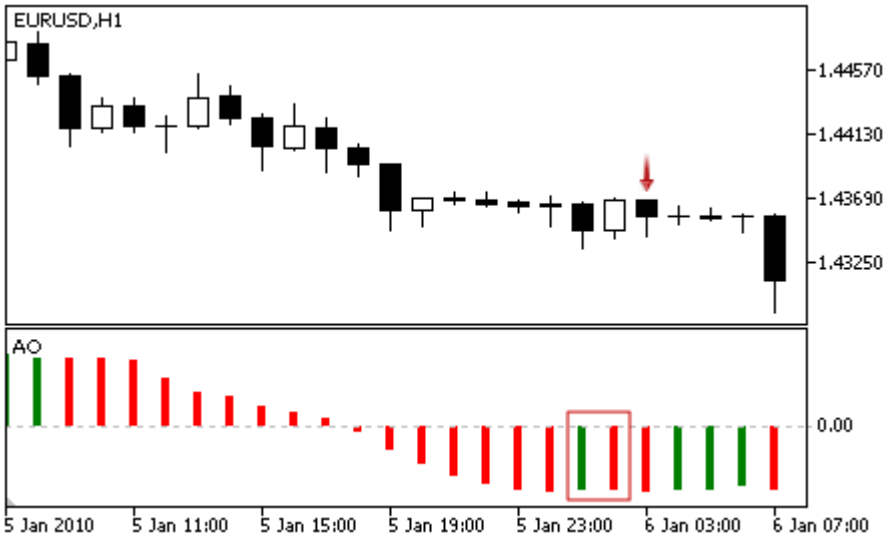
Dieses Signalmodul basiert auf Marktmodelle des Indikators Awesome Oscillator. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

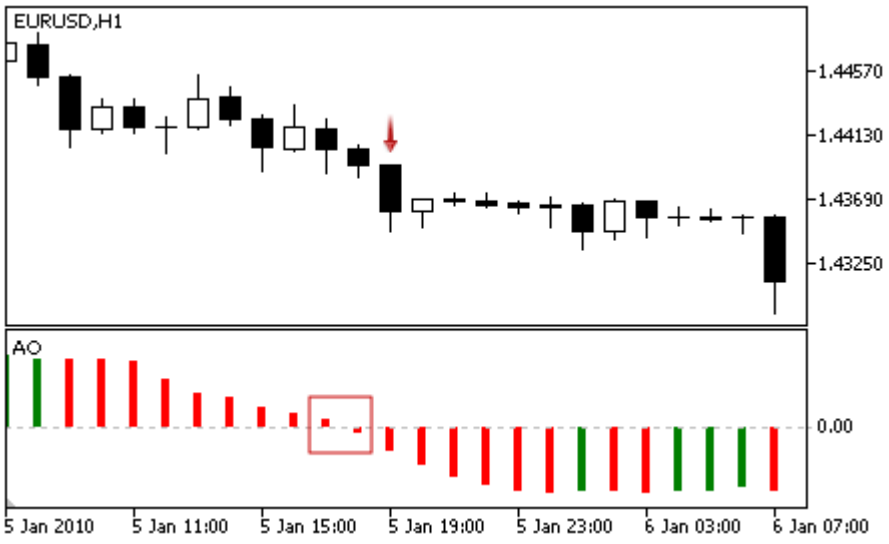
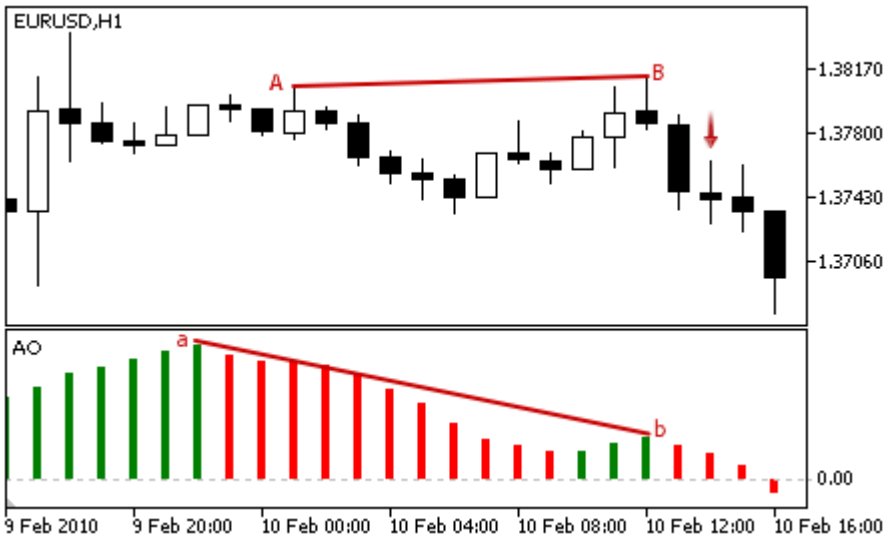
Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Untertasse** — der Indikatorwert wächst am analysierten Balken, und er fiel am vorherigen Balken mit beiden Werten größer als 0.  • **Nulldurchgang** — der Indikatorwert am analysierten Balken ist größer als 0, und es war kleiner als 0 am vorherigen Balken.  • **Divergenz** — das erste analysierte Tal des Indikators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige. Der Indikator sollte nicht über Null steigen.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	
Zum Verkauf	• **Untertasse** — der Indikatorwert fällt am analysierten Balken, und er wuchs am vorherigen Balken mit beiden Werten kleiner als 0.  • **Nulldurchgang** — der Indikatorwert am analysierten Balken ist kleiner als 0, und es war größer als 0 am vorherigen Balken.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Divergenz** — die erste analysierte Höhe des Indikators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige. Der Indikator sollte nicht unter Null fallen. 
Nicht gegen den Kauf	Der Indikatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Indikatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

© 2000-2025, MetaQuotes Ltd.

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.

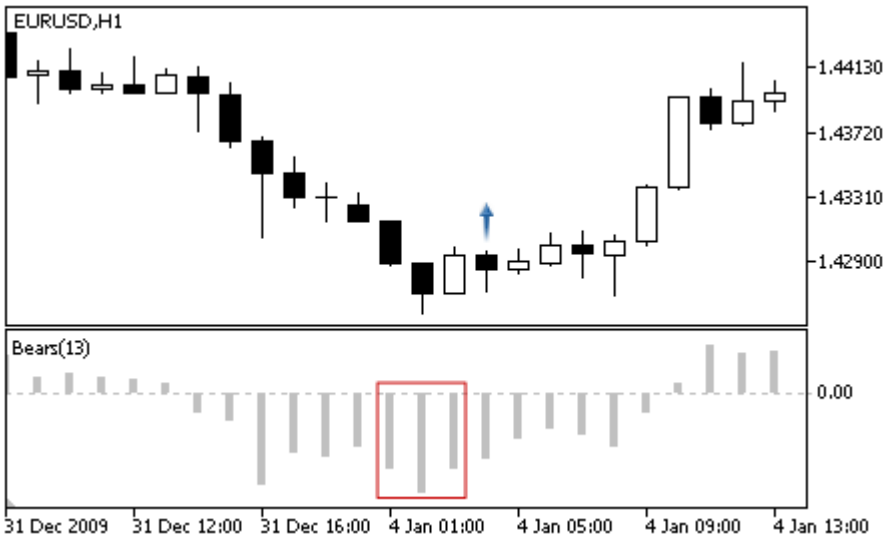
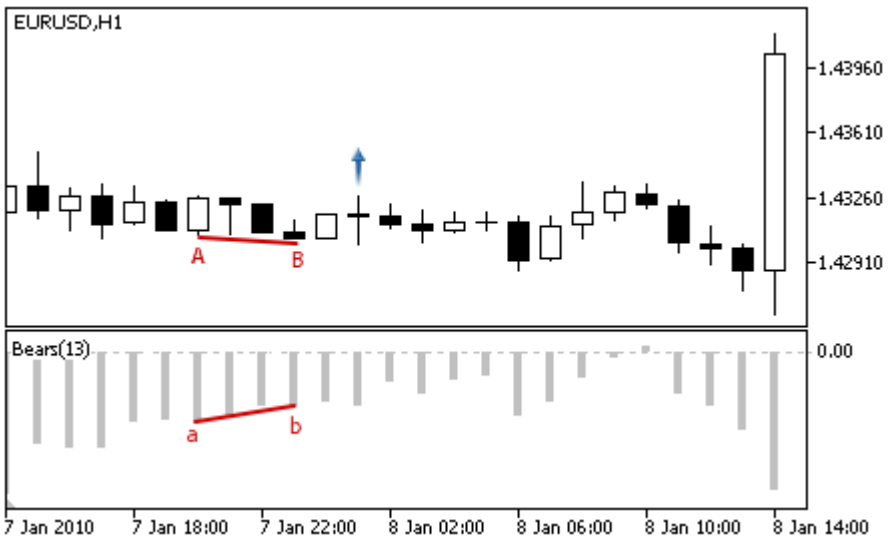
© 2000-2025, MetaQuotes Ltd.

Signale von Oszillator Bears Power

Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators Bears Power. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr** — Oszillator hat sich nach oben umgekehrt und sein Wert am analysierten Balken ist unter 0.  • **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige. Der Oszillator sollte nicht über Null steigen. 
Zum Verkauf	Keine Verkaufssignale.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
Nicht gegen den Kauf	Der Wert des Oszillators unter 0.
Nicht gegen den Verkauf	Keine Signale.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodBears	Der Berechnungszeitraum des Oszillators.

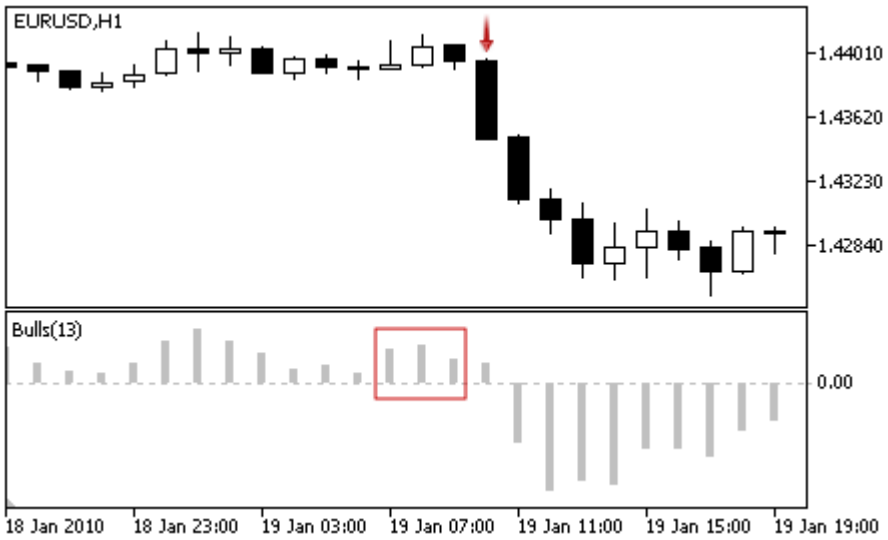
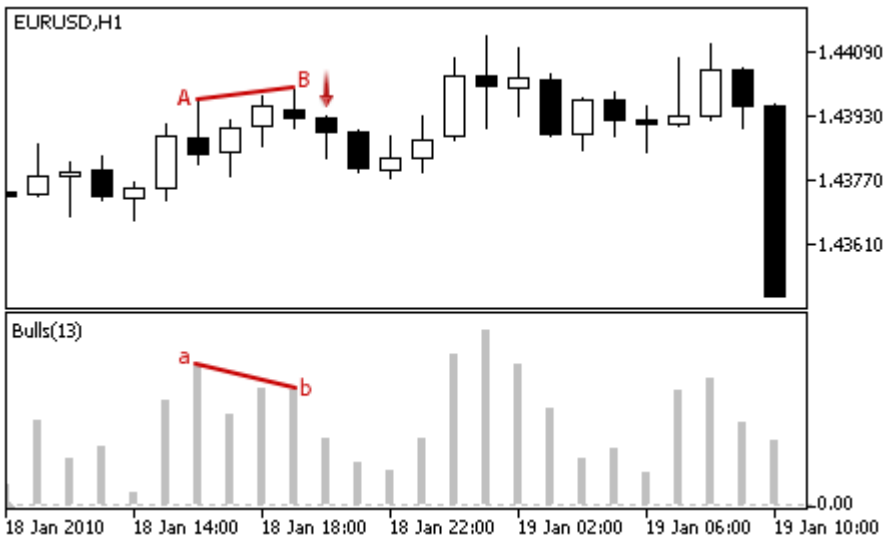
© 2000-2025, MetaQuotes Ltd.

Signale von Oszillator Bulls Power

Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators Bulls Power. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	Keine Kaufsignale.
Zum Verkauf	• **Umkehr** — Oszillator hat sich nach unten umgekehrt und sein Wert am analysierten Balken ist über 0.  • **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige. Der Oszillator sollte nicht unter Null fallen. 

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
Nicht gegen den Kauf	Keine Signale.
Nicht gegen den Verkauf	Der Wert des Oszillators über 0.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodBulls	Der Berechnungszeitraum des Oszillators.

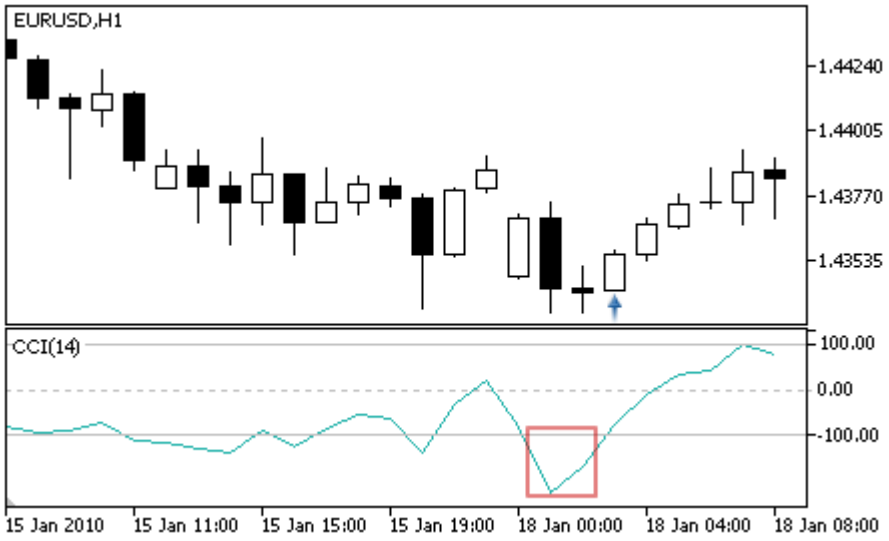
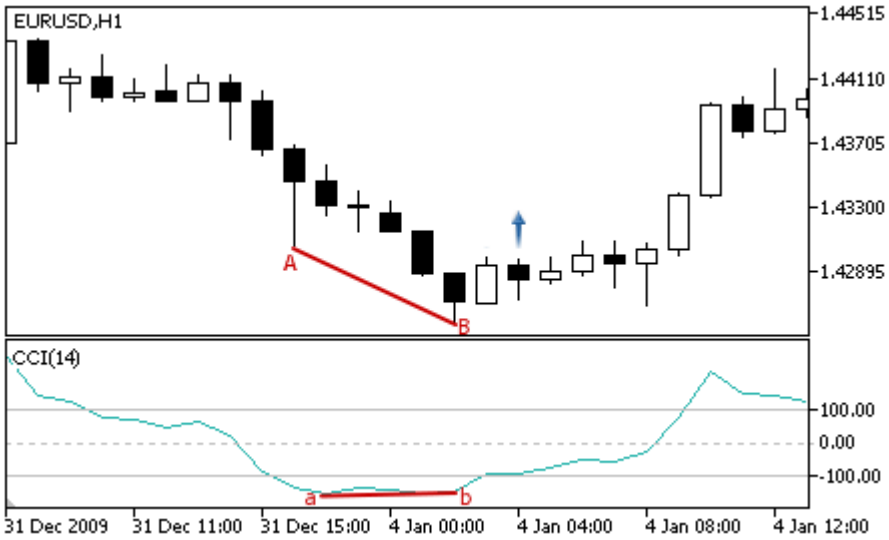
© 2000-2025, MetaQuotes Ltd.

Signale von Oszillator Commodity Channel Index

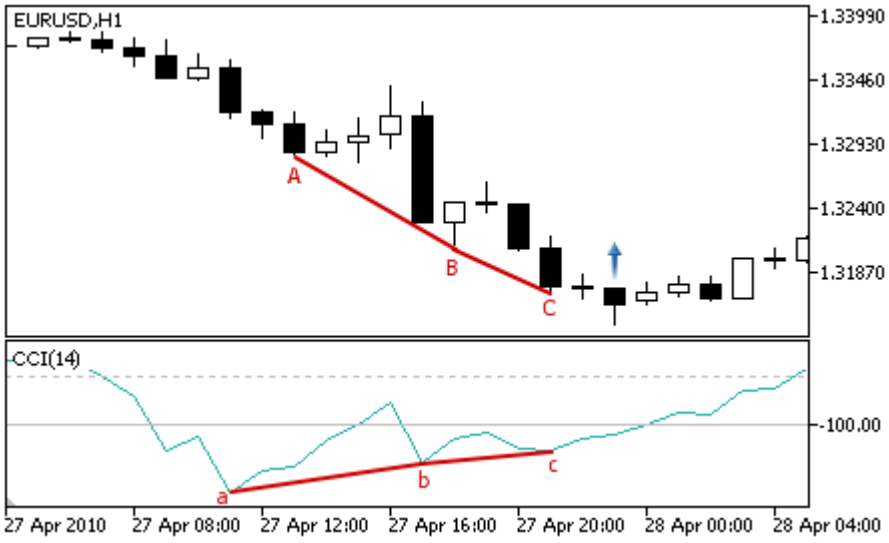
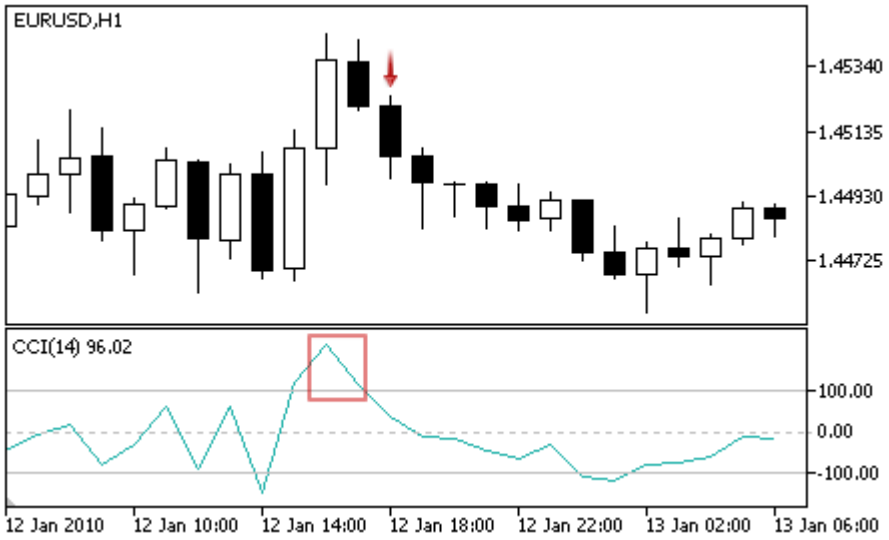
Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators Commodity Channel Index. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

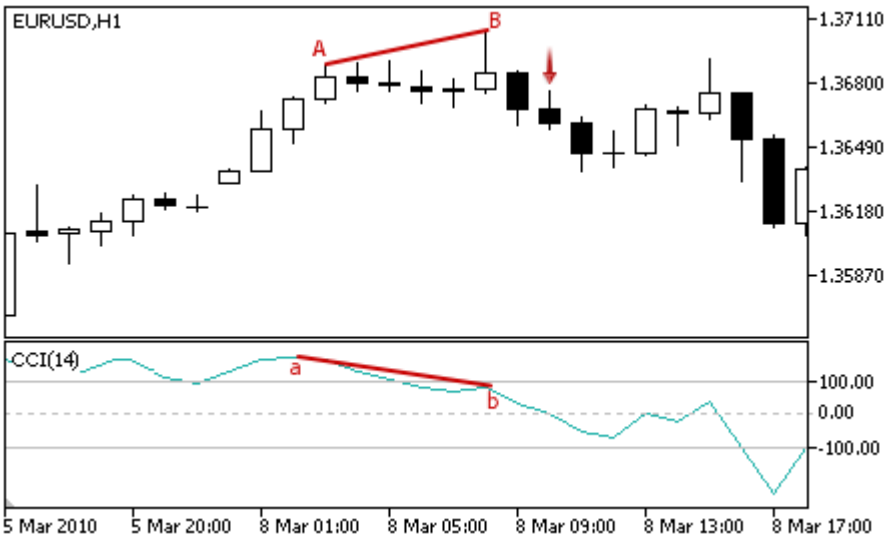
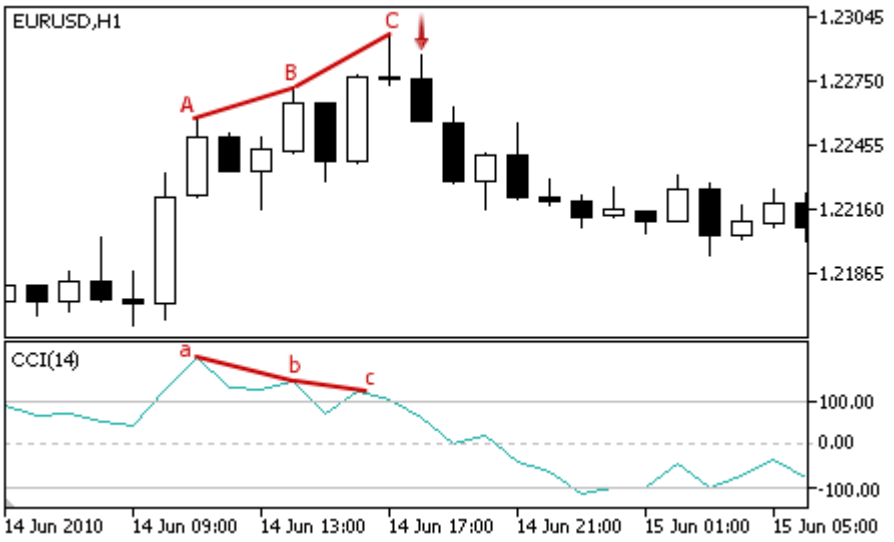
Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr nach dem überverkauften Niveau** — Oszillator hat sich nach oben umgekehrt und sein Wert am analysierten Balken ist hinter dem überverkauften Niveau (standardmäßig -100).  • **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige. 

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	• **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Täler gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Täler gebildet, von denen jede tiefer als der vorherige ist. 
Zum Verkauf	• **Umkehr nach dem überkauften Niveau** — Oszillator hat sich nach unten umgekehrt und sein Wert am analysierten Balken ist hinter dem überkauften Niveau (standardmäßig 100).  • **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Höhen gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Höhen gebildet, von denen jede größer als der vorherige ist. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

© 2000-2025, MetaQuotes Ltd.

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodCCI	Der Berechnungszeitraum des Oszillators.
Applied	Preistyp für die Berechnung von Oszillatorwerten.

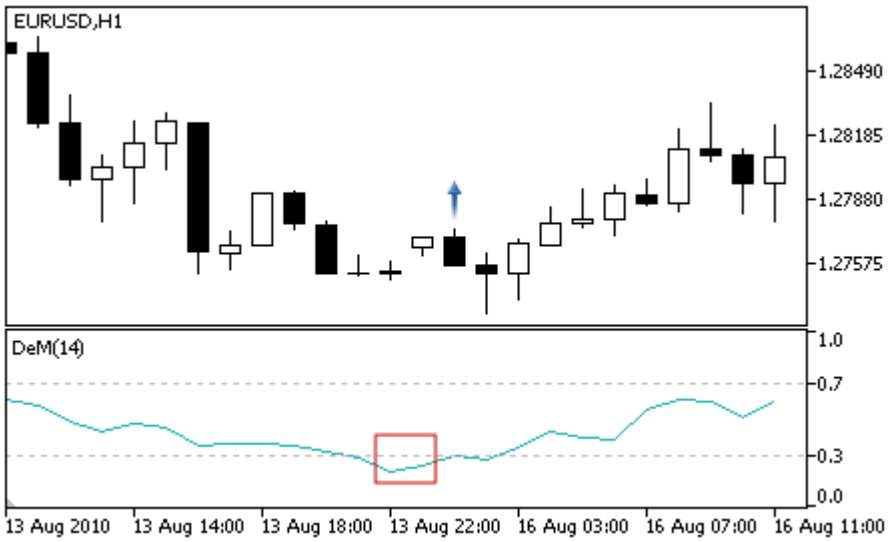
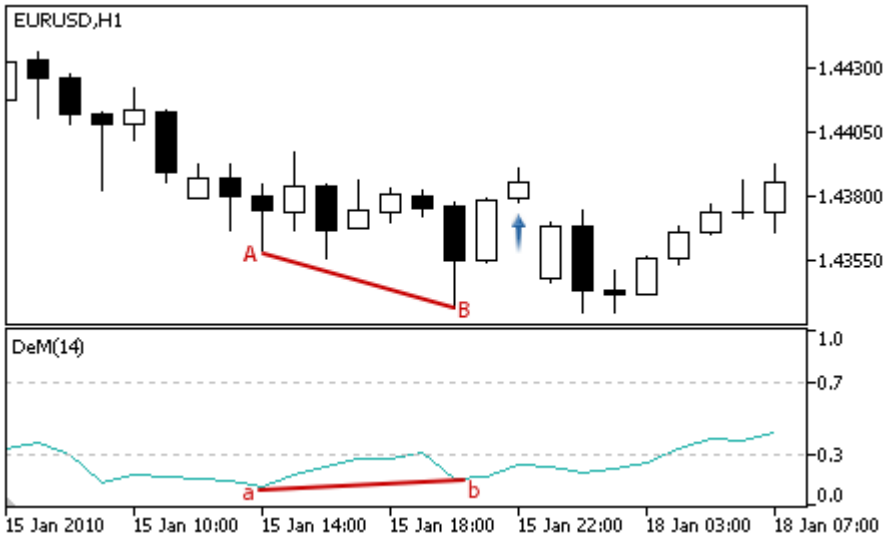
© 2000-2025, MetaQuotes Ltd.

Signale von DeMarker

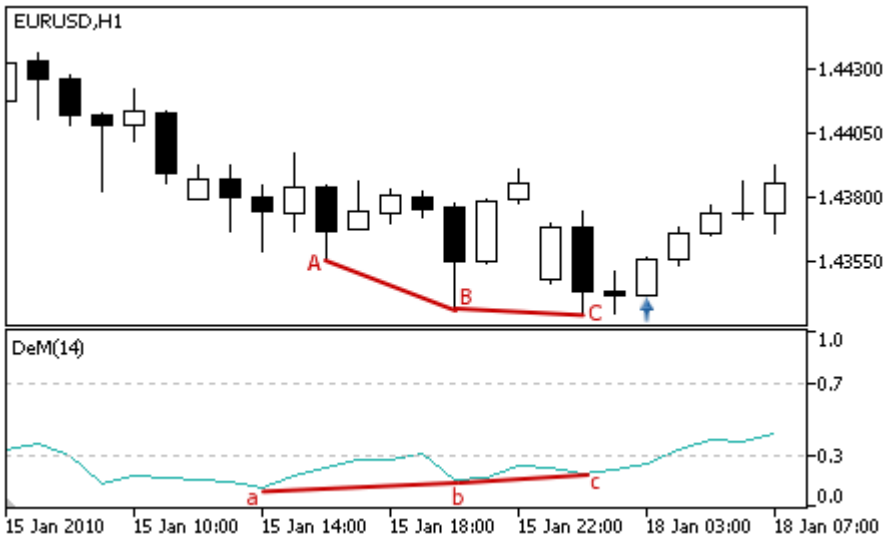
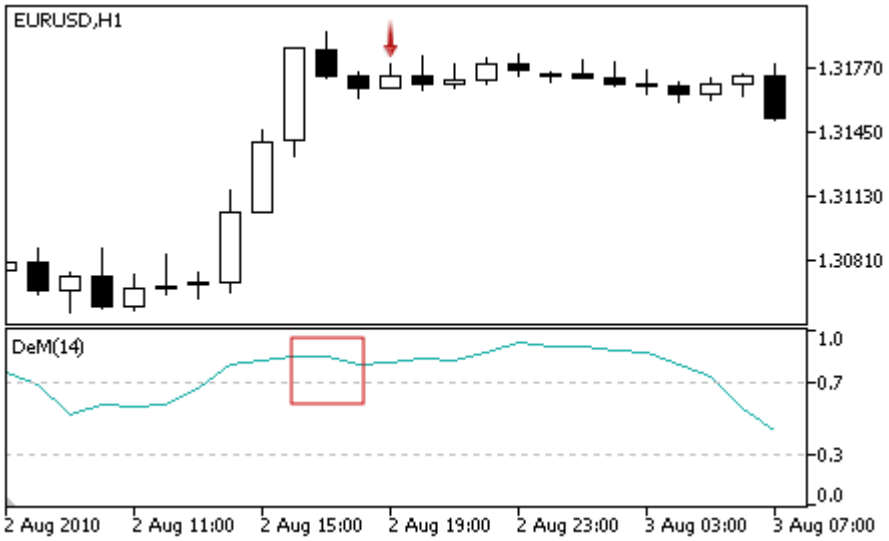
Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators DeMarker. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

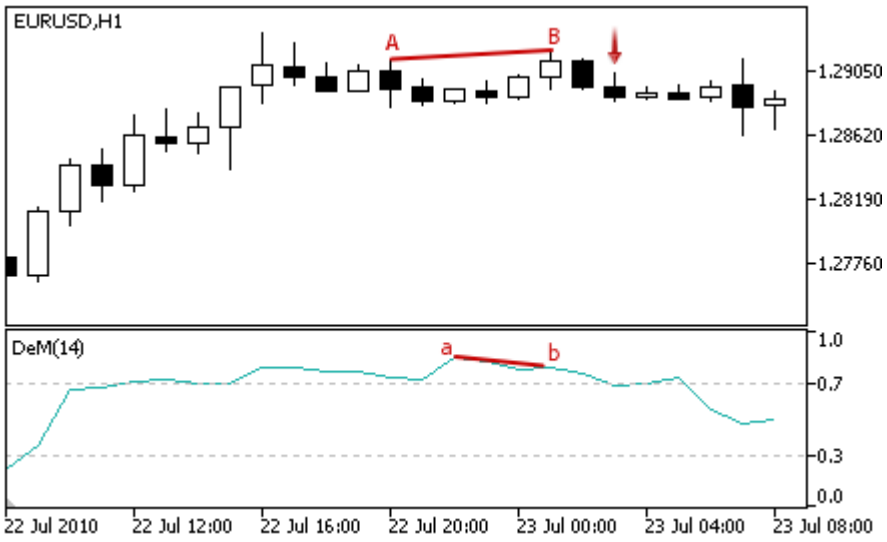
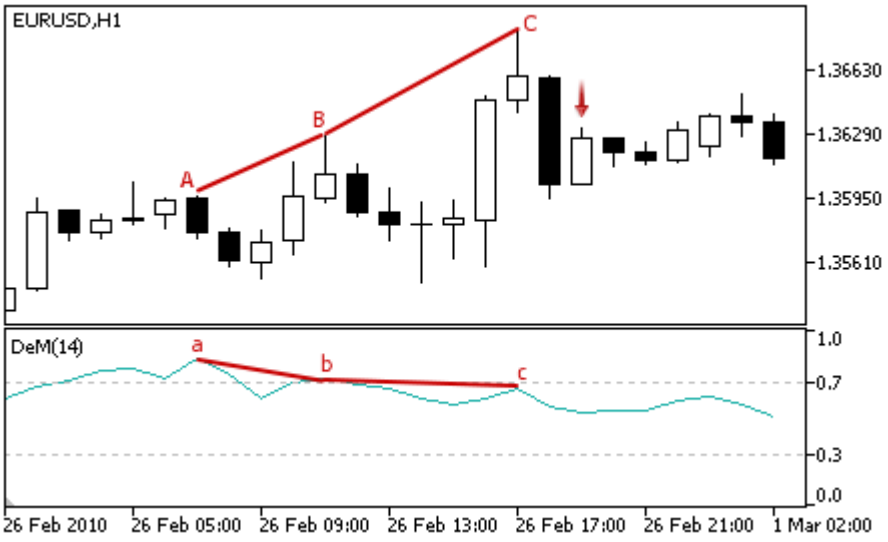
Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr nach dem überverkauften Niveau** — Oszillator hat sich nach oben umgekehrt und sein Wert am analysierten Balken ist hinter dem überverkauften Niveau (standardmäßig 0.3).  • **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige.  • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Täler gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	drei entsprechenden Täler gebildet, von denen jede tiefer als der vorherige ist. 
Zum Verkauf	• **Umkehr nach dem überkauften Niveau** — Oszillator hat sich nach unten umgekehrt und sein Wert am analysierten Balken ist hinter dem überkauften Niveau (standardmäßig 0.7).  • **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Höhen gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Höhen gebildet, von denen jede größer als der vorherige ist. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

© 2000-2025, MetaQuotes Ltd.

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodDeM	Der Berechnungszeitraum des Oszillators.

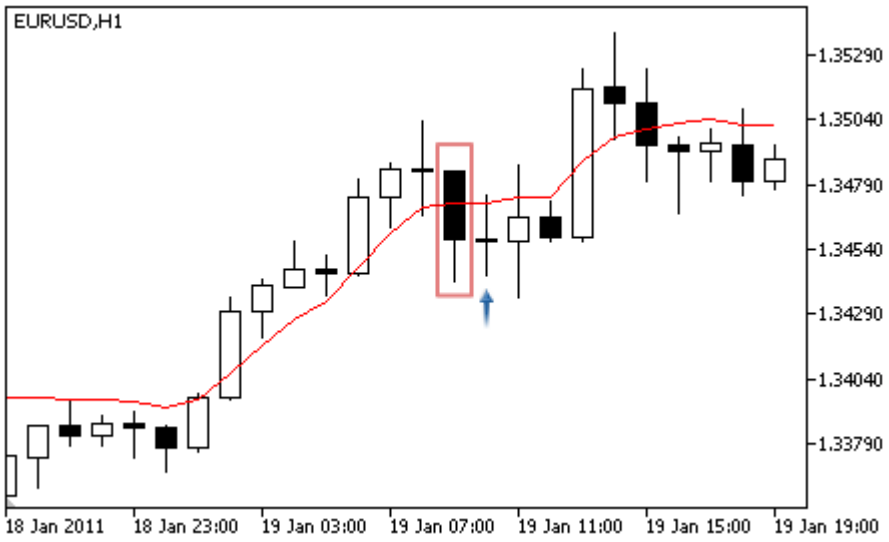
© 2000-2025, MetaQuotes Ltd.

Signale von Exponential Moving Average

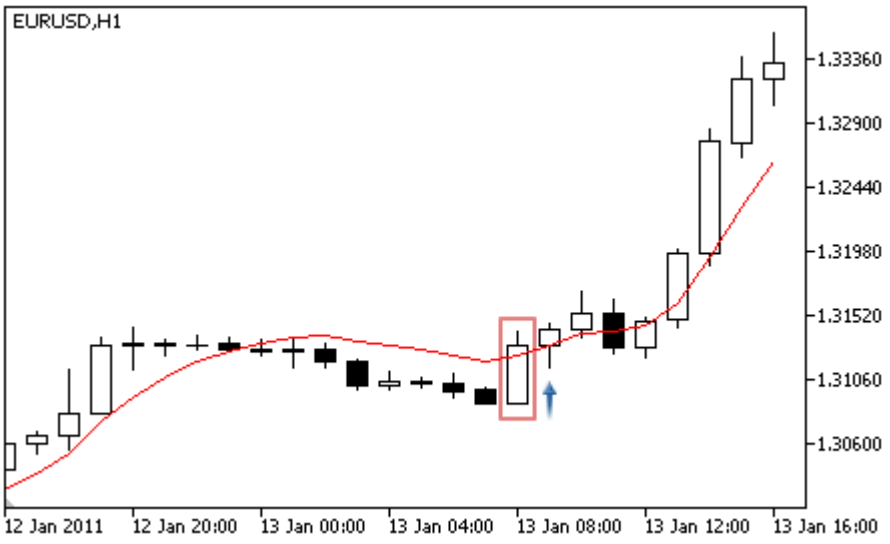
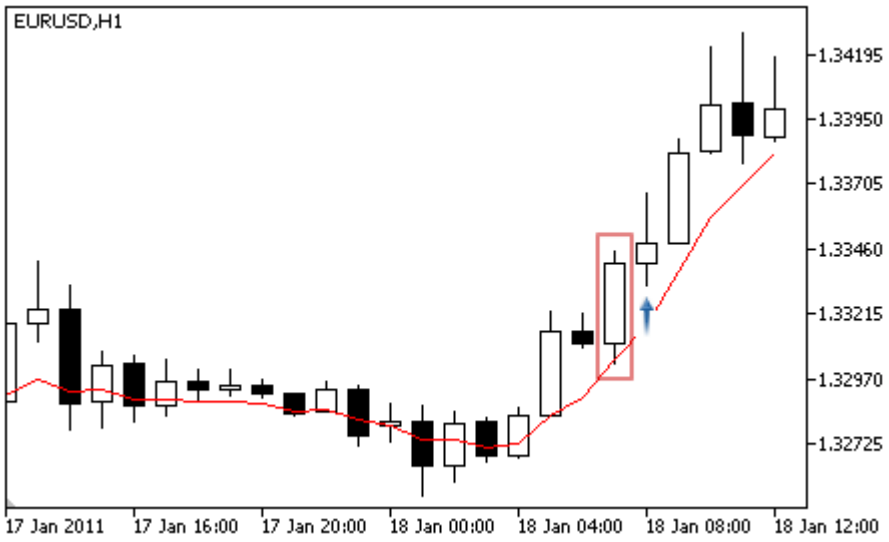
Dieses Signalmodul basiert auf Marktmodelle des Indikators Double Exponential Moving Average. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

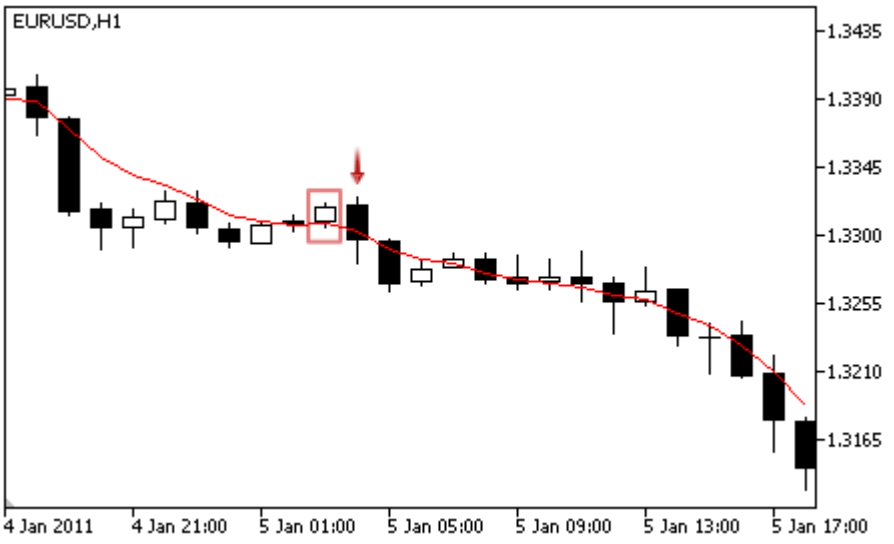
Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator wächst (schwaches Signal auf dem Rückzug aus der Indikatorlinie).  • **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator wächst (starkes Signal auf).

© 2000-2025, MetaQuotes Ltd.

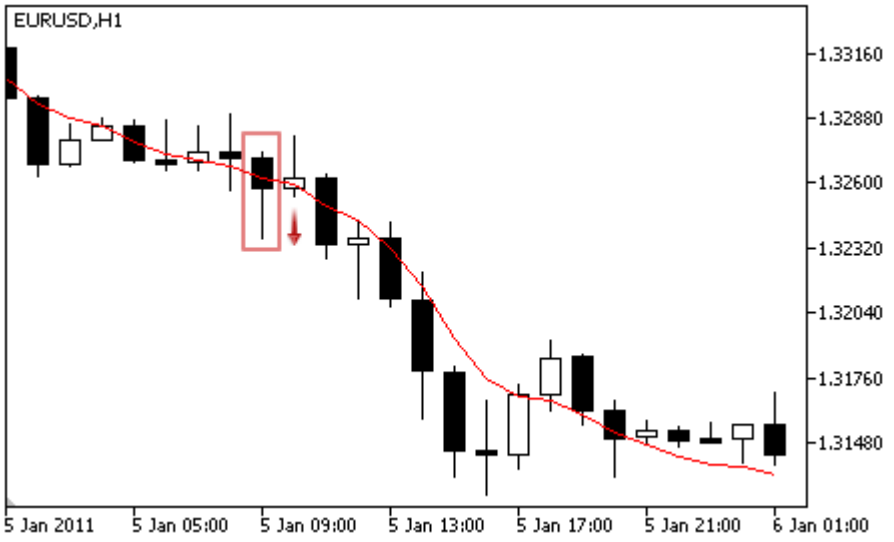
Signaltyp	Beschreibung der Bedingungen
	 • **Geformter Durchstich.** Der Preis hat den Indikator mit dem unteren Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind über der Indikatorlinie, der Low-Preis ist darunter), aber der Indikator wächst (Signal auf dem Rückzug aus der Indikatorlinie). 
Zum Verkauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator fällt (schwaches Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

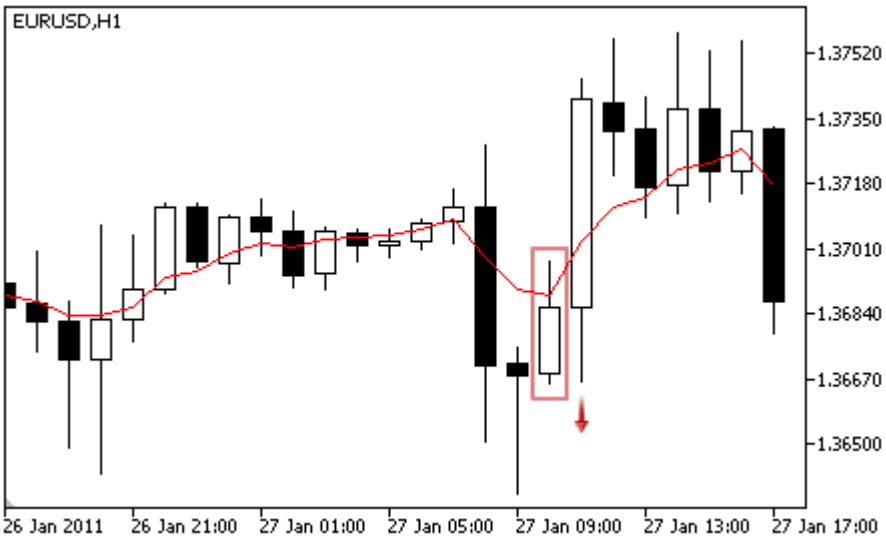


- **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator fällt (starkes Signal auf).



- **Geformter Durchstich.** Der Preis hat den Indikator mit dem oberen Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind unter der Indikatorlinie, der High-Preis ist darüber), aber der Indikator fällt (Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	
Nicht gegen den Kauf	Der Preis ist über dem Indikator.
Nicht gegen den Verkauf	Der Preis ist unter dem Indikator.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodMA	Mittelwertzeitraum des Indikators.
Shift	Verschiebung des Indikators entlang der Zeitachse (in Balken).
Method	Mittelungsverfahren.
Applied	Preistyp für die Berechnung von Indikatorwerten.

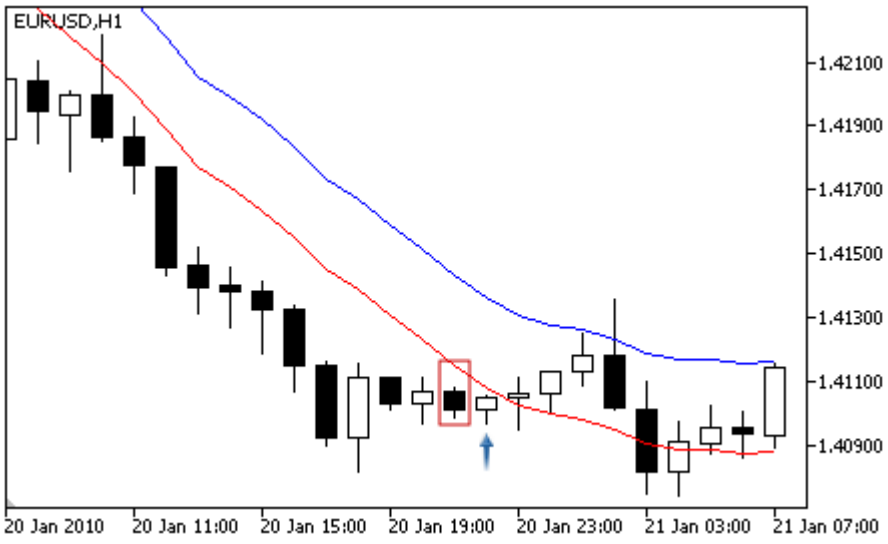
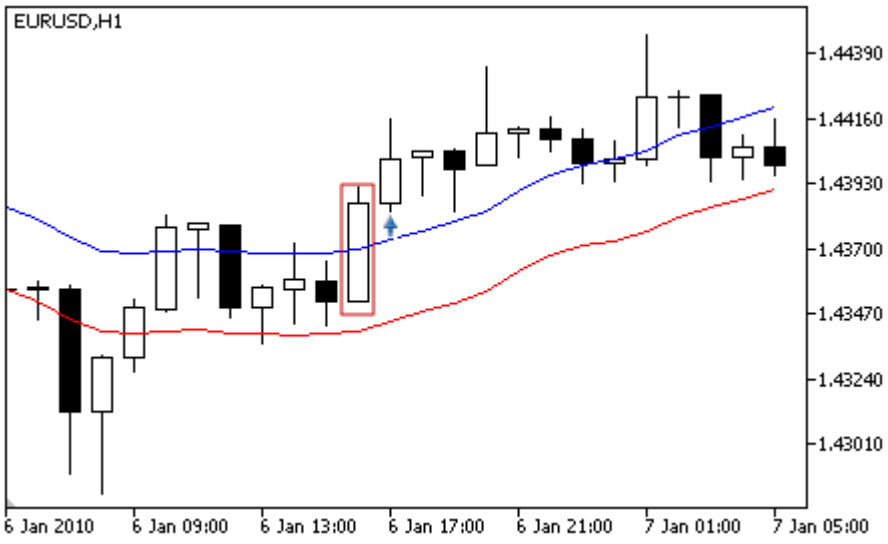
© 2000-2025, MetaQuotes Ltd.

Signale des Indikators Envelopes

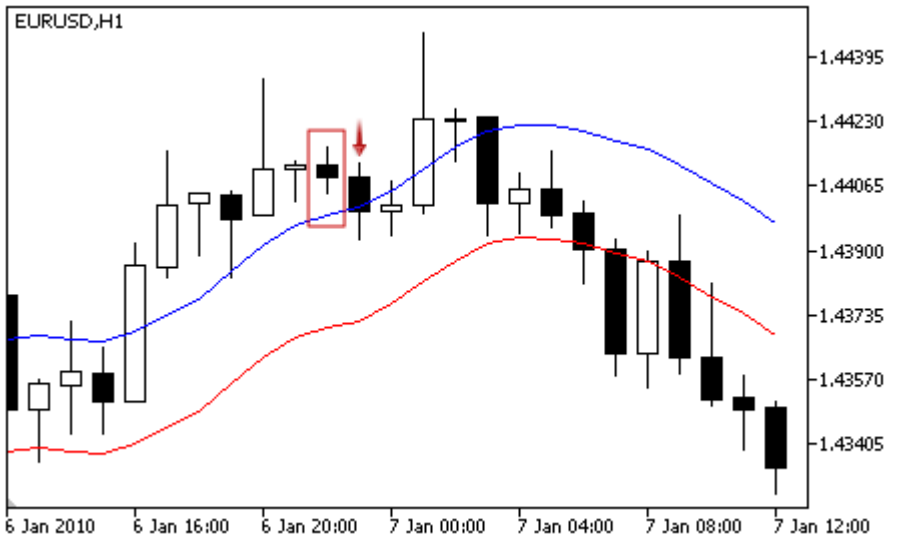
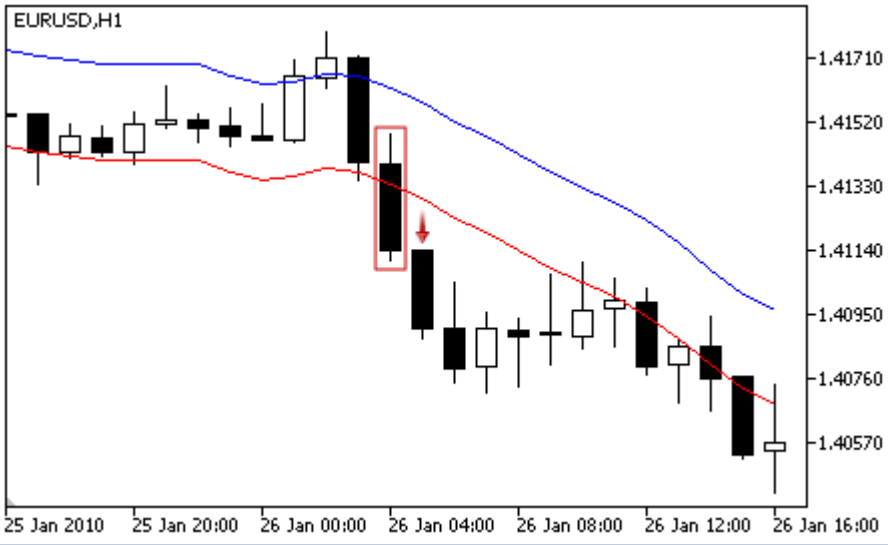
Dieses Signalmodul basiert auf Marktmodelle des Indikators Envelopes. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• Auf dem analysierten Balken ist die Preis nah an der unteren Linie des Indikators.  • Auf dem analysierten Balken hat die Preis die obere Linie des Indikators überschritten. 
Zum Verkauf	• Auf dem analysierten Balken ist die Preis nah an der oberen Linie des Indikators.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • Auf dem analysierten Balken hat die Preis die untere Linie des Indikators überschritten. 
Nicht gegen den Kauf	Keine Signale.
Nicht gegen den Verkauf	Keine Signale.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

© 2000-2025, MetaQuotes Ltd.

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodMA	Berechnungszeitraum des Indikators.
Shift	Verschiebung des Indikators entlang der Zeitachse (in Balken).
Method	Mittelungsverfahren.
Applied	Preistyp für die Berechnung von Indikatorwerten.
Deviation	Die Abweichung der Envelope-Grenzen von der Mittellinie (MA) als Prozentsatz.

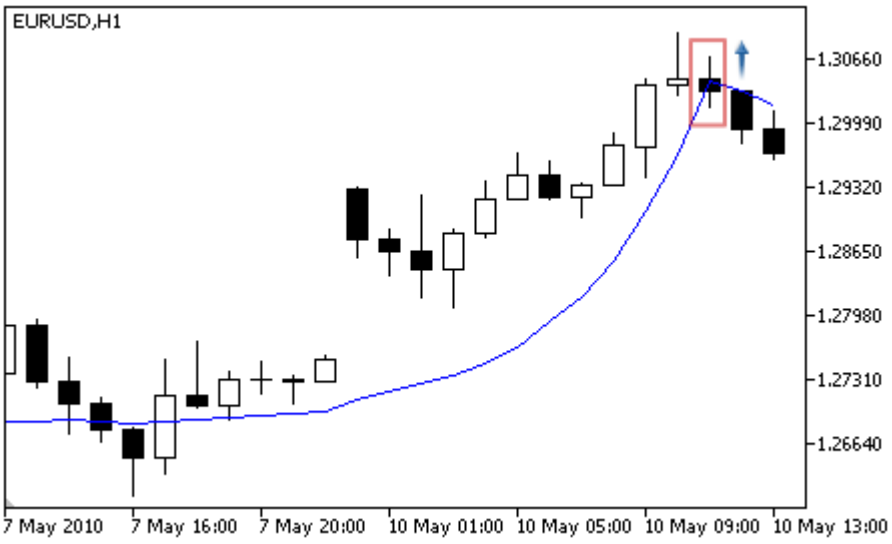
© 2000-2025, MetaQuotes Ltd.

Signale von Fractal Adaptive Moving Average

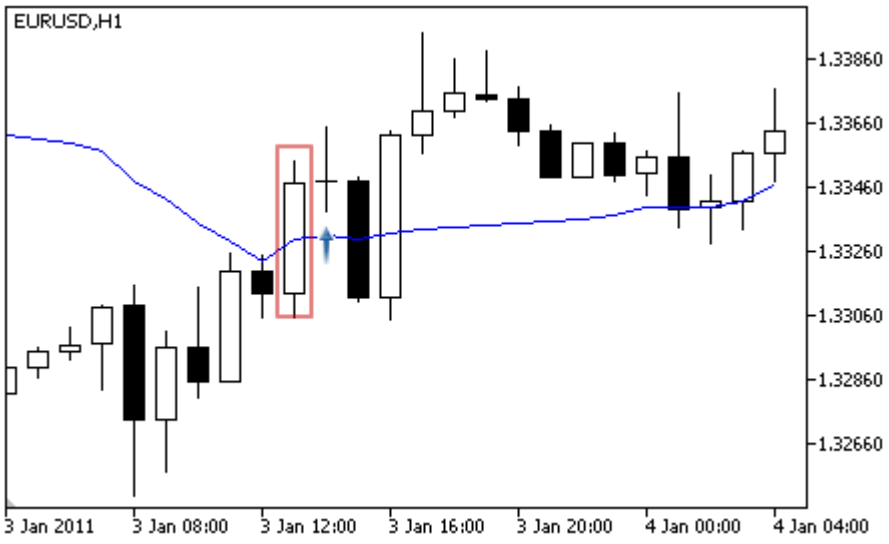
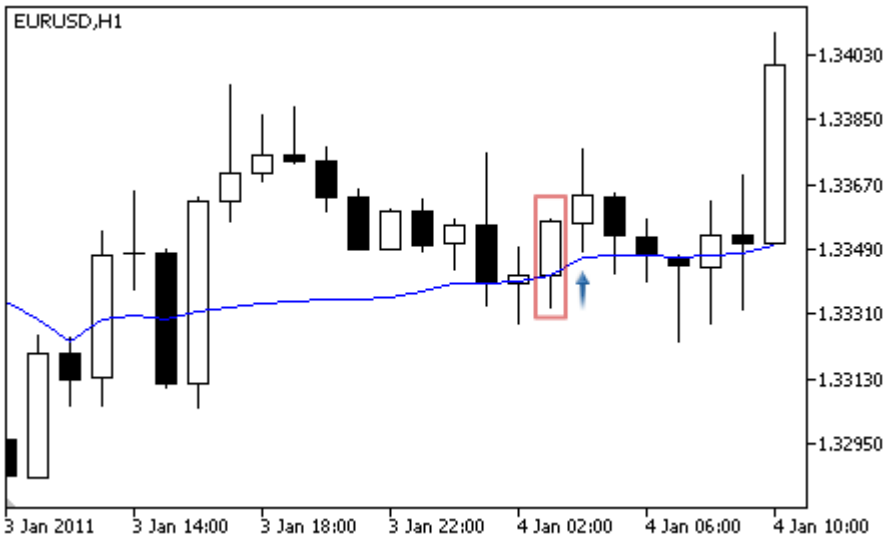
Dieses Signalmodul basiert auf Marktmodelle des Indikators Fractal Adaptive Moving Average. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

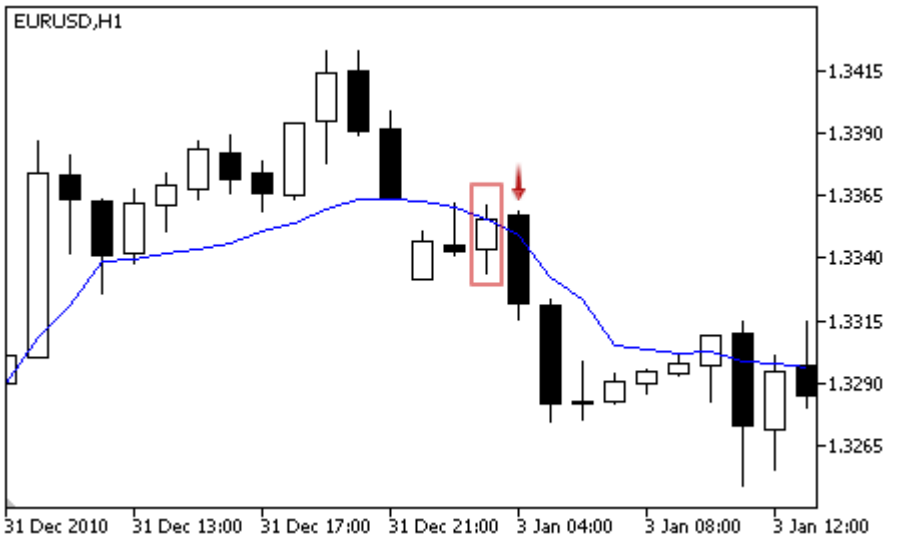
Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator wächst (schwaches Signal auf dem Rückzug aus der Indikatorlinie).  • **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator wächst (starkes Signal auf).

© 2000-2025, MetaQuotes Ltd.

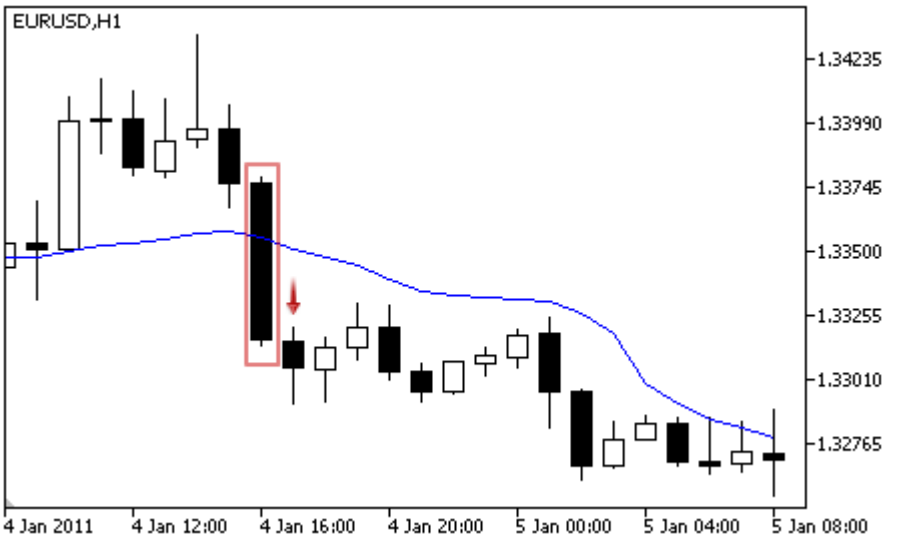
Signaltyp	Beschreibung der Bedingungen
	 • **Geformter Durchstich.** Der Preis hat den Indikator mit dem unteren Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind über der Indikatorlinie, der Low-Preis ist darunter), aber der Indikator wächst (Signal auf dem Rückzug aus der Indikatorlinie). 
Zum Verkauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator fällt (schwaches Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

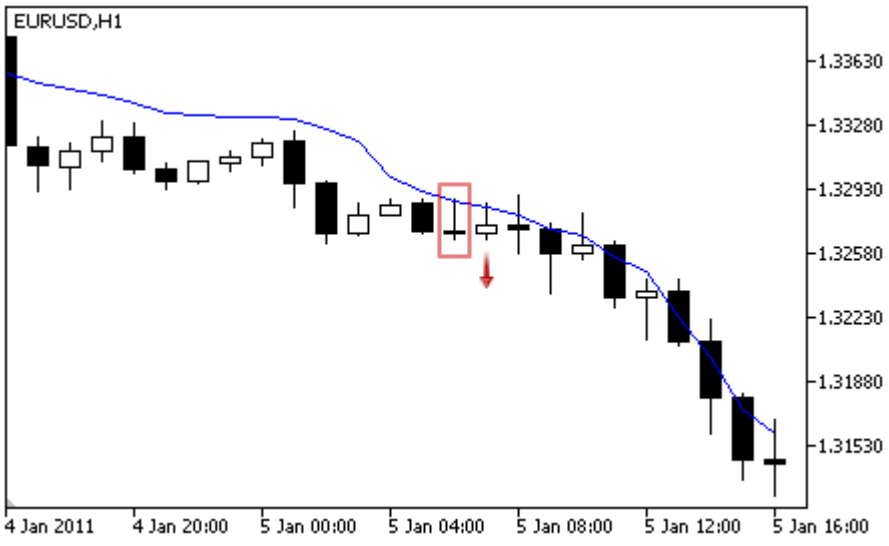


- **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator fällt (starkes Signal auf).



- **Geformter Durchstich.** Der Preis hat den Indikator mit dem oberen Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind unter der Indikatorlinie, der High-Preis ist darüber), aber der Indikator fällt (Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	
Nicht gegen den Kauf	Der Preis ist über dem Indikator.
Nicht gegen den Verkauf	Der Preis ist unter dem Indikator.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodMA	Mittelwertzeitraum des Indikators.
Shift	Verschiebung des Indikators entlang der Zeitachse (in Balken).
Method	Mittelungsverfahren.
Applied	Preistyp für die Berechnung von Indikatorwerten.

© 2000-2025, MetaQuotes Ltd.

Signale von Intraday-Zeitfilter

Dieses Signal-Modul basiert darauf, dass die Wirksamkeit der Marktmodelle im Laufe der Zeit ändert. Mit diesem Modul kann man die von anderen Module empfangenen Signale auf Stunden und Tage der Woche herausfiltern. Dies ermöglicht die Verbesserung der Qualität generierter Signale durch Filterung von der ungünstigen Zeit. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	Keine Signale.
Zum Verkauf	Keine Signale.
Nicht gegen den Kauf	Die aktuelle Uhrzeit und Wochentag entsprechen den angegebenen Parametern.
Nicht gegen den Verkauf	Die aktuelle Uhrzeit und Wochentag entsprechen den angegebenen Parametern.

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
GoodHourOfDay	Die einzige Stunde des Tages (0-23), in welcher die Handelssignale erlaubt sind. Bei einem Wert von -1 sind die Signale rund um die Uhr erlaubt.
BadHoursOfDay	Ein Bit-Feld, wobei jedes Bit einer Stunde des Tages entspricht (0. Bit - 0 Uhr, ..., 23. Bit - 23 Uhr). Wenn der Bit-Wert 0 ist, werden die entsprechenden Handelssignale in dieser Stunde erlaubt. Wenn der Bit-Wert 1 ist, werden die entsprechenden Handelssignale in dieser Stunde verboten. Die angegebene Anzahl wird in binärer Form dargestellt und wird in der Form einer Bitmaske verwendet. Verbotene Stunden sind der höhere Priorität.
GoodDayOfWeek	Der einzige Wochentag (0-6, 0 ist Sonntag), an welchem die Handelssignale erlaubt sind. Bei einem Wert von -1 sind die Signale an jedem Tag der Woche erlaubt.
BadDaysOfWeek	Ein Bit-Feld, wobei jedes Bit einem Wochentag entspricht (0. Bit - Sonntag, ..., 6. Bit - Samstag). Wenn der Bit-Wert 0 ist, werden die entsprechenden Handelssignale an diesem Tag erlaubt. Wenn der Bit-Wert 1 ist, werden die entsprechenden Handelssignale an diesem Tag verboten.

© 2000-2025, MetaQuotes Ltd.

Parameter	Beschreibung
	Die angegebene Anzahl wird in binärer Form dargestellt und wird in der Form einer Bitmaske verwendet. Verbotene Tage sind der höhere Priorität.

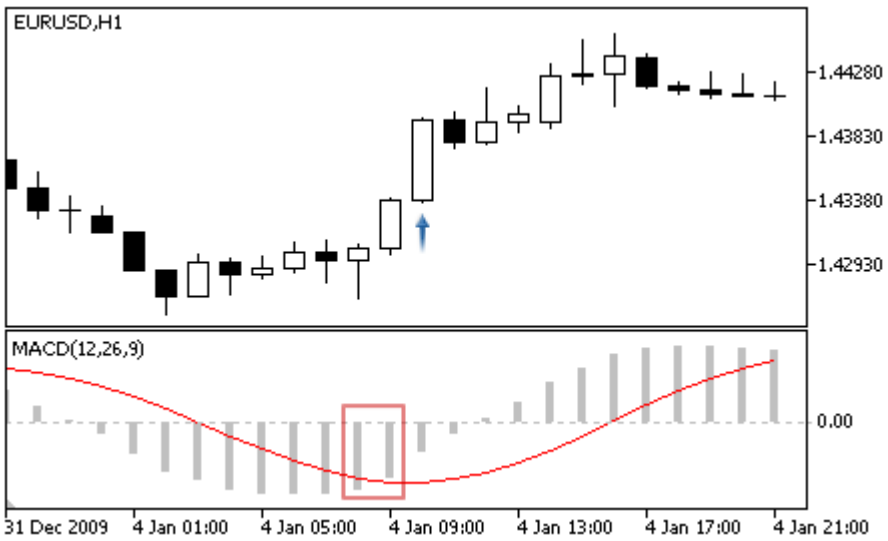
© 2000-2025, MetaQuotes Ltd.

Signale des Oszillators MACD

Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators MACD. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

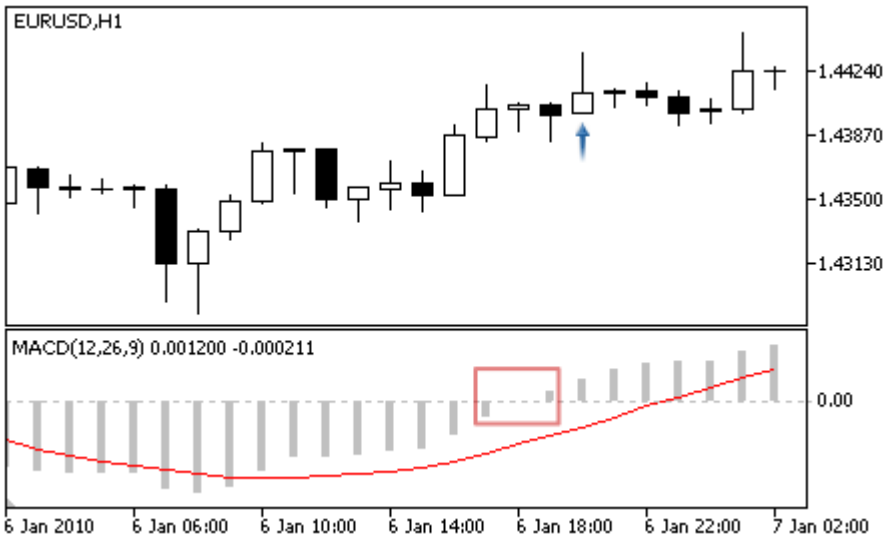
Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

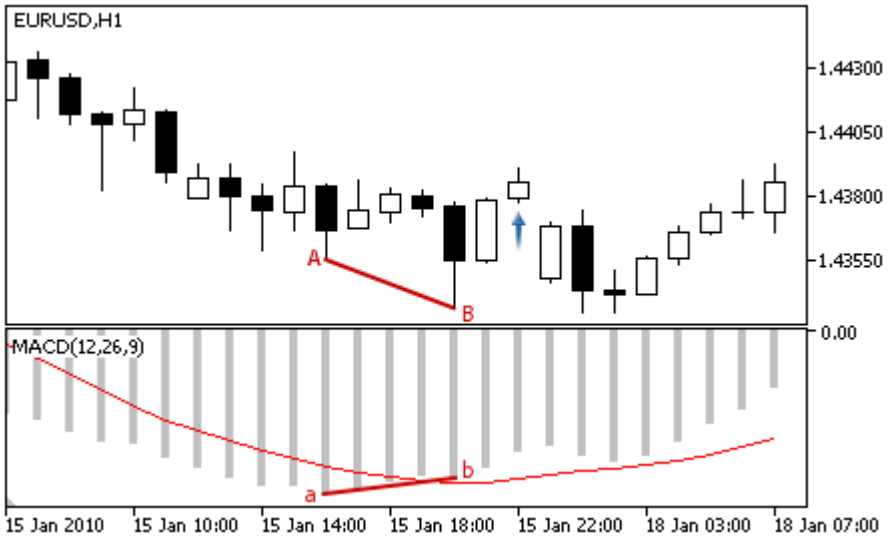
Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr** — der Oszillator hat sich nach oben umgekehrt (der Oszillator wächst auf dem analysierten Balken und fiel auf dem vorherigen).  • **Die Haupt- und Signallinie schneiden** — auf dem analysierten Balken ist die Hauptlinie über der Signallinie und war darunter auf dem vorherigen Balken.  • **Nulldurchgangs** — auf dem analysierten Balken ist die Hauptlinie über dem Nullwert und war darunter auf dem vorherigen Balken.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

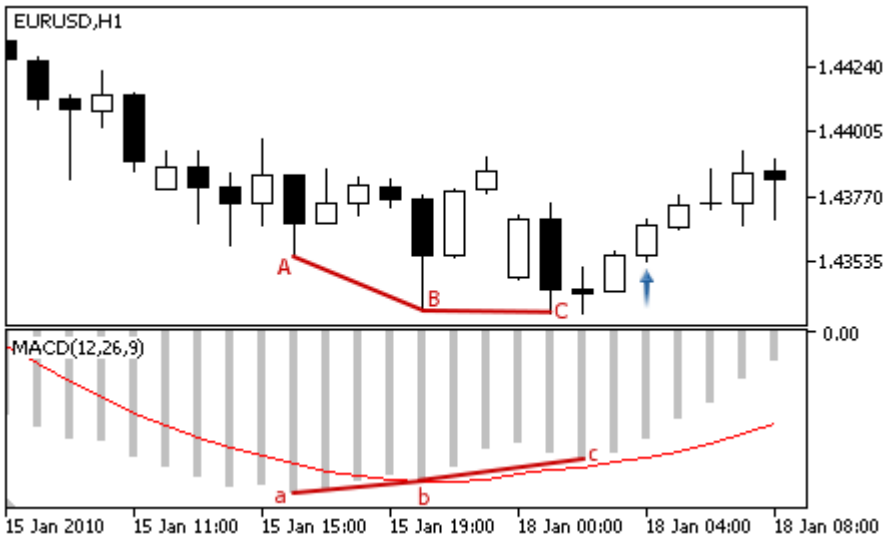
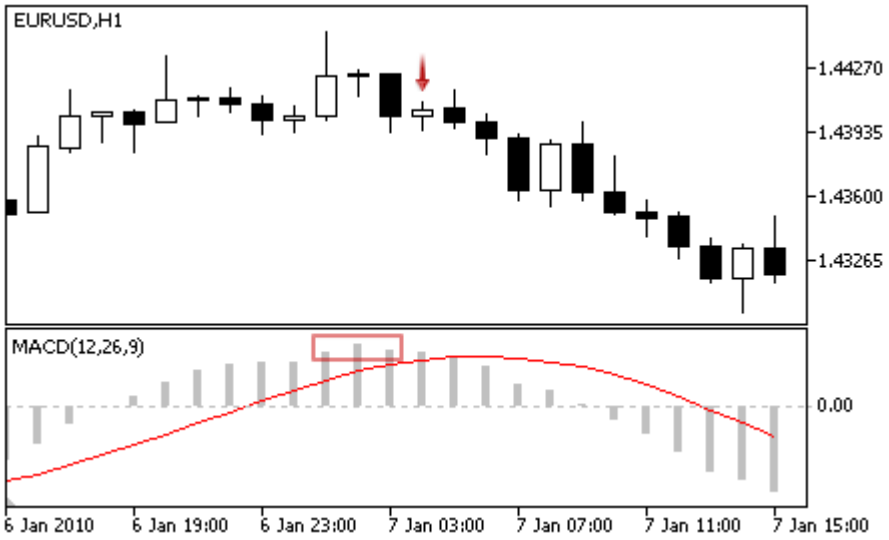


- **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige.



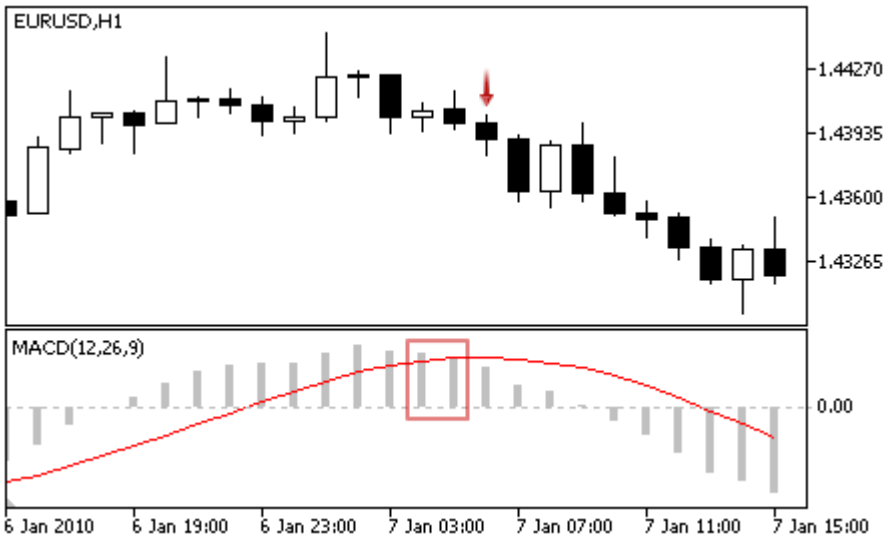
- **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Täler gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Täler gebildet, von denen jede tiefer als der vorherige ist.

© 2000-2025, MetaQuotes Ltd.

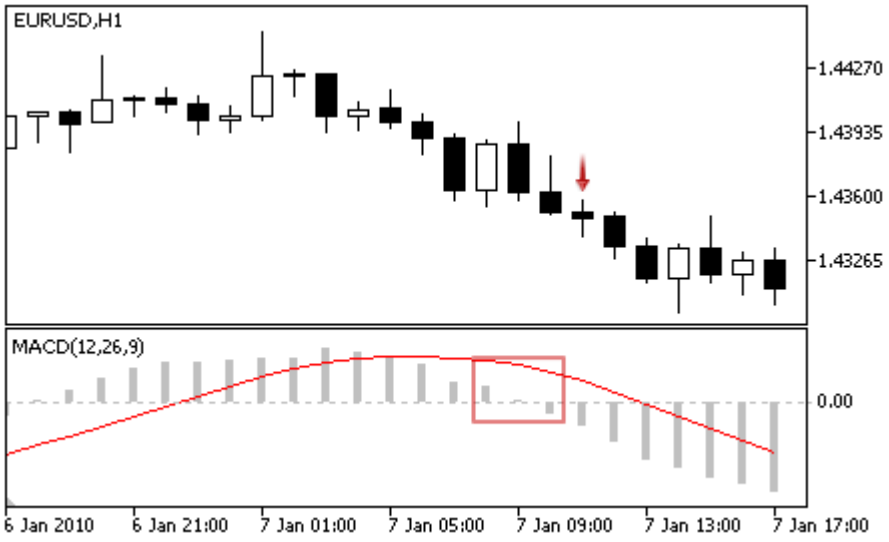
Signaltyp	Beschreibung der Bedingungen
	
Zum Verkauf	• **Umkehr** — der Oszillator hat sich nach unten umgekehrt (der Oszillator fällt auf dem analysierten Balken und wuchs auf dem vorherigen).  • **Die Haupt- und Signallinie schneiden** — auf dem analysierten Balken ist die Hauptlinie unter der Signallinie und war darüber auf dem vorherigen Balken.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

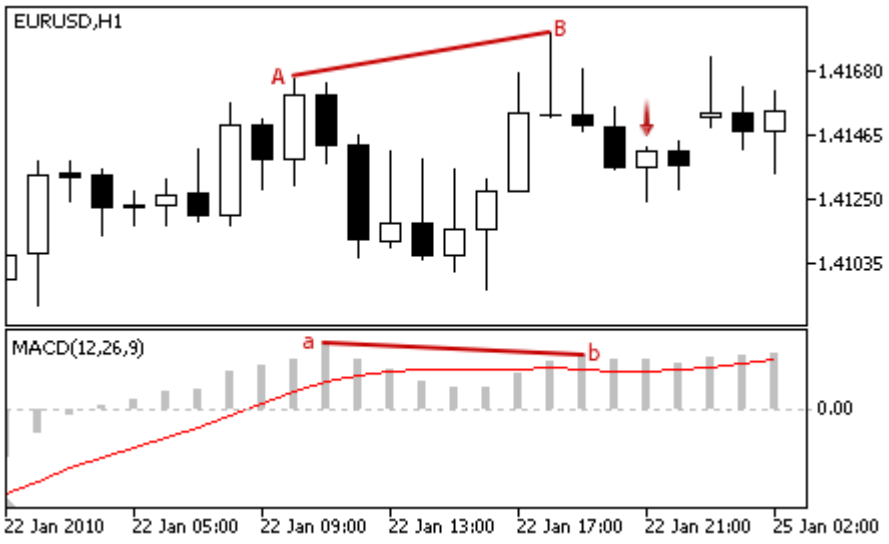
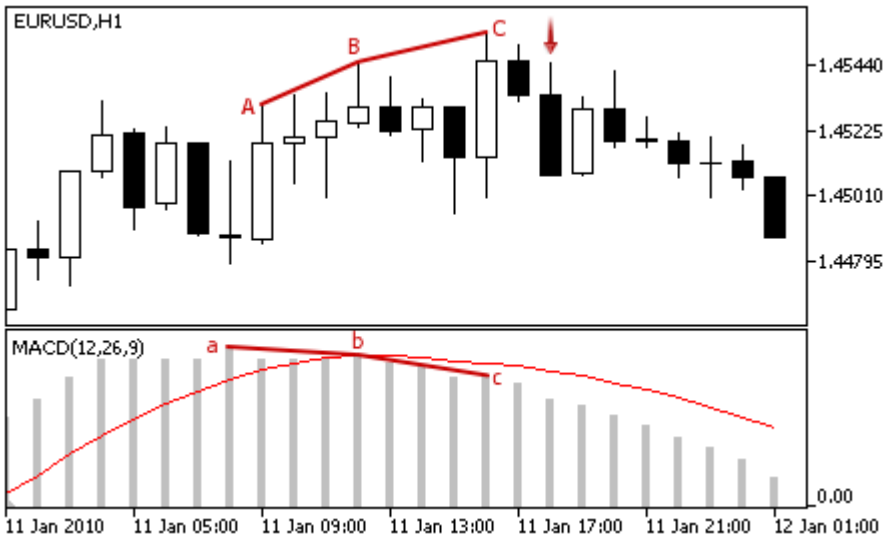


- **Nulldurchgangs** — auf dem analysierten Balken ist die Hauptlinie unter dem Nullwert und war darüber auf dem vorherigen Balken.



- **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Höhen gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Höhen gebildet, von denen jede größer als der vorherige ist. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

© 2000-2025, MetaQuotes Ltd.

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodFast	Die Berechnungsperiode der schnellen EMA.
PeriodSlow	Die Berechnungsperiode der langsamen EMA.
PeriodSignal	Glättungszeitraum.
Applied	Preistyp für die Berechnung von Oszillatorwerten.

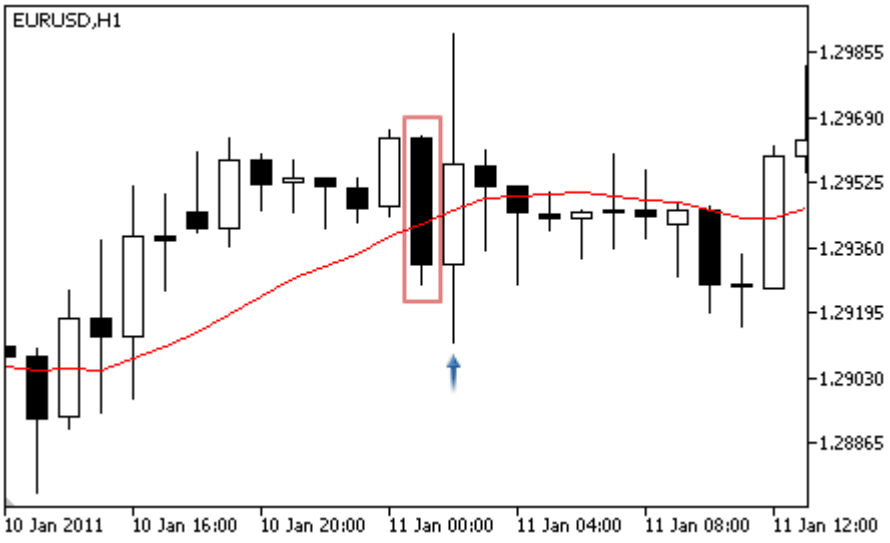
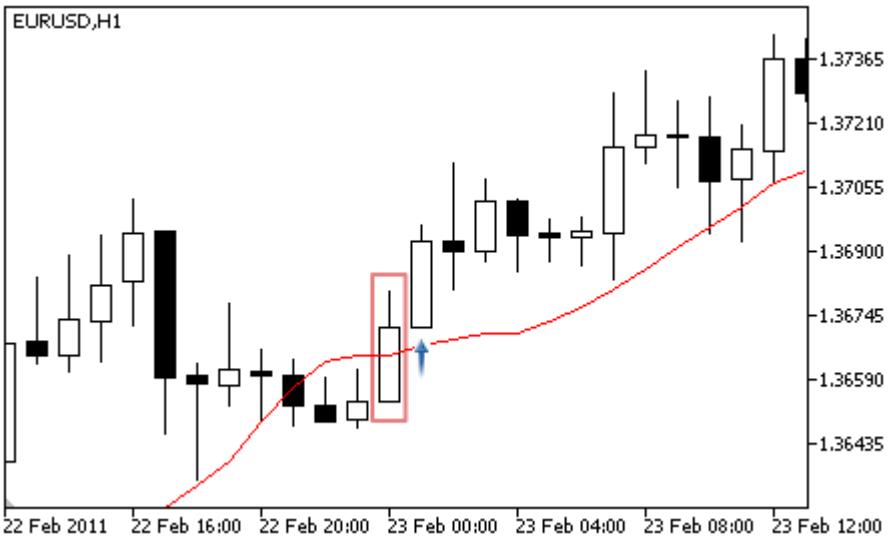
© 2000-2025, MetaQuotes Ltd.

Signale von Moving Average

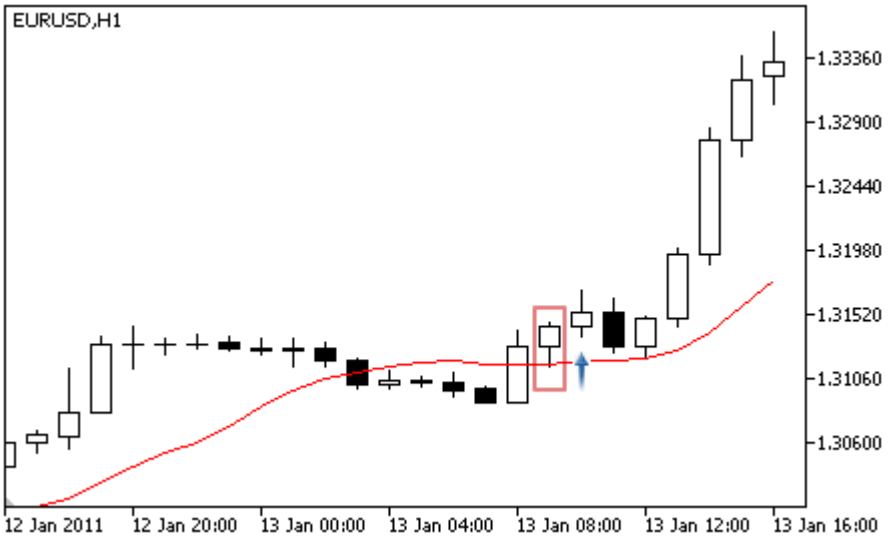
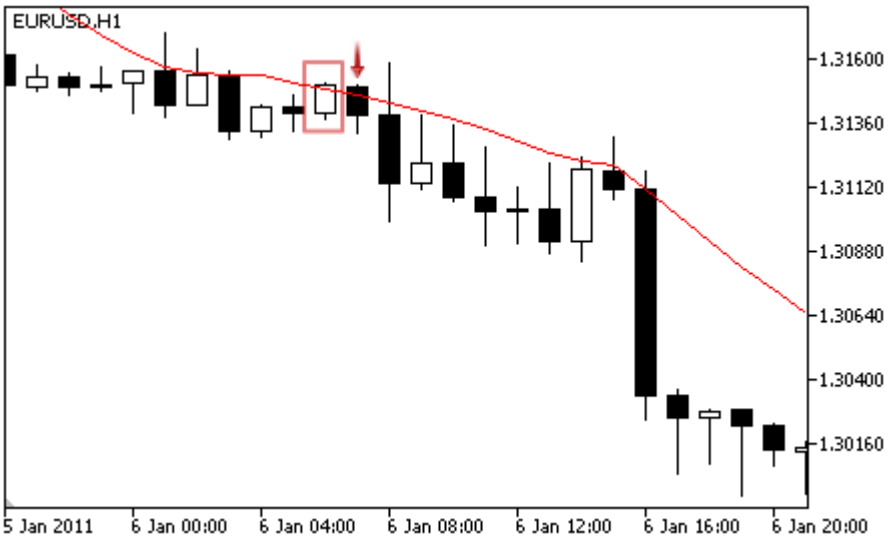
Dieses Signalmodul basiert auf Marktmodelle des Indikators Moving Average. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

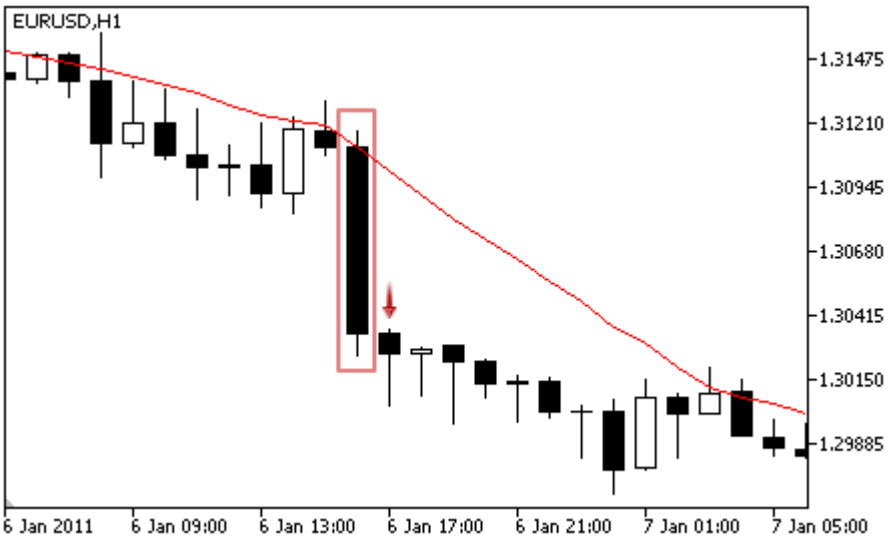
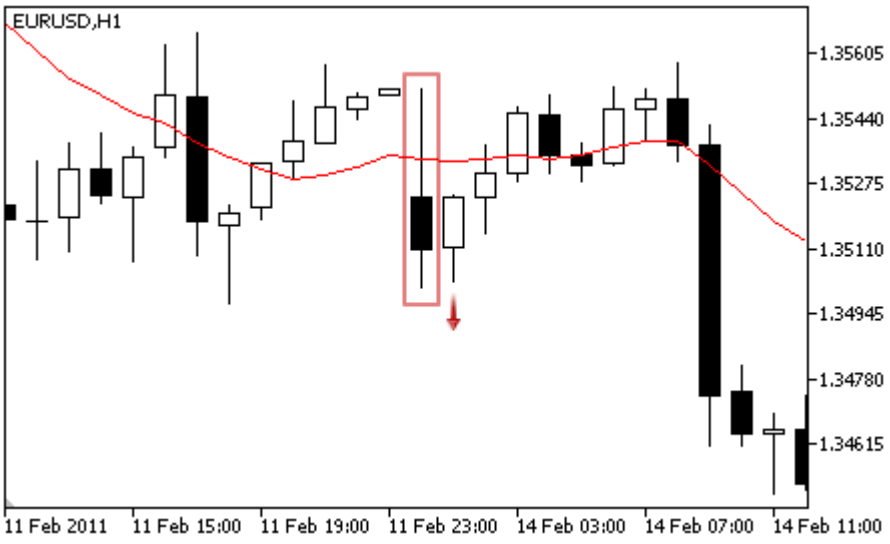
Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator wächst (schwaches Signal auf dem Rückzug aus der Indikatorlinie).  • **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator wächst (starkes Signal auf). 

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	• **Geformter Durchstich.** Der Preis hat den Indikator mit dem unteren Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind über der Indikatorlinie, der Low-Preis ist darunter), aber der Indikator wächst (Signal auf dem Rückzug aus der Indikatorlinie). 
Zum Verkauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator fällt (schwaches Signal auf dem Rückzug aus der Indikatorlinie).  • **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator fällt (starkes Signal auf).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Geformter Durchstich.** Der Preis hat den Indikator mit dem oberen Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind unter der Indikatorlinie, der High-Preis ist darüber), aber der Indikator fällt (Signal auf dem Rückzug aus der Indikatorlinie). 
Nicht gegen den Kauf	Der Preis ist über dem Indikator.
Nicht gegen den Verkauf	Der Preis ist unter dem Indikator.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

© 2000-2025, MetaQuotes Ltd.

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodMA	Mittelwertzeitraum des Indikators.
Shift	Verschiebung des Indikators entlang der Zeitachse (in Balken).
Method	Mittelungsverfahren.
Applied	Preistyp für die Berechnung von Indikatorwerten.

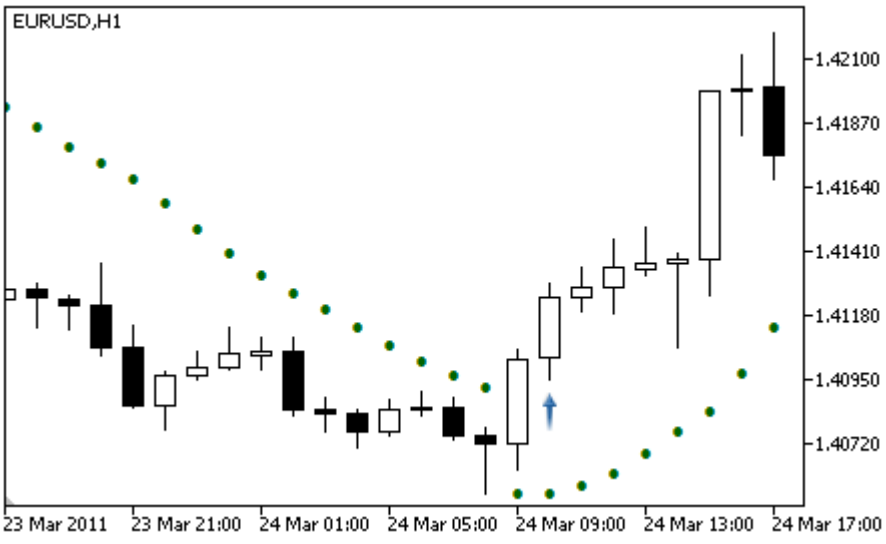
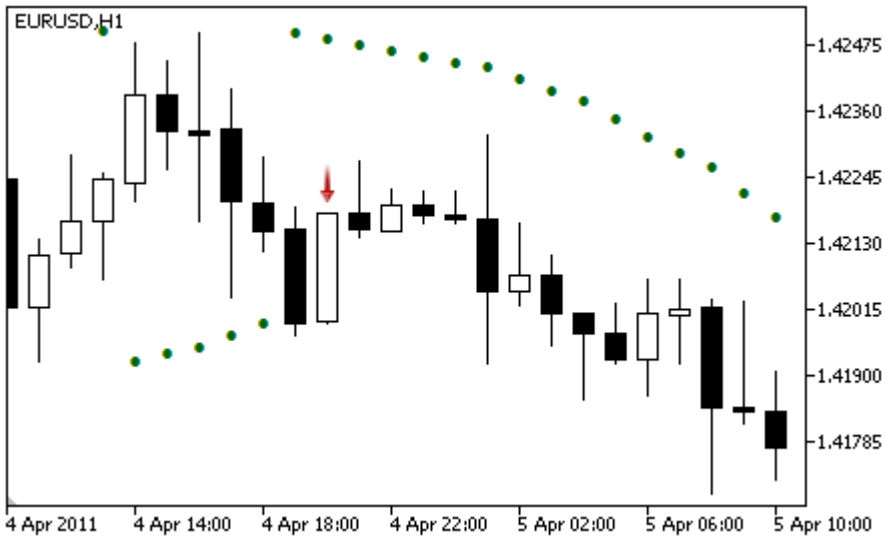
© 2000-2025, MetaQuotes Ltd.

Signale von Parabolic SAR

Dieses Signal-Modul basiert auf die Marktmodelle des Indikators Parabolic SAR. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	**Swing** — auf dem analysierten Balken ist der Indikator unter dem Preis, und war darüber auf dem vorherigen. 
Zum Verkauf	**Swing** — auf dem analysierten Balken ist der Indikator über dem Preis, und war darunter auf dem vorherigen. 
Nicht gegen den Kauf	Der Preis ist über dem Indikator.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
Nicht gegen den Verkauf	Der Preis ist unter dem Indikator.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
Step	Schritt der Geschwindigkeitssteigerung.
Maximum	Maximaler Geschwindigkeitsfaktor der Annäherung des Indikators mit dem Preis.

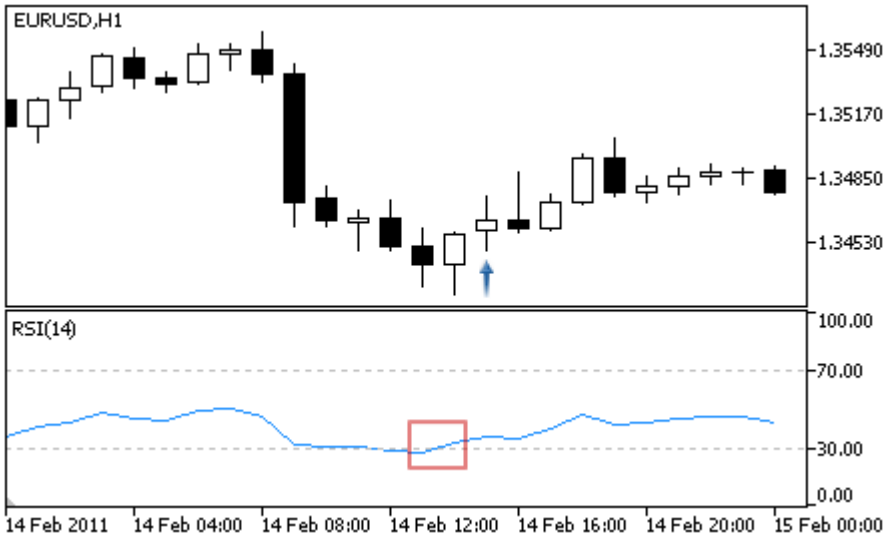
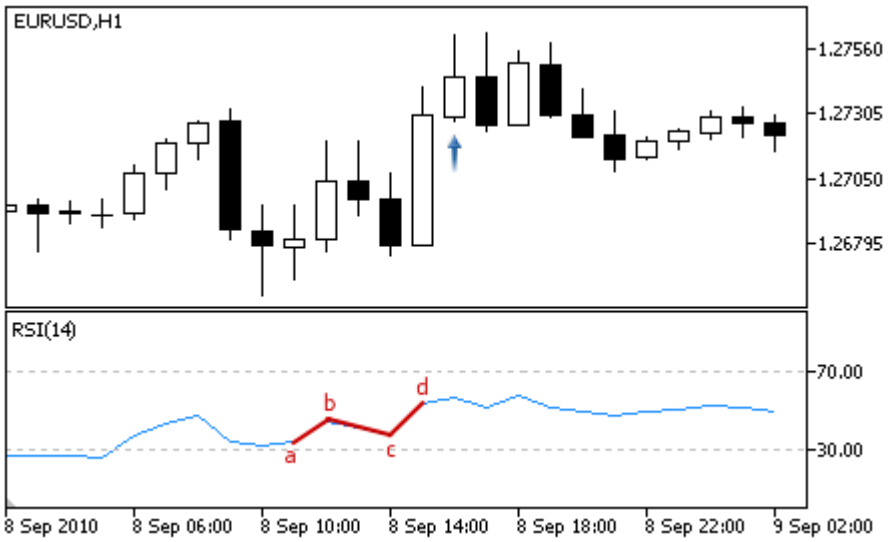
© 2000-2025, MetaQuotes Ltd.

Signale des Oszillators Relative Strength Index

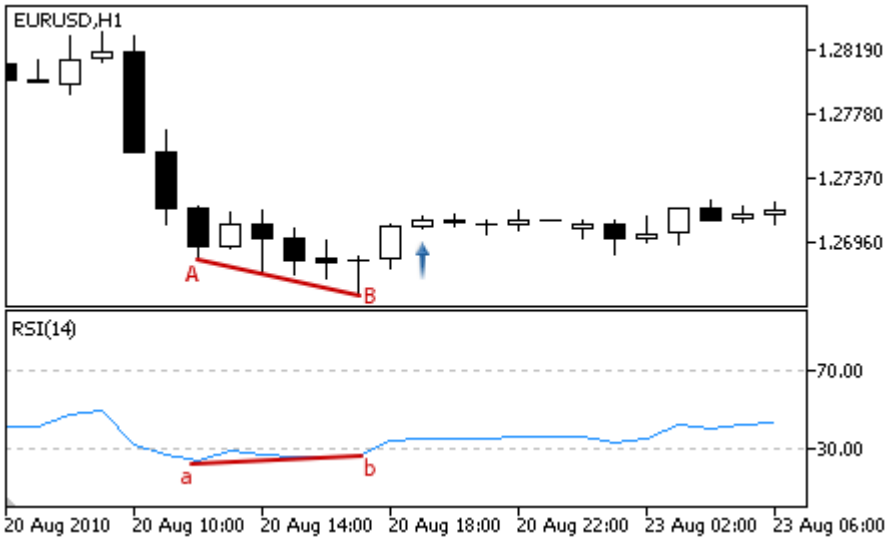
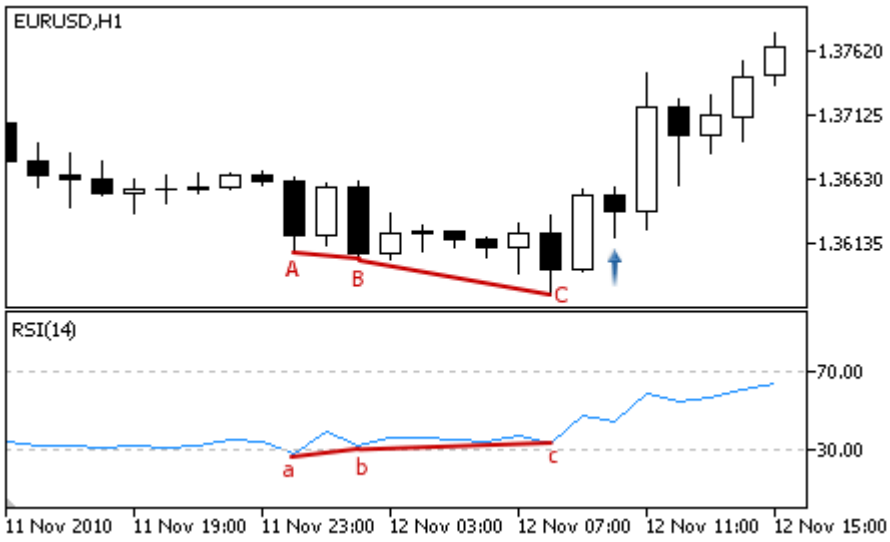
Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators Relative Strength Index. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

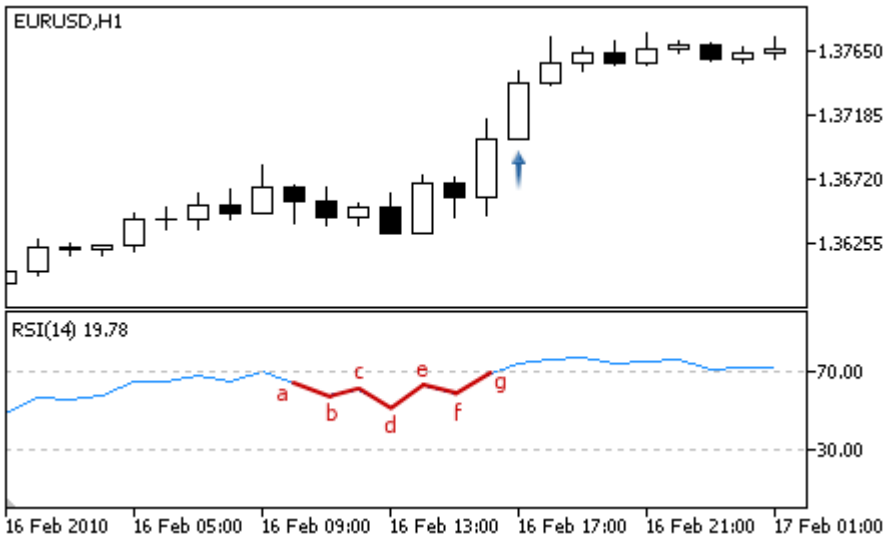
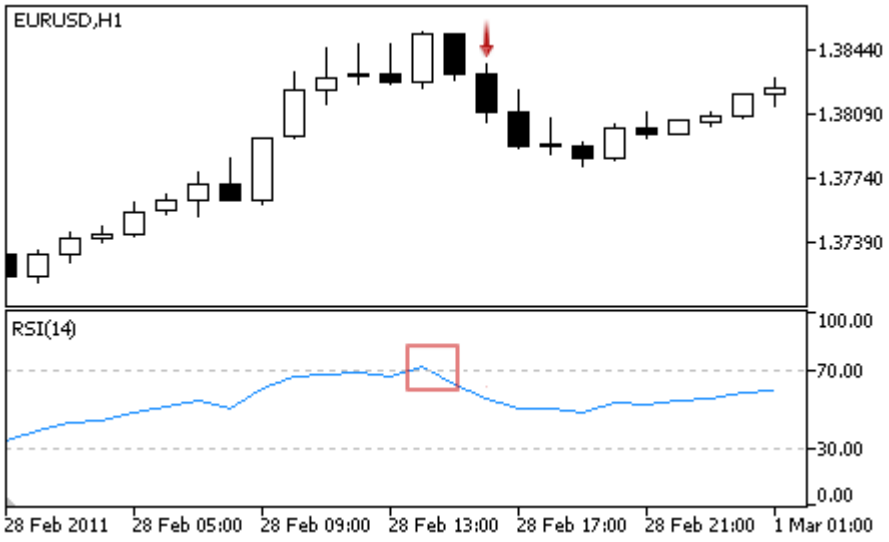
Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr nach dem überverkauften Niveau** — Oszillator hat sich nach oben umgekehrt und sein Wert am analysierten Balken ist hinter dem überverkauften Niveau (der Standardwert ist 30).  • **Failure Swings** — auf dem analysierten Balken stieg der Oszillator über dem bisherigen Höhepunkt. 

© 2000-2025, MetaQuotes Ltd.

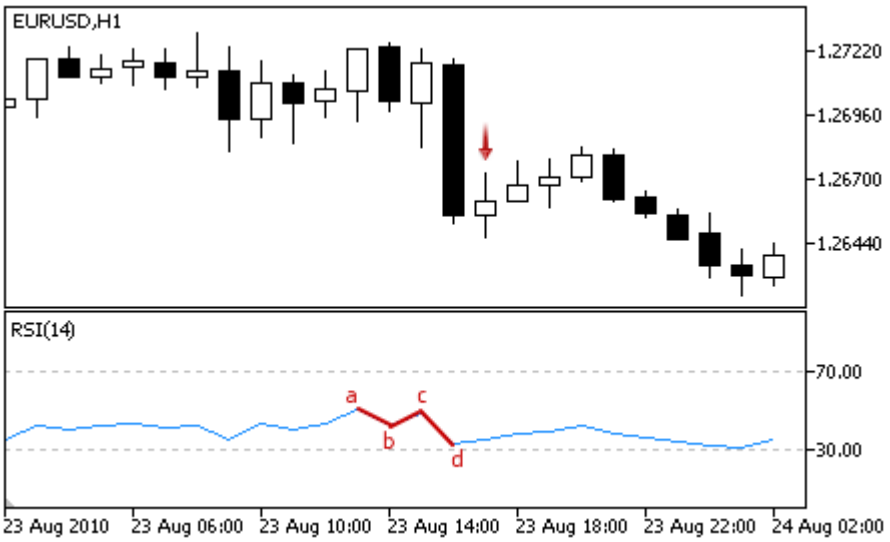
Signaltyp	Beschreibung der Bedingungen
	• **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige.  • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Täler gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Täler gebildet, von denen jede tiefer als der vorherige ist.  • **Kopf/Schulter** — Oszillator hat drei aufeinanderfolgenden Täler gebildet, von denen zweite tiefer als die anderen ist.

© 2000-2025, MetaQuotes Ltd.

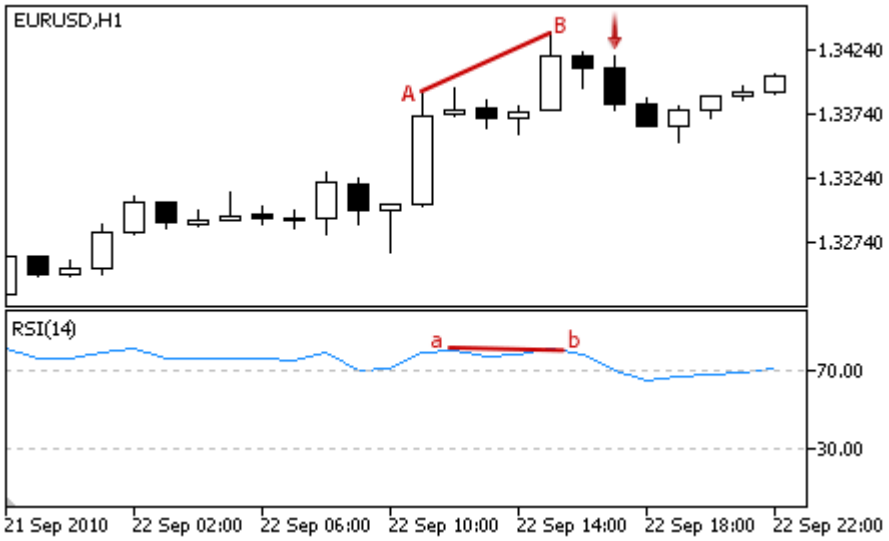
Signaltyp	Beschreibung der Bedingungen
	
Zum Verkauf	• **Umkehr nach dem überkauften Niveau** — Oszillator hat sich nach unten umgekehrt und sein Wert am analysierten Balken ist hinter dem überkauften Niveau (der Standardwert ist 70).  • **Failure Swings** — auf dem analysierten Balken fiel der Oszillator unter dem bisherigen Tal.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

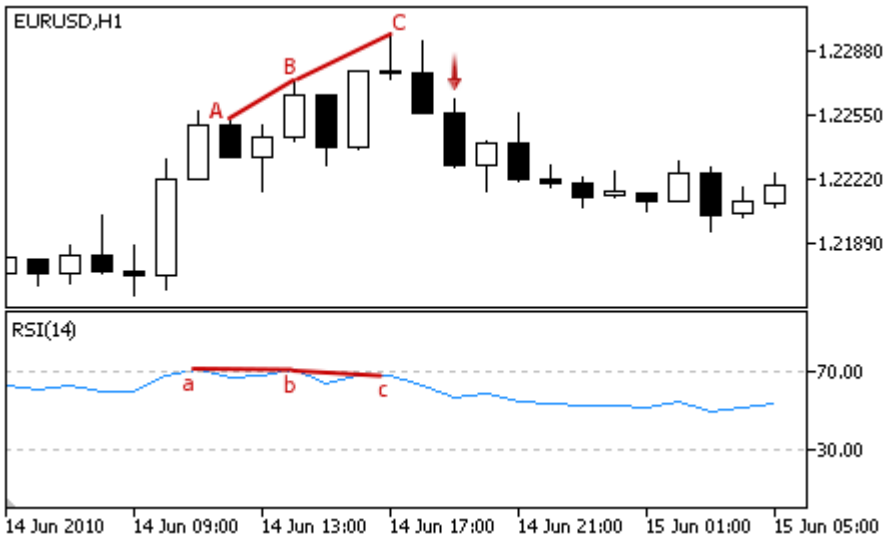
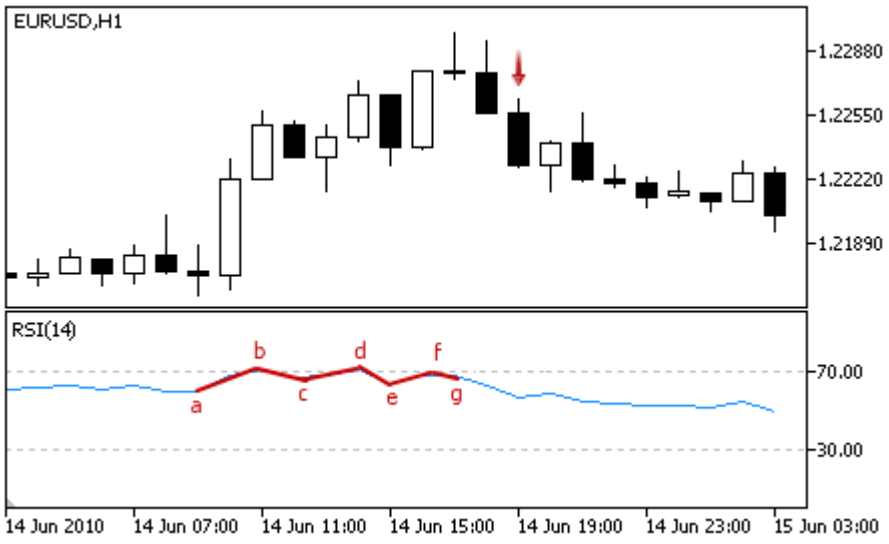


- **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige.



- **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Höhen gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Höhen gebildet, von denen jede größer als der vorherige ist.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Kopf/Schulter** — Oszillator hat drei aufeinanderfolgenden Scheitelpunkte gebildet, von denen zweite höher als die anderen ist. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

© 2000-2025, MetaQuotes Ltd.

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodRSI	Der Berechnungszeitraum des Oszillators.
Applied	Preistyp für die Berechnung von Oszillatorwerten.

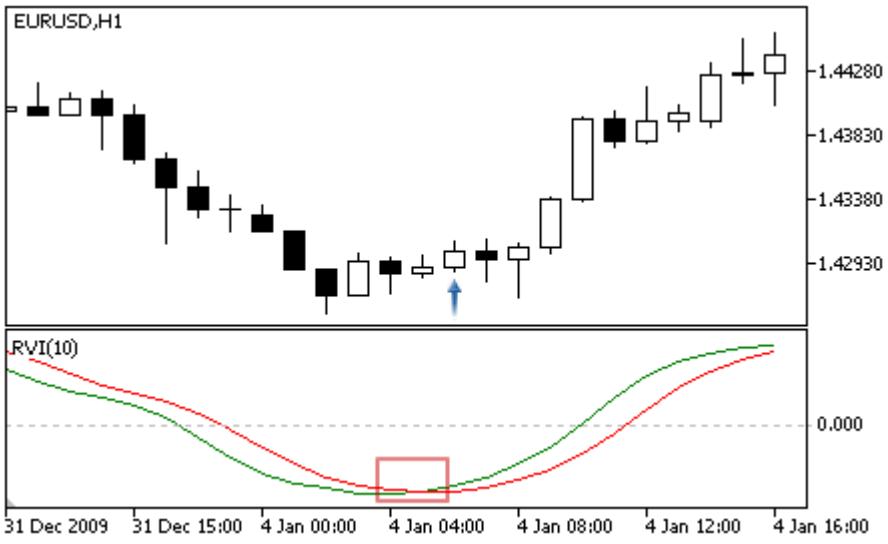
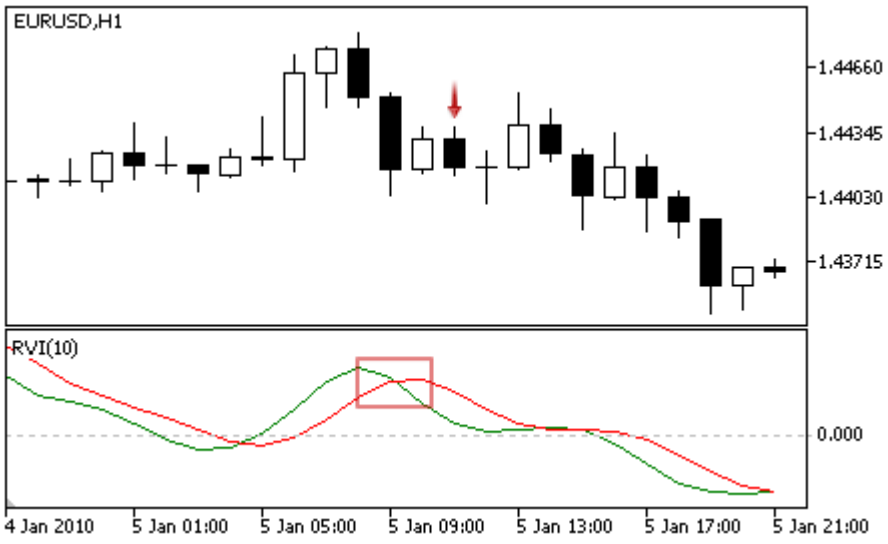
© 2000-2025, MetaQuotes Ltd.

Signale des Oszillators Relative Vigor Index

Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators Relative Vigor Index. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	**Die Haupt- und Signallinie schneiden** — auf dem analysierten Balken ist die Hauptlinie über der Signallinie und war darunter auf dem vorherigen Balken. 
Zum Verkauf	**Die Haupt- und Signallinie schneiden** — auf dem analysierten Balken ist die Hauptlinie unter der Signallinie und war darüber auf dem vorherigen Balken. 

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodRVI	Der Berechnungszeitraum des Oszillators.

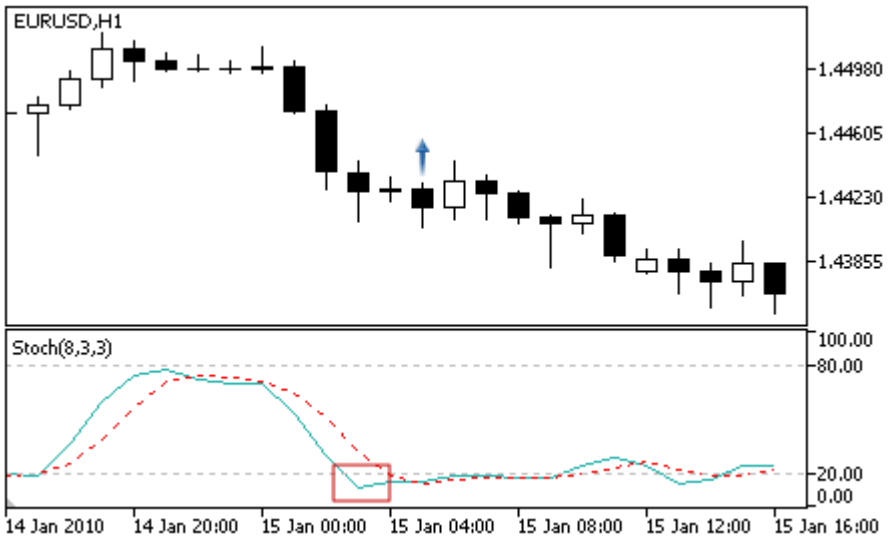
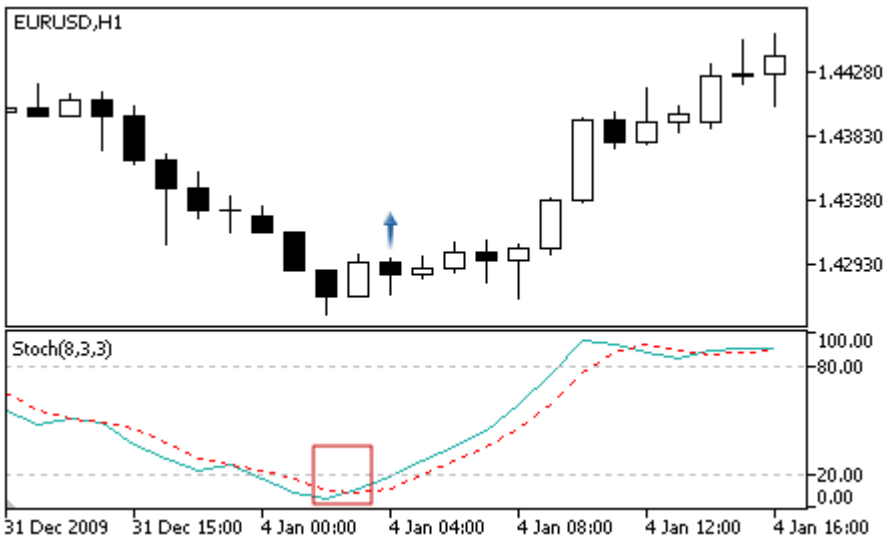
© 2000-2025, MetaQuotes Ltd.

Signale des Oszillators Stochastic

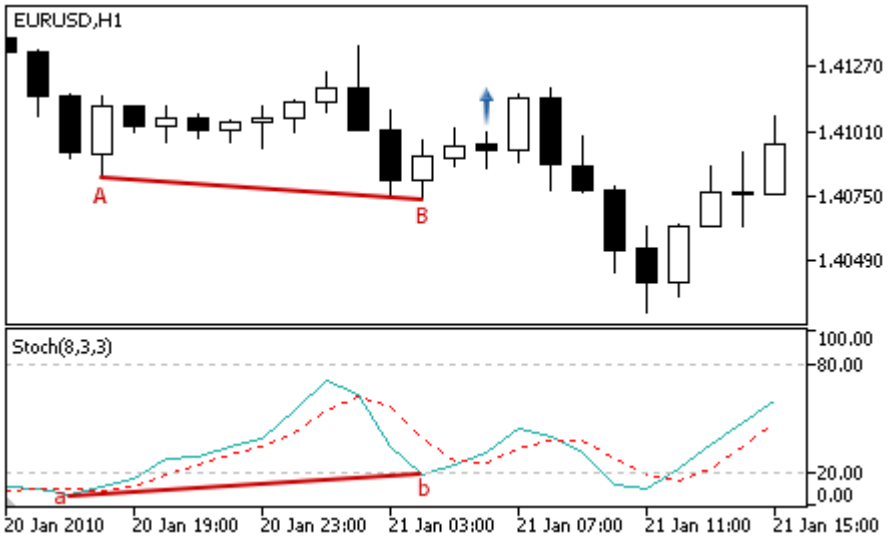
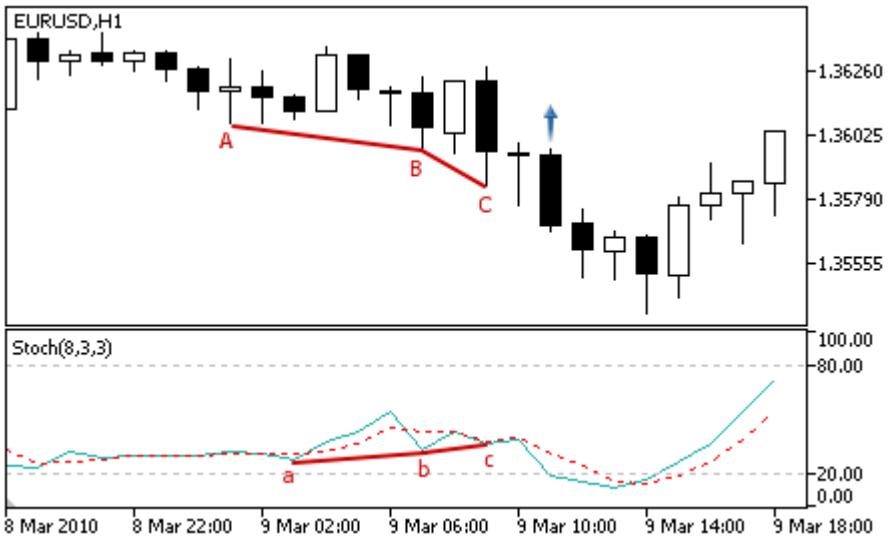
Dieses Signal-Modul basiert auf die Marktmodelle des Oszillators Stochastic. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

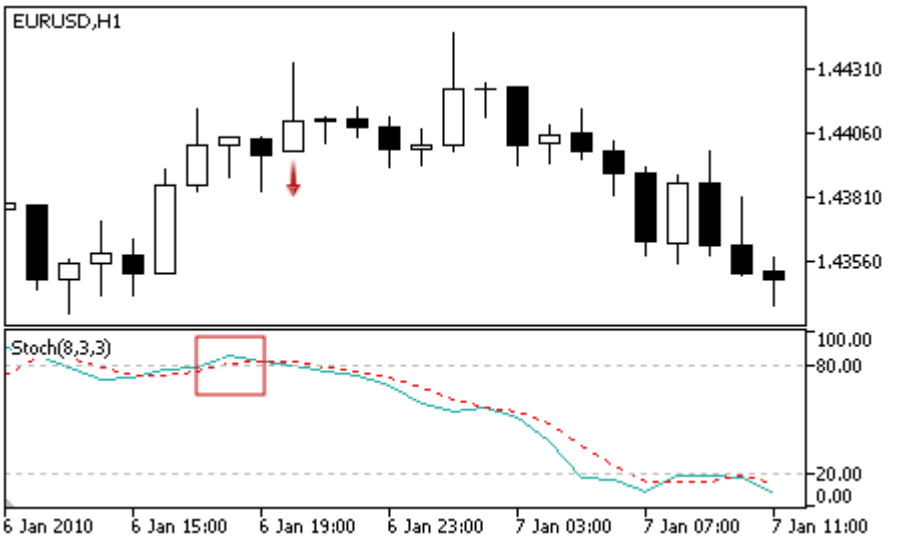
Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr** — der Oszillator hat sich nach oben umgekehrt (der Oszillator wächst auf dem analysierten Balken und fiel auf dem vorherigen).  • **Die Haupt- und Signallinie schneiden** — auf dem analysierten Balken ist die Hauptlinie über der Signallinie und war darunter auf dem vorherigen Balken. 

© 2000-2025, MetaQuotes Ltd.

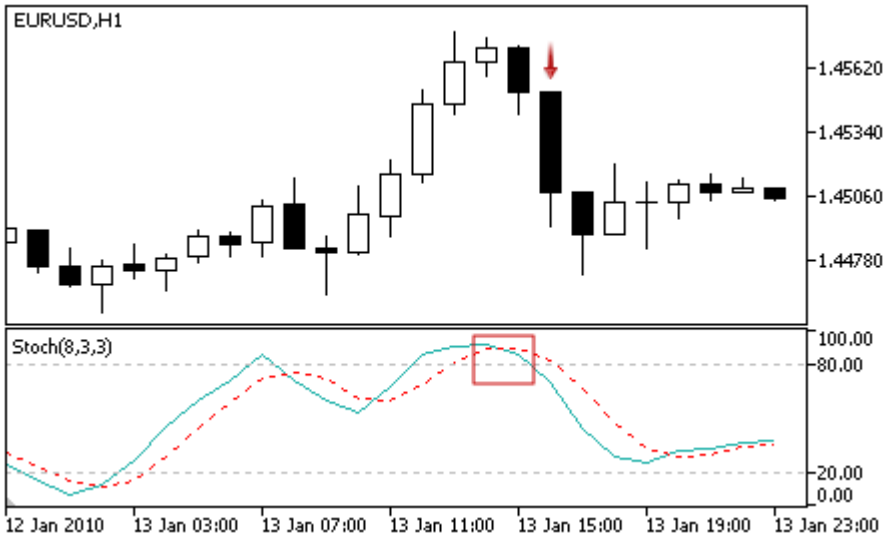
Signaltyp	Beschreibung der Bedingungen
	• **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige.  • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Täler gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Täler gebildet, von denen jede tiefer als der vorherige ist. 
Zum Verkauf	• **Umkehr** — der Oszillator hat sich nach unten umgekehrt (der Oszillator fällt auf dem analysierten Balken und wuchs auf dem vorherigen).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

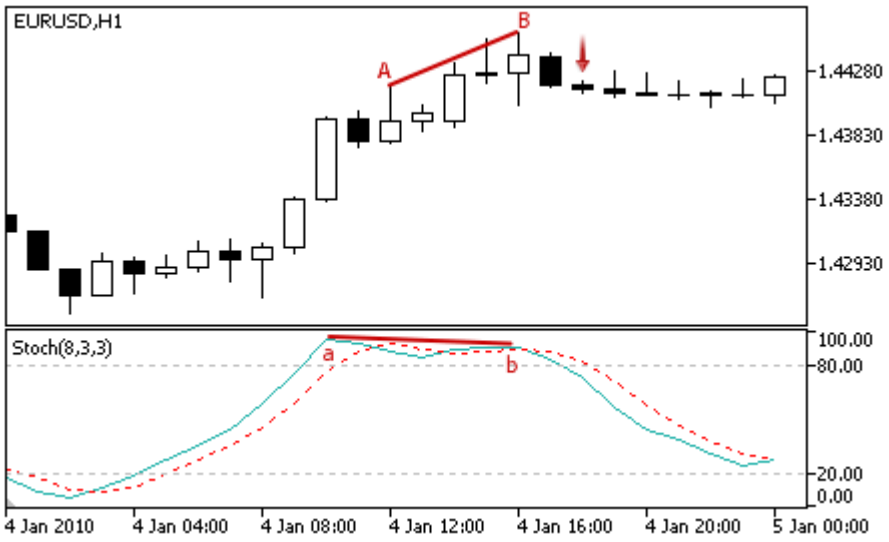
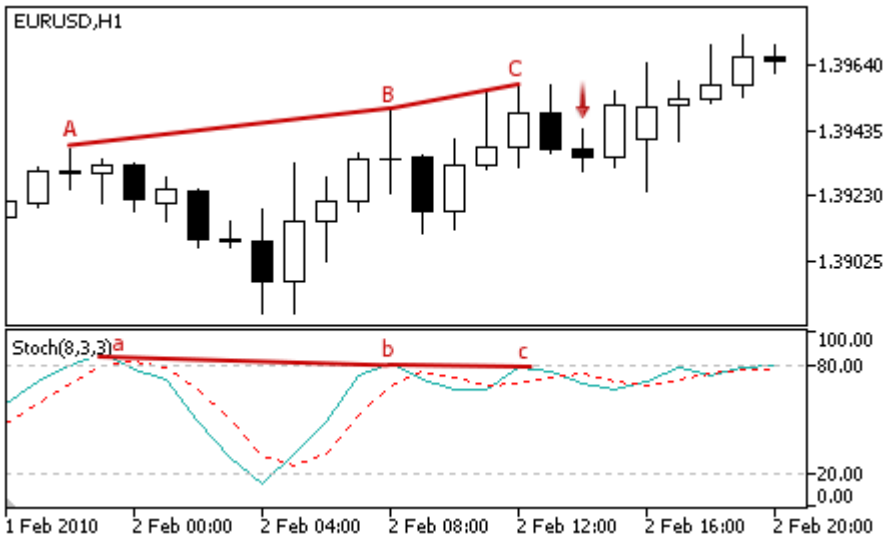


- **Die Haupt- und Signallinie schneiden** — auf dem analysierten Balken ist die Hauptlinie unter der Signallinie und war darüber auf dem vorherigen Balken.



- **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Doppeldivergenz** — Oszillator hat drei aufeinanderfolgenden Höhen gebildet, von denen jede kleiner als das vorherige ist, und der Preis hat drei entsprechenden Höhen gebildet, von denen jede größer als der vorherige ist. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

© 2000-2025, MetaQuotes Ltd.

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodK	Die Berechnungsperiode der Hauptlinie des Oscillators.
PeriodD	Die Durchschnittsperiode der Hauptlinie des Oscillators.
PeriodSlow	Die Periode der Verzögerung.
Applied	Preistyp für die Berechnung von Oszillatorwerten.

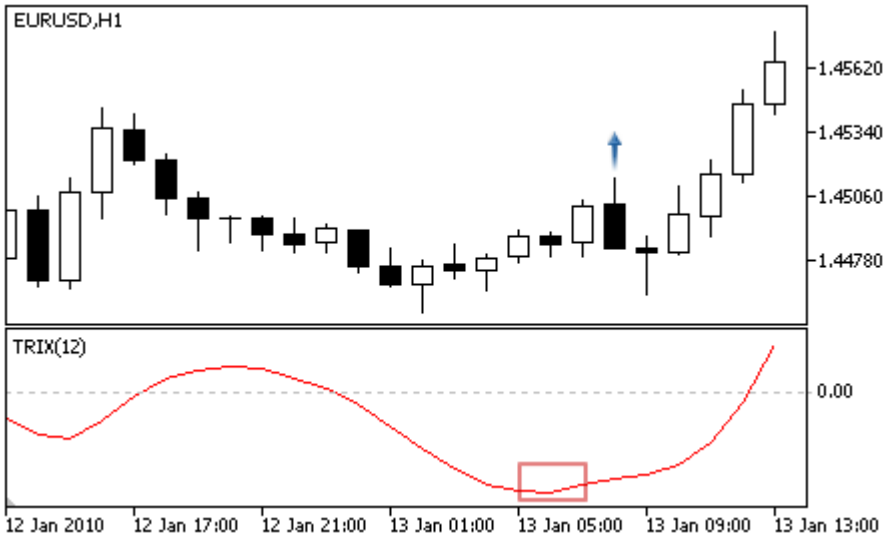
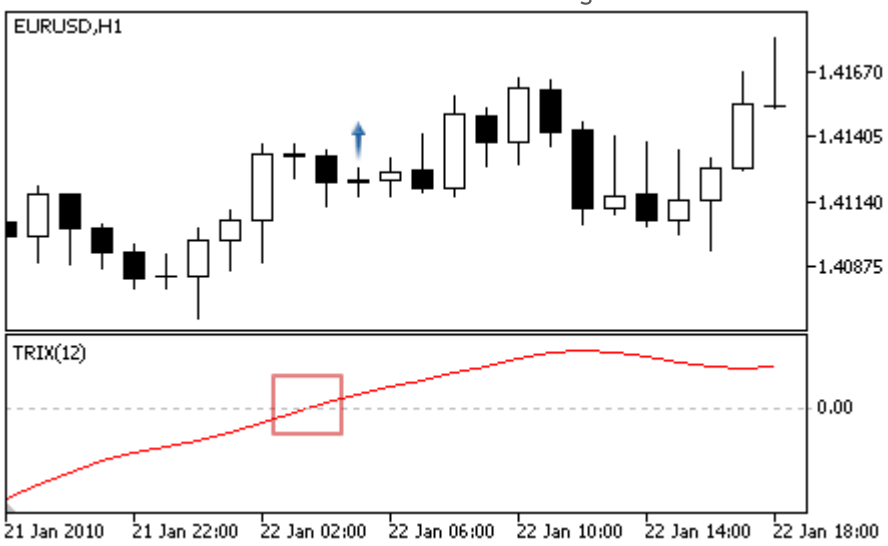
© 2000-2025, MetaQuotes Ltd.

Signale des Oszillators Triple Exponential Average

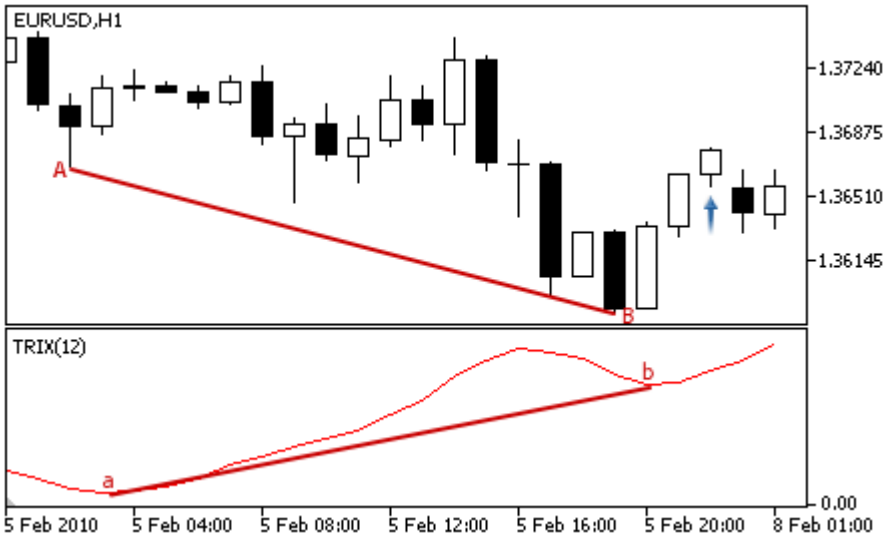
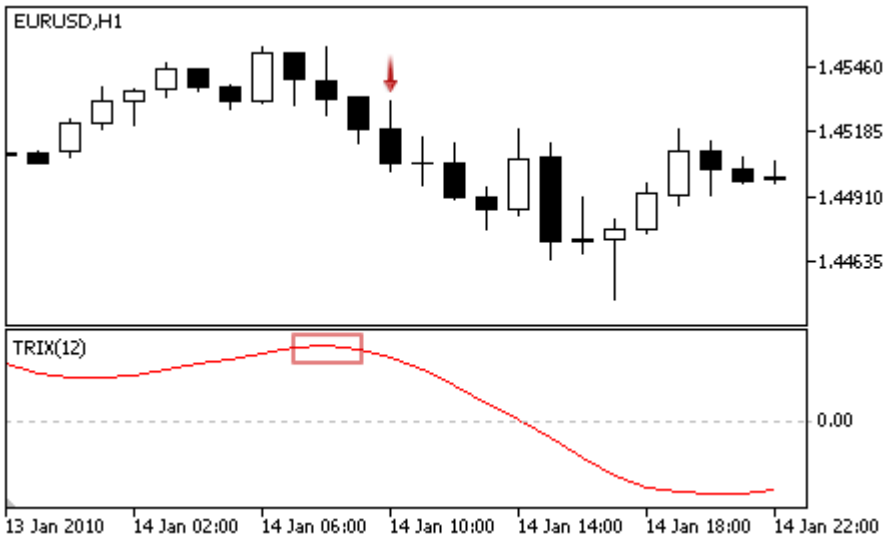
Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Triple Exponential Average. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

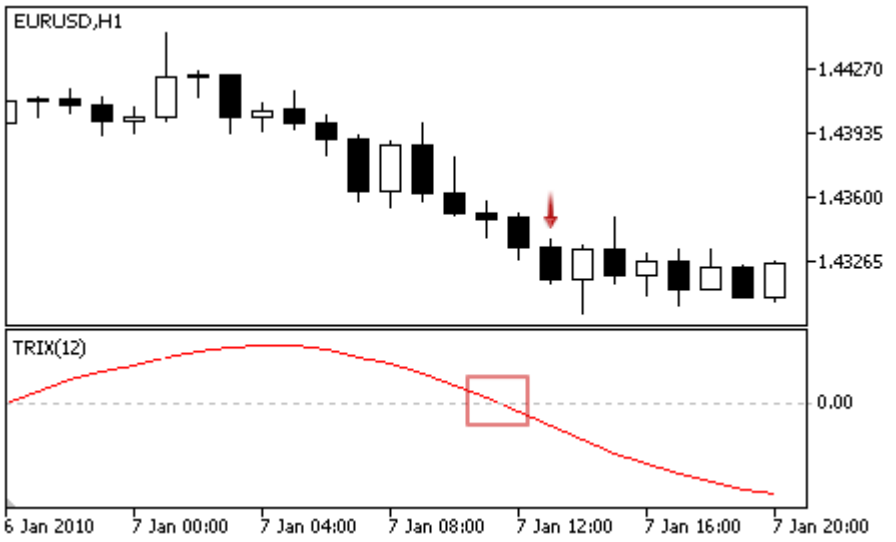
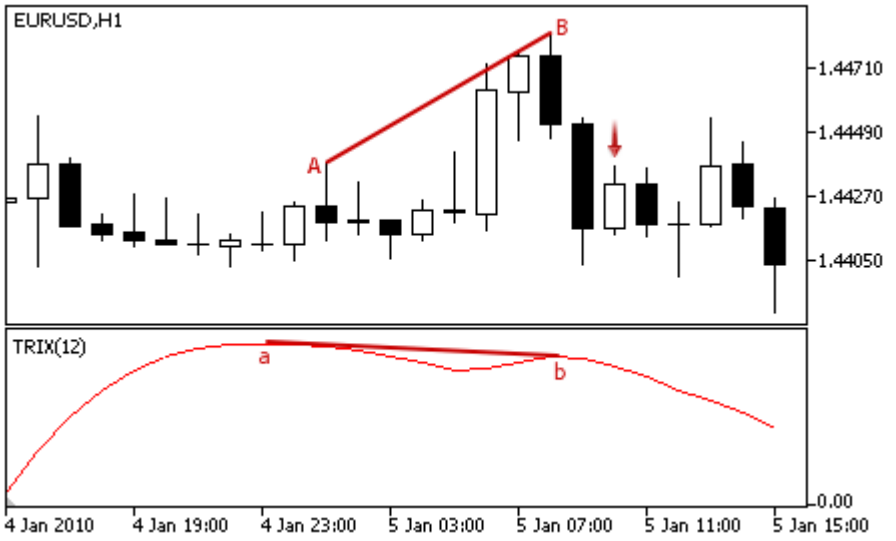
Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr** — der Oszillator hat sich nach oben umgekehrt (der Oszillator wächst auf dem analysierten Balken und fiel auf dem vorherigen).  • **Nulldurchgangs** — auf dem analysierten Balken ist die Hauptlinie über dem Nullwert und war darunter auf dem vorherigen Balken. 

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	• **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige. 
Zum Verkauf	• **Umkehr** — der Oszillator hat sich nach unten umgekehrt (der Oszillator fällt auf dem analysierten Balken und wuchs auf dem vorherigen).  • **Nulldurchgangs** — auf dem analysierten Balken ist die Hauptlinie unter dem Nullwert und war darüber auf dem vorherigen Balken.

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	 • **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

© 2000-2025, MetaQuotes Ltd.

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodTriX	Der Berechnungszeitraum des Oszillators.
Applied	Preistyp für die Berechnung von Oszillatorwerten.

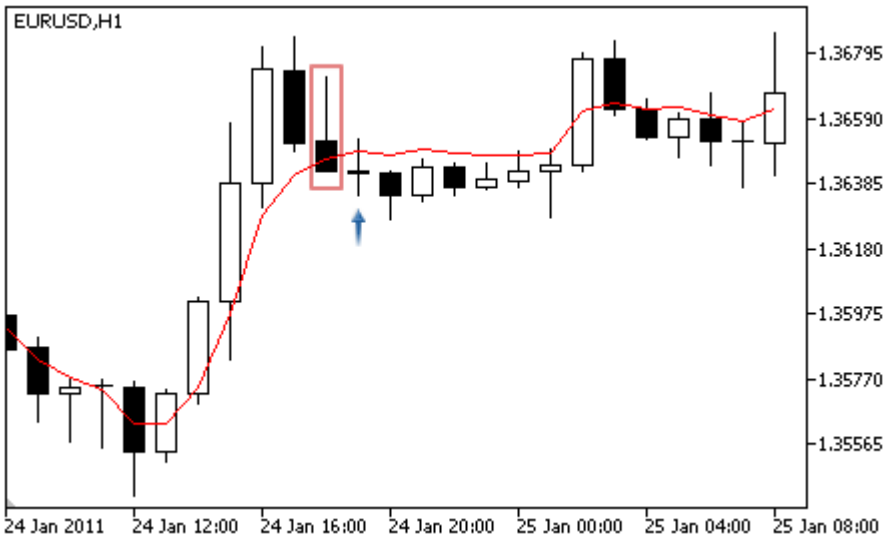
© 2000-2025, MetaQuotes Ltd.

Signale von Triple Exponential Moving Average

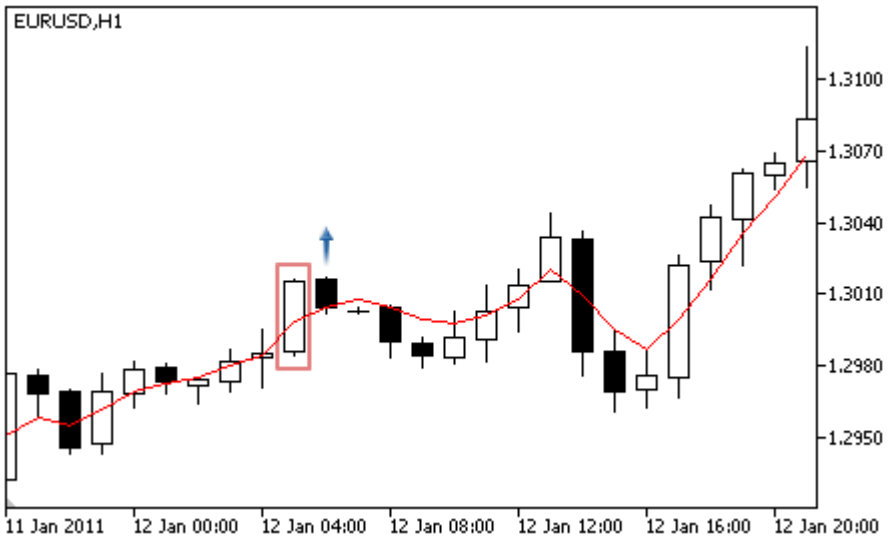
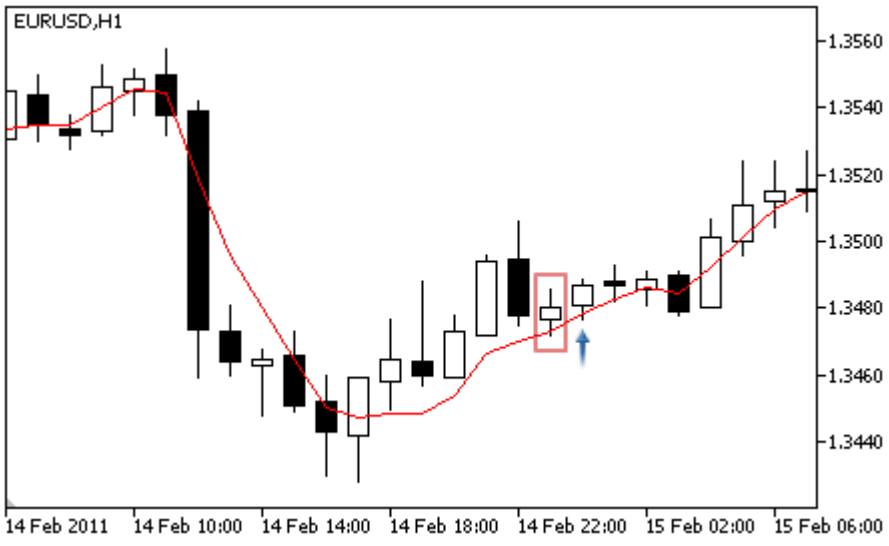
Dieses Signalmodul basiert auf Marktmodelle des Indikators Tripple Exponential Moving Average. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

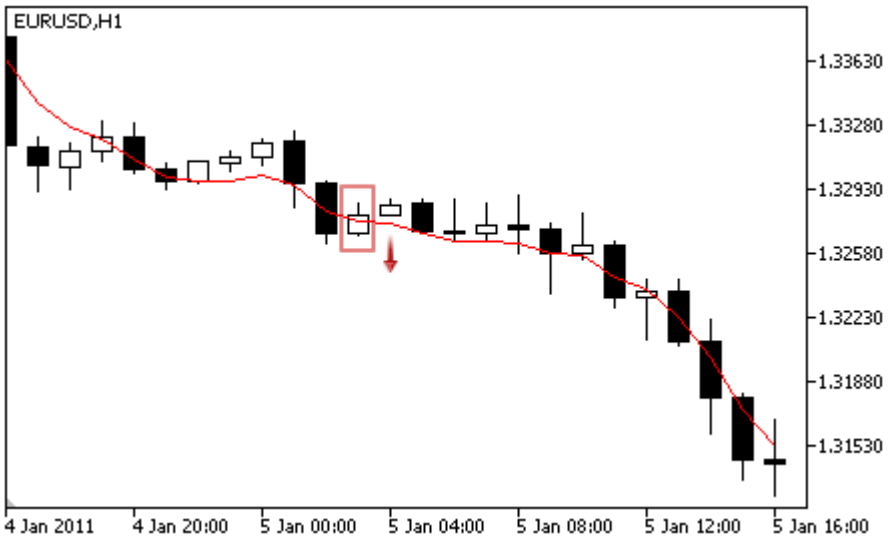
Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator wächst (schwaches Signal auf dem Rückzug aus der Indikatorlinie).  • **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator wächst (starkes Signal auf).

© 2000-2025, MetaQuotes Ltd.

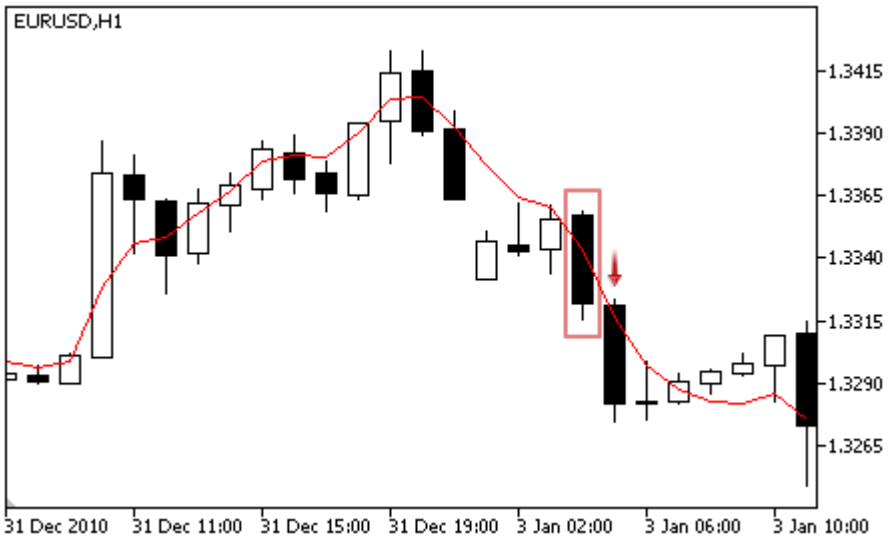
Signaltyp	Beschreibung der Bedingungen
	 • **Geformter Durchstich.** Der Preis hat den Indikator mit dem unteren Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind über der Indikatorlinie, der Low-Preis ist darunter), aber der Indikator wächst (Signal auf dem Rückzug aus der Indikatorlinie). 
Zum Verkauf	• **Ungeformter Durchstich.** Der Preis hat durch den Indikator nach oben gegangen (Open-Preis des analysierten Balkens ist unter der Indikatorlinie, der Close-Preis ist darüber), aber der Indikator fällt (schwaches Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen

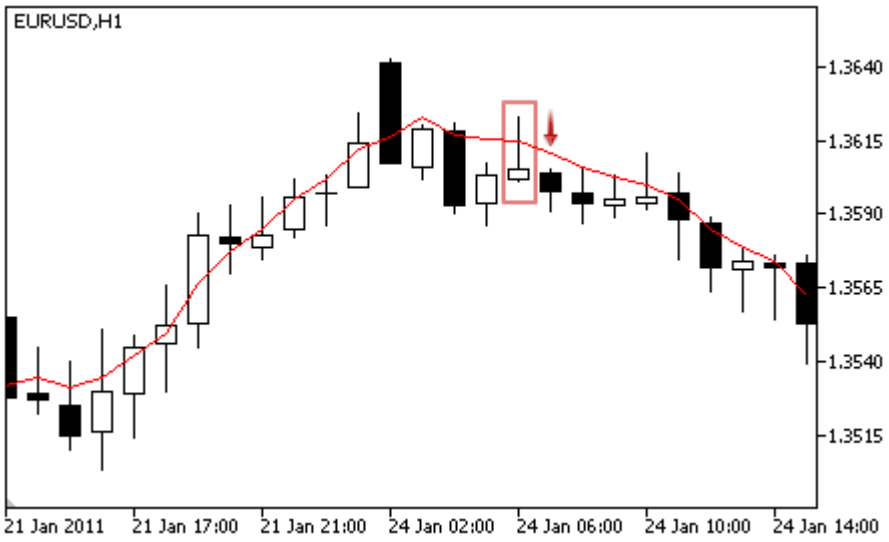


- **Durchgang von Moving Average.** Der Preis hat durch den Indikator nach unten gegangen (Open-Preis des analysierten Balkens ist über der Indikatorlinie, der Close-Preis ist darunter), aber der Indikator fällt (starkes Signal auf).



- **Geformter Durchstich.** Der Preis hat den Indikator mit dem oberen Schatten gekreuzt (Preise Open und Close des analysierten Balkens sind unter der Indikatorlinie, der High-Preis ist darüber), aber der Indikator fällt (Signal auf dem Rückzug aus der Indikatorlinie).

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
	
Nicht gegen den Kauf	Der Preis ist über dem Indikator.
Nicht gegen den Verkauf	Der Preis ist unter dem Indikator.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodMA	Mittelwertzeitraum des Indikators.
Shift	Verschiebung des Indikators entlang der Zeitachse (in Balken).
Method	Mittelungsverfahren.
Applied	Preistyp für die Berechnung von Indikatorwerten.

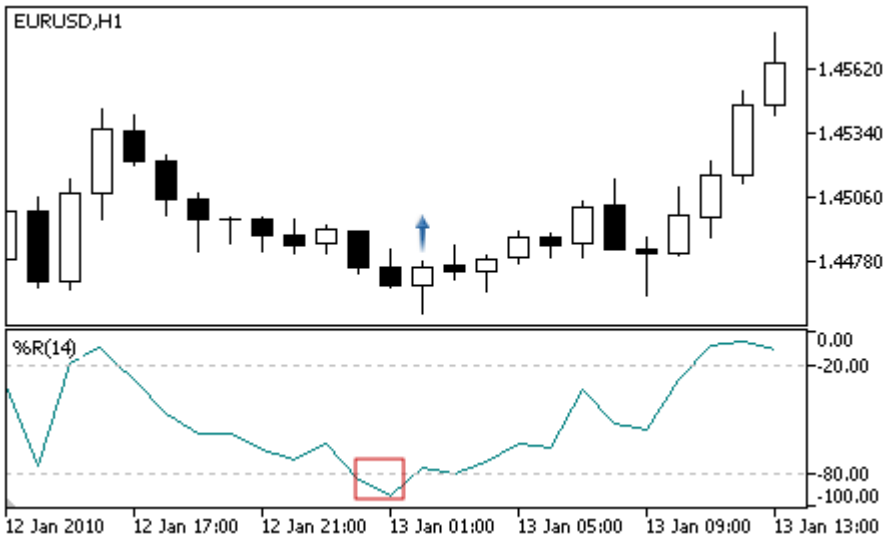
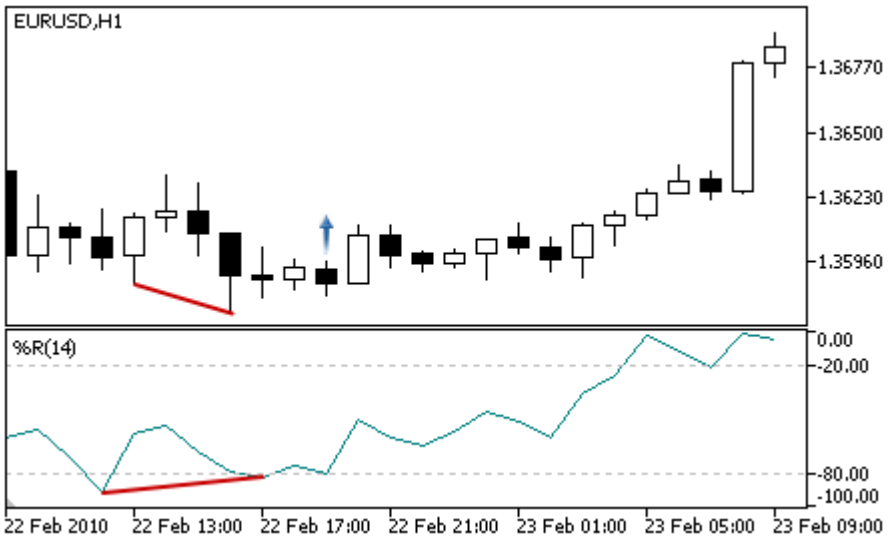
© 2000-2025, MetaQuotes Ltd.

Signale des Oszillators Williams Percent Range

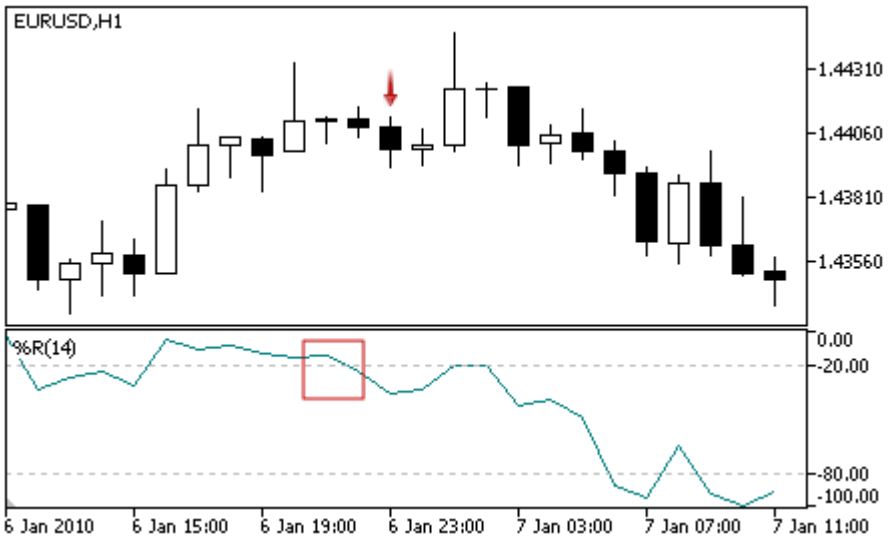
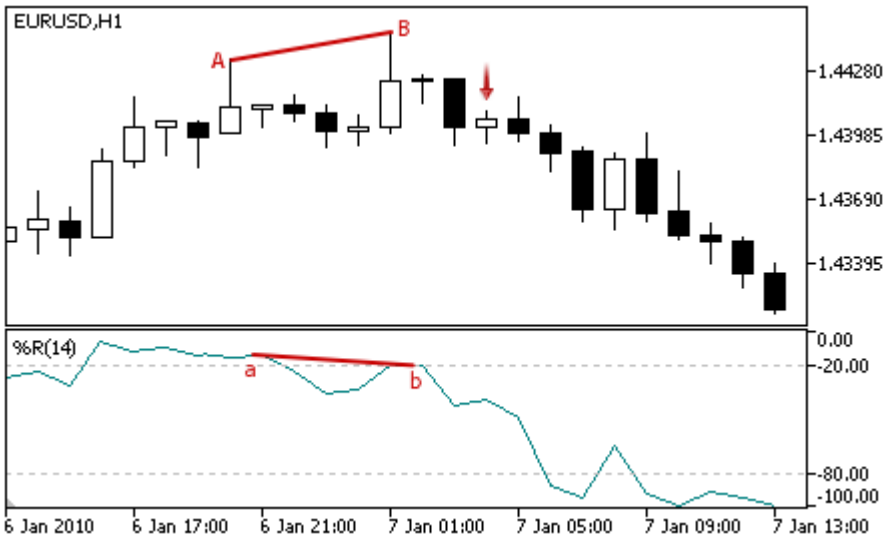
Das Modul von Signalen auf Basis der Marktmodelle des Oszillators Williams Percent Range. Der Mechanismus der Handelsentscheidungen auf Grundlage von Signalmodule ist im separaten Bereich beschrieben.

Bedingungen zur Signalgenerierung

Nachfolgend finden Sie eine Beschreibung der Bedingungen, unter denen der Modul ein Signal dem Expert Advisor überträgt.

Signaltyp	Beschreibung der Bedingungen
Zum Kauf	• **Umkehr nach dem überverkauften Niveau** — Oszillator hat sich nach oben umgekehrt und sein Wert am analysierten Balken ist hinter dem überverkauften Niveau (der Standardwert ist -80).  • **Divergenz** — das erste analysierte Tal des Oszillators ist kleiner als das vorherige, und das entsprechende Preistal ist tiefer, als das vorherige. 

© 2000-2025, MetaQuotes Ltd.

Signaltyp	Beschreibung der Bedingungen
Zum Verkauf	• **Umkehr nach dem überkauften Niveau** — Oszillator hat sich nach unten umgekehrt und sein Wert am analysierten Balken ist hinter dem überkauften Niveau (der Standardwert ist -20).  • **Divergenz** — die erste analysierte Höhe des Oszillators ist kleiner als die vorherige, und das entsprechende Preishöhe ist höher, als die vorherige. 
Nicht gegen den Kauf	Der Oszillatorwert wächst am analysierten Balken.
Nicht gegen den Verkauf	Der Oszillatorwert fällt am analysierten Balken.

Hinweis

In Abhängigkeit von der Arbeitsmodus des Expert Advisor ("Every Tick" oder "Open Price") wird der analysierte Balken entweder der aktuelle Balken (mit Index 0), oder der letzte gebildete Balken (mit Index 1).

© 2000-2025, MetaQuotes Ltd.

Beachten Sie, dass der Oszillator Williams Percent Range eine invertierte Skala hat. Sein Maximalwert ist -100, und der Minimalwert ist 0.

Anpassbare Einstellungen

Dieses Modul hat die folgende anpassbare Einstellungen:

Parameter	Beschreibung
Weight	Gewicht des Modulsignals ist im Bereich von 0 bis 1.
PeriodWPR	Der Berechnungszeitraum des Oszillators.

© 2000-2025, MetaQuotes Ltd.

Eine Reihe von vorgefertigten Implementierungen von Algorithmen für Steuerung offenen Positionen

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen von vorgefertigten Implementierungen von Algorithmen für Steuerung offenen Positionen und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen von vorgefertigten Implementierungen von Algorithmen für Steuerung offenen Positionen sparen Sie Zeit bei der Entwicklung (und vor allem beim Testen) von Handelsstrategien.

Die Standardbibliothek MQL5 (in Bezug auf die Klassen von vorgefertigten Implementierungen von Algorithmen für Steuerung offenen Positionen) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Expert\Trailing.

Klasse	Beschreibung
CTrailingFixedPips	Die Klasse implementiert einen Algorithmus für Steuerung offenen Positionen in einer festgelegten "Distanz"
CTrailingMA	Die Klasse implementiert einen Algorithmus für Steuerung offenen Positionen aufgrund den Werten vom gleitenden Durchschnitt
CTrailingNone	Stub-Klasse von Algorithmen für Steuerung offenen Positionen
CTrailingPSAR	Die Klasse implementiert einen Algorithmus für Steuerung offenen Positionen aufgrund den Werten des Indikators Parabolic SAR

© 2000-2025, MetaQuotes Ltd.

CTrailingFixedPips-Klasse

CTrailingFixedPips ist eine Klasse für die Implementierung von Algorithmen für Position-Trailing an der Festdistanz (in Punkten).

CTrailingFixedPips-Klasse implementiert den folgenden Trailing-Algorithmus: wenn Stop Loss gleich Null ist, wird angenommen, dass der Zustand für Änderung von Stop Loss nicht erfüllt ist und Änderung von Stop Loss ist nicht vorgeschlagen, andernfalls Preisänderung in der Gewinn-Richtung wird überprüft.

Wenn die Position schon ein Stop Loss hat, wird Distanz zwischen Preis und Stop Loss überprüft. Wenn die Position kein Stop Loss hat, wird Distanz zwischen Preis und Eröffnungspreis überprüft. Wenn die Distanz größer als Stop Loss ist, wird vorgeschlagen eine neue preis für Stop Loss festzulegen.

Wenn die Änderung-Bedingung erfüllt ist und Take Profit nicht Null ist, ein neuer Preis für Take Profit wird vorgeschlagen.

Wenn die Expert-Klasse mit dem Flag every_tick=false initialisiert war, führt Expert Advisor alle Handelsoperationen (und auch Trailing-Operationen) nur auf einem neuen Balken des Symbols auf dem Arbeitsperiode durch. In diesem Fall ermöglicht die Verwendung von Take Profit eine Position zu schließen, wenn der Preis sich in Richtung der Position bewegt bis eine neue Bar.

Beschreibung

CTrailingFixedPips implementiert einen Algorithmus für Position-Trailing an der Festdistanz (in Punkten).

Deklaration

```
class CTrailingFixedPips: public CExpertTrailing
```

Kopf

```
   #include <Expert\Trailing\CTrailingFixedPips.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertTrailing
      CTrailingFixedPips

Gruppen der Klassenmethode

Initialisierung	
StopLevel	Setzt ein Stop Loss
ProfitLevel	Setzt ein Take Profit
virtual ValidationSettings	Überprüft die Einstellungen

© 2000-2025, MetaQuotes Ltd.

Initialisierung	
Methoden zur Überprüfung der Trailing-Bedingungen	
virtual CheckTrailingStopLong	Findet hinzu, ob die Long-Position geändert werden soll
virtual CheckTrailingStopShort	Findet heraus, ob die Short-Position geändert werden soll

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

© 2000-2025, MetaQuotes Ltd.

StopLevel

Setzt einen Trailing-Wert für Stop Loss.

```
void  StopLevel(
   int  stop_level      // Wert
   )
```

Parameter

stop_level

[in] Der Trailing-Wert für Stop Loss in "normalen" (2/4-stelligen) Punkten.

Hinweis

Wenn man dem Wert 0 zuweist, wird Trailing (Änderung) nicht durchgeführt.

© 2000-2025, MetaQuotes Ltd.

ProfitLevel

Setzt einen Trailing-Wert für Take Profit.

```
void  ProfitLevel(
   int  profit_level      // Wert
   )
```

Parameter

profit_level

[in] Trailing-Wert für Take Profit in "normalen" (2/4-stelligen) Punkten.

Hinweis

Wenn man dem Wert 0 zuweist, wird Trailing (Änderung) nicht durchgeführt.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Überprüft den Trailing-Wert. Gültige Werte sind 0 und die Werte größer als "Mindest Distanz Vertiefung in Punkte vom aktuellen Preis für Stop Order" des Arbeitssymbols.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopLong

Findet hinzu, ob die Long-Position geändert werden soll.

```
virtual bool  CheckTrailingStopLong(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz für Stop Loss
   double&         tp            // Referenz für Take Profit
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Wenn das Trailing-Niveau von Stop Loss gleich Null ist, ist die Bedingung nicht erfüllt (ausgehen). Wenn die Position schon ein Stop Loss hat, ist ihr Preis ein Basispreis, andernfalls wird Eröffnungspreis verwendet. Wenn der aktuelle Bid-Preis über dem Basispreis um mehr als der Trailing-Wert ist, wird es vorgeschlagen Stop Loss bei der Trailing-Distanz unter dem aktuellen Bid-Preis zu verschieben. Wenn Take Profit auch nicht Null ist, wird es vorgeschlagen Take Profit bei der entsprechenden Trailing-Distanz über dem aktuellen Bid-Preis zu verschieben.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopShort

Findet heraus, ob die Short-Position geändert werden soll.

```
virtual bool  CheckTrailingStopShort(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz für Stop Loss
   double&         tp            // Referenz für Take Profit
   )
```

Parameter

position

[in] Zeiger auf das CPositionInfo-Objekt.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Wenn das Trailing-Niveau von Stop Loss gleich Null ist, ist die Bedingung nicht erfüllt (ausgehen). Wenn die Position schon ein Stop Loss hat, ist ihr Preis ein Basispreis, andernfalls wird Eröffnungspreis verwendet. Wenn der aktuelle Ask-Preis unter dem Basispreis um mehr als der Trailing-Wert ist, wird es vorgeschlagen Stop Loss bei der Trailing-Distanz über dem aktuellen Ask-Preis zu verschieben. Wenn Take Profit auch nicht Null ist, wird es vorgeschlagen Take Profit bei der entsprechenden Trailing-Distanz unter dem aktuellen Ask-Preis zu verschieben.

© 2000-2025, MetaQuotes Ltd.

CTrailingMA-Klasse

CTrailingMA ist eine Klasse für die Implementierung von Algorithmen für Position-Trailing aufgrund den Werten von Moving Average.

Beschreibung

CTrailingMA implementiert Einen Algorithmus für Position-Trailing aufgrund den Werten von Moving Average mit den angegebenen Parameter auf dem vorherigen Balken.

Deklaration

```
class CTrailingMA: public CExpertTrailing
```

Kopf

```
   #include <Expert\Trailing\TrailingMA.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertTrailing
      CTrailingMA

Gruppen der Klassenmethode

Initialisierung	
Period	Setzt den period-Parameter von Moving Average
Shift	Setzt den shift-Parameter von Moving Average
Method	Setzt den method-Parameter von Moving Average
Applied	Setzt den applied-Parameter von Moving Average
virtual InitIndicators	Initialisiert Indikatoren und Zeitreihen
virtual ValidationSettings	Überprüft die Einstellungen
Methoden zur Überprüfung der Trailing-Bedingungen	
virtual CheckTrailingStopLong	Findet hinzu, ob die Long-Position geändert werden soll
virtual CheckTrailingStopShort	Findet heraus, ob die Short-Position geändert werden soll

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries

© 2000-2025, MetaQuotes Ltd.

Period

Setzt den period-Parameter von Moving Average.

```
void  Period(
   int  period      // Parameter
   )
```

Parameter

period

[in] Wert des period-Parameters für Moving Average.

© 2000-2025, MetaQuotes Ltd.

Shift

Setzt den shift-Parameter von Moving Average.

```
void  Shift(
   int  shift      // Parameter
   )
```

Parameter

shift

[in] Wert des shift-Parameters für Moving Average.

© 2000-2025, MetaQuotes Ltd.

Method

Setzt den method-Parameter von Moving Average.

```
void  Method(
   ENUM_MA_METHOD  method      // Parameter
   )
```

Parameter

method

[in] Ein Wert des method-Parameters (aus Enumeration ENUM_MA_METHOD) für Moving Average.

© 2000-2025, MetaQuotes Ltd.

Applied

Setzt den applied-Parameter von Moving Average.

```
void  Applied(
   ENUM_APPLIED_PRICE  applied      // Parameter
   )
```

Parameter

applied

[in] Ein Wert des applied-Parameters (aus Enumeration ENUM_APPLIED_PRICE) für Moving Average.

© 2000-2025, MetaQuotes Ltd.

InitIndicators

Initialisiert Indikatoren und Zeitreihen.

```
virtual bool  InitIndicators(
   CIndicators*  indicators      // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf die Sammlung von Indikatoren und Zeitreihen, Mitglied der CExpert-Klasse.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Überprüft den Wert der Periode von Moving Average. Die gültige Werte sind größer als 0.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopLong

Findet hinzu, ob die Long-Position geändert werden soll.

```
virtual bool  CheckTrailingStopLong(
   CPositionInfo*  position,      // Zeiger
   double&         sl,            // Referenz
   double&         tp             // Referenz
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Berechnet den Wert der minimalen Position von Stop Loss. Berechnet den neuen Wert vom Stop Loss (aufgrund dem Wert von Moving Average auf dem vorherigen Balken). Wenn die Position schon ein Stop Loss hat, ist ihr Preis ein Basispreis, andernfalls wird Eröffnungspreis verwendet. Wenn der neue Wert von Stop Loss größer als Basispreis und kleiner als Minimalwert von Stop Loss ist, wird vorgeschlagen Stop Loss auf einen neuen Wert zu setzen.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopShort

Findet heraus, ob die Short-Position geändert werden soll.

```
virtual bool  CheckTrailingStopShort(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz
   double&         tp            // Referenz
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Berechnet den Wert der minimalen Position von Stop Loss. Berechnet den neuen Wert vom Stop Loss (aufgrund dem Wert von Moving Average auf dem vorherigen Balken einschließlich dem Spred-Wert). Wenn die Position schon ein Stop Loss hat, ist ihr Preis ein Basispreis, andernfalls wird Eröffnungspreis verwendet. Wenn der neue Wert von Stop Loss unter dem Basispreis und über dem Minimalwert von Stop Loss ist, wird vorgeschlagen Stop Loss auf einen neuen Wert zu setzen.

© 2000-2025, MetaQuotes Ltd.

CTrailingNone-Klasse

CTrailingNone ist eine Stub-Klasse. Man muss sie für Initialisierung eines Trailing-Objekts in der CExpert-Klasse verwenden, wenn die Handelsstrategie keinen Position-Trailing-Algorithmus hat.

Beschreibung

CTrailingNone implementiert keine Trailing-Algorithmen. The Methoden zur Überprüfung von Trailing-Bedingungen geben immer false zurück.

Deklaration

```
class CTrailingNone: public CExpertTrailing
```

Kopf

```
   #include <Expert\Trailing\TrailingNone.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertTrailing
      CTrailingNone

Gruppen der Klassenmethode

Methoden zur Überprüfung der Trailing-Bedingungen	
virtual CheckTrailingStopLong	Eine Stub-Methode zur Überprüfung, ob die Long-Position geändert werden soll
virtual CheckTrailingStopShort	Eine Stub-Methode zur Überprüfung, ob die Short-Position geändert werden soll

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, ValidationSettings, SetPriceSeries, SetOtherSeries, InitIndicators

Methoden geerbt von der Klasse CExpertTrailing

CheckTrailingStopLong, CheckTrailingStopShort

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopLong

Findet hinzu, ob die Long-Position geändert werden soll.

```
virtual bool  CheckTrailingStopLong(
   CPositionInfo*  position,      // Zeiger
   double&         sl,            // Referenz
   double&         tp             // Referenz
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Die Methode gibt immer false zurück.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopShort

Findet heraus, ob die Short-Position geändert werden soll.

```
virtual bool  CheckTrailingStopShort(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz
   double&         tp            // Referenz
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Die Methode gibt immer false zurück.

© 2000-2025, MetaQuotes Ltd.

CTrailingPSAR-Klasse

CTrailingPSAR ist eine Klasse für die Implementierung von Algorithmen für Position-Trailing aufgrund den Werten von Parabolic SAR-Indikator.

Beschreibung

CTrailingPSAR implementiert Einen Algorithmus für Position-Trailing aufgrund den Werten von Parabolic SAR-Indikator mit den angegebenen Parameter auf dem vorherigen Balken.

Deklaration

```
class CTrailingPSAR: public CExpertTrailing
```

Kopf

```
   #include <Expert\Trailing\TrailingParabolicSAR.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertTrailing
      CTrailingPSAR

Gruppen der Klassenmethode

Initialisierung	
Step	Setzt den step-Parameter von Parabolic SAR
Maximum	Setzt den maximum-Parameter von Parabolic SAR
virtual InitIndicators	Initialisiert Indikatoren und Zeitreihen
Methoden zur Überprüfung der Trailing-Bedingungen	
virtual CheckTrailingStopLong	Findet hinzu, ob die Long-Position geändert werden soll
virtual CheckTrailingStopShort	Findet heraus, ob die Short-Position geändert werden soll

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, ValidationSettings, SetPriceSeries, SetOtherSeries

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Step

Setzt den step-Parameter von Parabolic SAR.

```
void  Step(
   double  step      // Parameter
   )
```

Parameter

step

[in] Wert des step-Parameters für Parabolic SAR.

© 2000-2025, MetaQuotes Ltd.

Maximum

Setzt den maximum-Parameter von Parabolic SAR.

```
void  Maximum(
   double  maximum      // Parameter
   )
```

Parameter

maximum

[in] Wert des maximum-Parameters für Parabolic SAR.

© 2000-2025, MetaQuotes Ltd.

InitIndicators

Initialisiert Indikatoren und Zeitreihen.

```
virtual bool  InitIndicators(
   CIndicators*  indicators      // Zeiger
   )
```

Parameter

indicators

[in] Ein Zeiger auf die Sammlung von Indikatoren und Zeitreihen, Mitglied der CExpert-Klasse.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopLong

Findet hinzu, ob die Long-Position geändert werden soll.

```
virtual bool  CheckTrailingStopLong(
   CPositionInfo*  position,      // Zeiger
   double&         sl,            // Referenz
   double&         tp             // Referenz
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Berechnet den Wert der minimalen Position von Stop Loss. Berechnet den neuen Wert vom Stop Loss (aufgrund dem Wert von Parabolic SAR auf dem vorherigen Balken). Wenn die Position schon ein Stop Loss hat, ist ihr Preis ein Basispreis, andernfalls wird Eröffnungspreis verwendet. Wenn der neue Wert von Stop Loss größer als Basispreis und kleiner als Minimalwert von Stop Loss ist, wird vorgeschlagen Stop Loss auf einen neuen Wert zu setzen.

© 2000-2025, MetaQuotes Ltd.

CheckTrailingStopShort

Findet heraus, ob die Short-Position geändert werden soll.

```
virtual bool  CheckTrailingStopShort(
   CPositionInfo*  position,     // Zeiger
   double&         sl,           // Referenz
   double&         tp            // Referenz
   )
```

Parameter

position

[in] Zeiger auf das Objekt CPositionInfo.

sl

[in][out] Die Referenz an die Variable um den Wert des Preises von Stop Loss zu platzieren.

tp

[in][out] Die Referenz an die Variable um den Wert des Preises von Take Profit zu platzieren.

Rückgabewert

Gibt true zurück, wenn die Bedingung erfüllt ist, anderenfalls false.

Hinweis

Berechnet den Wert der minimalen Position von Stop Loss. Berechnet den neuen Wert vom Stop Loss (aufgrund dem Wert von Parabolic SAR auf dem vorherigen Balken einschließlich dem Spred-Wert). Wenn die Position schon ein Stop Loss hat, ist ihr Preis ein Basispreis, andernfalls wird Eröffnungspreis verwendet. Wenn der neue Wert von Stop Loss unter dem Basispreis und über dem Minimalwert von Stop Loss ist, wird vorgeschlagen Stop Loss auf einen neuen Wert zu setzen.

© 2000-2025, MetaQuotes Ltd.

Eine Reihe von vorgefertigten Implementierungen von Algorithmen für die Kapital- und Risikomanagement

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen von vorgefertigten Implementierungen von Algorithmen für Kapital- und Risikomanagement und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen von vorgefertigten Implementierungen von Algorithmen für Kapital- und Risikomanagement sparen Sie Zeit bei der Entwicklung (und vor allem beim Testen) von Handelsstrategien.

Die Standardbibliothek MQL5 (in Bezug auf die Klassen von vorgefertigten Implementierungen von Algorithmen für Kapital- und Risikomanagement) befindet sich im Arbeitsverzeichnis des Terminals im Ordner Include\Expert\Money.

Klasse	Beschreibung
CMoneyFixedLot	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Lot, das beim Einrichten angegeben war
CMoneyFixedMargin	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Marginniveau, das beim Einrichten angegeben war
CMoneyFixedRisk	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Risikowert, der beim Einrichten angegeben war
CMoneyNone	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit einem minimalen Lot
CMoneySizeOptimized	Die Klasse implementiert einen Algorithmus für Eintritt in den Markt mit dem Volumen, das durch früheren Transaktionen definiert wird

© 2000-2025, MetaQuotes Ltd.

Klasse CMoneyFixedLot

CMoneyFixedLot ist eine Klasse für die Implementierung von Algorithmus für den Handel mit festem Lot.

Beschreibung

Klasse CMoneYFixedLot implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Lot, das beim Einrichten angegeben war.

Deklaration

```
class CMoneyFixedLot: public CExpertMoney
```

Kopf

```
#include <Expert\Money\MoneyFixedLot.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertMoney
      CMoneyFixedLot

Gruppen der Klassenmethode

Initialisierung	
Lots	Setzt Handelsvolumen
virtual ValidationSettings	Überprüft die Einstellungen
Methoden zu Risiko- und Geldmanagement.	
virtual CheckOpenLong	Bestimmt das Volumen für Eröffnung einer Long-Position
virtual CheckOpenShort	Bestimmt das Volumen für Eröffnung einer Short-Position

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

Methoden geerbt von der Klasse CExpertMoney

Percent, CheckReverse, CheckClose

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Lots

Setzt Handelsvolumen (in Lots).

```
void  Lots(
   double  lots      // Anzahl von Lots
   )
```

Parameter

lots

[in] Handelsvolumen.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Kontrolliert Festlot, das beim Einrichten angegeben war, ob sein Wert in den Bereich der akzeptablen Werte ist und durch Änderungsschritt teilbar ist.

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Bestimmt das Volumen für Eröffnung einer Long-Position.

```
virtual double  CheckOpenLong(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Long-Position.

Hinweis

Die Methode bietet immer ein festes Lot, das beim Einrichten angegeben war.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Bestimmt das Volumen für Eröffnung einer Short-Position.

```
virtual double  CheckOpenShort(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Short-Position.

Hinweis

Die Methode bietet immer ein festes Lot, das beim Einrichten angegeben war.

© 2000-2025, MetaQuotes Ltd.

Klasse CMoneyFixedMargin

CMoneyFixedMargin ist eine Klasse für die Implementierung von Algorithmus für den Handel mit der festen Margin.

Beschreibung

Klasse CMoneyFixedMargin implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Marginniveau, das beim Einrichten angegeben war.

Deklaration

```
class CMoneyFixedMargin: public CExpertMoney
```

Kopf

```
#include <Expert\Money\MoneyFixedMargin.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertMoney
      CMoneyFixedMargin

Gruppen der Klassenmethode

Methoden zu Risiko- und Geldmanagement.	
virtual CheckOpenLong	Bestimmt das Volumen für Eröffnung einer Long-Position
virtual CheckOpenShort	Bestimmt das Volumen für Eröffnung einer Short-Position

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

Methoden geerbt von der Klasse CExpertMoney

Percent, ValidationSettings, CheckReverse, CheckClose

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Bestimmt das Volumen für Eröffnung einer Long-Position.

```
virtual double  CheckOpenLong(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Long-Position.

Hinweis

Die Methode bestimmt Eröffnungsvolumen für eine Short-Position mit einem festen Marginniveau, das beim Einrichten angegeben war. Das Marginniveau wird durch den Parameter Percent der Basisklasse CExpertMoney definiert.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Bestimmt das Volumen für Eröffnung einer Short-Position.

```
virtual double  CheckOpenShort(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Short-Position.

Hinweis

Die Methode bestimmt die Positionsgröße für die Eröffnung einer Short-Position mit einer festgelegten Margin, die bei der Einstellung gesetzt wurde. Das Marginniveau wird durch den Parameter Percent der Basisklasse CExpertMoney definiert.

© 2000-2025, MetaQuotes Ltd.

Klasse CMoneyFixedRisk

CMoneyFixedRisk ist eine Klasse für die Implementierung von Algorithmus für den Handel mit dem festen Risiko.

Beschreibung

Klasse CMoneyFixedRisk implementiert einen Algorithmus für Eintritt in den Markt mit einem festen Risikowert, der beim Einrichten angegeben war.

Deklaration

```
class CMoneyFixedRisk: public CExpertMoney
```

Kopf

```
#include <Expert\Money\MoneyFixedRisk.mqh>
```

Vererbungshierarchie

CObject

CExpertBase

CExpertMoney

CMoneyFixedRisk

Gruppen der Klassenmethode

Methoden zu Risiko- und Geldmanagement.	
virtual CheckOpenLong	Bestimmt das Volumen für Eröffnung einer Long-Position
virtual CheckOpenShort	Bestimmt das Volumen für Eröffnung einer Short-Position

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

Methoden geerbt von der Klasse CExpertMoney

Percent, ValidationSettings, CheckReverse

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Bestimmt das Volumen für Eröffnung einer Long-Position.

```
virtual double  CheckOpenLong(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Long-Position.

Hinweis

Die Methode bestimmt Eröffnungsvolumen für eine Short-Position mit einem festen Risikowert, der beim Einrichten angegeben war. Das Marginniveau wird durch den Parameter Percent der Basisklasse CExpertMoney definiert.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Bestimmt das Volumen für Eröffnung einer Short-Position.

```
virtual double  CheckOpenShort(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Short-Position.

Hinweis

Die Methode bestimmt das Volumen für die Eröffnung einer Short-Position mit einem festgelegten Risikoniveau, das bei der Einstellung gesetzt wurde. Das Marginniveau wird durch den Parameter Percent der Basisklasse CExpertMoney definiert.

© 2000-2025, MetaQuotes Ltd.

Klasse CMoneyNone

CMoneyNone ist eine Klasse für die Implementierung von "Kein" Risiko- und Geldmanagement.

Beschreibung

Klasse CMoneyNone implementiert einen Algorithmus für Eintritt in den Markt mit einem minimalen Lot.

Deklaration

```
class CMoneyNone: public CExpertMoney
```

Kopf

```
#include <Expert\Money\MoneyNone.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertMoney
      CMoneyNone

Gruppen der Klassenmethode

Initialisierung	
virtual ValidationSettings	Überprüft die Einstellungen
Methoden zu Risiko- und Geldmanagement.	
virtual CheckOpenLong	Bestimmt das Volumen für Eröffnung einer Long-Position
virtual CheckOpenShort	Bestimmt das Volumen für Eröffnung einer Short-Position

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

Methoden geerbt von der Klasse CExpertMoney

Percent, CheckReverse, CheckClose

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Die Methode gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Bestimmt das Volumen für Eröffnung einer Long-Position.

```
virtual double  CheckOpenLong(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Long-Position.

Hinweis

Die Methode bietet immer ein Minimallot.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Bestimmt das Volumen für Eröffnung einer Short-Position.

```
virtual double  CheckOpenShort(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Short-Position.

Hinweis

Die Methode bietet immer ein Minimallot.

© 2000-2025, MetaQuotes Ltd.

Klasse CMoneySizeOptimized

CMoneySizeOptimized ist eine Klasse für die Implementierung von Risiko- und Geldmanagement unter Berücksichtigung der Ergebnisse frühere Transaktionen.

Beschreibung

Klasse CMoneySizeOptimized implementiert einen Algorithmus für Eintritt in den Markt mit dem Volumen, das durch früheren Transaktionen definiert wird.

Deklaration

```
class CMoneySizeOptimized: public CExpertMoney
```

Kopf

```
#include <Expert\Money\MoneySizeOptimized.mqh>
```

Vererbungshierarchie

CObject
  CExpertBase
    CExpertMoney
      CMoneySizeOptimized

Gruppen der Klassenmethode

Initialisierung	
DecreaseFactor	Setzt den Wert des Parameters
virtual ValidationSettings	Überprüft die Einstellungen
Methoden zu Risiko- und Geldmanagement.	
virtual CheckOpenLong	Bestimmt das Volumen für Eröffnung einer Long-Position
virtual CheckOpenShort	Bestimmt das Volumen für Eröffnung einer Short-Position

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CExpertBase

InitPhase, TrendType, UsedSeries, EveryTick, Open, High, Low, Close, Spread, Time, TickVolume, RealVolume, Init, Symbol, Period, Magic, SetMarginMode, SetPriceSeries, SetOtherSeries, InitIndicators

Methoden geerbt von der Klasse CExpertMoney

Percent, CheckReverse, CheckClose

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

DecreaseFactor

Setzt den Parameter "Decrease Factor".

```
void  DecreaseFactor(
   double  decrease_factor      // Reduktionsfaktor
   )
```

Parameter

decrease_factor

[in] Ein Wert des Parameters "Reduktionsfaktor".

Hinweis

Der Parameter bestimmt den Faktor der Volumenreduktion für eine neue Position (bezogen auf das Volumen der vorangegangenen Position), wenn bevor es eine Serie von Verlust-Trades gab.

© 2000-2025, MetaQuotes Ltd.

ValidationSettings

Überprüft die Einstellungen.

```
virtual bool  ValidationSettings()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Überprüfungen der "non-Negativität" des Werts "Reduktionsfaktor", das beim Einrichten angegeben war.

© 2000-2025, MetaQuotes Ltd.

CheckOpenLong

Bestimmt das Volumen für Eröffnung einer Long-Position.

```
virtual double  CheckOpenLong(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Long-Position.

Hinweis

Die Methode bestimmt Eröffnungsvolumen für eine Long-Position unter Berücksichtigung von Ergebnisse der früheren Transaktionen.

© 2000-2025, MetaQuotes Ltd.

CheckOpenShort

Bestimmt das Volumen für Eröffnung einer Short-Position.

```
virtual double  CheckOpenShort(
   double  price,     // Preis
   double  sl         // Preis von Stop Loss
   )
```

Parameter

price

[in] Der geschätzte Eröffnungspreis.

sl

[in] Der geschätzte Preis von Stop-Loss-Order.

Rückgabewert

Volumen für Eröffnung einer Short-Position.

Hinweis

Die Methode bestimmt Eröffnungsvolumen für eine Short-Position unter Berücksichtigung von Ergebnisse der früheren Transaktionen.

© 2000-2025, MetaQuotes Ltd.

Klassen für die Erstellung von Anzeigeplatten und Steuerungsdialoge

Dieser Abschnitt enthält die technischen Details der Arbeit mit Klassen für die Erstellung von Anzeigeplatten und Steuerungsdialoge und die Beschreibungen der relevanten Komponenten der Standardbibliothek MQL5.

Durch die Verwendung der Klassen für die Erstellung der Anzeigeplatten und Steuerungsdialoge sparen Sie Zeit bei der Entwicklung von Ihren eigenen interaktiven MQL5-Anwendungen, z.B. Expert Advisors und Indikatoren.

Die Standardbibliothek MQL5 (in Bezug auf die Klassen für die Erstellung von Anzeigeplatten und Steuerungsdialoge) befindet sich im Arbeitsverzeichnis des Terminals im Ordner MQL5\Include\Controls.

Suchen Sie die Beispiele für das Arbeiten mit den Klassen in den folgenden Artikeln:

- Wie erstellt man ein grafisches Panel beliebiger Komplexität?
- Panels verbessern: Transparenz hinzufügen, Hintergrundfarbe ändern und von CAppDialog/CWndClient übernehmen
- Wie schnell ein Bedienfeld zu einem Indikator und Expert Advisor hinzugefügt werden kann
- Erstellen Ihrer eigenen grafischen Panels in MQL5
- Erstellen aktiver MQL5-Bedienfelder für den Handel

Ein Beispiel für einen Expert Advisor, welches illustriert die Arbeit dieser Klassen, ist in der Datei MQL5\Expert\Examples\Controls.

Zusatzstrukturen	Beschreibung
CRect	Die Struktur der rechteckigen Fläche des Charts
CDateTime	Die Struktur für die Arbeit mit dem Datum und der Zeit

Basisklassen	Beschreibung
CWnd	Die Basisklasse für alle Steuerelemente
CWndObj	Die Basisklasse für die Erstellung von Anzeigeplatten und Steuerungsdialoge
CWndContainer	Die Basisklasse für kombinierte Steuerelemente

Einfache Steuerelemente	Beschreibung
CLabel	Steuerelement basiert auf dem Chart-Objekt "Text-Label"

© 2000-2025, MetaQuotes Ltd.

Einfache Steuerelemente	Beschreibung
CBmpButton	Steuerelement basiert auf dem Chart-Objekt "Bitmap Label"
CButton	Steuerelement basiert auf dem Chart-Objekt "Taste"
CEdit	Steuerelement basiert auf dem Chart-Objekt "Eingabefeld"
CPanel	Steuerelement basiert auf dem Chart-Objekt "Rechteck-Label"
CPicture	Steuerelement basiert auf dem Chart-Objekt "Bitmap Label"

Kombinierte Steuerelemente	Beschreibung
CScroll	Die Basisklasse der Bildlaufleiste
CScrollV	Die vertikale Bildlaufleiste
CScrollH	Die horizontale Bildlaufleiste
CWndClient	Die Basisklasse der Bildlaufleiste
CListView	Listenansicht
CComboBox	Feld mit der Dropdown-Liste
CCheckBox	Schalter
CCheckGroup	Gruppenschalter mit unabhängigen Fixierung
CRadioButton	Ein Element des Gruppenschalters mit abhängigen Fixierung
CRadioGroup	Gruppenschalter mit abhängigen Fixierung
CSpinEdit	Feld von Inkrement/Dekrement
CDialog	Dialog
CAppDialog	Hauptdialog des MQL5-Programms

© 2000-2025, MetaQuotes Ltd.

Klasse CRect

Klasse CRect ist eine Implementierung der Klasse des rechteckigen Chart-Bereichs.

Beschreibung

Klasse CRect ist eine programmatische Implementierung des rechteckigen Chart-Bereichs in den kartesischen Koordinaten.

Deklaration

```
class CRect
```

Kopf

```
#include <Controls\Rect.mqh>
```

Gruppen der Klassenmethode

Eigenschaften	
Left	Erhält/setzt die X-Koordinate des oberen linken Punktes des Bereichs
Top	Erhält/setzt die Y-Koordinate des oberen linken Punktes des Bereichs
Right	Erhält/setzt die X-Koordinate des unteren rechten Punktes des Bereichs
Bottom	Erhält/setzt die Y-Koordinate des unteren rechten Punktes des Bereichs
Width	Erhält/setzt die Breite des Bereichs
Height	Erhält/setzt die Höhe des Bereichs
SetBound	Setzt neue Koordinaten des Bereichs
Move	Absolute Bewegung der Bereichskoordinaten
Shift	Relative Bewegung (Verschiebung) der Bereichskoordinaten
Contains	Überprüft ob ein Punkt innerhalb des Bereichs ist
Zusätzliche Methoden	
Format	Erhält die Koordinaten des Bereichs als String

© 2000-2025, MetaQuotes Ltd.

Left (Get-Methode)

Erhält die Y-Koordinate des oberen linken Punktes des rechteckigen Bereichs.

```
int  Left()
```

Rückgabewert

X-Koordinate des oberen linken Punktes.

Left (Set-Methode)

Setzt die Y-Koordinate des oberen linken Punktes des rechteckigen Bereichs.

```
void  Left(
   const int  x      // X-Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate des oberen linken Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Top (Get-Methode)

Erhält die X-Koordinate des oberen linken Punktes des rechteckigen Bereichs.

```
int  Top()
```

Rückgabewert

Y-Koordinate des oberen linken Punktes.

Top (Set-Methode)

Setzt die X-Koordinate des oberen linken Punktes des rechteckigen Bereichs.

```
void  Top(
   const int  y      // Y-Koordinate
   )
```

Parameter

y

[in] Der neue Wert der Y-Koordinate des oberen linken Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Right (Get-Methode)

Erhält die X-Koordinate des unteren rechten Punktes des Bereichs.

```
int  Right()
```

Rückgabewert

X-Koordinate des unteren rechten Punktes.

Right (Set-Methode)

Setzt die X-Koordinate des unteren rechten Punktes des Bereichs.

```
void  Right(
   const int  x      // X-Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate des unteren rechten Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Bottom (Get-Methode)

Erhält die Y-Koordinate des unteren rechten Punktes des Bereichs.

```
int  Bottom()
```

Rückgabewert

Y-Koordinate des unteren rechten Punktes.

Bottom (Set-Methode)

Setzt die Y-Koordinate des unteren rechten Punktes des Bereichs.

```
void  Bottom(
   const int  y      // Y-Koordinate
   )
```

Parameter

y

[in] Der neue Wert der Y-Koordinate des unteren rechten Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Width (Get-Methode)

Erhält die Breite des rechteckigen Bereichs.

```
int  Width()
```

Rückgabewert

Die Breite des rechteckigen Bereichs.

Width (Set-Methode)

Setzt die Breite des rechteckigen Bereichs.

```
virtual bool  Width(
   const int  w      // Breite
   )
```

Parameter

w

[in] Der neue Wert der Breite.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Height (Get-Methode)

Erhält die Höhe des rechteckigen Bereichs.

```
int  Height()
```

Rückgabewert

Die Höhe des rechteckigen Bereichs.

Height (Set-Methode)

Setzt die Höhe des rechteckigen Bereichs.

```
virtual bool  Height(
   const int  h      // Höhe
   )
```

Parameter

h

[in] Der neue Wert der Höhe.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

SetBound

Setzt die neuen Parameter des rechteckigen Bereichs des Steuerelements aus den Koordinaten der Klasse CRect.

```
void  SetBound(
   const & CRect  rect      // Klasse der Rechteckfläche
   )
```

Rückgabewert

Nichts.

SetBound

Setzt die neuen Parameter des rechteckigen Bereichs.

```
void  SetBound(
   const int  l      // left
   const int  t      // topt
   const int  r      // right
   const int  b      // bottom
   )
```

Parameter

l

[in] Die X-Koordinate des oberen linken Punktes.

t

[in] Die Y-Koordinate des oberen linken Punktes.

r

[in] Die X-Koordinate des unten rechten Punktes.

b

[in] Die Y-Koordinate des unten rechten Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Move

Absolute Bewegung der Koordinaten des rechteckigen Bereichs.

```
void  Move(
   const int  x,     // X-Koordinate
   const int  y      // Y-Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate.

y

[in] Der neue Wert der Y-Koordinate.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Shift

Relative Bewegung (Verschiebung) der Bereichskoordinaten.

```
void  Shift(
   const int  dx,     // Verschiebung entlang der X-Achse
   const int  dy      // Verschiebung entlang der Y-Achse
   )
```

Parameter

dx

[in] Die relative Verschiebung entlang der X-Achse.

dy

[in] Die relative Verschiebung entlang der Y-Achse.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Contains

Überprüft, ob ein Punkt mit angegebenen Koordinaten innerhalb des rechteckigen Bereichs ist.

```
bool  Contains(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] X-Koordinate des Punktes.

y

[in] Y-Koordinate des Punktes.

Rückgabewert

true, wenn der Punkt innerhalb des rechteckigen Bereichs (einschließlich der Grenze) ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Format

Erhält die Werte der Koordinaten des rechteckigen Bereichs als String

```
string  Format(
   string &  fmt,     // Format
   ) const
```

Parameter

fmt

[in] String mit der Formatbeschreibung.

Rückgabewert

Win String mit den Koordinaten des rechteckigen Bereichs.

© 2000-2025, MetaQuotes Ltd.

Struktur CDateTime

Die Struktur CDateTime wird für die Arbeit mit Datum und Zeit verwendet.

Beschreibung

Die Struktur CDateTime ist der Nachfolger von MqlDateTime, sie wird für die Arbeit mit Datum und Zeit in Steuerelemente verwendet.

Deklaration

```
struct CDateTime
```

Kopf

```
#include <Tools\DateTime.mqh>
```

Methoden

Eigenschaften	
MonthName	Gibt den Namen des Monats zurück
ShortMonthName	Gibt den Kurznamen des Monats zurück
DayName	Gibt den Vollnamen des Wochentags zurück
ShortDayName	Gibt den Kurznamen des Wochentags zurück
DaysInMonth	Gibt die Anzahl der Tage in einem Monat zurück
Methoden zum Setzen/Erhalt von Werten	
DateTime	Gibt zurück/setzt das Datum und die Zeit
Date	Setzt das Datum
Time	Setzt die Zeit
Sec	Setzt die Anzahl der Sekunden
Min	Setzt die Anzahl der Minuten
Hour	Setzt die Stunde
Day	Setzt den Tag des Monats
Mon	Setzt den Monat
Year	Setzt das Jahr
Zusätzliche Methoden	
SecDec	Zieht die angegebene Anzahl der Sekunden vom Datum ab
SecInc	Fügt die angegebene Anzahl der Sekunden zum Datum hinzu

© 2000-2025, MetaQuotes Ltd.

Eigenschaften	
MinDec	Zieht die angegebene Anzahl der Minuten vom Datum ab
MinInc	Fügt die angegebene Anzahl der Minuten zum Datum hinzu
HourDec	Zieht die angegebene Anzahl der Stunden vom Datum ab
HourInc	Fügt die angegebene Anzahl der Stunden zum Datum hinzu
DayDec	Zieht die angegebene Anzahl der Tage vom Datum ab
DayInc	Fügt die angegebene Anzahl der Tage zum Datum hinzu
MonDec	Zieht die angegebene Anzahl der Monaten vom Datum ab
MonInc	Fügt die angegebene Anzahl der Monaten zum Datum hinzu
YearDec	Zieht die angegebene Anzahl der Jahre vom Datum ab
YearInc	Fügt die angegebene Anzahl der Jahre zum Datum hinzu

© 2000-2025, MetaQuotes Ltd.

MonthName

Gibt den Namen des Monats zurück.

```
string  MonthName()  const
```

Gibt den Namen des Monats zurück nach seiner Nummer.

```
string  MonthName(
   const int      num         // Nummer des Monats
   )  const
```

Parameter

num

[in] Nummer des Monats (1-12).

Rückgabewert

Der Vollname des Monats.

© 2000-2025, MetaQuotes Ltd.

ShortMonthName

Gibt den Kurznamen des Monats zurück.

```
string  ShortMonthName()  const
```

Gibt den Kurznamen des Monats zurück nach seiner Nummer.

```
string  ShortMonthName(
   const int      num         // Nummer des Monats
   )  const
```

Parameter

num

[in] Nummer des Monats (1-12).

Rückgabewert

Der Kurzname des Monats.

© 2000-2025, MetaQuotes Ltd.

DayName

Gibt den Vollnamen des Wochentags zurück.

```
string  DayName()  const
```

Gibt den Vollnamen des Wochentags nach der Nummer zurück.

```
string  DayName(
   const int      num         // die Nummer des Wochentags
   )  const
```

Parameter

num

[in] Die Nummer des Wochentags (0-6).

Rückgabewert

Der Vollname des Wochentags.

© 2000-2025, MetaQuotes Ltd.

ShortDayName

Gibt den Kurznamen des Wochentags zurück.

```
string  ShortDayName()  const
```

Gibt den Kurznamen des Wochentags nach der Nummer zurück.

```
string  ShortDayName(
   const int      num         // die Nummer des Wochentags
   )  const
```

Parameter

num

[in] Die Nummer des Wochentags (0-6).

Rückgabewert

Der Kurzname des Wochentags.

© 2000-2025, MetaQuotes Ltd.

DaysInMonth

Gibt die Anzahl der Tage in einem Monat zurück.

```
int  DaysInMonth()  const
```

Rückgabewert

Die Anzahl der Tage in einem Monat.

© 2000-2025, MetaQuotes Ltd.

DateTime (Get-Methode)

Gibt das Datum und die Zeit zurück.

```
datetime  DateTime()
```

Rückgabewert

Wert vom Typ datetime.

DateTime (Methode Set datetime)

Setzt das Datum und die Zeit durch die Verwendung des Typs datetime.

```
void  DateTime(
   const datetime      value          // Datum und Zeit
   )
```

Parameter

value

[in] Wert vom Typ datetime.

Rückgabewert

Nichts.

DateTime (Methode Set MqlDateTime)

Setzt das Datum und die Zeit durch die Verwendung des Typs MqlDateTime.

```
void  DateTime(
   const MqlDateTime   &value         // Datum und Zeit
   )
```

Parameter

value

[in] Wert vom Typ MqlDateTime.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Date (Methode Set datetime)

Setzt das Datum durch die Verwendung des Typs datetime.

```
void  Date(
   const datetime      value          // Datum
   )
```

Parameter

value

[in] Wert vom Typ datetime.

Rückgabewert

Nichts.

Date (Methode Set MqlDateTime)

Setzt das Datum durch die Verwendung des Typs MqlDateTime.

```
void  Date(
   const MqlDateTime   &value         // Datum
   )
```

Parameter

value

[in] Wert vom Typ MqlDateTime.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Time (Methode Set datetime)

Setzt die Zeit durch die Verwendung des Typs datetime.

```
void  Time(
   const datetime      value          // Zeit
   )
```

Parameter

value

[in] Wert vom Typ datetime.

Rückgabewert

Nichts.

Time (Methode Set MqlDateTime)

Setzt die Zeit durch die Verwendung des Typs MqlDateTime.

```
void  Time(
   const MqlDateTime   &value         // Zeit
   )
```

Parameter

value

[in] Wert vom Typ MqlDateTime.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Sec

Setzt die Anzahl der Sekunden.

```
void  Sec(
   const int   value        // Anzahl der Sekunden
   )
```

Parameter

value

[in] Anzahl der Sekunden.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Min

Setzt die Anzahl der Minuten.

```
void  Min(
   const int   value        // Anzahl der Minuten
   )
```

Parameter

value

[in] Anzahl der Minuten.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Hour

Setzt die Stunde.

```
void  Hour(
   const int    value        // Stunde
   )
```

Parameter

value

[in] Stunde.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Day

Setzt den Tag des Monats.

```
void  Day(
   const int   value        // Tag des Monats
   )
```

Parameter

value

[in] Tag des Monats.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Mon

Setzt den Monat.

```
void  Mon(
   const int    value        // Nummer des Monats
   )
```

Parameter

value

[in] Nummer des Monats.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Year

Setzt das Jahr.

```
void  Year(
   const int    value        // Jahr
   )
```

Parameter

value

[in] Nummer des Jahres.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

SecDec

Zieht die angegebene Anzahl der Sekunden vom Datum ab.

```
void  SecDec(
   int   delta=1        // Anzahl der Sekunden
   )
```

Parameter

delta

[in] Anzahl der Sekunden.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

SecInc

Fügt die angegebene Anzahl der Sekunden zum Datum hinzu.

```
void  SecInc(
   int   delta=1        // Anzahl der Sekunden
   )
```

Parameter

delta

[in] Anzahl der Sekunden.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MinDec

Zieht die angegebene Anzahl der Minuten vom Datum ab.

```
void  MinDec(
   int   delta=1        // Anzahl der Minuten
   )
```

Parameter

delta

[in] Anzahl der Minuten .

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MinInc

Fügt die angegebene Anzahl der Minuten zum Datum hinzu.

```
void  MinInc(
   int   delta=1        // Anzahl der Minuten
   )
```

Parameter

delta

[in] Anzahl der Minuten .

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

HourDec

Zieht die angegebene Anzahl der Stunden vom Datum ab.

```
void  HourDec(
   int   delta=1        // Anzahl der Stunden
   )
```

Parameter

delta

[in] Anzahl der Stunden.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

HourInc

Fügt die angegebene Anzahl der Stunden zum Datum hinzu.

```
void  HourInc(
   int   delta=1        // Anzahl der Stunden
   )
```

Parameter

delta

[in] Anzahl der Stunden.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

DayDec

Zieht die angegebene Anzahl der Tage vom Datum ab.

```
void  DayDec(
   int   delta=1        // Anzahl der Tage
   )
```

Parameter

delta

[in] Anzahl der Tage.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

DayInc

Fügt die angegebene Anzahl der Tage zum Datum hinzu.

```
void  DayInc(
   int   delta=1        // days
   )
```

Parameter

delta

[in] Anzahl der Tage.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MonDec

Zieht die angegebene Anzahl der Monaten vom Datum ab.

```
void  MonDec(
   int   delta=1        // months
   )
```

Parameter

delta

[in] Anzahl der Monate.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MonInc

Fügt die angegebene Anzahl der Monaten zum Datum hinzu.

```
void  MonInc(
   int   delta=1        // Anzahl der Monate
   )
```

Parameter

delta

[in] Months to add.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

YearDec

Zieht die angegebene Anzahl der Jahre vom Datum ab.

```
void  YearDec(
   int   delta=1        // Anzahl der Jahre
   )
```

Parameter

delta

[in] Anzahl der Jahre.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

YearInc

Fügt die angegebene Anzahl der Jahre zum Datum hinzu.

```
void  YearInc(
   int   delta=1        // Anzahl der Jahre
   )
```

Parameter

delta

[in] Anzahl der Jahre.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Klasse CWnd

CWnd ist eine Basisklasse für alle Steuerelemente der Standardbibliothek.

Beschreibung

Klasse CWnd ist eine Software-Implementierung der Basisklasse des Steuerelements.

Deklaration

```
class CWnd : public CObject
```

Kopf

```
#include <Controls\Wnd.mqh>
```

Vererbungshierarchie

CObject

CWnd

Direkte Ableitungen

CDragWnd, CWndContainer, CWndObj

Gruppen der Klassenmethode

Erstellen und Löschen	
Create	Erstellt ein Element
Destroy	Löscht ein Element
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
OnMouseEvent	Behandelt nur das Chart-Ereignis CHARTEVENT_MOUSE_MOVE
Name	
Name	Erhält den Elementnamen
Der Mechanismus für den Zugang zum Container	
ControlsTotal	Erhält die Anzahl der Elementen im Container
Control	Erhält das Element des Containers am angegebenen Index
ControlFind	Erhält das Element des Containers mit der angegebenen ID
Geometrie	
Rect	Erhält die Koordinaten des Elements

© 2000-2025, MetaQuotes Ltd.

Erstellen und Löschen	
Left	Erhält/setzt die X-Koordinate der oberen linken Elementecke
Top	Erhält/setzt die Y-Koordinate der oberen linken Elementecke
Right	Erhält/setzt die X-Koordinate der unteren rechten Elementecke
Bottom	Erhält/setzt die Y-Koordinate der unteren rechten Elementecke
Width	Erhält/setzt die Elementbreite
Height	Erhält/setzt die Elementhöhe
Move	Absolute Bewegung des Elements
Shift	Relative Bewegung des Elements
Resize	Ändert die Elementgröße
Contains	Überprüft ob ein Punkt/Element innerhalb der Chartfläche, die das Element einnimmt, ist
Ausrichtung	
Alignment	Setzt die Parameter der Elementausrichtung
Align	Richtet ein Element in der angegebenen Chartfäche aus
Identifizierung	
Id	Erhält/setzt die Element-ID
Zustand	
IsEnabled	Überprüft die Fähigkeit des Elements auf Benutzeraktionen zu reagieren
Enable	Aktiviert die Fähigkeit des Elements auf Benutzeraktionen zu reagieren
Disable	Deaktiviert die Fähigkeit des Elements auf Benutzeraktionen zu reagieren
IsVisible	Prüft das Flag der Sichtbarkeit eines Elements
Visible	Setzt das Flag der Sichtbarkeit eines Elements auf dem Chart
Show	Blenden ein Steuerelement auf dem Chart ein
Hide	Blenden ein Steuerelement aus dem Chart aus
IsActive	Prüft Aktivität eines Elements
Activate	Macht ein Element aktiv
Deactivate	Macht ein Element inaktiv
Statusflags	
StateFlags	Erhält/ändert Statusflags eines Elements

© 2000-2025, MetaQuotes Ltd.

Erstellen und Löschen	
StateFlagsSet	Setzt Statusflags eines Elements
StateFlagsReset	Setzt Statusflags eines Elements zurück
Flags der Eigenschaften	
PropFlags	Erhält/ändert Flags der Eigenschaften eines Elements
PropFlagsSet	Setzt Flags der Eigenschaften eines Elements
PropFlagsReset	Setzt Flags der Eigenschaften eines Elements zurück
Für Mausoperationen	
MouseX	Erhält/speichert die X-Koordinate des Mauszeigers
MouseY	Erhält/speichert die Y-Koordinate des Mauszeigers
MouseFlags	Erhält/speichert den Status der Maustasten
MouseFocusKill	Löscht den Mausfokus
Behandlung von internen Ereignissen	
OnCreate	Handler des Ereignisses "Create" eines Steuerelements
OnDestroy	Handler des Ereignisses "Destroy" eines Steuerelements
OnMove	Handler des Ereignisses "Move" eines Steuerelements
OnResize	Handler des Ereignisses "Resize" eines Steuerelements
OnEnable	Handler des Ereignisses "Enable" eines Steuerelements
OnDisable	Handler des Ereignisses "Disable" eines Steuerelements
OnShow	Handler des Ereignisses "Show" eines Steuerelements
OnHide	Handler des Ereignisses "Hide" eines Steuerelements
OnActivate	Handler des Ereignisses "Activate" eines Steuerelements
OnDeactivate	Handler des Ereignisses "Deactivate" eines Steuerelements
OnClick	Handler des Ereignisses "Click" eines Steuerelements
OnChange	Handler des Ereignisses "Change" eines Steuerelements
Behandlung von Maus-Ereignissen	
OnMouseDown	Handler des Ereignisses "MouseDown" eines Steuerelements
OnMouseUp	Handler des Ereignisses "MouseUp" eines Steuerelements
Behandlung von Drag-Ereignissen	

© 2000-2025, MetaQuotes Ltd.

Erstellen und Löschen	
OnDragStart	Handler des Ereignisses "DragStart" eines Steuerelements
OnDragProcess	Handler des Ereignisses "DragProcess" eines Steuerelements
OnDragEnd	Handler des Ereignisses "DragEnd" eines Steuerelements
Arbeit mit dem Objekt	
DragObjectCreate	Erstellt ein Drag-Objekt
DragObjectDestroy	Löscht ein Drag-Objekt

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate x1
   const int      y1,          // Koordinate y1
   const int      x2,          // Koordinate x2
   const int      y2           // Koordinate y2
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Die Basisklassenmethode nur speichert die Erstellungsparameter und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

Destroy

Löscht ein Steuerelement.

```
virtual bool  Destroy()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Ereignisparameter
   const double&  dparam,      // Ereignisparameter
   const string&  sparam       // Ereignisparameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMouseEvent

Behandelt Mausereignisse (das Chart-Ereignis CHARTEVENT_MOUSE_MOVE).

```
virtual bool  OnMouseEvent(
   const int  x,          // X-Koordinate
   const int  y,          // Y-Koordinate
   const int  flags       // Flags
   )
```

Parameter

x

[in] X-Koordinate des Mauszeigers relativ zur oberen linken Ecke des Charts.

y

[in] Y-Koordinate des Mauszeigers relativ zur oberen linken Ecke des Charts.

flags

[in] Flags von Status der Maustasten.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Name

Erhält den Namen des Steuerobjekts.

```
string  Name()  const
```

Rückgabewert

Der Name des Steuerobjekts.

© 2000-2025, MetaQuotes Ltd.

ControlsTotal

Erhält die Anzahl der Steuerelementen im Container.

```
int  ControlsTotal()  const
```

Rückgabewert

Die Anzahl der Steuerelementen im Container.

Hinweis

Die Basisklassenmethode hat keinen Elementcontainer, sie bietet nur Zugriffsmechanismus für ihre Nachfolger und gibt immer 0 zurück.

© 2000-2025, MetaQuotes Ltd.

Control

Erhält das Steuerelement des Containers am angegebenen Index.

```
CWnd*  Control(
   const int  ind      // Index
   )  const
```

Parameter

ind

[in] Index des gewünschten Elements im Container.

Rückgabewert

Ein Zeiger auf ein Steuerelement aus dem Container.

Hinweis

Die Basisklassenmethode hat keinen Elementcontainer, sie bietet nur Zugriffsmechanismus für ihre Nachfolger und gibt immer NULL zurück.

© 2000-2025, MetaQuotes Ltd.

ControlFind

Erhält das Steuerelement des Containers an der angegebenen ID.

```
virtual CWnd*  ControlFind(
   const long  id      // ID
   )
```

Parameter

id

[in] Die ID des gewünschten Steuerelements.

Rückgabewert

Ein Zeiger auf ein Steuerelement aus dem Container.

Hinweis

Die Basisklassenmethode hat keinen Elementcontainer, sie bietet Zugriffsmechanismus für ihre Nachfolger. Wenn die gewünschte ID mit ihrer eigenen gleich ist, gibt ein Zeiger auf sie selbst (this) zurück.

© 2000-2025, MetaQuotes Ltd.

Rect

Erhält ein Zeiger auf das Objekt mit den Koordinaten des Steuerelements.

```
const CRect*  Rect()  const
```

Rückgabewert

Ein Zeiger auf das Objekt.

© 2000-2025, MetaQuotes Ltd.

Left (Get-Methode)

Erhält die X-Koordinate des oberen linken Punktes des Steuerelements.

```
int  Left()
```

Rückgabewert

X-Koordinate des oberen linken Punktes des Steuerelements.

Left (Set-Methode)

Setzt die X-Koordinate des oberen linken Punktes des Steuerelements.

```
virtual void  Left(
   const int  x      // Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate des oberen linken Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Top (Get-Methode)

Erhält die Y-Koordinate des oberen linken Punktes des Steuerelements.

```
int  Top()
```

Rückgabewert

Y-Koordinate des oberen linken Punktes des Steuerelements.

Top (Set-Methode)

Setzt die Y-Koordinate des oberen linken Punktes des Steuerelements.

```
virtual void  Top(
   const int  y      // Y-Koordinate
   )
```

Parameter

y

[in] Der neue Wert der Y-Koordinate des oberen linken Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Right (Get-Methode)

Erhält die X-Koordinate des unteren rechten Punktes eines Steuerelements.

```
int  Right()
```

Rückgabewert

X-Koordinate des unteren rechten Punktes des Steuerelements.

Right (Set-Methode)

Setzt die X-Koordinate des unteren rechten Punktes eines Steuerelements.

```
virtual void  Right(
   const int  x      // X-Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate des unteren rechten Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Bottom (Get-Methode)

Erhält die Y-Koordinate des unteren rechten Punktes eines Steuerelements.

```
int  Bottom()
```

Rückgabewert

Y-Koordinate des unteren rechten Punktes des Steuerelements.

Bottom (Set-Methode)

Setzt die Y-Koordinate des unteren rechten Punktes eines Steuerelements.

```
virtual void  Bottom(
   const int  y      // Y-Koordinate
   )
```

Parameter

y

[in] Der neue Wert der Y-Koordinate des unteren rechten Punktes.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Width (Get-Methode)

Erhält die Breite des Steuerelements.

```
int  Width()
```

Rückgabewert

Die Breite des Steuerelements.

Width (Set-Methode)

Setzt die Breite des Steuerelements.

```
virtual bool  Width(
   const int  w      // Breite
   )
```

Parameter

w

[in] Der neue Wert der Breite.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Height (Get-Methode)

Erhält die Höhe des Steuerelements.

```
int  Height()
```

Rückgabewert

Die Höhe des Steuerelements.

Height (Set-Methode)

Setzt die Höhe des Steuerelements.

```
virtual bool  Height(
   const int  h      // Höhe
   )
```

Parameter

h

[in] Der neue Wert der Höhe.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Move

Absolute Bewegung des Steuerelements.

```
virtual bool  Move(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate des oberen linken Punktes.

y

[in] Der neue Wert der Y-Koordinate des oberen linken Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Shift

Relative Bewegung des Steuerelements.

```
virtual bool  Shift(
   const int  dx,      // Verschiebung entlang der X-Achse
   const int  dy       // Verschiebung entlang der Y-Achse
   )
```

Parameter

dx

[in] Die relative Verschiebung entlang der X-Achse.

dy

[in] Die relative Verschiebung entlang der Y-Achse.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Resize

Ändert die Steuerelementgröße.

```
virtual bool  Resize(
   const int  w,
   const int  h
   )
```

Parameter

w

[in] Die neue Breite.

h

[in] Die neue Höhe.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Contains

Überprüft ob ein Punkt innerhalb der Chartfläche, die das Element einnimmt, ist.

```
bool  Contains(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] X-Koordinate des Punktes.

y

[in] Y-Koordinate des Punktes.

Rückgabewert

true, wenn der Punkt innerhalb der Chartfläche (einschließlich ihrer Grenze) ist, ansonsten false.

Contains

Überprüft ob das angegebene Element innerhalb der Chartfläche, die das Element einnimmt, ist.

```
bool  Contains(
   const CWnd*  control      // Zeiger
   )  const
```

Parameter

control

[in] Ein Zeiger auf das Element.

Rückgabewert

true, wenn das Element innerhalb der Chartfläche (einschließlich ihrer Grenze) ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Alignment

Setzt die Ausrichtung-Parameter eines Steuerelements.

```
void  Alignment(
   const int  flags,       // Flags
   const int  left,        // Einzug
   const int  top,         // Einzug
   const int  right,       // Einzug
   const int  bottom       // Einzug
   )
```

Parameter

flags

[in] Flags der Ausrichtung des Steuerelements.

left

[in] Fixierter Einzug vom linken Rand.

top

[in] Fixierter Einzug vom oberen Rand.

right

[in] Fixierter Einzug vom rechten Rand.

bottom

[in] Fixierter Einzug vom unteren Rand.

Rückgabewert

Nichts.

Hinweis

Ausrichtungflags:

```
enum WND_ALIGN_FLAGS
{
   WND_ALIGN_NONE=0,                                  // keine Ausrichtung
   WND_ALIGN_LEFT=1,                                  // links ausrichten
   WND_ALIGN_TOP=2,                                   // obere Ausrichtung
   WND_ALIGN_RIGHT=4,                                 // rechts ausrichten
   WND_ALIGN_BOTTOM=8,                                // untere Ausrichtung
   WND_ALIGN_WIDTH = WND_ALIGN_LEFT|WND_ALIGN_RIGHT,  // horizontale Ausrichtung
   WND_ALIGN_HEIGHT=WND_ALIGN_TOP|WND_ALIGN_BOTTOM,   // vertikale Ausrichtung
   WND_ALIGN_CLIENT=WND_ALIGN_WIDTH|WND_ALIGN_HEIGHT, // vertikale und horizontale Aus
   }
```

© 2000-2025, MetaQuotes Ltd.

Align

Richtet ein Element in der angegebenen Chartfäche aus.

```
virtual bool  Align(
   const CRect*  rect      // Zeiger
   )
```

Parameter

rect

[in] Ein Zeiger auf ein Objekt mit den Koordinaten der Chartfläche.

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

Hinweis

Ausrichtung-Parameter müssen im Voraus eingestellt werden (es gibt keine standardmäßige Ausrichtung).

© 2000-2025, MetaQuotes Ltd.

Id (Get-Methode)

Erhält die ID des Steuerelements.

```
long  Id()  const
```

Rückgabewert

Der Wert der Identifikator.

Id (Set-Methode)

Setzt den Wert der Identifikator des Steuerelements.

```
virtual long  Id(
   const long  id      // ID
   )
```

Parameter

x

[in] Der neue Wert der Identifikator des Steuerelements.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

IsEnabled

Überprüft die Fähigkeit des Steuerelements auf Benutzeraktionen zu reagieren.

```
bool  IsEnabled()  const
```

Rückgabewert

true wenn das Steuerelement auf Benutzeraktionen reagieren kann, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Enable

Aktiviert die Fähigkeit des Elements auf Benutzeraktionen zu reagieren.

```
virtual bool  Enable()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Wenn der Modus aktiv ist, behandelt das Element externe Ereignisse.

© 2000-2025, MetaQuotes Ltd.

Disable

Deaktiviert die Fähigkeit des Elements auf Benutzeraktionen zu reagieren.

```
virtual bool  Disable()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Wenn der Modus deaktiviert ist, behandelt das Element keine externe Ereignisse.

© 2000-2025, MetaQuotes Ltd.

IsVisible

Prüft die Sichtbarkeit eines Steuerelements.

```
bool  IsVisible()  const
```

Rückgabewert

true wenn das Steuerelement aktiv auf dem Chart angezeigt wird, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Visible

Setzt das SIchtbarkeit-Flag eines Steuerelements in den angegebenen Zustand

```
virtual bool  Visible(
   const bool  flag      // Flag
   )
```

Parameter

flag

[in] Der neue Flag-Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Show

Macht ein Steuerelement sichtbar.

```
virtual bool  Show()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Hide

Macht ein Steuerelement unsichtbar.

```
virtual bool  Hide()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

IsActive

Prüft Aktivität eines Steuerelements.

```
bool  IsActive()  const
```

Rückgabewert

true wenn das Steuerelement aktiv ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Activate

Macht ein Steuerelement aktiv.

```
virtual bool  Activate()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Das Element wird aktiv, wenn Sie den Mauszeiger darüber bewegen.

© 2000-2025, MetaQuotes Ltd.

Deactivate

Macht ein Steuerelement inaktiv.

```
virtual bool  Deactivate()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Das Element wird inaktiv, wenn Sie den Mauszeiger davor weg bewegen.

© 2000-2025, MetaQuotes Ltd.

StateFlags (Get-Methode)

Erhält Flags der Elementeigenschaften.

```
int  StateFlags()
```

Rückgabewert

Flags des Zustands des Steuerelements.

StateFlags (Set-Methode)

Setzt Flags des Zustands des Steuerelements.

```
virtual void  StateFlags(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Eine neue Kombination von Zustand-Flags.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

StateFlagsSet

Setzt Flags der Elementeigenschaften.

```
virtual void  StateFlagsSet(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Die Kombination von Eigenschaft-Flags, die gesetzt werden soll.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

StateFlagsReset

Setzt Flags der Elementeigenschaften zurück.

```
virtual void  StateFlagsReset(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Die Kombination von Eigenschaft-Flags, die zurückgesetzt werden soll.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

PropFlags (Get-Methode)

Erhält Flags der Elementeigenschaften.

```
void  PropFlags(
   const int  flags      // Flags
   )
```

Rückgabewert

Flags der Elementeigenschaften.

PropFlags (Set-Methode)

Setzt Flags der Elementeigenschaften.

```
virtual void  PropFlags(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Eine neue Kombination von Eigenschaft-Flags.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

PropFlagsSet

Setzt Flags der Elementeigenschaften.

```
virtual void  PropFlagsSet(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Die Kombination von Eigenschaft-Flags, die gesetzt werden soll.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

PropFlagsReset

Setzt Flags der Elementeigenschaften zurück.

```
virtual void  PropFlagsReset(
   const int  flags      // Flags
   )
```

Parameter

flags

[in] Die Kombination von Eigenschaft-Flags, die zurückgesetzt werden soll.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MouseX (Set-Methode)

Speichert die X-Koordinate des Mauszeigers.

```
void  MouseX(
   const int  value      // Koordinate
   )
```

Parameter

value

[in] X-Koordinate des Mauszeigers.

Rückgabewert

Nichts.

MouseX (Get-Methode)

Erhält die X-Koordinate des Mauszeigers.

```
int  MouseX()
```

Rückgabewert

X-Koordinate des Mauszeigers.

© 2000-2025, MetaQuotes Ltd.

MouseY (Set-Methode)

Speichert die Y-Koordinate des Mauszeigers.

```
void  MouseY(
   const int  value      // Koordinate
   )
```

Parameter

value

[in] Y-Koordinate des Mauszeigers.

Rückgabewert

Nichts.

MouseY (Get-Methode)

Erhält die Y-Koordinate des Mauszeigers.

```
int  MouseY()
```

Rückgabewert

Y-Koordinate des Mauszeigers.

© 2000-2025, MetaQuotes Ltd.

MouseFlags (Set-Methode)

Speichert den Status der Maustasten.

```
virtual void  MouseFlags(
   const int  value      // Status
   )
```

Parameter

value

[in] Status der Maustasten.

Rückgabewert

Nichts.

MouseFlags (Get-Methode)

Erhält den Status der Maustasten.

```
int  MouseFlags()
```

Rückgabewert

Status der Maustasten.

© 2000-2025, MetaQuotes Ltd.

MouseFocusKill

Löscht den gespeicherten Mausstatus und deaktiviert das Steuerelement.

```
bool  MouseFocusKill(
   const long  id=CONTROLS_INVALID_ID      // Identifikator
   )
```

Parameter

id=CONTROLS_INVALID_ID

[in] Die ID des Steuerelements, das den Mausfokus erhielt.

Rückgabewert

Das Ergebnis der Deaktivierung des Steuerelements.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnDestroy

Virtueller Handler des internen Ereignisses "Destroy" (löschen) des Steuerelements.

```
virtual bool  OnDestroy()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnEnable

Virtueller Handler des internen Ereignisses "Enable" (Aktivieren der Fähigkeit auf Benutzeraktionen zu reagieren) des Steuerelements.

```
virtual bool  OnEnable()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnDisable

Virtueller Handler des internen Ereignisses "Disable" (Deaktivieren der Fähigkeit auf Benutzeraktionen zu reagieren) des Steuerelements.

```
virtual bool  OnDisable()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnActivate

Virtueller Handler des internen Ereignisses "Activate" (Aktivieren) des Steuerelements.

```
virtual bool  OnActivate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnDeactivate

Virtueller Handler des internen Ereignisses "Deactivate" (Deaktivieren) des Steuerelements.

```
virtual bool  OnDeactivate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnClick

Virtueller Handler des internen Ereignisses "Click" (ein Mausklick) vom Steuerelement.

```
virtual bool  OnClick()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChange

Virtueller Handler des internen Ereignisses "Change" (Änderung) des Steuerelements.

```
virtual bool  OnChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnMouseDown

Virtueller Handler des Maus-Ereignisses "MouseDown" (das Drücken der Maustaste) von Steuerelement.

```
virtual bool  OnMouseDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Die Ereignis "MouseDown" erscheint beim Drücken auf die linke Maustaste, wenn sich der Mauszeiger über dem Steuerelement befindet.

© 2000-2025, MetaQuotes Ltd.

OnMouseUp

Virtueller Handler des Maus-Ereignisses "MouseUp" (Maustaste loslassen) von Steuerelement.

```
virtual bool  OnMouseUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Die Ereignis "MouseUp" erscheint beim Loslassen der linken Maustaste, wenn sich der Mauszeiger über dem Steuerelement befindet.

© 2000-2025, MetaQuotes Ltd.

OnDragStart

Virtueller Handler des Ereignisses "DragStart" (der Anfang der Drag-Operation) des Steuerelements.

```
virtual bool  OnDragStart()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "DragStart" tritt am Anfang des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnDragProcess

Virtueller Handler des Ereignisses "DragProcess" (Drag-Operation) des Steuerelements.

```
virtual bool  OnDragProcess(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] Der aktuelle Wert der X-Koordinate des Mauszeigers.

y

[in] Der aktuelle Wert der Y-Koordinate des Mauszeigers.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "DragProcess" tritt während des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnDragEnd

Virtueller Handler des Ereignisses "DragEnd" (das Ende der Drag-Operation) des Steuerelements.

```
virtual bool  OnDragEnd()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "DragEnd" tritt am Ende des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

DragObjectCreate

Erstellt ein Drag-Objekt.

```
virtual bool  DragObjectCreate()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

DragObjectDestroy

Löscht ein Drag-Objekt.

```
virtual bool  DragObjectDestroy()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Klasse CWndContainer

CWndContainer ist eine Basisklasse für die Gruppe der Steuerelemente der Standardbibliothek.

Beschreibung

Klasse CWndContainer ist eine Software-Implementierung der Gruppe von funktionell verwandten Steuerelementen.

Deklaration

```
class CWndContainer : public CWnd
```

Kopf

```
#include <Controls\WndContainer.mqh>
```

Vererbungshierarchie

CObject

CWnd

CWndContainer

Direkte Ableitungen

CCheckBox, CComboBox, CDateDropList, CDatePicker, CDialog, CRadioButton, CScroll, CSpinEdit, CWndClient

Gruppen der Klassenmethode

Löschen	
Destroy	Löscht eine Elementengruppe
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
OnMouseEvent	Behandelt nur das Chart-Ereignis CHARTEVENT_MOUSE_MOVE
Zugriff auf den Inhalt des Containers	
ControlsTotal	Erhält die Anzahl der Elementen im Container
Control	Das Element des Containers am angegebenen Index
ControlFind	Erhält das Element des Containers mit der angegebenen ID
Füllung	
Add	Fügt ein Element in der Gruppe (Container) hinzu
Delete	Löscht ein Element aus der Gruppe (Container) hinzu

© 2000-2025, MetaQuotes Ltd.

Löschen	
Geometrie	
Move	Absolute Bewegung einer Elementengruppe
Shift	Relative Bewegung einer Elementengruppe
Identifizierung	
Id	Setzt die Identifikatoren der Elementen in der Gruppe.
Zustand	
Enable	Aktiviert die Fähigkeit der Gruppe auf Benutzeraktionen zu reagieren
Disable	Deaktiviert die Fähigkeit der Gruppe auf Benutzeraktionen zu reagieren
Show	Macht eine Elementengruppe sichtbar
Hide	Macht eine Elementengruppe unsichtbar
Für Mausoperationen	
MouseFocusKill	Löscht den Mausfokus
Arbeiten mit Dateien	
Save	Schreibt Gruppeninformationen in eine Datei
Load	Liest Gruppeninformationen aus einer Datei
Behandlung von internen Ereignissen	
OnResize	Handler des Ereignisses "Resize" der Elementengruppe
OnActivate	Handler des Ereignisses "Activate" der Elementengruppe
OnDeactivate	Handler des Ereignisses "Deactivate" der Elementengruppe

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Create, Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

© 2000-2025, MetaQuotes Ltd.

Destroy

Löscht eine Gruppe von Steuerelementen.

```
virtual bool  Destroy()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMouseEvent

Behandelt Maus-Ereignisse.

```
virtual bool  OnMouseEvent(
   const int  x,          // Koordinate
   const int  y,          // Koordinate
   const int  flags       // Flags
   )
```

Parameter

x

[in] X-Koordinate des Mauszeigers relativ zur oberen linken Ecke des Charts.

y

[in] Y-Koordinate des Mauszeigers relativ zur oberen linken Ecke des Charts.

flags

[in] Flags von Status der Maustasten.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

ControlsTotal

Erhält die Anzahl der Steuerelementen im Container.

```
int  ControlsTotal()  const
```

Rückgabewert

Die Anzahl der Steuerelementen im Container.

© 2000-2025, MetaQuotes Ltd.

Control

Erhält das Steuerelement des Containers am angegebenen Index.

```
CWnd*  Control(
   const int  ind      // Index
   )  const
```

Parameter

ind

[in] Index des gewünschten Elements im Container.

Rückgabewert

Ein Zeiger auf ein Steuerelement oder NULL, wenn kein Element gefunden war.

© 2000-2025, MetaQuotes Ltd.

ControlFind

Erhält das Steuerelement des Containers an der angegebenen ID.

```
virtual CWnd*  ControlFind(
   const long  id      // ID
   )
```

Parameter

id

[in] Die ID des gewünschten Steuerelements.

Rückgabewert

Ein Zeiger auf ein Steuerelement oder NULL, wenn kein Element gefunden war.

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Element in der Gruppe (Container) hinzu.

```
bool  Add(
   CWnd&  control      // Referenz
   )
```

Parameter

control

[in] Referenz auf das hinzugefügte Steuerelement.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Delete

Löscht ein Element aus der Gruppe (Container) hinzu.

```
bool  Delete(
   CWnd&  control      // Referenz
   )
```

Parameter

control

[in] Referenz auf das Steuerelement zu löschen.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Move

Absolute Bewegung einer Elementengruppe.

```
virtual bool  Move(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] Der neue Wert der X-Koordinate des oberen linken Punktes.

y

[in] Der neue Wert der Y-Koordinate des oberen linken Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Shift

Relative Bewegung einer Elementengruppe.

```
virtual bool  Shift(
   const int  dx,      // Verschiebung entlang der X-Achse
   const int  dy       // Verschiebung entlang der Y-Achse
   )
```

Parameter

dx

[in] Die relative Verschiebung entlang der X-Achse.

dy

[in] Die relative Verschiebung entlang der Y-Achse.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Id

Setzt die Identifikatoren der Elementen in der Gruppe.

```
virtual long  Id(
   const long  id      // ID
   )
```

Parameter

id

[in] Basisindikator der Gruppe.

Rückgabewert

Die Anzahl der von der Gruppe genutzten IDs.

© 2000-2025, MetaQuotes Ltd.

Enable

Aktiviert die Fähigkeit der Gruppe auf Benutzeraktionen zu reagieren.

```
virtual bool  Enable()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Disable

Deaktiviert die Fähigkeit der Gruppe auf Benutzeraktionen zu reagieren.

```
virtual bool  Disable()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Show

Macht eine Elementengruppe sichtbar.

```
virtual bool  Show()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Hide

Macht eine Elementengruppe unsichtbar.

```
virtual bool  Hide()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

MouseFocusKill

Löscht den gespeicherten Mausstatus und deaktiviert die Elementengruppe.

```
virtual bool  MouseFocusKill(
   const long  id=CONTROLS_INVALID_ID      // Identifikator
   )
```

Parameter

id=CONTROLS_INVALID_ID

[in] Die ID des Steuerelements, das den Mausfokus erhielt.

Rückgabewert

Das Ergebnis der Deaktivierung der Elementengruppe.

© 2000-2025, MetaQuotes Ltd.

Save

Schreibt Gruppeninformationen in eine Datei.

```
virtual bool  Save(
   const int  file_handle      // Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die früher für Schreiben geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Load

Liest Gruppeninformationen aus einer Datei.

```
virtual bool  Load(
   const int  file_handle      // Handle
   )
```

Parameter

file_handle

[in] Handle einer Datei, die früher für Lesen geöffnet wurde.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnActivate

Virtueller Handler des internen Ereignisses "Activate" (Aktivieren) des Steuerelements.

```
virtual bool  OnActivate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnDeactivate

Virtueller Handler des internen Ereignisses "Deactivate" (Deaktivieren) des Steuerelements.

```
virtual bool  OnDeactivate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

Klasse CWndObj

CWndObj ist eine Basisklasse für die einfache (auf CHartobjekte basierende) Steuerelemente der Standardbibliothek.

Beschreibung

Klasse CWndObj ist eine Software-Implementierung der Basis des einfachen Steuerelements.

Deklaration

```
class CWndObj : public CWnd
```

Kopf

```
#include <Controls\WndObj.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndObj

Direkte Ableitungen

CBmpButton, CButton, CEdit, CLabel, CPanel, CPicture

Gruppen der Klassenmethode

Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Parameter	
Text	Erhält/setzt den Wert der Eigenschaft OBJPROP_TEXT eines Chartobjekts
Color	Erhält/setzt den Wert der Eigenschaft OBJPROP_COLOR eines Chartobjekts
ColorBackground	Erhält/setzt den Wert der Eigenschaft OBJPROP_BGCOLOR eines Chartobjekts
ColorBorder	Erhält/setzt den Wert der Eigenschaft OBJPROP_BORDER_COLOR eines Chartobjekts
Font	Erhält/setzt den Wert der Eigenschaft OBJPROP_FONT eines Chartobjekts
FontSize	Erhält/setzt den Wert der Eigenschaft OBJPROP_FONTSIZE eines Chartobjekts

© 2000-2025, MetaQuotes Ltd.

Behandlung von Chart-Ereignissen	
ZOrder	Erhält/setzt den Wert der Eigenschaft OBJPROP_ZORDER eines Chartobjekts
Behandlung der Ereignissen von Chartobjekten	
OnObjectCreate	Handler des Ereignisses CHARTEVENT_OBJECT_CREATE eines Chartobjekts
OnObjectChange	Handler des Ereignisses CHARTEVENT_OBJECT_CHANGE eines Chartobjekts
OnObjectDelete	Handler des Ereignisses CHARTEVENT_OBJECT_DELETE eines Chartobjekts
OnObjectDrag	Handler des Ereignisses CHARTEVENT_OBJECT_DRAG eines Chartobjekts
Parameteränderungenbehandlung	
OnSetText	Handler des Ereignisses "SetText" eines Steuerelements
OnSetColor	Handler des Ereignisses "SetColor" eines Steuerelements
OnSetColorBackground	Handler des Ereignisses "SetColorBackground" eines Steuerelements
OnSetFont	Handler des Ereignisses "SetFont" eines Steuerelements
OnSetFontSize	Handler des Ereignisses "SetFontSize" eines Steuerelements
OnSetZOrder	Handler des Ereignisses "SetZOrder" eines Steuerelements
Behandlung von internen Ereignissen	
OnDestroy	Handler des internen Ereignisses "Destroy" eines Steuerelements
OnChange	Handler des internen Ereignisses "Change" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Create, Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show, Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Text (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_TEXT (Text) eines Chartobjekts.

```
string  Text()
```

Rückgabewert

Der Wert OBJPROP_TEXT des Chartobjekts.

Text (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_TEXT (Text) eines Chartobjekts.

```
bool  Text(
   const string  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_TEXT.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Color (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_COLOR (Farbe) eines Chartobjekts.

```
color  Color()
```

Rückgabewert

Der Wert der Eigenschaft OBJPROP_COLOR des Chartobjekts.

Color (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_COLOR (Farbe) eines Chartobjekts.

```
bool  Color(
   const color  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_COLOR.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ColorBackground (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_BGCOLOR (Hintergrundfarbe) eines Chartobjekts.

```
color  ColorBackground()
```

Rückgabewert

Der Wert der Eigenschaft OBJPROP_BGCOLOR des Chartobjekts.

ColorBackground (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_BGCOLOR (Farbe) eines Chartobjekts.

```
bool  ColorBackground(
   const color  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_BGCOLOR.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ColorBorder (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_BORDER_COLOR (Rahmenfarbe) eines Chartobjekts.

```
color  ColorBorder()
```

Rückgabewert

Der Wert der Eigenschaft OBJPROP_BORDER_COLOR des Chartobjekts.

ColorBorder (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_BORDER_COLOR (Rahmenfarbe) eines Chartobjekts.

```
bool  ColorBorder(
   const color  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_BORDER_COLOR.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Font (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_FONT (Schrift) eines Chartobjekts.

```
string  Font()
```

Rückgabewert

Der Wert der Eigenschaft OBJPROP_FONT des Chartobjekts.

Font (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_FONT (Schrift) eines Chartobjekts.

```
bool  Font(
   const string  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_FONT.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

FontSize (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_FONTSIZE (Schriftgröße) eines Chartobjekts.

```
int  FontSize()
```

Rückgabewert

Der Wert der Eigenschaft OBJPROP_FONTSIZE des Chartobjekts.

FontSize (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_FONTSIZE (Schriftgröße) eines Chartobjekts.

```
bool  FontSize(
   const int  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_FONTSIZE.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ZOrder (Get-Methode)

Erhält den Wert der Eigenschaft OBJPROP_ZORDER eines Chartobjekts.

```
long  ZOrder()
```

Rückgabewert

Wert der Eigenschaft OBJPROP_ZORDER des Chartobjekts.

ZOrder (Set-Methode)

Setzt den Wert der Eigenschaft OBJPROP_ZORDER eines Chartobjekts.

```
bool  ZOrder(
   const long  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft OBJPROP_ZORDER.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnObjectCreate

Virtueller Handler des Ereignisses CHARTEVENT_OBJECT_CREATE des Chartobjekts.

```
virtual bool  OnObjectCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnObjectChange

Virtueller Handler des Ereignisses CHARTEVENT_OBJECT_CHANGE des Chartobjekts.

```
virtual bool  OnObjectChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnObjectDelete

Virtueller Handler des Ereignisses CHARTEVENT_OBJECT_DELETE des Chartobjekts.

```
virtual bool  OnObjectDelete()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnObjectDrag

Virtueller Handler des Ereignisses CHARTEVENT_OBJECT_DRAG des Chartobjekts.

```
virtual bool  OnObjectDrag()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetText

Virtueller Handler des Ereignisses "SetText" (Änderung der Eigenschaft OBJPROP_TEXT) des Steuerelements.

```
virtual bool  OnSetText()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetColor

Virtueller Handler des Ereignisses "SetColor" (Änderung der Eigenschaft OBJPROP_COLOR) des Steuerelements.

```
virtual bool  OnSetColor()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBackground

Virtueller Handler des Ereignisses "SetColorBackground" (Änderung der Eigenschaft OBJPROP_BGCOLOR) des Steuerelements.

```
virtual bool  OnSetColorBackground()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetFont

Virtueller Handler des Ereignisses "SetFont" (Änderung der Eigenschaft OBJPROP_FONT) des Steuerelements.

```
virtual bool  OnSetFont()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetFontSize

Virtueller Handler des Ereignisses "SetFontSize" (Änderung der Eigenschaft OBJPROP_FONTSIZE) des Steuerelements.

```
virtual bool  OnSetFontSize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetZOrder

Virtueller Handler des Ereignisses "SetZOrder" (Änderung des Parameters OBJPROP_ZORDER) des Steuerelements.

```
virtual bool  OnSetZOrder()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnDestroy

Virtueller Handler des internen Ereignisses "Destroy" (löschen) des Steuerelements.

```
virtual bool  OnDestroy()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChange

Virtueller Handler des internen Ereignisses "Change" (Änderung) des Steuerelements.

```
virtual bool  OnChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CLabel

Klasse CLabel ist eine Klasse des einfachen Steuerelements basierend auf das grafischen Objekt "Text-Label".

Beschreibung

Klasse CLabel wird für die Erstellung von einfachen nicht-bearbeitbaren Textfeldern verwendet.

Deklaration

```
class CLabel : public CWndObj
```

Kopf

```
#include <Controls\Label.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndObj
      CLabel

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Parameteränderungenbehandlung	
OnSetText	Handler des Ereignisses "SetText" eines Steuerelements
OnSetColor	Handler des Ereignisses "SetColor" eines Steuerelements
OnSetFont	Handler des Ereignisses "SetFont" eines Steuerelements
OnSetFontSize	Handler des Ereignisses "SetFontSize" eines Steuerelements
Behandlung von internen Ereignissen	
OnCreate	Handler des internen Ereignisses "Create" eines Steuerelements
OnShow	Handler des internen Ereignisses "Show" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnMove	Handler des internen Ereignisses "Move" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show, Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndObj

OnEvent, Text, Text, Color, Color, ColorBackground, ColorBackground, ColorBorder, ColorBorder, Font, Font, FontSize, FontSize, ZOrder, ZOrder

Ein Beispiel für die Erstellung eines Panels mit einem Text-Label:

```
//+------------------------------------------------------------------+
//|                                                ControlsLabel.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Control Panels and Dialogs. Demonstration class CLabel"
#include <Controls\Dialog.mqh>
#include <Controls\Label.mqh>
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CLabel            m_label;                        // CLabel object
public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons
protected:
   //--- create dependent controls
   bool              CreateLabel(void);
   //--- handlers of the dependent controls events
   void              OnClickLabel(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)

EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateLabel())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "CLabel"                                              |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateLabel(void)
  {
//--- coordinates
   int x1=INDENT_RIGHT;
   int y1=INDENT_TOP+CONTROLS_GAP_Y;
   int x2=x1+100;
   int y2=y1+20;
//--- create
   if(!m_label.Create(m_chart_id,m_name+"Label",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!m_label.Text("Label"))
      return(false);
   if(!Add(m_label))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CLabel.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate x1
   const int      y1,          // Koordinate y1
   const int      x2,          // Koordinate x2
   const int      y2           // Koordinate y2
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetText

Virtueller Handler des Ereignisses "SetText" (Änderung der Eigenschaft OBJPROP_TEXT) des Steuerelements CLabel.

```
virtual bool  OnSetText()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColor

Virtueller Handler des Ereignisses "SetColor" (Änderung der Eigenschaft OBJPROP_COLOR) des Steuerelements CLabel.

```
virtual bool  OnSetColor()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetFont

Virtueller Handler des Ereignisses "SetFont" (Änderung der Eigenschaft OBJPROP_FONT) des Steuerelements CLabel.

```
virtual bool  OnSetFont()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetFontSize

Virtueller Handler des Ereignisses "SetFontSize" (Änderung der Eigenschaft OBJPROP_FONTSIZE) des Steuerelements CLabel.

```
virtual bool  OnSetFontSize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements CLabel.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CLabel.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CLabel.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements CLabel.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CBmpButton

Klasse CBmpButton ist eine Klasse des einfachen Steuerelements basierend auf das grafischen Objekt "Bitmap Label".

Beschreibung

Klasse CBmpButton wird für die Erstellung von Tasten mit einem grafischen Bild verwendet.

Deklaration

```
class CBmpButton : public CWndObj
```

Kopf

```
#include <Controls\BmpButton.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndObj
      CBmpButton

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Steuerelement-Parameter	
Border	Erhält/setzt die Eigenschaft "Border" eines Steuerelements
BmpNames	Setzt die Namen der BMP-Dateien zur Anzeige eines Steuerelements
BmpOffName	Erhält/Setzt den Namen der BMP-Datei zur Anzeige eines Steuerelements im OFF-Zustand
BmpOnName	Erhält/Setzt den Namen der BMP-Datei zur Anzeige eines Steuerelements im ON-Zustand
BmpPassiveName	Erhält/Setzt den Namen der BMP-Datei zur Anzeige eines Steuerelements im inaktiven Zustand
BmpActiveName	Erhält/Setzt den Namen der BMP-Datei zur Anzeige eines Steuerelements im aktiven Zustand
Eigenschaften	
Pressed	Erhält/setzt den Zustand eines Steuerelements
Locking	Erhält/setzt den Wert der Eigenschaft "Locking" eines Steuerelements
Behandlung von internen Ereignissen	
OnSetZOrder	Handler des internen Ereignisses "SetZOrder" eines Steuerelements
OnCreate	Handler des internen Ereignisses "Create" eines Steuerelements
OnShow	Handler des internen Ereignisses "Show" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnMove	Handler des internen Ereignisses "Move" eines Steuerelements
OnChange	Handler des internen Ereignisses "Change" eines Steuerelements
OnActivate	Handler des internen Ereignisses "Activate" eines Steuerelements
OnDeactivate	Handler des internen Ereignisses "Deactivate" eines Steuerelements
OnMouseDown	Handler des internen Ereignisses "MouseDown" eines Steuerelements
OnMouseUp	Handler des internen Ereignisses "MouseUp" eines Steuerelements

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show, Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndObj

OnEvent, Text, Text, Color, Color, ColorBackground, ColorBackground, ColorBorder, ColorBorder, Font, Font, FontSize, FontSize, ZOrder, ZOrder

Ein Beispiel für die Erstellung eines Panels mit einem Bitmap-Label

```
//+------------------------------------------------------------------+
//|                                            ControlsBmpButton.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Control Panels and Dialogs. Demonstration class CBmpButton"
#include <Controls\Dialog.mqh>
#include <Controls\BmpButton.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allov
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CBmpButton        m_bmpbutton1;                    // CBmpButton object
   CBmpButton        m_bmpbutton2;                    // CBmpButton object

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateBmpButton1(void);
   bool              CreateBmpButton2(void);
   //--- handlers of the dependent controls events
   void              OnClickBmpButton1(void);
   void              OnClickBmpButton2(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CLICK,m_bmpbutton1,OnClickBmpButton1)
ON_EVENT(ON_CLICK,m_bmpbutton2,OnClickBmpButton2)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateBmpButton1())
      return(false);
   if(!CreateBmpButton2())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "BmpButton1" button                                   |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateBmpButton1(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+BUTTON_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_bmpbutton1.Create(m_chart_id,m_name+"BmpButton1",m_subwin,x1,y1,x2,y2))
      return(false);
//--- sets the name of bmp files of the control CBmpButton
   m_bmpbutton1.BmpNames("\\Images\\euro.bmp","\\Images\\dollar.bmp");
   if(!Add(m_bmpbutton1))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "BmpButton2" fixed button                             |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateBmpButton2(void)
  {
//--- coordinates
   int x1=INDENT_LEFT+2*(BUTTON_WIDTH+CONTROLS_GAP_X);
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+BUTTON_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_bmpbutton2.Create(m_chart_id,m_name+"BmpButton2",m_subwin,x1,y1,x2,y2))
      return(false);
//--- sets the name of bmp files of the control CBmpButton
   m_bmpbutton2.BmpNames("\\Images\\euro.bmp","\\Images\\dollar.bmp");
   if(!Add(m_bmpbutton2))
      return(false);
   m_bmpbutton2.Locking(true);
//--- succeed
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnClickBmpButton1(void)
  {
   Comment(__FUNCTION__);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnClickBmpButton2(void)
  {
   if(m_bmpbutton2.Pressed())
      Comment(__FUNCTION__+" State of the control is: On");
   else
      Comment(__FUNCTION__+" State of the control is: Off");
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
void OnChartEvent(const int id,          // event ID
                  const long& lparam,    // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CBmpButton.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate x1
   const int      y1,          // Koordinate y1
   const int      x2,          // Koordinate x2
   const int      y2           // Koordinate y2
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Border (Get-Methode)

Erhält den Wert des Parameters "Border" (Rahmenbreite) des Steuerelements CBmpButton.

```
int  Border()  const
```

Rückgabewert

Wert von "Border"-Parameter.

Border (Set-Methode)

Setzt den Wert des Parameters "Border" (Rahmenbreite) des Steuerelements CBmpButton.

```
bool  Border(
   const int  value      // Wert
   )
```

Parameter

value

[in] Ein neuer Wert des Parameters "Border".

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BmpNames

Setzt die Namen der BMP-Dateien zur Anzeige des Steuerelements CBmpButton.

```
bool  BmpNames(
   const string  off="",      // Dateiname
   const string  on=""        // Dateiname
   )
```

Parameter

off=""

[in] Name der BMP-Datei zur Anzeige des Steuerelements im OFF-Zustand.

on=""

[in] Name der BMP-Datei zur Anzeige des Steuerelements im ON-Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BmpOffName (Get-Methode)

Erhält den Namen der BMP-Datei zur Anzeige eines Steuerelements im OFF-Zustand.

```
string  BmpOffName()  const
```

Rückgabewert

Der Name der BMP-Datei zur Anzeige eines Steuerelements im OFF-Zustand.

BmpOffName (Set-Methode)

Setzt den Namen der BMP-Datei zur Anzeige eines Steuerelements im OFF-Zustand.

```
bool  BmpOffName(
   const string  name      // Dateiname
   )
```

Parameter

name

[in] Name der BMP-Datei zur Anzeige des Steuerelements im OFF-Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BmpOnName (Get-Methode)

Erhält den Namen der BMP-Datei zur Anzeige eines Steuerelements im ON-Zustand.

```
string  BmpOnName()  const
```

Rückgabewert

Der Name der BMP-Datei zur Anzeige eines Steuerelements im ON-Zustand.

BmpOnName (Set-Methode)

Setzt den Namen der BMP-Datei zur Anzeige eines Steuerelements im ON-Zustand.

```
bool  BmpOnName(
   const string  name      // Dateiname
   )
```

Parameter

name

[in] Name der BMP-Datei zur Anzeige des Steuerelements im ON-Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BmpPassiveName (Get-Methode)

Erhält den Namen der BMP-Datei zur Anzeige des Steuerelements CBmpButton im inaktiven Zustand.

```
string  BmpPassiveName()  const
```

Rückgabewert

Der Name der BMP-Datei zur Anzeige eines Steuerelements im inaktiven Zustand.

BmpPassiveName (Set Methode)

Setzt den Namen der BMP-Datei zur Anzeige des Steuerelements CBmpButton im inaktiven Zustand.

```
bool  BmpPassiveName(
   const string  name      // Dateiname
   )
```

Parameter

name

[in] Der Name der BMP-Datei zur Anzeige eines Steuerelements im inaktiven Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BmpActiveName (Get-Methode)

Erhält den Namen der BMP-Datei zur Anzeige des Steuerelements CBmpButton im aktiven Zustand.

```
string  BmpActiveName()  const
```

Rückgabewert

Der Name der BMP-Datei zur Anzeige eines Steuerelements im aktiven Zustand.

Hinweis

Ein Steuerelement ist aktiv, wenn sich der Mauszeiger über dem Steuerelement befindet.

BmpActiveName (Set-Methode)

Setzt den Namen der BMP-Datei zur Anzeige des Steuerelements CBmpButton im aktiven Zustand.

```
bool  BmpActiveName(
   const string  name      // Dateiname
   )
```

Parameter

name

[in] Der Name der BMP-Datei zur Anzeige eines Steuerelements im aktiven Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Pressed (Get-Methode)

Erhält den Zustand (Eigenschaft Pressed) des Steuerelements CBmpButton.

```
bool  Pressed()  const
```

Rückgabewert

Element-Zustand.

Pressed (Set-Methode)

Setzt den Zustand (Eigenschaft Pressed) des Steuerelements CBmpButton.

```
bool  Pressed(
   const bool  pressed      // Zustand
   )
```

Parameter

pressed

[in] Der neue Zustand des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Locking (Get-Methode)

Erhält den Wert der Eigenschaft "Locking" (Fixieren) des Steuerelements CBmpButton.

```
bool  Locking()  const
```

Rückgabewert

Wert der Eigenschaft "Locking".

Locking (Set-Methode)

Setzt die Eigenschaft "Locking" (Fixieren) des Steuerelements CBmpButton.

```
void  Locking(
   const bool  locking      // Wert
   )
```

Parameter

locking

[in] Der neue Wert der Eigenschaft "Locking".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnSetZOrder

Virtueller Handler des Ereignisses "SetZOrder" (Änderung des Parameters OBJPROP_ZORDER) des Steuerelements CBmpButton.

```
virtual bool  OnSetZOrder()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements CBmpButton.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CBmpButton.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CBmpButton.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements CBmpButton.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChange

Virtueller Handler des internen Ereignisses "Change" (Änderung) des Steuerelements CBmpButton.

```
virtual bool  OnChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnActivate

Virtueller Handler des internen Ereignisses "Activate" (Aktivieren) des Steuerelements CBmpButton.

```
virtual bool  OnActivate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnDeactivate

Virtueller Handler des internen Ereignisses "Deactivate" (Deaktivieren) des Steuerelements CBmpButton.

```
virtual bool  OnDeactivate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMouseDown

Virtueller Handler des Maus-Ereignisses "MouseDown" (das Drücken der Maustaste) von Steuerelement CBmpButton.

```
virtual bool  OnMouseDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMouseUp

Virtueller Handler des Maus-Ereignisses "MouseUp" (Maustaste loslassen) von Steuerelement CBmpButton.

```
virtual bool  OnMouseUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CButton

Klasse CButton ist eine Klasse des einfachen Steuerelements basierend auf das grafischen Objekt "Taste".

Beschreibung

Klasse CButton wird für die Erstellung von einfachen Tasten verwendet.

Deklaration

```
class CButton : public CWndObj
```

Kopf

```
#include <Controls\Button.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndObj
CButton

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Eigenschaften	
Pressed	Erhält/Setzt den Wert der Eigenschaft "Pressed".
Locking	Erhält/Setzt den Wert der Eigenschaft "Locking".
Parameteränderungenbehandlung	
OnSetText	Handler des Ereignisses "SetText" eines Steuerelements
OnSetColor	Handler des Ereignisses "SetColor" eines Steuerelements
OnSetColorBackground	Handler des Ereignisses "SetColorBackground" eines Steuerelements
OnSetColorBorder	Handler des Ereignisses "SetColorBorder" eines Steuerelements
OnSetFont	Handler des Ereignisses "SetFont" eines Steuerelements
OnSetFontSize	Handler des Ereignisses "SetFontSize" eines Steuerelements
Behandlung von internen Ereignissen	
OnCreate	Handler des internen Ereignisses "Create" eines Steuerelements
OnShow	Handler des internen Ereignisses "Show" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnMove	Handler des internen Ereignisses "Move" eines Steuerelements
OnResize	Handler des internen Ereignisses "Resize" eines Steuerelements
OnMouseDown	Handler des internen Ereignisses "MouseDown" eines Steuerelements
OnOnMouseUp	Handler des internen Ereignisses "MouseUp" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show, Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndObj

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

OnEvent, Text, Text, Color, Color, ColorBackground, ColorBackground, ColorBorder, ColorBorder, Font, Font, FontSize, FontSize, ZOrder, ZOrder

Ein Beispiel für die Erstellung eines Panels mit einem Button:

```
//+------------------------------------------------------------------+
//|                                                ControlsButton.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CButto
#include <Controls\Dialog.mqh>
#include <Controls\Button.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CButton           m_button1;                     // the button object
   CButton           m_button2;                     // the button object
   CButton           m_button3;                     // the fixed button object
```

© 2000-2025, MetaQuotes Ltd.

```
public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateButton1(void);
   bool              CreateButton2(void);
   bool              CreateButton3(void);
   //--- handlers of the dependent controls events
   void              OnClickButton1(void);
   void              OnClickButton2(void);
   void              OnClickButton3(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CLICK,m_button1,OnClickButton1)
ON_EVENT(ON_CLICK,m_button2,OnClickButton2)
ON_EVENT(ON_CLICK,m_button3,OnClickButton3)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateButton1())
      return(false);
```

© 2000-2025, MetaQuotes Ltd.

```
   if(!CreateButton2())
      return(false);
   if(!CreateButton3())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "Button1" button                                      |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateButton1(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+BUTTON_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_button1.Create(m_chart_id,m_name+"Button1",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!m_button1.Text("Button1"))
      return(false);
   if(!Add(m_button1))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "Button2" button                                      |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateButton2(void)
  {
//--- coordinates
   int x1=INDENT_LEFT+(BUTTON_WIDTH+CONTROLS_GAP_X);
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+BUTTON_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_button2.Create(m_chart_id,m_name+"Button2",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!m_button2.Text("Button2"))
      return(false);
   if(!Add(m_button2))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "Button3" fixed button                                |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
bool CControlsDialog::CreateButton3(void)
  {
//--- coordinates
   int x1=INDENT_LEFT+2*(BUTTON_WIDTH+CONTROLS_GAP_X);
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+BUTTON_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_button3.Create(m_chart_id,m_name+"Button3",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!m_button3.Text("Locked"))
      return(false);
   if(!Add(m_button3))
      return(false);
   m_button3.Locking(true);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnClickButton1(void)
  {
   Comment(__FUNCTION__);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnClickButton2(void)
  {
   Comment(__FUNCTION__);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnClickButton3(void)
  {
   if(m_button3.Pressed())
      Comment(__FUNCTION__+" Status des Elements: On");
   else
      Comment(__FUNCTION__+" Status des Elements: Off");
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CButton.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Pressed (Get-Methode)

Erhält den Wert der Eigenschaft "Pressed" (Zustand) des Steuerelements CButton.

```
bool  Pressed()
```

Rückgabewert

Die Eigenschaft "Pressed" (des Steuerelements CButton.

Pressed (Set-Methode)

Setzt den Wert der Eigenschaft "Pressed" (Zustand) des Steuerelements CButton.

```
bool  Pressed(
   const bool  pressed      // Zustand
   )
```

Parameter

pressed

[in] Der neue Zustand des Steuerelements CButton.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Locking (Get-Methode)

Erhält den Wert der Eigenschaft "Locking" (Fixieren) des Steuerelements CButton.

```
bool  Locking()
```

Rückgabewert

Die Eigenschaft "Locking" (des Steuerelements CButton.

Locking (Set-Methode)

Setzt den Wert der Eigenschaft "Locking" (Fixieren) des Steuerelements CButton.

```
void  Locking(
   const bool  flag      // Wert
   )
```

Parameter

flag

[in] Der neue Wert der Eigenschaft "Locking" des Steuerelements CButton.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnSetText

Virtueller Handler des Ereignisses "SetText" (Änderung der Eigenschaft OBJPROP_TEXT) des Steuerelements CButton.

```
virtual bool  OnSetText()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColor

Virtueller Handler des Ereignisses "SetColor" (Änderung der Eigenschaft OBJPROP_COLOR) des Steuerelements CButton.

```
virtual bool  OnSetColor()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBackground

Virtueller Handler des Ereignisses "SetColorBackground" (Änderung der Eigenschaft OBJPROP_BGCOLOR) des Steuerelements CButton.

```
virtual bool  OnSetColorBackground()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBorder

Virtueller Handler des Ereignisses "SetColorBorder" (Änderung der Eigenschaft OBJPROP_BORDER_COLOR) des Steuerelements CButton.

```
virtual bool  OnSetColorBackground()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetFont

Virtueller Handler des Ereignisses "SetFont" (Änderung des Parameters OBJPROP_FONT) des Steuerelements CButton.

```
virtual bool  OnSetFont()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetFontSize

Virtueller Handler des Ereignisses "SetFontSize" (Änderung des Parameters OBJPROP_FONTSIZE) des Steuerelements CButton.

```
virtual bool  OnSetFontSize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements CButton.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CButton.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CButton.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements CButton.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CButton.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMouseDown

Virtueller Handler des Maus-Ereignisses "MouseDown" (das Drücken der Maustaste) von Steuerelement CButton.

```
virtual bool  OnMouseDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Die Ereignis "MouseDown" erscheint beim Drücken auf die linke Maustaste, wenn sich der Mauszeiger über dem Steuerelement befindet.

© 2000-2025, MetaQuotes Ltd.

OnMouseUp

Virtueller Handler des Maus-Ereignisses "MouseUp" (Maustaste loslassen) von Steuerelement CButton.

```
virtual bool  OnMouseUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Die Ereignis "MouseUp" erscheint beim Loslassen der linken Maustaste, wenn sich der Mauszeiger über dem Steuerelement befindet.

© 2000-2025, MetaQuotes Ltd.

Klasse CEdit

Klasse CEdit ist eine Klasse des einfachen Steuerelements basierend auf das grafischen Objekt "Eingabefeld".

Beschreibung

Klasse CEdit wird für die Erstellung von bearbeitbaren Textfeldern verwendet.

Deklaration

```
class CEdit : public CWndObj
```

Kopf

```
#include <Controls\Edit.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndObj
CEdit

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Eigenschaften	
ReadOnly	Erhält/setzt den Wert der Eigenschaft "ReadOnly" eines Steuerelements
TextAlign	Erhält/setzt den Wert der Eigenschaft "TextAlign" eines Steuerelements
Behandlung von Objekt-Ereignissen	
OnObjectEndEdit	Virtueller Handler des Ereignisses CHARTEVENT_OBJECT_ENDEDIT des Chartobjekts
Parameteränderungenbehandlung	
OnSetText	Handler des Ereignisses "SetText" eines Steuerelements
OnSetColor	Handler des Ereignisses "SetColor" eines Steuerelements
OnSetColorBackground	Handler des Ereignisses "SetColorBackground" eines Steuerelements
OnSetColorBorder	Handler des Ereignisses "SetColorBorder" eines Steuerelements
OnSetFont	Handler des Ereignisses "SetFont" eines Steuerelements
OnSetFontSize	Handler des Ereignisses "SetFontSize" eines Steuerelements
OnSetZOrder	Handler des Ereignisses "SetZOrder" eines Steuerelements
Behandlung von internen Ereignissen	
OnCreate	Handler des internen Ereignisses "Create" eines Steuerelements
OnShow	Handler des internen Ereignisses "Show" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnMove	Handler des internen Ereignisses "Move" eines Steuerelements
OnResize	Handler des internen Ereignisses "Resize" eines Steuerelements
OnChange	Handler des internen Ereignisses "Change" eines Steuerelements
OnClick	Handler des internen Ereignisses "Click" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndObj

Text, Text, Color, Color, ColorBackground, ColorBackground, ColorBorder, ColorBorder, Font, Font, FontSize, FontSize, ZOrder, ZOrder

Ein Beispiel für die Erstellung eines Panels mit einem editierbaren Textfeld:

```
//+------------------------------------------------------------------+
//|                                                 ControlsEdit.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CEdit
#include <Controls\Dialog.mqh>
#include <Controls\Edit.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allov
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
```

© 2000-2025, MetaQuotes Ltd.

```
  {
private:
   CEdit             m_edit;                          // CEdit Objekt

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler

protected:
   //--- create dependent controls
   bool              CreateEdit(void);
  };
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateEdit())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the display field                                         |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateEdit(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP;
   int x2=ClientAreaWidth()-INDENT_RIGHT;
```

© 2000-2025, MetaQuotes Ltd.

```
   int y2=y1+EDIT_HEIGHT;
//--- create
   if(!m_edit.Create(m_chart_id,m_name+"Edit",m_subwin,x1,y1,x2,y2))
      return(false);
//--- erlauben, den Inhalt zu bearbeiten
   if(!m_edit.ReadOnly(false))
      return(false);
   if(!Add(m_edit))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt das Steuerelement CEdit.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ReadOnly (Get-Methode)

Erhält den Wert der Eigenschaft "ReadOnly" (nur lesen) des Steuerelements CEdit.

```
bool  ReadOnly()
```

Rückgabewert

Die Eigenschaft "ReadOnly" des Steuerelements CEdit.

ReadOnly (Set-Methode)

Setyt den Wert der Eigenschaft "ReadOnly" (nur lesen) des Steuerelements CEdit.

```
bool  ReadOnly(
   const bool  flag      // Wert
   )
```

Parameter

flag

[in] Der neue Wert der Eigenschaft "ReadOnly" des Steuerelements CEdit.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

TextAlign (Get-Methode)

Erhält den Wert der Eigenschaft "TextAlign" (Typ der horizontalen Textausrichtung) des Steuerelements CEdit.

```
ENUM_ALIGN_MODE  TextAlign() const
```

Rückgabewert

Ein Wert der Eigenschaft "TextAlign" des Steuerelements CEdit.

TextAlign (Set-Methode)

Setzt den Wert der Eigenschaft "TextAlign" (Typ der horizontalen Textausrichtung) des Steuerelements CEdit.

```
bool  TextAlign(
   ENUM_ALIGN_MODE  align      // Eigenschaftswert
   )
```

Parameter

align

[in] Der neue Wert der Eigenschaft "TextAlign".

Rückgabewert

Gibt bei Erfolg true zurück, false wenn die Eigenschaft nicht geändert werden konnte.

© 2000-2025, MetaQuotes Ltd.

OnObjectEndEdit

Virtueller Handler des Ereignisses CHARTEVENT_OBJECT_ENDEDIT des Chartobjekts.

```
virtual bool  OnObjectEndEdit()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetText

Virtueller Handler des Ereignisses "SetText" (Änderung der Eigenschaft OBJPROP_TEXT) des Steuerelements CEdit.

```
virtual bool  OnSetText()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColor

Virtueller Handler des Ereignisses "SetColor" (Änderung der Eigenschaft OBJPROP_COLOR) des Steuerelements CEdit.

```
virtual bool  OnSetColor()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBackground

Virtueller Handler des Ereignisses "SetColorBackground" (Änderung der Eigenschaft OBJPROP_BGCOLOR) des Steuerelements CEdit.

```
virtual bool  OnSetColorBackground()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBorder

Virtueller Handler des Ereignisses "SetColorBorder" (Änderung der Eigenschaft OBJPROP_BORDER_COLOR) des Steuerelements CEdit.

```
virtual bool  OnSetColorBorder()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetFont

Virtueller Handler des Ereignisses "SetFont" (Änderung der Eigenschaft OBJPROP_FONT) des Steuerelements CEdit.

```
virtual bool  OnSetFont()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetFontSize

Virtueller Handler des Ereignisses "SetFontSize" (Änderung der Eigenschaft OBJPROP_FONTSIZE) des Steuerelements CEdit.

```
virtual bool  OnSetFontSize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetZOrder

Virtueller Handler des Ereignisses "SetZOrder" (Änderung der Eigenschaft OBJPROP_ZORDER) des Steuerelements CEdit.

```
virtual bool  OnSetZOrder()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements CEdit.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CEdit.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CEdit.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements CEdit.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CEdit.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChange

Virtueller Handler des internen Ereignisses "Change" (Änderung) des Steuerelements CEdit.

```
virtual bool  OnChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnClick

Virtueller Handler des internen Ereignisses "Click" (ein Mausklick) vom Steuerelement CEdit.

```
virtual bool  OnClick()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CPanel

Klasse CPanel ist eine Klasse des einfachen Steuerelements basierend auf das grafischen Objekt "Rechteck-Label".

Beschreibung

Klasse CPanel wird für die visuelle Gruppierung von funktionell verwandten homogenen Elementen verwendet.

Deklaration

```
class CPanel : public CWndObj
```

Kopf

```
#include <Controls\Panel.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndObj
CPanel

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Create	Erstellt ein Steuerelement
Parameter des Chartobjekts	
BorderType	Erhält den Wert des Parameters "Border" (Rahmentyp) des Chartobjekts
Behandlung von Objekt-Ereignissen	
OnSetText	Handler des Ereignisses "SetText" eines Steuerelements
OnSetColorBackground	Handler des Ereignisses "SetColorBackground" eines Steuerelements
OnSetColorBorder	Handler des Ereignisses "SetColorBorder" eines Steuerelements
Behandlung von internen Ereignissen	
OnCreate	Handler des internen Ereignisses "Create" eines Steuerelements
OnShow	Handler des internen Ereignisses "Show" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnMove	Handler des internen Ereignisses "Move" eines Steuerelements
OnResize	Handler des internen Ereignisses "Resize" eines Steuerelements
OnChange	Handler des internen Ereignisses "Change" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show, Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndObj

OnEvent, Text, Text, Color, Color, ColorBackground, ColorBackground, ColorBorder, ColorBorder, Font, Font, FontSize, FontSize, ZOrder, ZOrder

Ein Beispiel für die Erstellung eines Panels mit einem Rechteck-Label:

```
//+------------------------------------------------------------------+
//|                                                ControlsPanel.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd.      |
//|                                             https://www.mql5.com |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CPanel
#include <Controls\Dialog.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const

protected:
   //--- create dependent controls
   bool              CreatePanel(void);
  };
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreatePanel())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "CPanel"                                              |
//+------------------------------------------------------------------+
bool CControlsDialog::CreatePanel(void)
  {
//--- coordinates
   int x1=20;
   int y1=20;
   int x2=ExtDialog.Width()/3;
   int y2=ExtDialog.Height()/3;
//--- create
   if(!my_white_border.Create(0,ExtDialog.Name()+"MyWhiteBorder",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!my_white_border.ColorBackground(CONTROLS_DIALOG_COLOR_BG))
      return(false);
   if(!my_white_border.ColorBorder(CONTROLS_DIALOG_COLOR_BORDER_LIGHT))
      return(false);
   if(!ExtDialog.Add(my_white_border))
      return(false);
   my_white_border.Alignment(WND_ALIGN_CLIENT,0,0,0,0);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//---
CPanel   my_white_border;        // object CPanel
bool     pause=true;             // true - Pause
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//---
   EventSetTimer(3);
   pause=true;
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
//+------------------------------------------------------------------+
//|  Timer                                                           |
//+------------------------------------------------------------------+
void OnTimer()
  {
   pause=!pause;
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CPanel.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BorderType (Get-Methode)

Erhält den Wert des Parameters "Border" (Rahmentyp) des Chartobjekts.

```
ENUM_BORDER_TYPE  BorderType()
```

Rückgabewert

Wert von "Border"-Parameter.

BorderType (Set-Methode)

Setzt den Wert des Parameters "Border" (Rahmentyp) des Chartobjekts.

```
bool  BorderType(
   const ENUM_BORDER_TYPE  type      // Wert
   )
```

Parameter

type

[in] Ein neuer Wert des Parameters "Border".

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnSetText

Virtueller Handler des Ereignisses "SetText" (Änderung der Eigenschaft OBJPROP_TEXT) des Steuerelements CPanel.

```
virtual bool  OnSetText()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBackground

Virtueller Handler des Ereignisses "SetColorBackground" (Änderung der Eigenschaft OBJPROP_BGCOLOR) des Steuerelements CPanel.

```
virtual bool  OnSetColorBackground()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnSetColorBorder

Virtueller Handler des Ereignisses "SetColorBorder" (Änderung der Eigenschaft OBJPROP_BORDER_COLOR) des Steuerelements CPanel.

```
virtual bool  OnSetColorBorder()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements CPanel.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CPanel.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CPanel.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements CPanel.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CPanel.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChange

Virtueller Handler des internen Ereignisses "Change" (Änderung) des Steuerelements CPanel.

```
virtual bool  OnChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CPicture

Klasse CPicture ist eine Klasse des einfachen Steuerelements basierend auf das grafischen Objekt "Bitmap Label".

Beschreibung

Klasse CPicture wird für die Erstellung von einfachen Grafiken verwendet.

Deklaration

```
class CPicture : public CWndObj
```

Kopf

```
#include <Controls\Picture.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndObj
CPicture

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Eigenschaften des Chartobjekts	
Border	Erhält/setzt die Rahmenbreite eines Chartobjekts
BmpName	Erhält/setzt den Namen der BMP-Dateien zur Anzeige eines Steuerelements
Behandlung von internen Ereignissen	
OnCreate	Handler des internen Ereignisses "Create" eines Steuerelements
OnShow	Handler des internen Ereignisses "Show" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnMove	Handler des internen Ereignisses "Move" eines Steuerelements
OnChange	Handler des internen Ereignisses "Change" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Save, Load, Type, Compare

Methoden geerbt von der Klasse CWnd

Destroy, OnMouseEvent, Name, ControlsTotal, Control, ControlFind, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Move, Move, Shift, Contains, Contains, Alignment, Align, Id, Id, IsEnabled, Enable, Disable, IsVisible, Visible, Show, Hide, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndObj

OnEvent, Text, Text, Color, Color, ColorBackground, ColorBackground, ColorBorder, ColorBorder, Font, Font, FontSize, FontSize, ZOrder, ZOrder

Ein Beispiel für die Erstellung eines Panels mit einem Bitmap-Label:

```
//+------------------------------------------------------------------+
//|                                                ControlsPicture.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CPictu
#include <Controls\Dialog.mqh>
#include <Controls\Picture.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CPicture          m_picture;                      // CPicture object

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreatePicture(void);
   //--- handlers of the dependent controls events
   void              OnClickPicture(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CLICK,m_picture,OnClickPicture)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreatePicture())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "Picture"                                             |
//+------------------------------------------------------------------+
bool CControlsDialog::CreatePicture(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+32;
   int y2=y1+32;
//--- create
   if(!m_picture.Create(m_chart_id,m_name+"Picture",m_subwin,x1,y1,x2,y2))
      return(false);
//--- benennen wir die bmp-Dateien für die Anzeige des CPicture Steuerelements
   m_picture.BmpName("\\Images\\euro.bmp");

   if(!Add(m_picture))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnClickPicture(void)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
   Comment(__FUNCTION__);
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CPicture.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Border (Get-Methode)

Erhält den Wert der Eigenschaft "Border" (Rahmenbreite) des Steuerelements CPicture.

```
int  Border()  const
```

Rückgabewert

Der Wert der Eigenschaft "Border".

Border (Set-Methode)

Setzt den Wert der Eigenschaft "Border" (Rahmenbreite) des Steuerelements CPicture.

```
bool  Border(
   const int  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert der Eigenschaft "Border".

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BmpName (Get-Methode)

Erhält den Namen der BMP-Dateien zur Anzeige des Steuerelements CPicture.

```
string  BmpName()  const
```

Rückgabewert

Der Name der BMP-Datei zur Anzeige eines Steuerelements.

BmpName (Set-Methode)

Setzt den Namen der BMP-Dateien zur Anzeige des Steuerelements CPicture.

```
bool  BmpName(
   const string  name      // Dateiname
   )
```

Parameter

name

[in] Der Name der BMP-Datei zur Anzeige eines Steuerelements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnCreate

Virtueller Handler des internen Ereignisses "Create" (Erstellen) des Steuerelements CPicture.

```
virtual bool  OnCreate()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CPicture.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CPicture.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnMove

Virtueller Handler des internen Ereignisses "Move" (Bewegen) des Steuerelements CPicture.

```
virtual bool  OnMove()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChange

Virtueller Handler des internen Ereignisses "Change" (Änderung) des Steuerelements CPicture.

```
virtual bool  OnChange()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CScroll

CScroll ist eine Basisklasse für die Erstellung der Bildlaufleisten.

Beschreibung

Klasse CScroll ist ein kombiniertes Steuerelement, welches die grundlegenden Mechanismen für die Erstellung der Bildlaufleisten enthält. Die Basisklasse wird nicht direkt verwendet, nur ihre abgeleitete Klassen CScrollV und CScrollH haben die funktionelle Belastung.

Deklaration

```
class CScroll : public CWndContainer
```

Kopf

```
#include <Controls\Scrolls.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CScroll

Direkte Ableitungen

CScrollH, CScrollV

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Eigenschaften	
MinPos	Erhält/setzt den Wert der minimalen Position
MaxPos	Erhält/setzt den Wert der maximalen Position
CurrPos	Erhält/setzt den Wert der aktuellen Position
Erstellung von unterlegenden Steuerelementen	
CreateBack	Erstellt eine Hintergrundtaste eines Steuerelements
CreateInc	Erstellt eine Taste, die die Position der Bildlaufleiste erhöht

© 2000-2025, MetaQuotes Ltd.

Erstellung	
CreateDec	Erstellt eine Taste, die die Position der Bildlaufleiste reduziert
CreateThumb	Erstellt einen Schieberegler der aktuellen Position der Bildlaufleiste
Behandlung von unterlegenden Steuerelementen	
OnClickInc	Handler des Ereignisses der Erhöhung der Position von Bildlaufleiste
OnClickDec	Handler des Ereignisses der Reduzierung der Position von Bildlaufleiste
Behandlung von internen Ereignissen	
OnShow	Handler des internen Ereignisses "Create" eines Steuerelements
OnHide	Handler des internen Ereignisses "Hide" eines Steuerelements
OnChangePos	Handler des internen Ereignisses "ChangePosition" eines Steuerelements
Ereignisse von Objekt-Bewegung	
OnThumbDragStart	Handler des Ereignisses "ThumbDragStart" eines Steuerelements
OnThumbDragProcess	Handler des Ereignisses "ThumbDragProcess" eines Steuerelements
OnThumbDragEnd	Handler des Ereignisses "ThumbDragEnd" eines Steuerelements
Die Berechnung der Position nach den Koordinaten	
CalcPos	Erhält die Position der Bildlaufleiste nach den Koordinaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide, Save, Load

© 2000-2025, MetaQuotes Ltd.

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CScroll.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

MinPos (Get-Methode)

Erhält den Wert des Parameters "MinPos" (minimale Position) des Steuerelements CScroll.

```
int  MinPos()  const
```

Rückgabewert

Ein Wert des Parameters "MinPos".

MinPos (Set-Methode)

Setzt den Wert des Parameters "MinPos" (minimale Position) des Steuerelements CScroll.

```
void  MinPos(
   const int  value      // Wert
   )
```

Parameter

value

[in] Ein neuer Wert des Parameters "MinPos".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MaxPos (Get-Methode)

Erhält den Wert des Parameters "MaxPos" (maximale Position) des Steuerelements CScroll.

```
int  MaxPos()  const
```

Rückgabewert

Ein Wert des Parameters "MaxPos".

MaxPos (Set-Methode)

Setzt den Wert des Parameters "MaxPos" (maximale Position) des Steuerelements CScroll.

```
void  MaxPos(
   const int  value      // Wert
   )
```

Parameter

value

[in] Ein neuer Wert des Parameters "MaxPos".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CurrPos (Get-Methode)

Erhält den Wert des Parameters "CurrPos" (aktuelle Position) des Steuerelements CScroll.

```
int  CurrPos()  const
```

Rückgabewert

Ein Wert des Parameters "CurrPos".

CurrPos (Set-Methode)

Setzt den Wert des Parameters "CurrPos" (aktuelle Position) des Steuerelements CScroll.

```
void  CurrPos(
   const int  value      // Wert
   )
```

Parameter

value

[in] Ein neuer Wert des Parameters "CurrPos".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CreateBack

Erstellt eine Hintergrundtaste des Steuerelements CScroll.

```
virtual bool  CreateBack()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateInc

Erstellt eine Taste, die die Position der Bildlaufleiste erhöht CScroll.

```
virtual bool  CreateInc()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateDec

Erstellt eine Taste, die die Position der Bildlaufleiste reduziert CScroll.

```
virtual bool  CreateDec()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateThumb

Erstellt einen Schieberegler der aktuellen Position der Bildlaufleiste CScroll.

```
virtual bool  CreateThumb()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickInc

Virtueller Handler des internen Ereignisses "ClickInc" (Mausklick auf den Schieberegler, der Position erhöht) vom Steuerelement CScroll.

```
virtual bool  OnClickInc()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnClickDec

Virtueller Handler des internen Ereignisses "ClickDec" (Mausklick auf den Schieberegler, der Position reduziert) vom Steuerelement CScroll.

```
virtual bool  OnClickDec()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnShow

Virtueller Handler des internen Ereignisses "Show" (Einblenden) des Steuerelements CScroll.

```
virtual bool  OnShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnHide

Virtueller Handler des internen Ereignisses "Hide" (Ausblenden) des Steuerelements CScroll.

```
virtual bool  OnHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChangePos

Virtueller Handler des internen Ereignisses "ChangePos" (Positionsänderung) des Steuerelements CScroll.

```
virtual bool  OnChangePos()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragStart

Virtueller Handler des Ereignisses "ThumbDragStart" (der Anfang der Drag-Operation) des Steuerelements CScroll.

```
virtual bool  OnThumbDragStart()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragStart" tritt am Anfang des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragProcess

Virtueller Handler des Ereignisses "ThumbDragProcess" (Drag-Operation) des Steuerelements CScroll.

```
virtual bool  OnThumbDragProcess(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] Der aktuelle Wert der X-Koordinate des Mauszeigers.

y

[in] Der aktuelle Wert der Y-Koordinate des Mauszeigers.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragProcess" tritt während des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragEnd

Virtueller Handler des Ereignisses "ThumbDragEnd" (das Ende der Drag-Operation) des Steuerelements CScroll.

```
virtual bool  OnThumbDragEnd()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragEnd" tritt am Ende des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

CalcPos

Erhält die Position der Bildlaufleiste des Steuerelements CScroll nach den Koordinaten

```
virtual int  CalcPos(
   const int      coord      // Koordinate
   )
```

Parameter

coord

[in] Koordinate der Position der Bildlaufleiste.

Rückgabewert

Ein Wert der Position der Bildlaufleiste

© 2000-2025, MetaQuotes Ltd.

Klasse CScrollV

CScrollV ist eine Klasse des kombinierten Steuerelements "Senkrechte Scrollleiste".

Beschreibung

Klasse CScrollV wird für die Erstellung von senkrechten Scrollleisten verwendet.

Deklaration

```
class CScrollV : public CScroll
```

Kopf

```
#include <Controls\Scrolls.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndContainer
CScroll
CScrollV

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

© 2000-2025, MetaQuotes Ltd.

Erstellung von unterlegenden Steuerelementen	
CreateInc	Erstellt eine Taste, die die Position der Bildlaufleiste erhöht
CreateDec	Erstellt eine Taste, die die Position der Bildlaufleiste reduziert
CreateThumb	Erstellt einen Schieberegler der aktuellen Position der Bildlaufleiste
Behandlung von internen Ereignissen	
OnResize	Handler des internen Ereignisses "Resize" eines Steuerelements
OnChangePos	Handler des internen Ereignisses "ChangePosition" eines Steuerelements
Ereignisse von Objekt-Bewegung	
OnThumbDragStart	Handler des Ereignisses "ThumbDragStart" eines Steuerelements
OnThumbDragProcess	Handler des Ereignisses "ThumbDragProcess" eines Steuerelements
OnThumbDragEnd	Handler des Ereignisses "ThumbDragEnd" eines Steuerelements
Die Berechnung der Position nach den Koordinaten	
CalcPos	Erhält die Position der Bildlaufleiste nach den Koordinaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide, Save, Load

Methoden geerbt von der Klasse CScroll

Create, OnEvent, MinPos, MinPos, MaxPos, MaxPos, CurrPos, CurrPos

Ein Beispiel für die Erstellung eines Panels mit der waagerechten Scroll-Funktion:

```
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                ControlsScrollV.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CScrol
#include <Controls\Dialog.mqh>
#include <Controls\Scrolls.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CScrollV          m_scroll_v;                     // CScrollV Objekt

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
```

© 2000-2025, MetaQuotes Ltd.

```
   bool              CreateScrollV(void);
   //--- handlers of the dependent controls events
   void              OnScrollInc(void);
   void              OnScrollDec(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_SCROLL_INC,m_scroll_v,OnScrollInc)
ON_EVENT(ON_SCROLL_DEC,m_scroll_v,OnScrollDec)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateScrollV())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the CScrollsV object                                      |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateScrollV(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP;
   int x2=x1+18;
   int y2=y1+LIST_HEIGHT;
//--- create
   if(!m_scroll_v.Create(m_chart_id,m_name+"ScrollV",m_subwin,x1,y1,x2,y2))
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
//--- set up the scrollbar
   m_scroll_v.MinPos(0);
//--- set up the scrollbar
   m_scroll_v.MaxPos(10);
   if(!Add(m_scroll_v))
      return(false);
   Comment("Position der Scrollleiste ",m_scroll_v.CurrPos());
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnScrollInc(void)
  {
   Comment("Position der Scrollleiste ",m_scroll_v.CurrPos());
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnScrollDec(void)
  {
   Comment("Position der Scrollleiste ",m_scroll_v.CurrPos());
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
```

© 2000-2025, MetaQuotes Ltd.

```
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

CreateInc

Erstellt eine Taste, die die Position der Bildlaufleiste erhöht CScrollV.

```
virtual bool  CreateInc()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateDec

Erstellt eine Taste, die die Position der Bildlaufleiste reduziert CScrollV.

```
virtual bool  CreateDec()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateThumb

Erstellt einen Schieberegler der aktuellen Position der Bildlaufleiste CScrollV.

```
virtual bool  CreateThumb()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CScrollV.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChangePos

Virtueller Handler des internen Ereignisses "ChangePos" (Positionsänderung) des Steuerelements CScrollV.

```
virtual bool  OnChangePos()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragStart

Virtueller Handler des Ereignisses "ThumbDragStart" (der Anfang der Drag-Operation) des Steuerelements CScrollV.

```
virtual bool  OnThumbDragStart()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragStart" tritt am Anfang des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragProcess

Virtueller Handler des Ereignisses "ThumbDragProcess" (Drag-Operation) des Steuerelements CScrollV.

```
virtual bool  OnThumbDragProcess(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] Der aktuelle Wert der X-Koordinate des Mauszeigers.

y

[in] Der aktuelle Wert der Y-Koordinate des Mauszeigers.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragProcess" tritt während des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragEnd

Virtueller Handler des Ereignisses "ThumbDragEnd" (das Ende der Drag-Operation) des Steuerelements CScrollV.

```
virtual bool  OnThumbDragEnd()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragEnd" tritt am Ende des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

CalcPos

Erhält die Position der Bildlaufleiste des Steuerelements CScrollV nach den Koordinaten

```
virtual int  CalcPos(
   const int      coord      // Koordinate
   )
```

Parameter

coord

[in] Koordinate der Position der Bildlaufleiste.

Rückgabewert

Ein Wert der Position der Bildlaufleiste

© 2000-2025, MetaQuotes Ltd.

Klasse CScrollH

CScrollH ist eine Klasse des kombinierten Steuerelements "Waagerechte Scrollleiste".

Beschreibung

Klasse CScrollH wird für die Erstellung von waagerechten Scrollleisten verwendet.

Deklaration

```
class CScrollH : public CScroll
```

Kopf

```
#include <Controls\Scrolls.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndContainer
CScroll
CScrollH

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

© 2000-2025, MetaQuotes Ltd.

Erstellung von unterlegenden Steuerelementen	
CreateInc	Erstellt eine Taste, die die Position der Bildlaufleiste erhöht
CreateDec	Erstellt eine Taste, die die Position der Bildlaufleiste reduziert
CreateThumb	Erstellt einen Schieberegler der aktuellen Position der Bildlaufleiste
Behandlung von internen Ereignissen	
OnResize	Handler des internen Ereignisses "Resize" eines Steuerelements
OnChangePos	Handler des internen Ereignisses "ChangePosition" eines Steuerelements
Ereignisse von Objekt-Bewegung	
OnThumbDragStart	Handler des Ereignisses "ThumbDragStart" eines Steuerelements
OnThumbDragProcess	Handler des Ereignisses "ThumbDragProcess" eines Steuerelements
OnThumbDragEnd	Handler des Ereignisses "ThumbDragEnd" eines Steuerelements
Die Berechnung der Position nach den Koordinaten	
CalcPos	Erhält die Position der Bildlaufleiste nach den Koordinaten

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide, Save, Load

Methoden geerbt von der Klasse CScroll

Create, OnEvent, MinPos, MinPos, MaxPos, MaxPos, CurrPos, CurrPos

Ein Beispiel für die Erstellung eines Panels mit der horizontalen Scroll-Funktion:

```
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//|                                                 ControlsScrollH.mq5 |
//|                          Copyright 2000-2024, MetaQuotes Ltd. |
//|                                                 https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Die Demonstration der CS
#include <Controls\Dialog.mqh>
#include <Controls\Scrolls.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CScrollH          m_scroll_v;                     // CScrollH Objekt

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
```

© 2000-2025, MetaQuotes Ltd.

```
   bool              CreateScrollsH(void);
   //--- handlers of the dependent controls events
   void              OnScrollInc(void);
   void              OnScrollDec(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_SCROLL_INC,m_scroll_v,OnScrollInc)
ON_EVENT(ON_SCROLL_DEC,m_scroll_v,OnScrollDec)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateScrollsH())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the CScrollsH object                                      |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateScrollsH(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP;
   int x2=x1+3*BUTTON_WIDTH;
   int y2=y1+18;
//--- create
   if(!m_scroll_v.Create(m_chart_id,m_name+"ScrollsH",m_subwin,x1,y1,x2,y2))
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
//--- set up the scrollbar
   m_scroll_v.MinPos(0);
//--- set up the scrollbar
   m_scroll_v.MaxPos(10);
   if(!Add(m_scroll_v))
      return(false);
   Comment("Position der Scrollleiste ",m_scroll_v.CurrPos());
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnScrollInc(void)
  {
   Comment("Position der Scrollleiste ",m_scroll_v.CurrPos());
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnScrollDec(void)
  {
   Comment("Position der Scrollleiste ",m_scroll_v.CurrPos());
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
```

© 2000-2025, MetaQuotes Ltd.

```
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

CreateInc

Erstellt eine Taste, die die Position der Bildlaufleiste erhöht CScrollH.

```
virtual bool  CreateInc()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateDec

Erstellt eine Taste, die die Position der Bildlaufleiste reduziert CScrollH.

```
virtual bool  CreateDec()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateThumb

Erstellt einen Schieberegler der aktuellen Position der Bildlaufleiste CScrollH.

```
virtual bool  CreateThumb()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CScrollH.

```
virtual bool  OnResize()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChangePos

Virtueller Handler des internen Ereignisses "ChangePos" (Positionsänderung) des Steuerelements CScrollH.

```
virtual bool  OnChangePos()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragStart

Virtueller Handler des Ereignisses "ThumbDragStart" (der Anfang der Drag-Operation) des Steuerelements CScrollH.

```
virtual bool  OnThumbDragStart()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragStart" tritt am Anfang des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragProcess

Virtueller Handler des Ereignisses "ThumbDragProcess" (Drag-Operation) des Steuerelements CScrollH.

```
virtual bool  OnThumbDragProcess(
   const int  x,      // X-Koordinate
   const int  y       // Y-Koordinate
   )
```

Parameter

x

[in] Der aktuelle Wert der X-Koordinate des Mauszeigers.

y

[in] Der aktuelle Wert der Y-Koordinate des Mauszeigers.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragProcess" tritt während des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

OnThumbDragEnd

Virtueller Handler des Ereignisses "ThumbDragEnd" (das Ende der Drag-Operation) des Steuerelements CScrollH.

```
virtual bool  OnThumbDragEnd()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "ThumbDragEnd" tritt am Ende des Ziehens des Steuerelements auf.

© 2000-2025, MetaQuotes Ltd.

CalcPos

Erhält die Position der Bildlaufleiste des Steuerelements CScrollH nach den Koordinaten

```
virtual int  CalcPos(
   const int      coord      // Koordinate
   )
```

Parameter

coord

[in] Koordinate der Position der Bildlaufleiste.

Rückgabewert

Ein Wert der Position der Bildlaufleiste

© 2000-2025, MetaQuotes Ltd.

Klasse CWndClient

Klasse CWndClient ist ein kombiniertes Steuerelement "Kundenbereich" und ist eine Basisklasse für die Erstellung von Bereichen mit Bildlaufleisten.

Beschreibung

Klasse CWndClient ist eine Software-Implementierung der Funktionen für die Erstellung von Bereichen mit Bildlaufleisten.

Deklaration

```
class CWndClient : public CWndContainer
```

Kopf

```
#include <Controls\WndClient.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CWndClient

Direkte Ableitungen

CCheckGroup, CListView, CRadioGroup

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Parameter	
ColorBackground	Setzt die Hintergrundfarbe eines Elements.
ColorBorder	Setzt die Rahmenfarbe eines Elements.
BorderType	Setzt den Rahmentyp eines Elements.
Einstellungen	
VScrolled	Erhält/setzt das Zeichen der Nutzung von der vertikalen Bildlaufleiste
HScrolled	Erhält/setzt das Zeichen der Nutzung von der horizontale Bildlaufleiste

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Unterlegende Steuerelemente	
CreateBack	Erstellt eine Hintergrundtaste der Bildlaufleiste
CreateScrollV	Erstellt eine vertikale Bildlaufleiste
CreateScrollH	Erstellt eine horizontale Bildlaufleiste
Behandlung von internen Ereignissen	
OnResize	Virtueller Handler des internen Ereignisses "Resize" eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnVScrollShow	Virtueller Handler des internen Ereignisses "Show" des unterlegenden Steuerelements VScroll
OnVScrollHide	Virtueller Handler des internen Ereignisses "Hide" des unterlegenden Steuerelements VScroll
OnHScrollShow	Virtueller Handler des internen Ereignisses "Show" des unterlegenden Steuerelements HScroll
OnHScrollHide	Virtueller Handler des internen Ereignisses "Hide" des unterlegenden Steuerelements HScroll
OnScrollLineDown	Virtueller Handler des internen Ereignisses "ScrollLineDown" des unterlegenden Steuerelements VScroll
OnScrollLineUp	Virtueller Handler des internen Ereignisses "ScrollLineUp" des unterlegenden Steuerelements VScroll
OnScrollLineLeft	Virtueller Handler des internen Ereignisses "ScrollLineLeft" des unterlegenden Steuerelements HScroll
OnScrollLineRight	Virtueller Handler des internen Ereignisses "ScrollLineRight" des unterlegenden Steuerelements HScroll
Größenänderung	
Rebound	Setzt die neuen Parameter des Bereichs des Steuerelements aus den Koordinaten der Klasse CRect

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Enable, Disable, Hide, Save, Load

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CWndClient.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

ColorBackground

Setzt die Hintergrundfarbe eines Steuerelements.

```
bool  ColorBackground(
   const color  value      // Hintergrundfarbe
   )
```

Parameter

value

[in] Hintergrundfarbe des Steuerelements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ColorBorder

Setzt die Rahmenfarbe eines Steuerelements.

```
bool  ColorBorder(
   const color  value      // Farbe
   )
```

Parameter

value

[in] Rahmenfarbe des Steuerelements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

BorderType

Setzt den Rahmentyp eines Steuerelements.

```
bool  BorderType(
   const ENUM_BORDER_TYPE  type      // Wert
   )
```

Parameter

type

[in] Rahmentyp des Steuerelements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

VScrolled (Get-Methode)

Erhält das Zeichen der Nutzung von der vertikalen Bildlaufleiste.

```
bool  VScrolled()
```

Rückgabewert

true - wenn die vertikale Bildlaufleiste verwendet wird, ansonsten false.

VScrolled (Set-Methode)

Setzt das Zeichen der Nutzung von der vertikalen Bildlaufleiste.

```
bool  VScrolled(
   const bool  flag      // Flag
   )
```

Parameter

flag

[in] Flag.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

HScrolled (Get-Methode)

Erhält das Zeichen der Nutzung von der horizontale Bildlaufleiste.

```
bool  HScrolled()
```

Rückgabewert

true - wenn die horizontale Bildlaufleiste verwendet wird, ansonsten false.

HScrolled (Set-Methode)

Setzt das Zeichen der Nutzung von der horizontale Bildlaufleiste

```
bool  HScrolled(
   const bool  flag      // Flag
   )
```

Parameter

flag

[in] Flag.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateBack

Erstellt eine Hintergrundtaste der Bildlaufleiste.

```
virtual bool  CreateBack()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateScrollV

Erstellt eine vertikale Bildlaufleiste.

```
virtual bool  CreateScrollV()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateScrollH

Erstellt eine horizontale Bildlaufleiste.

```
virtual bool  CreateScrollH()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CWinClient.

```
virtual bool  OnResize()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnVScrollShow

Virtueller Handler des internen Ereignisses "Show" (einblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnVScrollHide

Virtueller Handler des internen Ereignisses "Hide" (ausblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnHScrollShow

Virtueller Handler des internen Ereignisses "Show" (einblenden) des unterlegenden Steuerelements HScroll (horizontale Bildlaufleiste).

```
virtual bool  OnHScrollShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnHScrollHide

Virtueller Handler des internen Ereignisses "Hide" (ausblenden) des unterlegenden Steuerelements HScroll (horizontale Bildlaufleiste).

```
virtual bool  OnHScrollHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineDown

Virtueller Handler des internen Ereignisses "ScrollLineDown" (scrollen eine Zeile nach unten) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineUp

Virtueller Handler des internen Ereignisses "ScrollLineUp" (scrollen eine Zeile nach oben) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineLeft

Virtueller Handler des internen Ereignisses "ScrollLineLeft" (nach links scrollen) des unterlegenden Steuerelements HScroll (horizontale Bildlaufleiste).

```
virtual bool  OnScrollLineLeft()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineRight

Virtueller Handler des internen Ereignisses "ScrollLineRight" (nach rechts scrollen) des unterlegenden Steuerelements HScroll (horizontale Bildlaufleiste).

```
virtual bool  OnScrollLineRight()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Basisklassenmethode tut nichts und gibt immer true zurück.

© 2000-2025, MetaQuotes Ltd.

ReBound

Setzt die neuen Parameter des Bereichs des Steuerelements CWndClient aus den Koordinaten der Klasse CRect.

```
void  ReBound(
   const & CRect  rect      // Klasse der Rechteckfläche
   )
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Klasse CListView

CListView ist eine Klasse des kombinierten Steuerelements "Liste einblenden".

Beschreibung

Die CListView Klasse ermöglicht Anzeige einer Liste mit Datenelementen.

Deklaration

```
class CListView : public CWndClient
```

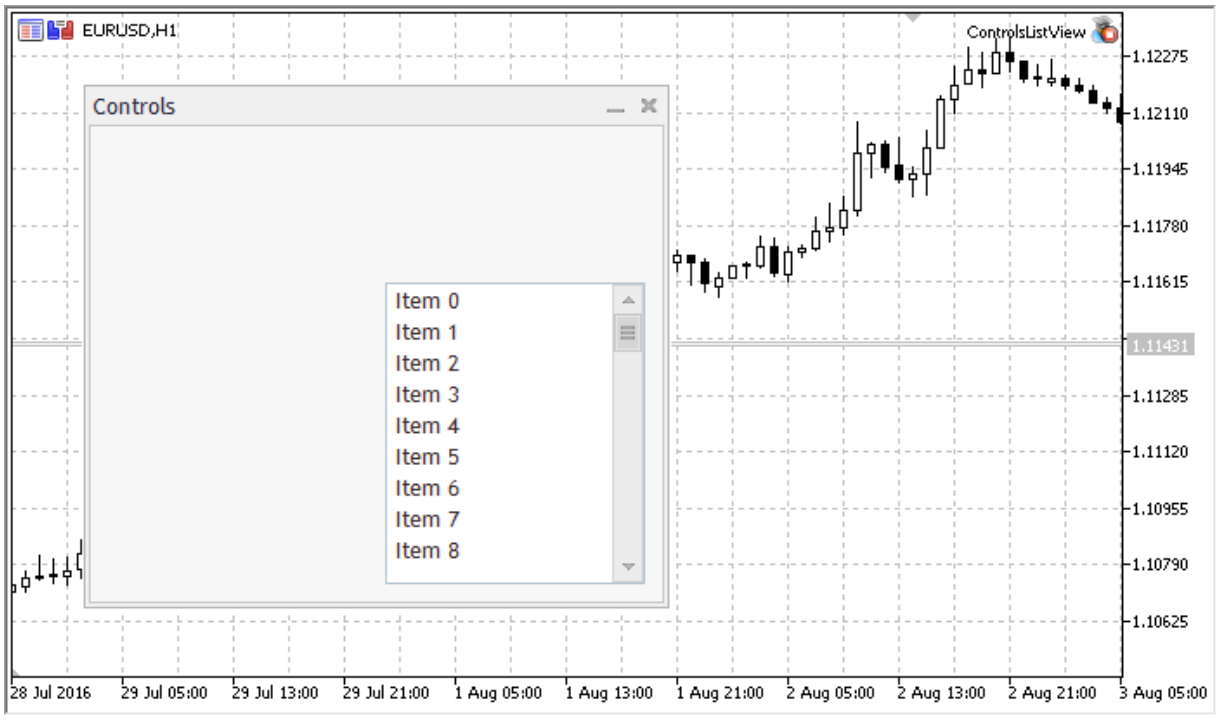
Kopf

```
#include <Controls\ListView.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndContainer
CWndClient
CListView

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Einstellung	
TotalView	Setzt einen Parameter, der die Anzahl der angezeigten Elemente bestimmt
Füllung	
AddItem	Fügt ein Element (Zeile) in die Liste eines Steuerelements hinzu
Daten	
Select	Setzt das aktuelle Listenelement nach dem Index
SelectByText	Setzt das aktuelle Listenelement nach dem Text
SelectByValue	Setzt das aktuelle Listenelement nach dem Wert
Daten (nur lesen)	
Value	Erhält den Wert des aktuellen Listenelements
Unterlegende Steuerelemente	
CreateRow	Erstellt eine "Zeile" des Elements in der Liste
Behandlung von internen Ereignissen	
OnResize	Virtueller Handler des internen Ereignisses "Resize" eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnVScrollShow	Virtueller Handler des internen Ereignisses "Show" des unterlegenden Steuerelements VScroll
OnVScrollHide	Virtueller Handler des internen Ereignisses "Hide" des unterlegenden Steuerelements VScroll
OnScrollLineDown	Virtueller Handler des internen Ereignisses "ScrollLineDown" des unterlegenden Steuerelements VScroll
OnScrollLineUp	Virtueller Handler des internen Ereignisses "ScrollLineUp" des unterlegenden Steuerelements VScroll
OnItemClick	Virtueller Handler des internen Ereignisses "ItemClick" in der angegebenen Zeile eines Steuerelements

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Neuzeichnung	
Redraw	Neuzeichnung eines Steuerelements
RowState	Änderung des Zustands der angegebenen Zeile des Steuerelements
CheckView	Überprüfung der "Sichtbarkeit" der ausgewählten Zeile des Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Enable, Disable, Hide, Save, Load

Methoden geerbt von der Klasse CWndClient

ColorBackground, ColorBorder, BorderType, VScrolled, VScrolled, HScrolled, HScrolled, Id

Ein Beispiel für die Erstellung eines Panels mit einer Liste von Werten:

```
//+------------------------------------------------------------------+
//|                                                ControlsListView.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CListV
#include <Controls\Dialog.mqh>
#include <Controls\ListView.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
```

© 2000-2025, MetaQuotes Ltd.

```
#define CONTROLS_GAP_X                    (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                    (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                      (100)     // size by X coordinate
#define BUTTON_HEIGHT                     (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                       (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                       (150)     // size by X coordinate
#define LIST_HEIGHT                       (179)     // size by Y coordinate
#define RADIO_HEIGHT                      (56)      // size by Y coordinate
#define CHECK_HEIGHT                      (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CListView         m_list_view;                    // CListView Objekt

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateListView(void);
   //--- handlers of the dependent controls events
   void              OnChangeListView(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CHANGE,m_list_view,OnChangeListView)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateListView())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "ListView" element                                    |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateListView(void)
  {
//--- coordinates
   int x1=INDENT_LEFT+GROUP_WIDTH+2*CONTROLS_GAP_X;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (BUTTON_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+2*CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+LIST_HEIGHT-CONTROLS_GAP_Y;
//--- create
   if(!m_list_view.Create(m_chart_id,m_name+"ListView",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!Add(m_list_view))
      return(false);
//--- fill out with strings
   for(int i=0;i<16;i++)
      if(!m_list_view.AddItem("Item "+IntegerToString(i)))
         return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeListView(void)
  {
   Comment(__FUNCTION__+" \""+m_list_view.Select()+"\"");
  }
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CListView.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

TotalView

Setzt die Anzahl der sichtbaren Elemente in der Liste des Steuerelements CListView.

```
bool  TotalView(
   const int  value      // Anzahl der angezeigten Elemente
   )
```

Parameter

value

[in] Anzahl der sichtbaren Elemente in der Liste.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Die Anzahl der angezeigten Elemente kann nur einmal eingegeben werden.

© 2000-2025, MetaQuotes Ltd.

AddItem

Fügt ein Element (Zeile) in die Liste des Steuerelements CListView hinzu.

```
bool  AddItem(
   const string  item,      // Text
   const long    value      // Wert
   )
```

Parameter

item

[in] Text.

value

[in] Wert.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Select

Setzt das aktuelle Listenelement nach dem Index.

```
bool  Select(
   const int  index     // Index
   )
```

Parameter

index

[in] Der Index des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

SelectByText

Setzt das aktuelle Listenelement nach dem Text.

```
bool  SelectByText(
   const string  text     // Text
   )
```

Parameter

text

[in] Der Text des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

SelectByValue

Setzt das aktuelle Listenelement nach dem Wert.

```
bool  SelectByValue(
   const long  value     // Wert
   )
```

Parameter

value

[in] Wert.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Value

Erhält den Wert des aktuellen Listenelements.

```
long  Value()
```

Rückgabewert

Der Wert des aktuellen Listenelements.

© 2000-2025, MetaQuotes Ltd.

CreateRow

Erstellt eine "Zeile" des Elements in der Liste.

```
bool  CreateRow(
   const int  index     // Index
   )
```

Parameter

index

[in] Der Index des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnResize

Virtueller Handler des internen Ereignisses "Resize" (Größenänderung) des Steuerelements CListView.

```
virtual bool  OnResize()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnVScrollShow

Virtueller Handler des internen Ereignisses "Show" (einblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnVScrollHide

Virtueller Handler des internen Ereignisses "Hide" (ausblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineDown

Virtueller Handler des internen Ereignisses "ScrollLineDown" (scrollen eine Zeile nach unten) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineUp

Virtueller Handler des internen Ereignisses "ScrollLineUp" (scrollen eine Zeile nach oben) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnItemClick

Virtueller Handler des internen Ereignisses "ItemClick" (Mausklick) in der angegebenen Zeile des Steuerelements CListView.

```
virtual bool  OnItemClick()
   const int    index     // Index
   )
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Redraw

Neuzeichnung des Steuerelements CListView.

```
bool  Redraw()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

RowState

Änderung des Zustands der angegebenen Zeile des Steuerelements CListView.

```
bool  RowState(
   const int   index      // Index
   const bool  select     // Zustand
   )
```

Parameter

index

[in] Der Index der Zeile.

select

[in] Der Zustand der Zeile.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CheckView

Überprüfung der "Sichtbarkeit" der ausgewählten Zeile des Steuerelements CListView.

```
bool  CheckView()
```

Rückgabewert

true - wenn die ausgewählte Zeile eingeblendet ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CComboBox

CComboBox ist eine Klasse des kombinierten Steuerelements "Feld mit einer Dropdown-Liste".

Beschreibung

Klasse CComboBox ist ein Steuerelement mit einer Dropdown-Liste für die Auswahl eines Werts.

Deklaration

```
class CComboBox : public CWndContainer
```

Kopf

```
#include <Controls\ComboBox.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CComboBox

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Füllung	
AddItem	Fügt ein Element (Zeile) in die Liste eines Steuerelements hinzu
Einstellung	
ListViewItems	Setzt die Anzahl der Elemente in der Dropdown-Liste des Steuerelements
Daten	
Select	Setzt das aktuelle Listenelement nach dem Index
SelectByText	Setzt das aktuelle Listenelement nach dem Text
SelectByValue	Setzt das aktuelle Listenelement nach dem Wert
Daten (nur lesen)	
Value	Erhält den Wert des aktuellen Listenelements
Unterlegende Steuerelemente	
CreateEdit	Erstellt ein unterlegendes Steuerelement (Eingabefeld) eines Steuerelements
CreateButton	Erstellt ein unterlegendes Steuerelement (Taste) eines Steuerelements
CreateList	Erstellt ein unterlegendes Steuerelement (Dropdown-Liste) eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnClickEdit	Virtueller Handler des internen Ereignisses "ClickEdit" eines Steuerelements
OnClickButton	Virtueller Handler des internen Ereignisses "ClickButton" eines Steuerelements
OnChangeList	Virtueller Handler des internen Ereignisses "ChangeList" eines Steuerelements
Dropdown-Liste einblenden	
ListShow	Einblendet die Dropdown-Liste eines Steuerelements
ListHide	Ausblendet die Dropdown-Liste eines Steuerelements

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Hide

Ein Beispiel für die Erstellung eines Panel mit dem Steuerelement "Feld mit einer Dropdown-Liste":

```
//+------------------------------------------------------------------+
//|                                              ControlsComboBox.mq5 |
//|                        Copyright 2015, MetaQuotes Software Corp. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2015, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Control Panels and Dialogs. Demonstration class CComboBox"
#include <Controls\Dialog.mqh>
#include <Controls\ComboBox.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allov
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CComboBox         m_combo_box;                     // CComboBox object

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateComboBox(void);
   //--- handlers of the dependent controls events
   void              OnChangeComboBox(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CHANGE,m_combo_box,OnChangeComboBox)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateComboBox())
```

© 2000-2025, MetaQuotes Ltd.

```
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "ComboBox" element                                    |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateComboBox(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (BUTTON_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+EDIT_HEIGHT;
//--- create
   if(!m_combo_box.Create(m_chart_id,m_name+"ComboBox",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!Add(m_combo_box))
      return(false);
//--- fill out with strings
   for(int i=0;i<16;i++)
      if(!m_combo_box.ItemAdd("Item "+IntegerToString(i)))
         return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeComboBox(void)
  {
   Comment(__FUNCTION__+" \""+m_combo_box.Select()+"\"");
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
```

© 2000-2025, MetaQuotes Ltd.

```
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,          // event ID
                  const long& lparam,    // event parameter of the long type
                  const double& dparam,  // event parameter of the double type
                  const string& sparam)  // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CComboBox.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

AddItem

Fügt ein Element (Zeile) in die Liste des Steuerelements CComboBox hinzu.

```
bool  AddItem(
   const string  item,      // Text
   const long    value      // Wert
   )
```

Parameter

item

[in] Text.

value=0

[in] Wert.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ListViewItems

Setzt die Anzahl der Elemente in der Dropdown-Liste des Steuerelements CComboBox.

```
void  ListViewItems(
   const int    value     // Anzahl der Elemente
   )
```

Parameter

value

[in] Anzahl der Elemente in der Dropdown-Liste.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Select

Setzt das aktuelle Listenelement nach dem Index.

```
bool  Select(
   const int  index     // Index
   )
```

Parameter

index

[in] Der Index des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

SelectByText

Setzt das aktuelle Listenelement nach dem angegebenen Text.

```
bool  SelectByText(
   const string  text     // Text
   )
```

Parameter

text

[in] Der Text des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

SelectByValue

Setzt das aktuelle Listenelement nach dem angegebenen Wert.

```
bool  SelectByValue(
   const long  value     // Wert
   )
```

Parameter

value

[in] Wert.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Value

Erhält den Wert des aktuellen Listenelements.

```
long  Value()
```

Rückgabewert

Der Wert des aktuellen Listenelements.

© 2000-2025, MetaQuotes Ltd.

CreateEdit

Erstellt ein unterlegendes Steuerelement (Eingabefeld) des Steuerelements CComboBox.

```
virtual bool  CreateEdit()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateButton

Erstellt ein unterlegendes Steuerelement (Taste) des Steuerelements CComboBox.

```
virtual bool  CreateButton()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateList

Erstellt ein unterlegendes Steuerelement (Dropdown-Liste) des Steuerelements CComboBox.

```
virtual bool  CreateList()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickEdit

Virtueller Handler des internen Ereignisses "ClickEdit" (Mausklick auf das Eingabefeld) von CComboBox.

```
virtual bool  OnClickEdit()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnClickButton

Virtueller Handler des internen Ereignisses "ClickButton" (Mausklick auf die Taste) von CComboBox.

```
virtual bool  OnClickButton()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChangeList

Virtueller Handler des internen Ereignisses "ChangeList" (Änderung der Dropdown-Liste) des Steuerelements CComboBox.

```
virtual bool  OnChangeList()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

ListShow

Einblendet die Dropdown-Liste des Steuerelements CComboBox.

```
virtual bool  ListShow()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ListHide

Ausblendet die Dropdown-Liste des Steuerelements CComboBox.

```
virtual bool  ListHide()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Klasse CCheckBox

CCheckBox ist eine Klasse des kombinierten Steuerelements "Schalter mit Fixierung".

Beschreibung

Klasse CCheckBox wird für Erstellung eines Steuerelements, welches Zustand auf den Gegenzustand mit dem Mausklick ändert, verwendet.

Deklaration

```
class CCheckBox : public CWndContainer
```

Kopf

```
#include <Controls\CheckBox.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CCheckBox

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Einstellungen	
Text	Erhält/setzt den Erläuterungstext eines Steuerelements
Color	Erhält/setzt die Farbe des Erläuterungstextes eines Steuerelements
Zustand	
Checked	Erhält/setzt den Zustand eines Steuerelements
Daten	
Value	Erhält/setzt den mit dem Steuerelement verbundenen Wert
Unterlegende Steuerelemente	
CreateButton	Erstellt ein unterlegendes Steuerelement (Taste) eines Steuerelements
CreateLabel	Erstellt ein unterlegendes Steuerelement (Erläuterung-Label) eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
ClickButton	Virtueller Handler des internen Ereignisses "ClickButton" eines Steuerelements
ClickLabel	Virtueller Handler des internen Ereignisses "ClickLabel" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide

© 2000-2025, MetaQuotes Ltd.

Ein Beispiel für die Erstellung eines Panels mit dem Steuerelement "Umschalter mit Fixierung":

```
//+------------------------------------------------------------------+
//|                                                ControlsCheckBox.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                                https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Control Panels and Dialogs. Demonstration class CCheckBox"
#include <Controls\Dialog.mqh>
#include <Controls\CheckBox.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CCheckBox         m_check_box1;                    // CCheckBox object
   CCheckBox         m_check_box2;                    // CCheckBox object
public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons
```

© 2000-2025, MetaQuotes Ltd.

```
protected:
   //--- create dependent controls
   bool              CreateCheckBox1(void);
   bool              CreateCheckBox2(void);
   //--- handlers of the dependent controls events
   void              OnChangeCheckBox1(void);
   void              OnChangeCheckBox2(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CHANGE,m_check_box1,OnChangeCheckBox1)
ON_EVENT(ON_CHANGE,m_check_box2,OnChangeCheckBox2)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateCheckBox1())
      return(false);
   if(!CreateCheckBox2())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "CheckBox" element                                    |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateCheckBox1(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
```

© 2000-2025, MetaQuotes Ltd.

```
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (BUTTON_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (RADIO_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_check_box1.Create(m_chart_id,m_name+"CheckBox1",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!m_check_box1.Text("CheckBox1"))
      return(false);
   if(!m_check_box1.Color(clrBlue))
      return(false);
   if(!Add(m_check_box1))
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "CheckBox" element                                    |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateCheckBox2(void)
  {
//--- coordinates
   int x1=INDENT_LEFT+GROUP_WIDTH+CONTROLS_GAP_X;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (BUTTON_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (RADIO_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+BUTTON_HEIGHT;
//--- create
   if(!m_check_box2.Create(m_chart_id,m_name+"CheckBox2",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!m_check_box2.Text("CheckBox2"))
      return(false);
   if(!m_check_box2.Color(clrBlue))
      return(false);
   if(!Add(m_check_box2))
      return(false);
   m_check_box2.Checked(true);
   Comment(__FUNCTION__+" : Checked="+IntegerToString(m_check_box2.Checked()));
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeCheckBox1(void)
  {
   Comment(__FUNCTION__+" : Checked="+IntegerToString(m_check_box1.Checked()));
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeCheckBox2(void)
  {
   Comment(__FUNCTION__+" : Checked="+IntegerToString(m_check_box2.Checked()));
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CCheckBox.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Text (Get-Methode)

Erhält den Erläuterungstext des Steuerelements CCheckBox.

```
string  Text()
```

Rückgabewert

Erläuterungstext.

Text (Set-Methode)

Setzt den Erläuterungstext des Steuerelements CCheckBox.

```
bool  Text(
   const string  value      // Wert
   )
```

Parameter

value

[in] Der neue Erläuterungstext.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Ein Erläuterungstext wird durch die Eingabe des Werts der Eigenschaft OBJPROP_TEXT (Text) des Chartobjekts eingegeben.

© 2000-2025, MetaQuotes Ltd.

Color (Get-Methode)

Erhält die Farbe des Erläuterungstextes des Steuerelements CCheckBox.

```
color  Color()  const
```

Rückgabewert

Die Farbe des Erläuterungstextes.

Color (Set-Methode)

Setzt die Farbe des Erläuterungstextes des Steuerelements CCheckBox.

```
bool  Color(
   const color  value      // Farbe
   )
```

Parameter

value

[in] Die neue Farbe Farbe des Erläuterungstextes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Hinweis

Die Farbe des Erläuterungstextes wird durch Werteingabe der Eigenschaft OBJPROP_COLOR (Farbe) des Chartobjekts eingegeben.

© 2000-2025, MetaQuotes Ltd.

Checked (Get-Methode)

Erhält den Zustand des Steuerelements CCheckBox.

```
bool  Checked()  const
```

Rückgabewert

Der Zustand des Steuerelements CCheckBox.

Checked (Set-Methode)

Setzt den Zustand des Steuerelements CCheckBox.

```
bool  Checked(
   const bool  flag      // Zustand
   )
```

Parameter

flag

[in] Der neue Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Value (Get-Methode)

Erhält den mit dem Steuerelement CCheckBox verbundenen Wert.

```
int  Value()  const
```

Rückgabewert

Der Wert, der mit dem Steuerelement CCheckBox verbunden ist.

Value (Set-Methode)

Setzt den mit dem Steuerelement CCheckBox verbundenen Wert.

```
void  Value(
   const int  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CreateButton

Erstellt ein unterlegendes Steuerelement (Taste) des Steuerelements CCheckBox.

```
virtual bool  CreateButton()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateLabel

Erstellt ein unterlegendes Steuerelement (Erläuterung-Label) des Steuerelements CCheckBox.

```
virtual bool  CreateLabel()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickButton

Virtueller Handler des internen Ereignisses "ClickButton" (Mausklick auf die Taste) von CCheckBox.

```
virtual bool  OnClickButton()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnClickLabel

Virtueller Handler des internen Ereignisses "ClickLabel" (Mausklick auf den Label) von CCheckBox.

```
virtual bool  OnClickLabel()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CCheckGroup

CCheckGroup ist eine Klasse des kombinierten Steuerelements "Schalter mit einer unabhängigen Fixierung".

Beschreibung

Klasse CCheckGroup wird für Erstellung eines Steuerelements, der einen Satz von Flags anzeigen und bearbeiten erlaubt, verwendet.

Deklaration

```
class CCheckGroup : public CWndClient
```

Kopf

```
#include <Controls\CheckGroup.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CWndClient
        CCheckGroup

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Füllung	
AddItem	Fügt ein neues Element in der Gruppe hinzu
Daten (nur lesen)	
Value	Erhält den mit dem Steuerelement verbundenen Wert
Unterlegende Steuerelemente	
CreateButton	Erstellt ein neues Element CCheckBox in der Gruppe am angegebenen Index
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnVScrollShow	Virtueller Handler des internen Ereignisses "Show" des unterlegenden Steuerelements VScroll.
OnVScrollHide	Virtueller Handler des internen Ereignisses "Hide" des unterlegenden Steuerelements VScroll.
OnScrollLineDown	Virtueller Handler des internen Ereignisses "ScrollLineUp" des unterlegenden Steuerelements VScroll.
OnScrollLineUp	Virtueller Handler des internen Ereignisses "ScrollLineDown" des unterlegenden Steuerelements VScroll.
OnChangeItem	Virtueller Handler des internen Ereignisses "ChangeItem" eines Steuerelements
Neuzeichnung	
Redraw	Zeichnet eine Elementgruppe neu
RowState	Ändert den Zustand eines Elements der Gruppe

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWndContainer

OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Enable, Disable, Hide

Methoden geerbt von der Klasse CWndClient

ColorBackground, ColorBorder, BorderType, VScrolled, VScrolled, HScrolled, HScrolled, Id

Ein Beispiel für die Erstellung eines Panels mit dem Steuerelement "Umschalter mit einer unabhängigen Fixierung":

```
//+------------------------------------------------------------------+
//|                                            ControlsCheckGroup.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Control Panels and Dialogs. Demonstration class CCheckGroup"
#include <Controls\Dialog.mqh>
#include <Controls\CheckGroup.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
```

© 2000-2025, MetaQuotes Ltd.

```
class CControlsDialog : public CAppDialog
  {
private:
   CCheckGroup       m_check_group;                    // CCheckGroup object

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateCheckGroup(void);
   //--- handlers of the dependent controls events
   void              OnChangeCheckGroup(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CHANGE,m_check_group,OnChangeCheckGroup)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateCheckGroup())
      return(false);
//--- succeed
   return(true);
```

© 2000-2025, MetaQuotes Ltd.

```
  }
//+------------------------------------------------------------------+
//| Create the "CheckGroup" element                                  |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateCheckGroup(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (BUTTON_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (RADIO_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+CHECK_HEIGHT;
//--- create
   if(!m_check_group.Create(m_chart_id,m_name+"CheckGroup",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!Add(m_check_group))
      return(false);
//--- fill out with strings
   for(int i=0;i<5;i++)
      if(!m_check_group.AddItem("Item "+IntegerToString(i),1<<i))
         return(false);
   m_check_group.Check(0,1<<0);
   m_check_group.Check(2,1<<2);
   Comment(__FUNCTION__+" : Value="+IntegerToString(m_check_group.Value()));
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeCheckGroup(void)
  {
   Comment(__FUNCTION__+" : Value="+IntegerToString(m_check_group.Value()));
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(ChartID(),"Controls",0,40,40,380,344))
      return(INIT_FAILED);
```

© 2000-2025, MetaQuotes Ltd.

```
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//---
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CCheckGroup.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

AddItem

Fügt ein neues Gruppenelement (eine Zeile) hinzu.

```
virtual bool  AddItem(
   const string  item,         // Text
   const long    value=0       // Wert
   )
```

Parameter

item

[in] Erläuterungstext des hinzugefügten Steuerelements.

value=0

[in] Ein Wert, der mit dem hinzugefügten Steuerelement verbunden ist.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Value

Erhält den mit dem Steuerelement verbundenen Wert.

```
long  Value()
```

Rückgabewert

Ein mit dem Steuerelement verbundener Wert.

Hinweis

Der Wert hängt vom Zustand aller Elemente in der Gruppe CCheckGroup.

© 2000-2025, MetaQuotes Ltd.

CreateButton

Erstellt ein neues Element CCheckBox in der Gruppe am angegebenen Index.

```
bool  CreateButton(
   int  index      // Index
   )
```

Parameter

index

[in] Index des neuen Elements CheckBox in der Gruppe.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnVScrollShow

Virtueller Handler des internen Ereignisses "Show" (einblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnVScrollHide

Virtueller Handler des internen Ereignisses "Hide" (ausblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineDown

Virtueller Handler des internen Ereignisses "ScrollLineDown" (scrollen eine Position nach unten) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineUp

Virtueller Handler des internen Ereignisses "ScrollLineUp" (scrollen eine Position nach oben) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChangeItem

Virtueller Handler des internen Ereignisses "ChangeItem" (Änderung des Zustands des Gruppenelements) des Steuerelements CCheckGroup.

```
virtual bool  OnChangeItem(
   const int  index      // Index
   )
```

Parameter

index

[in] Der Index des Elements, dessen Zustand geändert wurde.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Redraw

Zeichnet eine Elementgruppe neu.

```
bool  Redraw()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

RowState

Ändert den Zustand eines Elements der Gruppe.

```
bool  RowState(
   const int   index,      // Index
   const bool  select      // Zustand
   )
```

Parameter

index

[in] Der Index des Elements, dessen Zustand geändert wird.

select

[in] Der neue Zustand des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Klasse CRadioButton

CRadioButton ist eine Klasse des kombinierten Steuerelements "Schalter".

Beschreibung

Klasse CRadioButton hat keine eigenständige Anwendung und wird verwendet, um Schalter mit abhängigen Fixierung zu erstellen.

Deklaration

```
class CRadioButton : public CWndContainer
```

Kopf

```
#include <Controls\RadioButton.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CRadioButton

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Einstellungen	
Text	Erhält/setzt den mit dem Steuerelement verbundenen Text
Color	Erhält/setzt die Farbe des Erläuterungstextes eines Steuerelements
Zustand	
State	Erhält/setzt den Zustand eines Steuerelements
Unterlegende Steuerelemente	
CreateButton	Erstellt eine Taste
CreateLabel	Erstellt einen Erläuterungstext für eine Taste
Behandlung der Ereignisse der unterlegenden	

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Steuerelemente	
OnClickButton	Virtueller Handler des internen Ereignisses "ClickButton" eines Steuerelements
OnClickLabel	Virtueller Handler des internen Ereignisses "ClickLabel" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide, Save, Load

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CRadioButton.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Text (Get-Methode)

Erhält den mit dem Steuerelement CRadioButton verbundenen Text

```
string  Text()  const
```

Rückgabewert

Text.

Text (Set-Methode)

Setzt den Erläuterungstext.

```
bool  Text(
   const string  value      // Text
   )
```

Parameter

value

[in] Der neue Text.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Color (Get-Methode)

Erhält die Farbe des Erläuterungstextes des Steuerelements CRadioButton.

```
color  Color()  const
```

Rückgabewert

Die Farbe des Erläuterungstextes.

Color (Set-Methode)

Setzt die Farbe des Erläuterungstextes des Steuerelements CRadioButton.

```
bool  Color(
   const color  value      // Farbe
   )
```

Parameter

value

[in] Die neue Farbe Farbe des Erläuterungstextes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

State (Get-Methode)

Erhält den Zustand des Steuerelements CRadioButton.

```
bool  State()  const
```

Rückgabewert

Zustand der Taste.

State (Set-Methode)

Setzt den Zustand des Steuerelements CRadioButton.

```
bool  State(
   const bool  flag      // Zustand
   )
```

Parameter

flag

[in] Der neue Zustand der Taste.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateButton

Erstellt eine Taste.

```
virtual bool  CreateButton()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateLabel

Erstellt einen Erläuterungstext für eine Taste.

```
virtual bool  CreateLabel()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickButton

Virtueller Handler des internen Ereignisses "ClickButton" (Mausklick auf die Taste) vom Steuerelement CRadioButton.

```
virtual bool  OnClickButton()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnClickLabel

Virtueller Handler des internen Ereignisses "ClickLabel" (Mausklick auf den Label) vom Steuerelement CRadioButton.

```
virtual bool  OnClickLabel()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CRadioGroup

CRadioGroup ist eine Klasse des kombinierten Steuerelements "Schalter mit abhängigen Fixierung".

Beschreibung

Klasse CRadioGroup wird für Erstellung eines Steuerelements, der ein Feld des Enumerationstyps anzeigen und bearbeiten erlaubt.

Deklaration

```
class CRadioGroup : public CWndClient
```

Kopf

```
#include <Controls\RadioGroup.mqh>
```

Vererbungshierarchie

CObject
CWnd
CWndContainer
CWndClient
CRadioGroup

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Füllung	
AddItem	Fügt ein neues Gruppenelement hinzu
Daten (nur lesen)	
Value	Erhält den mit dem Zustand eines Steuerelements verbundenen Wert
Unterlegende Steuerelemente	
CreateButton	Erstellt ein neues Element in der Gruppe am angegebenen Index
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnVScrollShow	Virtueller Handler des internen Ereignisses "Show" des unterlegenden Steuerelements VScroll.
OnVScrollHide	Virtueller Handler des internen Ereignisses "Hide" des unterlegenden Steuerelements VScroll.
OnScrollLineDown	Virtueller Handler des internen Ereignisses "ScrollLineDown" des unterlegenden Steuerelements VScroll
OnScrollLineUp	Virtueller Handler des internen Ereignisses "ScrollLineUp" des unterlegenden Steuerelements VScroll.
OnChangeItem	Virtueller Handler des internen Ereignisses "ChangeItem" eines Steuerelements
Neuzeichnung	
Redraw	Zeichnet eine Elementgruppe neu
RowState	Ändert den Zustand eines Elements der Gruppe
Select	Wählt das aktuelle Element aus

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet,

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Enable, Disable, Hide

Methoden geerbt von der Klasse CWndClient

ColorBackground, ColorBorder, BorderType, VScrolled, VScrolled, HScrolled, HScrolled, Id

Ein Beispiel für die Erstellung eines Panels mit einer Gruppe von "Radio Buttons":

```
//+------------------------------------------------------------------+
//|                                            ControlsRadioGroup.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                             https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CRadio
#include <Controls\Dialog.mqh>
#include <Controls\RadioGroup.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
```

© 2000-2025, MetaQuotes Ltd.

```
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CRadioGroup       m_radio_group;                    // CRadioGroup Objekt

public:
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateRadioGroup(void);
   //--- handlers of the dependent controls events
   void              OnChangeRadioGroup(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
ON_EVENT(ON_CHANGE,m_radio_group,OnChangeRadioGroup)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateRadioGroup())
      return(false);
//--- succeed
```

© 2000-2025, MetaQuotes Ltd.

```
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "RadioGroup" element                                  |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateRadioGroup(void)
  {
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (BUTTON_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y)+
          (EDIT_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+RADIO_HEIGHT;
//--- create
   if(!m_radio_group.Create(m_chart_id,m_name+"RadioGroup",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!Add(m_radio_group))
      return(false);
//--- fill out with strings
   for(int i=0;i<3;i++)
      if(!m_radio_group.AddItem("Item "+IntegerToString(i),1<<i))
         return(false);
   m_radio_group.Value(1<<2);
   Comment(__FUNCTION__+" : Value="+IntegerToString(m_radio_group.Value()));
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeRadioGroup(void)
  {
   Comment(__FUNCTION__+" : Value="+IntegerToString(m_radio_group.Value()));
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CRadioGroup.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

AddItem

Fügt ein neues Gruppenelement hinzu.

```
virtual bool  AddItem(
   const string  item,         // Text
   const long    value=0       // Wert
   )
```

Parameter

item

[in] Erläuterungstext des hinzugefügten Steuerelements.

value=0

[in] Ein Wert, der mit dem hinzugefügten Steuerelement verbunden ist.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Value

Erhält den mit dem Zustand eines Steuerelements verbundenen Wert.

```
long  Value()
```

Rückgabewert

Ein wert, der mit dem Zustand eines Steuerelements verbundenen ist.

Hinweis

Der Wert hängt vom Zustand aller Elemente CRadioGroup in der Gruppe.

© 2000-2025, MetaQuotes Ltd.

CreateButton

Erstellt ein neues Element in der Gruppe am angegebenen Index.

```
bool  CreateButton(
   const int  index      // Index
   )
```

Parameter

index

[in] Index des neuen Elements in der Gruppe.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnVScrollShow

Virtueller Handler des internen Ereignisses "Show" (einblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollShow()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnVScrollHide

Virtueller Handler des internen Ereignisses "Hide" (ausblenden) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnVScrollHide()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineDown

Virtueller Handler des internen Ereignisses "ScrollLineDown" (scrollen eine Position nach unten) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineDown()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnScrollLineUp

Virtueller Handler des internen Ereignisses "ScrollLineUp" (scrollen eine Position nach oben) des unterlegenden Steuerelements VScroll (vertikale Bildlaufleiste).

```
virtual bool  OnScrollLineUp()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

OnChangeItem

Virtueller Handler des internen Ereignisses "ChangeItem" (Änderung des Zustands des Gruppenelements) des Steuerelements CRadioGroup.

```
virtual bool  OnChangeItem(
   const int  index      // Index
   )
```

Parameter

index

[in] Der Index des Elements, dessen Zustand geändert wurde.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Redraw

Zeichnet eine Elementgruppe neu.

```
bool  Redraw()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

RowState

Ändert den Zustand eines Elements der Gruppe.

```
bool  RowState(
   const int   index,      // Index
   const bool  select      // Zustand
   )
```

Parameter

index

[in] Der Index des Elements, dessen Zustand geändert wird.

select

[in] Der neue Zustand des Elements.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Select

Wählt das aktuelle Element aus.

```
void  Select(
   const int  index      // Index
   )
```

Parameter

index

[in] Der Index des ausgewählten Elements.

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Klasse CSpinEdit

CSpinEdit ist eine Klasse des kombinierten Steuerelements "Feld von Inkrement-Dekrement".

Beschreibung

Mit der CSpinEdit-Klasse erstellen Sie ein Steuerelement, um den Wert einer Integer-Variable mit den angegebenen Schritten in den festgelegten Grenzen zu bearbeiten.

Deklaration

```
class CSpinEdit : public CWndContainer
```

Kopf

```
#include <Controls\SpinEdit.mqh>
```

Vererbungshierarchie

COBject
CWnd
CWndContainer
CSpinEdit

Das Ergebnis des unten angeführten Codes:



Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement

© 2000-2025, MetaQuotes Ltd.

Erstellung	
Handler von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Einstellung	
MinValue	Erhält/setzt den Minimalwert eines Steuerelements
MaxValue	Erhält/setzt den Maximalwert eines Steuerelements
Zustand	
Value	Erhält/setzt den aktuellen Wert eines Steuerelements
Unterlegende Steuerelemente	
CreateEdit	Erstellt ein unterlegendes Steuerelement CEdit eines Steuerelements
CreateInc	Erstellt eine Taste für Erhöhung des Werts (Inkrement) eines Steuerelements
CreateDec	Erstellt eine Taste für Reduzierung des Werts (Dekrement) eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnClickInc	Virtueller Handler des internen Ereignisses "ClickInc" eines Steuerelements
OnClickDec	Virtueller Handler des internen Ereignisses "ClickDec" eines Steuerelements
Behandlung von internen Ereignissen	
OnChangeValue	Virtueller Handler des internen Ereignisses "ChangeValue" eines Steuerelements

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide

Ein Beispiel für die Erstellung eines Panels mit dem Scrollen von Werten:

```
//+------------------------------------------------------------------+
//|                                              ControlsSpinEdit.mq5 |
//|                        Copyright 2000-2024, MetaQuotes Ltd. |
//|                                              https://www.mql5.com |
//+------------------------------------------------------------------+
#property copyright "Copyright 2017, MetaQuotes Software Corp."
#property link      "https://www.mql5.com"
#property version   "1.00"
#property description "Das Panel der Anzeige und der Dialoge. Demonstration der CSpinE
#include <Controls\Dialog.mqh>
#include <Controls\SpinEdit.mqh>
//+------------------------------------------------------------------+
//| defines                                                          |
//+------------------------------------------------------------------+
//--- indents and gaps
#define INDENT_LEFT                         (11)      // indent from left (with allowa
#define INDENT_TOP                          (11)      // indent from top (with allowar
#define INDENT_RIGHT                        (11)      // indent from right (with allow
#define INDENT_BOTTOM                       (11)      // indent from bottom (with allo
#define CONTROLS_GAP_X                      (5)       // gap by X coordinate
#define CONTROLS_GAP_Y                      (5)       // gap by Y coordinate
//--- for buttons
#define BUTTON_WIDTH                        (100)     // size by X coordinate
#define BUTTON_HEIGHT                       (20)      // size by Y coordinate
//--- for the indication area
#define EDIT_HEIGHT                         (20)      // size by Y coordinate
//--- for group controls
#define GROUP_WIDTH                         (150)     // size by X coordinate
#define LIST_HEIGHT                         (179)     // size by Y coordinate
#define RADIO_HEIGHT                        (56)      // size by Y coordinate
#define CHECK_HEIGHT                        (93)      // size by Y coordinate
//+------------------------------------------------------------------+
//| Class CControlsDialog                                            |
//| Usage: main dialog of the Controls application                   |
//+------------------------------------------------------------------+
class CControlsDialog : public CAppDialog
  {
private:
   CSpinEdit         m_spin_edit;                    // CSpinEdit Objekt

public:
```

© 2000-2025, MetaQuotes Ltd.

```
                     CControlsDialog(void);
                    ~CControlsDialog(void);
   //--- create
   virtual bool      Create(const long chart,const string name,const int subwin,const
   //--- chart event handler
   virtual bool      OnEvent(const int id,const long &lparam,const double &dparam,cons

protected:
   //--- create dependent controls
   bool              CreateSpinEdit(void);
   //--- handlers of the dependent controls events
   void              OnChangeSpinEdit(void);
  };
//+------------------------------------------------------------------+
//| Event Handling                                                   |
//+------------------------------------------------------------------+
EVENT_MAP_BEGIN(CControlsDialog)
   ON_EVENT(ON_CHANGE,m_spin_edit,OnChangeSpinEdit)
EVENT_MAP_END(CAppDialog)
//+------------------------------------------------------------------+
//| Constructor                                                      |
//+------------------------------------------------------------------+
CControlsDialog::CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Destructor                                                       |
//+------------------------------------------------------------------+
CControlsDialog::~CControlsDialog(void)
  {
  }
//+------------------------------------------------------------------+
//| Create                                                           |
//+------------------------------------------------------------------+
bool CControlsDialog::Create(const long chart,const string name,const int subwin,const
  {
   if(!CAppDialog::Create(chart,name,subwin,x1,y1,x2,y2))
      return(false);
//--- create dependent controls
   if(!CreateSpinEdit())
      return(false);
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Create the "SpinEdit" element                                    |
//+------------------------------------------------------------------+
bool CControlsDialog::CreateSpinEdit(void)
  {
```

© 2000-2025, MetaQuotes Ltd.

```
//--- coordinates
   int x1=INDENT_LEFT;
   int y1=INDENT_TOP+(EDIT_HEIGHT+CONTROLS_GAP_Y)+(BUTTON_HEIGHT+CONTROLS_GAP_Y);
   int x2=x1+GROUP_WIDTH;
   int y2=y1+EDIT_HEIGHT;
//--- create
   if(!m_spin_edit.Create(m_chart_id,m_name+"SpinEdit",m_subwin,x1,y1,x2,y2))
      return(false);
   if(!Add(m_spin_edit))
      return(false);
   m_spin_edit.MinValue(10);
   m_spin_edit.MaxValue(100);
   m_spin_edit.Value(50);
   Comment(__FUNCTION__+" : Value="+IntegerToString(m_spin_edit.Value()));
//--- succeed
   return(true);
  }
//+------------------------------------------------------------------+
//| Event handler                                                    |
//+------------------------------------------------------------------+
void CControlsDialog::OnChangeSpinEdit(void)
  {
   Comment(__FUNCTION__+" : Value="+IntegerToString(m_spin_edit.Value()));
  }
//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
CControlsDialog ExtDialog;
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
//--- create application dialog
   if(!ExtDialog.Create(0,"Controls",0,40,40,380,344))
      return(INIT_FAILED);
//--- run application
   ExtDialog.Run();
//--- succeed
   return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
  {
//--- Kommentare löschen
   Comment("");
//--- destroy dialog
```

© 2000-2025, MetaQuotes Ltd.

```
   ExtDialog.Destroy(reason);
  }
//+------------------------------------------------------------------+
//| Expert chart event function                                      |
//+------------------------------------------------------------------+
void OnChartEvent(const int id,         // event ID
                  const long& lparam,   // event parameter of the long type
                  const double& dparam, // event parameter of the double type
                  const string& sparam) // event parameter of the string type
  {
   ExtDialog.ChartEvent(id,lparam,dparam,sparam);
  }
```

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt einen Steuerelement CSpinEdit.

```
virtual bool  Create(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

MinValue (Get-Methode)

Erhält den Wert des Parameters "MinValue" (Minimalwert) des Steuerelements CSpinEdit.

```
int  MinValue()  const
```

Rückgabewert

Wert des Parameters "MinValue".

MinValue (Set-Methode)

Setzt den Wert des Parameters "MinValue" (Minimalwert) des Steuerelements CSpinEdit.

```
void  MinValue(
   const int  value      // Wert
   )
```

Parameter

value

[in] Ein neuer Wert des Parameters "MinPos".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

MaxValue (Get-Methode)

Erhält den Wert des Parameters "MaxValue" (Maximalwert) des Steuerelements CSpinEdit.

```
int  MaxValue()  const
```

Rückgabewert

Wert des Parameters "MaxValue".

MaxValue (Set-Methode)

Setzt den Wert des Parameters "MaxValue" (Maximalwert) des Steuerelements CSpinEdit.

```
void  MaxValue(
   const int  value      // Wert
   )
```

Parameter

value

[in] Der neue Wert des Parameters "MaxValue".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Value (Get-Methode)

Erhält den Wert des Parameters "Value" (Wert) des Steuerelements CSpinEdit.

```
int  Value()  const
```

Rückgabewert

Wert des Parameters "Value".

Value (Set-Methode)

Setzt den Wert des Parameters "Value" (Wert) des Steuerelements CSpinEdit.

```
void  Value(
   const int  value      // Wert
   )
```

Parameter

value

[in] Ein neuer Wert des Parameters "Value".

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

CreateEdit

Erstellt ein unterlegendes Steuerelement (Eingabefeld) des Steuerelements CSpinEdit.

```
virtual bool  CreateEdit()
```

Rückgabewert

Gibt bei Erfolg true zurück, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

CreateInc

Erstellt eine Taste für Erhöhung des Werts (Inkrement) des Steuerelements CSpinEdit.

```
virtual bool  CreateInc()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateDec

Erstellt eine Taste für Reduzierung des Werts (Dekrement) des Steuerelements CSpinEdit.

```
virtual bool  CreateDec()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickInc

Virtueller Handler des internen Ereignisses "ClickInc" (Mausklick auf die Taste für Werterhöhung) vom Steuerelement CSpinEdit.

```
virtual bool  OnClickInc()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickDec

Virtueller Handler des internen Ereignisses "ClickDec" (Mausklick auf die Taste für Wertreduzierung) vom Steuerelement CSpinEdit.

```
virtual bool  OnClickDec()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnChangeValue

Virtueller Handler des internen Ereignisses "ChangeValue" (Änderung des aktuellen Werts) des Steuerelements CSpinEdit.

```
virtual bool  OnChangeValue()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Klasse CDialog

CDialog ist eine Klasse des kombinierten Steuerelements "Dialog".

Beschreibung

Klasse CDialog wird für die visuelle Gruppierung von funktionell verwandten heterogenen Elementen verwendet.

Deklaration

```
class CDialog : public CWndContainer
```

Kopf

```
#include <Controls\Dialog.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CDialog

Direkte Ableitungen

CAppDialog

Gruppen der Klassenmethode

Erstellung	
Create	Erstellt ein Steuerelement
Behandlung von Chart-Ereignissen	
OnEvent	Behandelt alle Chart-Ereignisse
Einstellung	
Caption	Erhält/setzt den Wert der Eigenschaft "Caption" eines Steuerelements
Füllung	
Add	Fügt ein Steuerelement in den Kundenbereich nach dem Zeiger/Referenz hinzu
Unterlegende Steuerelemente	
CreateWhiteBorder	Erstellt ein unterlegendes Steuerelement (weißer Rahmen) eines Steuerelements

© 2000-2025, MetaQuotes Ltd.

Erstellung	
CreateBackground	Erstellt ein unterlegendes Steuerelement (Hintergrund) eines Steuerelements
CreateCaption	Erstellt ein unterlegendes Steuerelement (Fenstertitel) eines Steuerelements
CreateButtonClose	Erstellt ein unterlegendes Steuerelement (Fenster-Schließtaste) eines Steuerelements
CreateClientArea	Erstellt ein unterlegendes Steuerelement (Kundenbereich) eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnClickCaption	Virtueller Handler des internen Ereignisses "ClickCaption" eines Steuerelements
OnClickButtonClose	Virtueller Handler des internen Ereignisses "ClickButtonClose" eines Steuerelements
Zugriff auf die Eigenschaften des Kundenbereichs	
ClientAreaVisible	Setzt das Flag der Sichtbarkeit des unterlegenden Steuerelements (Kundenbereich) eines Steuerelements
ClientAreaLeft	Erhält die X-Koordinate des oberen linken Punktes des Kundenbereichs eines Steuerelements.
ClientAreaTop	Erhält die Y-Koordinate des oberen linken Punktes des Kundenbereichs eines Steuerelements.
ClientAreaRight	Erhält die X-Koordinate des unteren rechten Punktes des Kundenbereichs eines Steuerelements.
ClientAreaBottom	Erhält die Y-Koordinate des unteren rechten Punktes des Kundenbereichs eines Steuerelements.
ClientAreaWidth	Erhält die Breite des Kundenbereichs des Steuerelements
ClientAreaHeight	Erhält die Höhe des Kundenbereichs des Steuerelements
Behandlung von Ziehen	
OnDialogDragStart	Virtueller Handler des Ereignisses "DialogDragStart" eines Steuerelements
OnDialogDragProcess	Virtueller Handler des Ereignisses "DialogDragProcess" eines Steuerelements
OnDialogDragEnd	Virtueller Handler des Ereignisses "DialogDragEnd" eines Steuerelements

© 2000-2025, MetaQuotes Ltd.

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

Destroy, OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CDialog.

```
virtual bool  Create(
   const long    chart,      // Chart-ID
   const string  name,       // Name
   const int     subwin,     // Chart-Unterfenster
   const int     x1,         // Koordinate
   const int     y1,         // Koordinate
   const int     x2,         // Koordinate
   const int     y2          // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Caption (Get-Methode)

Erhält den Wert der Eigenschaft "Caption" (Titel) des Steuerelements CDialog.

```
string  MinValue()  const
```

Rückgabewert

Der Wert der Eigenschaft "Caption".

Caption (Set-Methode)

Setzt den Wert der Eigenschaft "Caption" (Titel) des Steuerelements CDialog.

```
bool  Caption(
   const string  text      // Text
   )
```

Parameter

text

[in] Der neue Wert der Eigenschaft "Caption".

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Add

Fügt ein Steuerelement in den Kundenbereich nach dem Zeiger hinzu.

```
bool  Add(
   CWnd   *control,         // Zeiger
   )
```

Parameter

control

[in] Zeiger auf ein Steuerelement.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

Add

Fügt ein Steuerelement in den Kundenbereich nach Referenz hinzu

```
bool  Add(
   CWnd   &control,         // Referenz
   )
```

Parameter

control

[in] Referenz auf ein Steuerelement.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateWhiteBorder

Erstellt ein unterlegendes Steuerelement (weißer Rahmen) des Steuerelements CDialog.

```
virtual bool  CreateWhiteBorder()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateBackground

Erstellt ein unterlegendes Steuerelement (Hintergrund) des Steuerelements CDialog.

```
virtual bool  CreateBackground()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateCaption

Erstellt ein unterlegendes Steuerelement (Fenstertitel) des Steuerelements CDialog.

```
virtual bool  CreateCaption()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateButtonClose

Erstellt ein unterlegendes Steuerelement (Fenster-Schließtaste) des Steuerelements CDialog.

```
virtual bool  CreateButtonClose()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateClientArea

Erstellt ein unterlegendes Steuerelement (Kundenbereich) des Steuerelements CDialog.

```
virtual bool  CreateClientArea()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickCaption

Virtueller Handler des internen Ereignisses "ClickCaption" (Mausklick auf die Fenstertitel) vom Steuerelement CDialog.

```
virtual bool  OnClickCaption()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

OnClickButtonClose

Virtueller Handler des internen Ereignisses "ClickButtonClose" (Mausklick auf die Schließen-Taste) vom Steuerelement CDialog.

```
virtual bool  OnClickButtonClose()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ClientAreaVisible

Setzt das Flag der Sichtbarkeit des unterlegenden Steuerelements (Kundenbereich) des Steuerelements CDialog.

```
bool  ClientAreaVisible(
   const bool  visible      // Flag der Sichtbarkeit
   )
```

Parameter

visible

[in] Flag der Sichtbarkeit.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ClientAreaLeft

Erhält die X-Koordinate des oberen linken Punktes des Kundenbereichs des Steuerelements CDialog.

```
int  ClientAreaLeft()
```

Rückgabewert

Die X-Koordinate des oberen linken Punktes des Kundenbereichs.

© 2000-2025, MetaQuotes Ltd.

ClientAreaTop

Erhält die Y-Koordinate des oberen linken Punktes des Kundenbereichs des Steuerelements CDialog.

```
int  ClientAreaTop()
```

Rückgabewert

Die Y-Koordinate des oberen linken Punktes des Kundenbereichs.

© 2000-2025, MetaQuotes Ltd.

ClientAreaRight

Erhält die X-Koordinate des unteren rechten Punktes des Kundenbereichs des Steuerelements CDialog.

```
int  ClientAreaTop()
```

Rückgabewert

Die X-Koordinate des unteren rechten Punktes des Kundenbereichs.

© 2000-2025, MetaQuotes Ltd.

ClientAreaBottom

Erhält die Y-Koordinate des unteren rechten Punktes des Kundenbereichs des Steuerelements CDialog.

```
int  ClientAreaBottom()
```

Rückgabewert

Die Y-Koordinate des unteren rechten Punktes des Kundenbereichs.

© 2000-2025, MetaQuotes Ltd.

ClientAreaWidth

Erhält die Breite des Kundenbereichs des Steuerelements CDialog.

```
int  ClientAreaWidth()
```

Rückgabewert

Die Breite des Kundenbereichs.

© 2000-2025, MetaQuotes Ltd.

ClientAreaHeight

Erhält die Höhe des Kundenbereichs des Steuerelements CDialog.

```
int  ClientAreaHeight()
```

Rückgabewert

Die Höhe des Kundenbereichs.

© 2000-2025, MetaQuotes Ltd.

OnDialogDragStart

Virtueller Handler des Ereignisses "DialogDragStart" (der Anfang der Drag-Operation) des Steuerelements CDialog.

```
virtual bool  OnDialogDragStart()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "DialogDragStart" tritt am Anfang des Ziehens des Steuerelements CDialog auf.

© 2000-2025, MetaQuotes Ltd.

OnDialogDragProcess

Virtueller Handler des Ereignisses "DialogDragProcess" (Drag-Operation) des Steuerelements CDialog.

```
virtual bool  OnDialogDragProcess()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "DialogDragProcess" erscheint während des Ziehens des Steuerelements CDialog.

© 2000-2025, MetaQuotes Ltd.

OnDialogDragEnd

Virtueller Handler des Ereignisses "DialogDragEnd" (das Ende der Drag-Operation) des Steuerelements CDialog.

```
virtual bool  OnDialogDragEnd()
```

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

Hinweis

Das Ereignis "DialogDragEnd" tritt am Ende des Ziehens des Steuerelements CDialog auf.

© 2000-2025, MetaQuotes Ltd.

Klasse CAppDialog

CAppDialog ist eine Klasse des kombinierten Steuerelements "Anwendung-Dialog".

Beschreibung

Klasse CAppDialog wird für die visuelle Gruppierung von funktionell verwandten heterogenen Elementen innerhalb eines MQL5-Programms verwendet.

Deklaration

```
class CAppDialog : public CDialog
```

Kopf

```
#include <Controls\Dialog.mqh>
```

Vererbungshierarchie

CObject
  CWnd
    CWndContainer
      CDialog
        CAppDialog

Gruppen der Klassenmethode

Erstellen und Löschen	
Create	Erstellt ein Steuerelement
Destroy	Löscht ein Steuerelement
Ereignisbehandlung	
OnEvent	Behandelt alle Ereignisse
Ablauf	
Run	Startet ein Steuerelement
Behandlung von Chart-Ereignissen	
ChartEvent	Behandelt alle Chart-Ereignisse
Einstellung	
Minimized	Setzt ein Flag für Minimieren eines Steuerelements
Zustand speichern/wiederherstellen	
IniFileSave	Speichert den Zustand des Steuerelements in einer Datei

© 2000-2025, MetaQuotes Ltd.

Erstellen und Löschen	
IniFileLoad	Lädt den Zustand des Steuerelements aus einer Datei
IniFileName	Setzt den Dateinamen für das Speichern/Laden des Zustands eines Steuerelements
IniFileExt	Setzt den Datei-Endung für das Speichern/Laden des Zustands eines Steuerelements
Initialisierung	
CreateCommon	Methode zur Initialisierung der allgemeinen Einstellungen eines Steuerelements
CreateExpert	Methode zur Initialisierung der Einstellungen eines Steuerelements für Arbeit in einem Expert Advisor
CreateIndicator	Methode zur Initialisierung der Einstellungen eines Steuerelements für Arbeit in einem Indikator
Unterlegende Steuerelemente	
CreateButtonMinMax	Erstellt unterlegende Steuerelemente (Tasten für Fenster-Minimieren/Wiederherstellen) eines Steuerelements
Behandlung der Ereignisse der unterlegenden Steuerelemente	
OnClickButtonClose	Virtueller Handler des internen Ereignisses "ClickButtonClose" eines Steuerelements
OnClickButtonMinMax	Virtueller Handler des internen Ereignisses "ClickButtonMinMax" eines Steuerelements
Behandlung von externen Ereignissen	
OnAnotherApplicationClose	Virtueller Handler von externen Ereignissen eines Steuerelements
Methoden	
Rebound	Setzt die neuen Parameter des rechteckigen Bereichs des Steuerelements aus den Koordinaten der Klasse CRect
Minimize	Setzt den Fenster des Steuerelements in einen minimierten Zustand
Maximize	Setzt den Fenster des Steuerelements in einen wiederhergestellten Zustand
CreateInstanceId	Erstellt ein eindeutiges Präfix für die Namen der Objekte eines Steuerelements
ProgramName	Erhält den Namen des Programms, das das Steuerelement verwendet

© 2000-2025, MetaQuotes Ltd.

Erstellen und Löschen	
SubwinOff	Erhält die Verschiebung entlang der Y-Achse des Fensters, in dem das Steuerelement sich befindet

Methoden geerbt von der Klasse CObject

Prev, Prev, Next, Next, Type, Compare

Methoden geerbt von der Klasse CWnd

Name, ControlsTotal, Control, Rect, Left, Left, Top, Top, Right, Right, Bottom, Bottom, Width, Width, Height, Height, Size, Size, Size, Contains, Contains, Alignment, Align, Id, IsEnabled, IsVisible, Visible, IsActive, Activate, Deactivate, StateFlags, StateFlags, StateFlagsSet, StateFlagsReset, PropFlags, PropFlags, PropFlagsSet, PropFlagsReset, MouseX, MouseX, MouseY, MouseY, MouseFlags, MouseFlags, MouseFocusKill, BringToTop

Methoden geerbt von der Klasse CWndContainer

OnMouseEvent, ControlsTotal, Control, ControlFind, MouseFocusKill, Add, Add, Delete, Delete, Move, Move, Shift, Id, Enable, Disable, Show, Hide

Methoden geerbt von der Klasse CDialog

Caption, Caption, Add, Add

© 2000-2025, MetaQuotes Ltd.

Create

Erstellt ein Steuerelement CAppDialog.

```
virtual bool  Create(
   const long    chart,      // Chart-ID
   const string  name,       // Name
   const int     subwin,     // Chart-Unterfenster
   const int     x1,         // Koordinate
   const int     y1,         // Koordinate
   const int     x2,         // Koordinate
   const int     y2          // Koordinate
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Destroy

CAppDialog, Methode zur Deinitialisierung des Steuerelements.

```
virtual void  Destroy(
   const int  reason=REASON_PROGRAM    // Ursachen-Code
)
```

Parameter

reason

[in] Deinitialisierung Ursachen-Code. REASON_PROGRAM wird standardmäßig gesetzt.

Rückgabewert

Keiner.

© 2000-2025, MetaQuotes Ltd.

OnEvent

Behandelt alle Chart-Ereignisse.

```
virtual bool  OnEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Run

Methode der Start des Steuerelements CAppDialog.

```
bool  Run()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

ChartEvent

Virtueller Handler von Ereignissen des Steuerelements CAppDialog.

```
virtual bool  ChartEvent(
   const int      id,          // Identifikator
   const long&    lparam,      // Parameter
   const double&  dparam,      // Parameter
   const string&  sparam       // Parameter
   )
```

Parameter

id

[in] Identifikator des Ereignisses.

lparam

[in] Referenz auf den Parameter eines Ereignisses von long-Typ.

dparam

[in] Referenz auf den Parameter eines Ereignisses von double-Typ.

sparam

[in] Referenz auf den Parameter eines Ereignisses von string-Typ.

Rückgabewert

true - wenn das Ereignis behandelt ist, ansonsten false.

© 2000-2025, MetaQuotes Ltd.

Minimized

Setzt den Wert des Flags "Minimized" (Fenster-Zustand) des Steuerelements CAppDialog.

```
bool  Minimized(
   const bool  flag      // Zustand
   )
```

Parameter

flag

[in] Der neue Zustand.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

IniFileSave

Speichert den Zustand des Steuerelements CAppDialog in eine Datei.

```
void  IniFileSave()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

IniFileLoad

Lädt den Zustand des Steuerelements CAppDialog in eine Datei.

```
void  IniFileLoad()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

IniFileName

Erhält den Namen der Datei für das Speichern/Laden des Zustands des Steuerelements CAppDialog.

```
virtual string  IniFileName()  const
```

Rückgabewert

Name der Datei für das Speichern/Laden des Zustands des Steuerelements CAppDialog.

Hinweis

Dateiname enthält den Namen des Indikators/Expert Advisor, sowie ein Symbol, auf das das Programm ausgeführt wird.

© 2000-2025, MetaQuotes Ltd.

IniFileExt

Erhält den Datei-Endung für das Speichern/Laden des Zustands des Steuerelements CAppDialog.

```
virtual string  IniFileExt()  const
```

Rückgabewert

Endung der Datei für das Speichern/Laden des Zustands des Steuerelements CAppDialog.

© 2000-2025, MetaQuotes Ltd.

CreateCommon

Methode zur Initialisierung der allgemeinen Einstellungen des Steuerelements CAppDialog.

```
bool  CreateCommon(
   const long     chart,       // Chart-ID
   const string   name,        // Name
   const int      subwin,      // Chart-Unterfenster
   )
```

Parameter

chart

[in] ID des Charts, auf dem ein Steuerelement erstellt wird.

name

[in] Der eindeutige Name des Steuerelements.

subwin

[in] Der Unterfenster des Charts, in dem ein Steuerelement erstellt wird.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateExpert

Methode zur Initialisierung der Einstellungen des Steuerelements CAppDialog für Arbeit in einem Expert Advisor.

```
bool  CreateExpert(
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateIndicator

Methode zur Initialisierung der Einstellungen des Steuerelements CAppDialog für Arbeit in einem Indikator.

```
bool  CreateIndicator(
   const int      x1,          // Koordinate
   const int      y1,          // Koordinate
   const int      x2,          // Koordinate
   const int      y2           // Koordinate
   )
```

Parameter

x1

[in] Der Wert der X-Koordinate des oberen linken Punktes.

y1

[in] Der Wert der Y-Koordinate des oberen linken Punktes.

x2

[in] Der Wert der X-Koordinate des unten rechten Punktes.

y2

[in] Der Wert der Y-Koordinate des unten rechten Punktes.

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateButtonMinMax

Erstellt unterlegende Steuerelemente (Tasten für Minimieren/Wiederherstellen) des Steuerelements CAppDialog.

```
virtual void  CreateButtonMinMax()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnClickButtonClose

Virtueller Handler des internen Ereignisses "ClickButtonClose" (Mausklick auf die Schließen-Taste) von Steuerelement CAppDialog.

```
virtual void  OnClickButtonClose()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnClickButtonMinMax

Virtueller Handler des internen Ereignisses "ClickButtonMinMax" (Mausklick auf die Minimieren/Wiederherstellen-Taste) von Steuerelement CAppDialog.

```
virtual void  OnClickButtonClose()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

OnAnotherApplicationClose

Virtueller Handler von externen Ereignissen des Steuerelements CAppDialog.

```
virtual void  OnAnotherApplicationClose()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Rebound

Setzt die neuen Parameter des rechteckigen Bereichs des Steuerelements CAppDialog aus den Koordinaten der Klasse CRect.

```
bool  Rebound(
   const & CRect  rect      // Klasse der Rechteckfläche
   )
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Minimize

Setzt den Fenster des Steuerelements CAppDialog in einen minimierten Zustand.

```
virtual void  Minimize()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

Maximize

Setzt den Fenster des Steuerelements CAppDialog in einen wiederhergestellten Zustand.

```
virtual void  Maximize()
```

Rückgabewert

Gibt bei Erfolg true, ansonsten false zurück.

© 2000-2025, MetaQuotes Ltd.

CreateInstanceId

Erstellt ein eindeutiges Präfix für die Namen der Objekte des Steuerelements CAppDialog.

```
string  CreateInstanceId()
```

Rückgabewert

Das Präfix für die Objektnamen.

© 2000-2025, MetaQuotes Ltd.

ProgramName

Erhält den Namen des Programms, das das Steuerelement CAppDialog verwendet.

```
string  ProgramName()
```

Rückgabewert

Name des MQL5-Programms.

© 2000-2025, MetaQuotes Ltd.

SubwinOff

Erhält die Verschiebung entlang der Y-Achse des Fensters, in dem das Steuerelement CAppDialog sich befindet.

```
void  SubwinOff()
```

Rückgabewert

Nichts.

© 2000-2025, MetaQuotes Ltd.

Übergang von MQL4

Die SpracheMQL5 ist Entwicklung ihres Vorgaengers – der Sprache MQL4, in der viele Indikatoren, Scripts und Experten geschrieben werden. Obwohl die neue Programmiersprache mit der Sprache der früheren Generation maximal kompatibel ist. Es gibt es eine Reihe der Unterschiede zwischen diesen Sprachen. Bei der Portierung de Programme muss man diese Unterschiede kennen.

In diesem Abschnitt gibt es die notwendige Information, um die Adaptierung der Kodes zur neuen Sprache MQL5 für Programmierer, die MQL4 gut kennen, zu ermöglichen.

Vor allem muss bemerkt werden:

- Funktionen start(), init()und deinit() fehlen;
- Anzahl der Indikatorpuffer ist nicht begrenzt;
- Ladung dll erfolgt sofort nach der Ladung von Experten (oder eines anderen mql5-Programms);
- Pruefung der logischen Bedingungen ist verkuerzt;
- die laufende Durchführung wird bei der Grenzenüberschreitung des Feldes gestoppt (kritisch - mit Fehlerausgabe) ;
- Prioritaet der Operationen wie in C ++;
- Explizite Typenreduzierung (auch von der Zeile in die Zahl);
- Lokale Variablen werden automatisch nicht initialisiert (ausser Zeilen);
- Normale lokale Felder werden automatisch entfernt.

Sonderfunktionen init, start und deinit

In de Sprache MQL4 gab es nur drei vorbestimmte Funktionen, die im Kode eines Indikators, Ratgebers oder Scripts sein konnten (nicht beachtet Include-Dateien *.mqh und Bibliothekdateien). Diese Funktionen fehlen in MQL5, aber ihre Analoge gibt es. In der Tabelle ist die ungefaehre Entsprechnung dieser Funktionen dargestellt.

MQL4	MQL5
init	OnInit
start	OnStart
deinit	OnDeinit

Funktionen OnInit und OnDeinit spielen dieselbe Rolle wie die Funktionen init und deinit in MQL4 – sie sind bestimmt für Positionieren des Kodes, der bei der Initialisierung und bei der Deinitialisierung des mql5-Programms durchgeführt werden muss. Sie können diese Funktionen entsprechenderweise umbenennen oder lassen sie wie sie sind, aber in entsprechenden Stellen den Aufruf zuzufuegen.

Beispiel:

© 2000-2025, MetaQuotes Ltd.

```
void OnInit()
  {
//--- Funktion wird bei der Initialisierung aufgerufen
   init();
  }
void OnDeinit(const int reason)
  {
//--- Funktion wird bei der Deinitialisierung aufgerufen
   deinit();
//---
  }
```

Funktion start wird durch die Funktion OnStart nur in Scripts ersetzt, für Experten und Indikator muss sie in OnTick und OnCalculate umbenennen. Der Kode, der innerhalb des mql5-Programms durchzuführen ist, muss in diesen drei Funktionen sein :

mql5-Programm	Hauptfunktion
Script	OnStart
Indikator	OnCalculate
Expert	OnTick

Wenn die Hautfunktion im Kode des Indikators oder Scripts fehlt oder sich der Name dieser Funktion von dem eforderlichen Namen unterscheidet, wird diese Funktion nicht aufgerufen. D.H. wenn es im Ausgangskode des Scripts die Funktion OnStart fehlt, wird dieser Kode als Ratgeber compilert.

Wenn die Funktion OnCalculate im Kode des Indikators fehlt, ist die Compilierung dieses Indikators unmoeglich.

Vorbestimmte Variablen

In MQL5 gibt es keine solche Variablen wie Ask, Bid, Bars. Die Schreibweise der Variablen Digits und Point ist etwas verändert, wie es in der Tabelle gezeigt ist.

MQL4	MQL5
Digits	_Digits
Point	_Point
	_LastError
	_Period
	_Symbol
	_StopFlag
	_UninitReason

Zugang zu Zeitreihen

© 2000-2025, MetaQuotes Ltd.

In MQL5 gibt es keine vorbestimmte Zeitreihen Open[], High[], Low[], Close[], Volume[] und Time[]. Die notwendige Tiefe der Zeitreihe kann selbsstaendig durch die entsprechenden Funktionen für Zugang zu Zeitreihen vorgegeben werden.

Ratgeber (Experten)

Experten in MQL5 brauchen nicht unbedingt die Funktion-Bearbeiter des Ereignisses haben, wenn neues Tick OnTick empfangen wird,wie es in MQL4 der Fall war (Funktion start in MQL4 wird beim Empfang neues Ticks aufgerufen), denn jetzt können Experten in MQL5 vorbestimmte Funktionen-Bearbeiter mehrerer Typen der Ereignisse haben:

- OnTick – Empfang eines neuen Ticks;
- OnTimer – Ereignis des Timers;
- OnTrade - Handelsereignis;
- OnChartEvent – Ereignis der Eingabe von Keyboard und Maus, Ereignis der Verschiebung des graphischen Objektes, Ereignis der Beendigung der Textbearbeitung im Feld der Ogjektausgabe;
- OnBookEvent – Ereignis der DOM Veränderung (Depth of Market).

Benutzerindikatoren

In MQL4 ist die Anzahl von Indikatorpuffer begrenzt und kann nicht mehr als 8 sein. In MQL5 gibt es keine solche Einschraenkung, aber man muss sich daran erinnern, dass jeder Indikatorpuffer ein bestimmtes Volumen des Operativspeichers erfordert, darum muss man die neue Moeglichkeit nicht missbrauchen.

Außerdem gab es in MQL4 nur 6 Typen der Zeichnung des Benutzerindikators, in MQL5 gibt es nun 18 Zeichnungsstile. Obwohl sich die Namen der Zeichnungstypen nicht verändert haben, hat sich die Ideologie der graphischen Darstellung der Indikatoren wesentlich verändert.

Richtung des Indizierens in Indikatorpuffern unterscheidet sich auch. In MQL5 haben alle Puffer dasselbe Verhalten, wie normale Felder, d.h. Element mit dem Index 0 ist das aelteste in der Geschichte, beim Steigen des Indexes bewegen wir von den aeltesten Daten zu den letztesten.

Die einzige Funktion für die Arbeit mit Benutzerindikatoren, die sich aus MQL4 erhaelt ist SetIndexBuffer. Aber ihrer Aufruf hat sich verändert, nun muss man Typ der Daten, die im Feld aufbewahren werden angeben, verbunden mit dem Indikatorpuffer.

Außerdem haben sich die Eigenschaften der Benutzerindikatoren verändert und erweitert, Funktionen für Zugang zu Zeitreihen erschienen,darum muss der ganze Algorithmuss der Berechnung vollstaendig neu überlegt werden.

Graphische Objekte

Die Anzahl der graphischen Objekte in MQL5 ist wesentlich gestiegen. Außerdem ist Positionieren der graphischen Objekte in der Zeit mit der Genauigkeit bis auf die Sekunde ist auf dem Chart jeder Periode moeglich geworden - jetzt werden Bezugspunkte von graphischen Objekten mit der Genauigkeit bis auf die Eroeffnungszeit der Bar im laufenden Preischart nicht ausgerunden.

© 2000-2025, MetaQuotes Ltd.

für Objekte Arrow, Text und Label ist es moeglich geworden, Weg der Verbindung anzugeben, und für Obkekte Label, Button, Chart, Bitmap Label und Edit ist es moeglich, Chartwinkel, zu dem Objekt gebunden ist vorzugeben .

© 2000-2025, MetaQuotes Ltd.

MQL5 Funktionenliste

Alle MQL5 Funktionen in alphabetischer Reihenfolge.

Funktion	Aktion	Sektion
AccountInfoDouble	Gibt den wert des Typs double der entsprechenden Eigenschaft des Kontos zurück	Kontoinformation
AccountInfoInteger	Gibt Wert des ganzzahligen Typs (bool,int oder long) der entsprechenden Eigenschaft des Kontos zurück	Kontoinformation
AccountInfoString	Gibt den Wert des Typs string der entsprechenden Eigenschaft des Kontos zurück	Kontoinformation
acos	Gibt arccos x in Radianen zurück	Mathematische Funktionen
Alert	Gibt Meldungen in einem separaten Fenster aus	Allgemeine Funktionen
ArrayBsearch	Ermittelt den angegeben Wert in einem aufsteigend sortierten mehrdimensionalen Array	Operationen mit Arrays
ArrayCompare	Gibt das Ergebnis eines Vergleichs um zwei Arrays von einfachen Typen oder angepassten Strukturen ohne komplexe Objekte zurück	Operationen mit Arrays
ArrayCopy	Kopiert ein Array in das andere Array	Operationen mit Arrays
ArrayFill	Füllt das numerische Array mit den angegeben Wert	Operationen mit Arrays
ArrayFree	Befreit Puffer jedes dynamischen Arrays und stellt die Größe der ersten Null-Dimension auf 0.	Operationen mit Arrays
ArrayGetAsSeries	Prüft die Richtung des Arrayindizierens	Operationen mit Arrays
ArrayInitialize	Setzt alle Elementen des nummerischen Arrays auf Größe	Operationen mit Arrays
ArrayIsDynamic	Prüft, ob das Array dynamisches Array ist	Operationen mit Arrays
ArrayIsSeries	Prüft, ob das Array Timeserie ist	Operationen mit Arrays

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
ArrayMaximum	Ermittelt das maximale Element in der ersten Dimension eines mehrdimensionalen Arrays	Operationen mit Arrays
ArrayMinimum	Ermittelt das minimale Element in der ersten Dimension eines mehrdimensionalen Arrays	Operationen mit Arrays
ArrayRange	Gibt die Zahl der Elementen in der gegebenen Dimension des Arrays zurück	Operationen mit Arrays
ArrayResize	Stellt die neue Größe in der ersten Dimension des Arrays ein	Operationen mit Arrays
ArraySetAsSeries	Stellt die Richtung der Indizierung im Array ein	Operationen mit Arrays
ArraySize	Gibt die Zahl von Elementen im Array zurück	Operationen mit Arrays
ArraySort	Sortiert das mehrdimensionale Array aufsteigend in der ersten Dimension	Operationen mit Arrays
ArrayPrint	Gibt Arrays eines einfachen Typs oder einer einfachen Struktur im Journal aus	Operationen mit Arrays
ArrayInsert	Einfügen einer Anzahl von Elementen eines Quellarrays in ein Zielarray, beginnend mit dem angegebenen Index	Operationen mit Arrays
ArrayRemove	Entfernt die angegebene Anzahl von Elementen im Array, beginnend mit dem angegebenen Index	Operationen mit Arrays
ArrayReverse	Kehrt die angegebene Anzahl von Elementen im Array um, beginnend mit dem angegebenen Index	Operationen mit Arrays
ArraySwap	Vertauscht die Inhalte von zwei dynamischen Arrays eines Typs	Operationen mit Arrays
asin	Gibt arcsin x in Radianen zurück	Mathematische Funktionen
atan	Gibt arctg x in Radianen zurück	Mathematische Funktionen
Bars	Gibt die die entsprechende PeriodeAnzahl der Bars in der Geschichte für das entsprechende	Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Symbol und die entsprechende Periode	
BarsCalculated	Gibt die Anzahl der berechneten Daten im Anzeigerpuffer oder -1 beim Fehler (Daten sind noch nicht berechnet)	Zugang zu Zeitreihen und Indikatoren
CalendarCountryById	Abrufen der Länderbeschreibung nach seiner ID	Wirtschaftskalender
CalendarEventById	Abrufen der Ereignisbeschreibung nach seiner ID	Wirtschaftskalender
CalendarValueById	Abrufen der Ereigniswertbeschreibung nach seiner ID	Wirtschaftskalender
CalendarCountries	Abrufen des Arrays der Namen der im Kalender verfügbaren Länder	Wirtschaftskalender
CalendarEventByCountry	Abrufen des Arrays mit den Beschreibungen aller im Kalender verfügbaren Ereignisse nach dem angegebenen Landes-Code	Wirtschaftskalender
CalendarEventByCurrency	Abrufen des Arrays der Beschreibungen aller im Kalender verfügbaren Ereignisse in einer bestimmten Währung	Wirtschaftskalender
CalendarValueHistoryByEvent	Abrufen des Arrays der Werte für alle Ereignisse in einem bestimmten Zeitraum mit einem Ereignis-ID	Wirtschaftskalender
CalendarValueHistory	Abrufen des Arrays der Werte für alle Ereignisse in einem bestimmten Zeitbereich mit der Möglichkeit, nach Land und/oder Währung zu sortieren	Wirtschaftskalender
CalendarValueLastByEvent	Abrufen des Arrays der Ereigniswerte nach seiner ID seit dem Status der Kalenderdatenbank nach einer angegebenen change_id	Wirtschaftskalender
CalendarValueLast	Abrufen des Arrays der Werte für alle Ereignisse mit der Möglichkeit, nach Land und/oder Währung seit dem Status der Kalenderdatenbank mit einer	Wirtschaftskalender

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	angegebenen change_id zu sortieren	
ceil	Gibt den ganzzahligen Wert, der der naechste von oben ist, zurück	Mathematische Funktionen
CharArrayToString	Wandelt Symbolcode (ansi) in Einsymbolzeile um	Datenverarbeitung
ChartApplyTemplate	Verwendet Schablone zum bezeichneten Chart der bezeichneten Datei	Operationen mit Charts
ChartClose	Schliesst den bezeichneten Chart ab	Operationen mit Charts
ChartFirst	Gibt den Identifier des ersten Client-Terminals zurück	Operationen mit Charts
ChartGetDouble	Gibt den Wert der entsprechenden Eigenschaft des bezeichneten Charts zurück	Operationen mit Charts
ChartGetInteger	Gibt den Ganzwert der entsprechenden Eigenschaft des bezeichneten Charts zurück	Operationen mit Charts
ChartGetString	Gibt den Stringwert der entsprechenden Eigenschaft des bezeichneten Charts zurück	Operationen mit Charts
ChartID	Gibt den Indikator des laufenden Charts zurück	Operationen mit Charts
ChartIndicatorAdd	Setzt Indikator mit dem angegebenen handle ins angegebene Fenster der graphischen Darstellung zu	Operationen mit Charts
ChartIndicatorDelete	Löscht Indikator mit angegebenem Namen aus dem angegebenen Chartfenster	Operationen mit Charts
ChartIndicatorGet	Gibt das Handle des Indikators mit dem angegebenen Kurznamen im angegebenen Chart-Fenster zurück	Operationen mit Charts
ChartIndicatorName	Bringt Kurzname des Indikators beim Nummer in der Indikatorenliste des Chartfensters zurück	Operationen mit Charts

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
ChartIndicatorsTotal	Bringt die Anzahl aller Indikatoren am angegebenen Chartfenster	Operationen mit Charts
ChartNavigate	Verschiebt den angegebenen Chart um die angegebene Anzahl der Bars in Bezug auf die angegebene Position des Charts	Operationen mit Charts
ChartNext	Bringt Identifikator des Charts, das dem bezeichneten Chart folgt, zurück	Operationen mit Charts
ChartOpen	Oeffnet einen neuen Chart mit dem bezeichneten Symbol und mit der bezeichneten Periode	Operationen mit Charts
CharToString	Umwandlung des Symbolcodes in die Einsymbolzeile	Datenverarbeitung
ChartPeriod	Gibt den Periodenwert des bezeichneten Chart zurück	Operationen mit Charts
ChartPriceOnDropped	Gibt die Preiskoordinate des Punktes zurück, wohin dieser Script oder Expert mit der Maus versetzt wurde	Operationen mit Charts
ChartRedraw	Ruft Zwangsumzeichnen des bezeichneten Charts auf	Operationen mit Charts
ChartSaveTemplate	Speichert aktuelle Chart-Einstellungen in einer Schablone mit einem bestimmten Namen	Operationen mit Charts
ChartScreenShot	Screenshot des angegebenen Charts in Format GIF, PNG oder BMP je nach angegebenen Erweiterung	Operationen mit Charts
ChartSetDouble	Ordnet double-Typ Wert der entsprechenden Eigenschaft des bezeichneten Charts zu	Operationen mit Charts
ChartSetInteger	Ordnet Ganzwert (datetime, int, color, bool oder char) der entsprechenden Eigenschaft des bezeichneten Charts zu	Operationen mit Charts
ChartSetString	Ordnet string-Typ Wert der entsprechenden Eigenschaft des bezeichneten Charts zu	Operationen mit Charts
ChartSetSymbolPeriod	Verändert die Werte des Symbols und der Periode des angegebenen	Operationen mit Charts

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Charts	
ChartSymbol	Bringt den Symbolnamen des bezeichneten Chart zurück	Operationen mit Charts
ChartTimeOnDropped	Bringt die Preiskoordinate des Punktes zurück, wohin dieser Expert oder Script mit der Maus versetzt wurde.	Operationen mit Charts
ChartTimePriceToXY	Konvertiert die Koordinaten des Charts aus der Darstellung von Zeit/Geld in die X- und Y-Koordinaten	Operationen mit Charts
ChartWindowFind	Gibt die Nummer des Subfensters, in dem Indikator war, zurück	Operationen mit Charts
ChartWindowOnDropped	Gibt die Nummer des Subfensters zurück, wohin der vorliegende Expert, Script, Objekt oder Anzeiger mit der Maus versetzt wurden sind	Operationen mit Charts
ChartXOnDropped	Gibt die X-Koordinate zurück, wohin dieser Expert oder Script mit der Maus versetzt wurde	Operationen mit Charts
ChartXYToTimePrice	Konvertiert die X- und Y-Koordinate des Charts in Zeit und Preis	Operationen mit Charts
ChartYOnDropped	Gibt die Y-Koordinate zurück, wohin dieser Expert oder Script mit der Maus versetzt wurde	Operationen mit Charts
CheckPointer	Gibt den Typ eines Objekts zurück	Allgemeine Funktionen
CLBufferCreate	Erstellt einen OpenCL-Puffer	Arbeit mit OpenCL
CLBufferFree	Löscht einen OpenCL-Puffer	Arbeit mit OpenCL
CLBufferRead	Liest einen OpenCL-Puffer in ein Array	Arbeit mit OpenCL
CLBufferWrite	Schreibt ein Array in einen OpenCL-Puffer	Arbeit mit OpenCL
CLContextCreate	Erstellt einen Kontext für OpenCL	Arbeit mit OpenCL
CLContextFree	Löscht den Kontext von OpenCL	Arbeit mit OpenCL
CLExecute	Führt OpenCL-Programm	Arbeit mit OpenCL

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
CLGetDeviceInfo	Erhielt die Geräteeigenschaft aus dem OpenCL-Treiber	Arbeit mit OpenCL
CLGetInfoInteger	Gibt den Wer des ganzzahligen Eigenschafts für ein OpenCL-Objekt oder Gerät zurück	Arbeit mit OpenCL
CLHandleType	Gibt den Typ des OpenCL-Handles als ein Wert aus Enumeration ENUM_OPENCL_HANDLE_TYPE zurück	Arbeit mit OpenCL
CLKernelCreate	Erstellt eine Funktion des Starts von OpenCL	Arbeit mit OpenCL
CLKernelFree	Löscht die Funktion des Starts von OpenCL	Arbeit mit OpenCL
CLProgramCreate	Erstellt ein OpenCL-Programm aus dem Quellcode	Arbeit mit OpenCL
CLProgramFree	Löscht das OpenCL-Programm	Arbeit mit OpenCL
CLSetKernelArg	Setzt den Parameter für die OpenCL-Funktion	Arbeit mit OpenCL
CLSetKernelArgMem	Setzt einen OpenCL-Puffer als ein Parameter der OpenCL-Funktion	Arbeit mit OpenCL
ColorToARGB	Konvertiert den Typ color in den Typ uint, um Farbdarstellung ARGB zu erhalten.	Datenverarbeitung
ColorToString	Wandelt den Farbenwert in die Zeile der Art "R,G,B" um	Datenverarbeitung
Comment	Gibt eine Meldung in der linken oberen Ecke des Preischarts aus	Allgemeine Funktionen
CopyBuffer	Bekommt Daten des angegebenen Puffers vom angegebenen Anzeiger im Feld	Zugang zu Zeitreihen und Indikatoren
CopyClose	Bekommt im Feld historische Daten über den Schlusspreis der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopyHigh	Bekommt im Feld historische Daten über den maximalen Preis der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
CopyLow	Bekommt im Feld historische Daten über den minimalen Preis der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopyOpen	Bekommt im Feld historische Daten über die Eroeffnungszeit der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopyRates	Bekommt historische Daten der Struktur Rates für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopyRealVolume	Bekommt im Feld historische Daten über Handelsvolumen der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopySpread	Bekommt im Feld historische Daten über Spreads für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopyTicks	Erhält für die aktuelle Arbeitssitzung angesammelten Ticks ins Array ticks_array	Zugang zu Zeitreihen und Indikatoren
CopyTickVolume	Bekommt im Feld historische Daten über Tickvolumen der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
CopyTime	Bekommt im Feld historische Daten über die Eroeffnungszeit der Bars für das angegebene Symbol und Periode	Zugang zu Zeitreihen und Indikatoren
cos	Gibt Kosinus zurück	Mathematische Funktionen
CryptDecode	Führt eine Rückumwandlung der Daten eines Arrays aus	Allgemeine Funktionen
CryptEncode	Wandelt Daten eines Quell-Arrays in ein Empfänger-Array anhand der angegebenen Methode um	Allgemeine Funktionen
CustomSymbolCreate	Erstellt ein benutzerdefiniertes Symbol mit dem angegebenen Namen in der angegebenen Gruppe	Benutzerdefinierte Symbole
CustomSymbolDelete	Löscht ein benutzerdefinierte Symbol mit dem angegebenen	Benutzerdefinierte Symbole

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Namen	
CustomSymbolSetInteger	Setzt den Wert einer Eigenschaft vom ganzzahligen Typ für das benutzerdefinierte Symbol	Benutzerdefinierte Symbole
CustomSymbolSetDouble	Setzt den Wert einer Eigenschaft vom reellen Typ für das benutzerdefinierte Symbol	Benutzerdefinierte Symbole
CustomSymbolSetString	Setzt den Wert einer Eigenschaft vom String-Typ für das benutzerdefinierte Symbol	Benutzerdefinierte Symbole
CustomSymbolSetMarginRate	Setzt den Koeffizienten der Margin je nach Typ und Richtung einer Order für das benutzerdefinierte Symbol	Benutzerdefinierte Symbole
CustomSymbolSetSessionQuote	Setzt den Anfang und das Ende der angegebenen Notierungssitzung für das angegebene Symbol und Wochentag	Benutzerdefinierte Symbole
CustomSymbolSetSessionTrade	Setzt den Anfang und das Ende der angegebenen Notierungssitzungitzung für das angegebene Symbol und Wochentag	Benutzerdefinierte Symbole
CustomRatesDelete	Löscht alle Balken im angegebenen Zeitintervall aus der Preishistorie des benutzerdefinierten Symbols	Benutzerdefinierte Symbole
CustomRatesReplace	Ersetzt die komplette Preishistorie des benutzerdefinierten Symbols im angegebenen Zeitintervall durch die Daten aus dem Array vom Typ MqlRates	Benutzerdefinierte Symbole
CustomRatesUpdate	Fügt der Historie des benutzerdefinierten Symbols fehlende Balken hinzu und ersetzt die vorhandenen Balken durch die Daten aus dem Array vom Typ MqlRates	Benutzerdefinierte Symbole
CustomTicksAdd	Fügt Daten aus einem Array vom Typ MqlTick in die Preishistorie	Benutzerdefinierte Symbole

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	eines benutzerdefinierten Symbols hinzu. Das benutzerdefinierte Symbol muss im Fenster MarketWatch (Marktübersicht) ausgewählt werden	
CustomTicksDelete	Löscht alle Balken im angegebenen Zeitintervall aus der Preishistorie des benutzerdefinierten Symbols	Benutzerdefinierte Symbole
CustomTicksReplace	Ersetzt die komplette Preishistorie des benutzerdefinierten Symbols im angegebenen Zeitintervall durch die Daten aus dem Array vom Typ MqlTick	Benutzerdefinierte Symbole
CustomBookAdd	Überträgt den Status der Markttiefe (Depth of Market) für ein nutzerdefiniertes Symbol	Benutzerdefinierte Symbole
DatabaseOpen	Öffnet oder erstellt die angegebene Datenbankdatei	Arbeiten mit der Datenbank
DatabaseClose	Schließt die Datenbank	Arbeiten mit der Datenbank
DatabaseImport	Importieren von Daten aus einer Datei in eine Tabelle	Arbeiten mit der Datenbank
DatabaseExport	Exportiert eine Tabelle oder das Ergebnis einer SQL-Anfrage in eine CSV-Datei	Arbeiten mit der Datenbank
DatabasePrint	Druckt eine Tabelle oder ein Ergebnis der SQL-Anfrage im Experten-Journal aus	Arbeiten mit der Datenbank
DatabaseTableExists	Prüft das Vorhandensein der Tabelle in einer Datenbank	Arbeiten mit der Datenbank
DatabaseExecute	Ausführung einer Anfrage an die angegebene Datenbank	Arbeiten mit der Datenbank
DatabasePrepare	Erstellt das Handle für die Anfragen, die durch DatabaseRead() ausgeführt werden kann	Arbeiten mit der Datenbank
DatabaseReset	Rücksetzen einer Anfrage, wie nach dem Aufruf von DatabasePrepare()	Arbeiten mit der Datenbank

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
DatabaseBind	Setzt einen Parameterwert in einer Anfrage	Arbeiten mit der Datenbank
DatabaseBindArray	Setzt einen Parameterarray als Parameterwert	Arbeiten mit der Datenbank
DatabaseRead	Wechselt zum nächsten Eintrag als Ergebnis einer Anfrage	Arbeiten mit der Datenbank
DatabaseReadBind	Wechselt zum nächsten Datensatz und liest aus ihm Daten in die Struktur	Arbeiten mit der Datenbank
DatabaseFinalize	Entfernt eine Anfrage, die durch DatabasePrepare() erstellt wurde	Arbeiten mit der Datenbank
DatabaseTransactionBegin	Startet die Ausführung der Transaktion	Arbeiten mit der Datenbank
DatabaseTransactionCommit	Schließt die Ausführung der Transaktion ab	Arbeiten mit der Datenbank
RDatabaseTransactionRollback	Zurücksetzen der Transaktionen	Arbeiten mit der Datenbank
DatabaseColumnsCount	Abrufen der Felderanzahl einer Anfrage	Arbeiten mit der Datenbank
DatabaseColumnName	Abrufen des Feldnamens nach dem Index	Arbeiten mit der Datenbank
DatabaseColumnType	Abrufen des Feldtyps nach dem Index	Arbeiten mit der Datenbank
DatabaseColumnSize	Abrufen der Feldgröße in Bytes	Arbeiten mit der Datenbank
DatabaseColumnText	Abrufen des Feldwerts als Zeichenkette aus dem aktuellen Datensatz	Arbeiten mit der Datenbank
DatabaseColumnInteger	Abrufen des Integer-Werts aus dem aktuellen Datensatz	Arbeiten mit der Datenbank
DatabaseColumnLong	Abrufen des Long-Werts aus dem aktuellen Datensatz	Arbeiten mit der Datenbank
DatabaseColumnDouble	Abrufen des Double-Wertes aus dem aktuellen Datensatz	Arbeiten mit der Datenbank
DatabaseColumnBlob	Abrufen des Feldwerts als Array aus dem aktuellen Datensatz	Arbeiten mit der Datenbank
DebugBreak	Programmhaltepunkt beim Debugging	Allgemeine Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
Digits	Gibt die Anzahl der Dezimalzeichen, die Genauigkeit der Berechnung des Preiswertes des laufenden Charts bestimmt	Prüfung des Status
DoubleToString	Umwandlung des numerishen Wertes in die Textzeile mit der angegebenen Genauigkeit	Datenverarbeitung
DXContextCreate	Erzeugt einen grafischen Kontext zum Rendern von Frames einer bestimmten Größe.	Arbeiten mit DirectX
DXContextSetSize	Ändert die Frame-Größe eines mit DXContextCreate() erstellten Grafikkontextes.	Arbeiten mit DirectX
DXContextSetSize	Liefert die Rahmengröße eines mit DXContextCreate() erstellten Grafikkontextes.	Arbeiten mit DirectX
DXContextClearColors	Weist eine bestimmte Farbe allen Pixeln für den Rendering-Puffer.	Arbeiten mit DirectX
DXContextClearDepth	Löscht den Tiefenpuffer	Arbeiten mit DirectX
DXContextGetColors	Ruft ein Bild einer bestimmten Größe und eines bestimmten Offsets aus einem Grafikkontext ab.	Arbeiten mit DirectX
DXContextGetDepth	Ruft den Tiefenpuffer eines gerenderten Frames ab	Arbeiten mit DirectX
DXBufferCreate	Erstellt einen Puffer eines bestimmten Typs basierend auf einem Daten-Array	Arbeiten mit DirectX
DXTextureCreate	Erstellt eine 2D-Textur aus einem Rechteck mit einer bestimmten Größe, das aus einem übergebenen Bild ausgeschnitten wurde.	Arbeiten mit DirectX
DXInputCreate	Erstellt die Eingaben des Shaders	Arbeiten mit DirectX
DXInputSet	Setzt die Eingaben des Shaders	Arbeiten mit DirectX
DXShaderCreate	Erstellt einen Shader eines bestimmten Typs	Arbeiten mit DirectX
DXShaderSetLayout	Legt das Vertex-Layout für den Vertex-Shader fest	Arbeiten mit DirectX

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
DXShaderInputsSet	Setzt die Eingaben des Shaders	Arbeiten mit DirectX
DXShaderTexturesSet	Legt die Texturen des Shaders fest	Arbeiten mit DirectX
DXDraw	Rendert die Eckpunkte des in DXBufferSet() gesetzten Vertex-Puffers	Arbeiten mit DirectX
DXDrawIndexed	Rendert grafische Primitive, die durch den Indexpuffer von DXBufferSet() beschrieben werden	Arbeiten mit DirectX
DXPrimiveTopologySet	Setzt den Typ der Primitive für das Rendern mit DXDrawIndexed()	Arbeiten mit DirectX
DXBufferSet	Setzt einen Puffer für das aktuelle Rendering	Arbeiten mit DirectX
DXShaderSet	Setzt einen Shader für das Rendern	Arbeiten mit DirectX
DXHandleType	Liefert einen Handle-Typ zurück	Arbeiten mit DirectX
DXRelease	Gibt ein Handle frei	Arbeiten mit DirectX
EnumToString	Umwandlung der Enumerationwert beliebigen Typs in ein String	Datenverarbeitung
EventChartCustom	Generiert das Benutzerereignis für den angegebenen Chart.	Arbeit mit Ereignissen
EventKillTimer	Setzt auf dem laufenden Chart die Zeitgebergenerierung von Ereignissen still	Arbeit mit Ereignissen
EventSetMillisecondTimer	Startet den Generator der hochauflösenden Ereignissetimer mit einer Periode von weniger als 1 Sekunde für den aktuellen Chart	Arbeit mit Ereignissen
EventSetTimer	Läuft Generierer der Timerereignisse mit dem angegebenen Abruf für den laufenden Chart ab	Arbeit mit Ereignissen
exp	Gibt Exponente der Zahl zurück	Mathematische Funktionen
ExpertRemove	Stoppt den Expert Advisor und entfernt ihn vom Chart	Allgemeine Funktionen
fabs	Gibt absoluten Wert (Modulus) der angegebenen Zahl zurück	Mathematische Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
FileClose	Schließt die früher geöffnete Datei	Dateioperationen
FileCopy	Kopiert die Ausgangsdatei aus dem lokalen oder gesamten Ordner in eine andere Datei	Dateioperationen
FileDelete	Entfernt die angegebene Datei	Dateioperationen
FileFindClose	Schließt die Suche Handle	Dateioperationen
FileFindFirst	Fängt die Suche der Dateien im entsprechenden Verzeichnis nach dem angegebenen Filter an	Dateioperationen
FileFindNext	Setzt die Suche fort, die durch die Funktion FileFindFirst() angefangen wurde	Dateioperationen
FileFlush	Speicher auf der Platte alle Daten, die im Dateipuffer Eingabe-Ausgabe noch bleiben	Dateioperationen
FileGetInteger	Ruft eine ganzzahlige Eigenschaft der Datei	Dateioperationen
FileIsEnding	Bestimmt das Ende der Datei im während des Lesens	Dateioperationen
FileIsExist	Prüft, ob die Datei existiert	Dateioperationen
FileIsLineEnding	Bestimmt Zeilenende während des Lesens	Dateioperationen
FileMove	Bewegt oder benennt die Datei um	Dateioperationen
FileOpen	Öffnet die Datei mit dem angegebenen Namen und mit der angegebenen Flagge	Dateioperationen
FileReadArray	Liest Arrays verschiedener Typen außer Zeilentypen (es kann ein Strukturarray sein ohne Zeilen und die dynamische Arrays) aus der binären Datei von der laufenden Position des Dateiindikators	Dateioperationen
FileReadBool	Liest aus der Datei des Typs CSV die Zeile von der laufenden Position bis zum Begrenzer (oder bis zum Ende der Textzeile) und wandelt die gelesene Zeile in den Wert des Typs bool	Dateioperationen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
FileReadDatetime	Liest aus der Datei des Typs CSV die Zeile eines der Formate: "YYYY.MM.DD HH:MI:SS", "YYYY.MM.DD" oder "HH:MI:SS" - und wandelt sie in den Wert des Typs datetime um	Dateioperationen
FileReadDouble	Liest die Zahl der doppelten Genauigkeit mit dem Fließpunkt (double) aus der binären Datei von der laufenden Position des Dateiindikators	Dateioperationen
FileReadFloat	Liest den Wert float von der laufenden Position des Dateianzeigers	Dateioperationen
FileReadInteger	Liest den Wert des Typs int, short oder charvon der laufenden Position des Dateianzeigers abhängig von der angegebenen Länge in Bytes	Dateioperationen
FileReadLong	Liest den Wert des Typs long von der laufenden Position des Dateianzeigers	Dateioperationen
FileReadNumber	Liest aus der Datei des Typs CSV die Zeile von der laufenden Position zum Begrenzer (oder bis zum Ende der Textzeile) und wandelt die gelesene Zeile in den Wert des Typs double um	Dateioperationen
FileReadString	Liest die Zeile aus der Datei von der laufenden Position des Dateianzeigers	Dateioperationen
FileReadStruct	Liest Inhalt aus der binären Datei in die Struktur, die als Parameter übertragen wurde	Dateioperationen
FileSeek	Bewegt die Position des Dateiindikators um die angegebene Zahl der Bytes gegen die angegebene Position	Dateioperationen
FileSize	Gibt die Größe der entsprechenden geöffneten Datei zurück	Dateioperationen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
FileTell	Gibt die laufende Position des Indikators der entsprechenden geöffneten Datei zurück	Dateioperationen
FileWrite	Schreibt Daten in die Datei des Typs CSV oder TXT	Dateioperationen
FileWriteArray	Schreibt in die Datei des Typs BIN Arrays aller Typen außer Zeilentypen	Dateioperationen
FileWriteDouble	Schreibt den Parameterwert des Typs double von der angegebenen Position des Dateianzeigers in die binäre Datei	Dateioperationen
FileWriteFloat	Schreibt den Parameterwert des Typs float von der angegebenen Position des Dateianzeigers	Dateioperationen
FileWriteInteger	Schreibt den Parameterwert des Typs int in die binäre Datei von der laufenden Position des Dateianzeigers	Dateioperationen
FileWriteLong	Schreibt den Parameterwert des Typs long in die binäre Datei von der laufenden Position des Dateianzeigers	Dateioperationen
FileWriteString	Schreibt den Parameterwert des Typs string in die Datei des Typs BIN oder TXT von der laufenden Position des Dateianzeigers	Dateioperationen
FileWriteStruct	Schreibt den Inhalt der Struktur, die als Parameter übertragen wurde, in die binäre Datei von der laufenden Position des Dateianzeigers	Dateioperationen
floor	Gibt den ganzzahligen Wert, der der naechste von unten ist, zurück	Mathematische Funktionen
fmax	Gibt maximalen Wert der zwei numerischen Werte	Mathematische Funktionen
fmin	Gibt minimalen Wert der zwei numerischen Werte	Mathematische Funktionen
fmod	Gibt reellen überbleibsel der Division der zwei Zahlen	Mathematische Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
FolderClean	Entfernt alle Dateien im angegebenen Ordner	Dateioperationen
FolderCreate	Erzeugt ein Verzeichnis im Ordner Files (abhängig von dem Wert common_flag)	Dateioperationen
FolderDelete	Entfernt das angegebene Verzeichnis. Wenn der Ordner nicht leer ist, kann er entfernt werden	Dateioperationen
FrameAdd	Fügt einen Frame mit Daten hinzu	Arbeit mit Ergebnisse der Optimierung
FrameFilter	Setzt den Filter der Lesung des Frames und verschiebt den Zeiger auf den Anfang	Arbeit mit Ergebnisse der Optimierung
FrameFirst	Verschiebt einen Zeiger der Lesung des Frames auf den Anfang und rücksetzt den zuvor angegebenen Filter	Arbeit mit Ergebnisse der Optimierung
FrameInputs	Empfängt Input-Parameter, auf die der Frame gebildet ist	Arbeit mit Ergebnisse der Optimierung
FrameNext	Liest einen Frame und verschiebt den Zeiger auf das nächste	Arbeit mit Ergebnisse der Optimierung
GetLastError	Gibt den wert des letzten Fehlers zurück	Prüfung des Status
GetPointer	Gibt den Anzeiger des Objektes zurück.	Allgemeine Funktionen
GetTickCount	Gibt die Anzahl der Millisekunden zurück, die seit dem Start des Systems vergangen sind	Allgemeine Funktionen
GlobalVariableCheck	Prueft, ob die globale Variable mit dem angegebenen Namen existiert	Globalvariablen des Client-Terminals
GlobalVariableDel	entfernt globale Variable	Globalvariablen des Client-Terminals
GlobalVariableGet	Fordert den Wert der globalen Variable an	Globalvariablen des Client-Terminals
GlobalVariableName	Gibt den Namen der globalen Variable nach der laufenden Nummer in der Liste der globalen Variable	Globalvariablen des Client-Terminals

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
GlobalVariablesDeleteAll	Entfernt globale Variablen mit dem angegebenen Praefix im Namen	Globalvariablen des Client-Terminals
GlobalVariableSet	Stellt den neuen Wert der globalen Variable fest	Globalvariablen des Client-Terminals
GlobalVariableSetOnCondition	Stellt den neuen Wert der existierenden globalen Variable nach Bedingung	Globalvariablen des Client-Terminals
GlobalVariablesFlush	Speichert den Inhalt aller Variablen auf der Platte zwangsmaessig	Globalvariablen des Client-Terminals
GlobalVariablesTotal	Gibt die ganze Zahl der globalen Variablen zurück	Globalvariablen des Client-Terminals
GlobalVariableTemp	Stellt den neuen wert der globalen Variable fest, die nur während der laufenden Terminalsitzung existiert	Globalvariablen des Client-Terminals
GlobalVariableTime	Gibt die Zeit des letzten Zuganges zur globalen Variable zurück	Globalvariablen des Client-Terminals
HistoryDealGetDouble	Gibt die angeforderte Eigenschaft des Deals in der Geschichte zurück (double)	Handelsfunktionen
HistoryDealGetInteger	Bringt die angeforderte Eigenschaft des Deals in der Geschichte zurück (datetime oder int)	Handelsfunktionen
HistoryDealGetString	Bringt die angeforderte Eigenschaft des Deals in der Geschichte zurück (string)	Handelsfunktionen
HistoryDealGetTicket	Wählt das Deal für die weitere Verarbeitung und bringt Ticket des Deals in der Geschichte zurück	Handelsfunktionen
HistoryDealSelect	Wählt das Deal in der Geschichte für ihre weitere Anwendung durch entsprechende Funktionen	Handelsfunktionen
HistoryDealsTotal	Bringt die Dealszahl in der Geschichte zurück	Handelsfunktionen
HistoryOrderGetDouble	Bringt die angeforderte Eigenschaft der Order in der Geschichte zurück (double)	Handelsfunktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
HistoryOrderGetInteger	Bringt die angeforderte Eigenschaft der Order in der Geschichte zurück (datetime oder int)	Handelsfunktionen
HistoryOrderGetString	Bringt die angeforderte Eigenschaft der Order in der Geschichte zurück (string)	Handelsfunktionen
HistoryOrderGetTicket	Bringt Ticket der entsprechenden Order in der Geschichte zurück	Handelsfunktionen
HistoryOrderSelect	Wählt Order in der Geschichte für die weitere Arbeit damit	Handelsfunktionen
HistoryOrdersTotal	Bringt die Orderzahl in der Geschichte zurück	Handelsfunktionen
HistorySelect	Fordert die Deals- und Ordergeschichte für die angegebene Periode der Severzeit an	Handelsfunktionen
HistorySelectByPosition	Fordert die Geschichte der Deals und der Order mit dem angegebenen Identifikator der Position an	Handelsfunktionen
iBars	Gibt die Anzahl der Balken in der Historie nach dem entsprechenden Symbol und Zeitrahmen zurück	Zugang zu Zeitreihen und Indikatoren
iBarShift	Gibt den Index des Balkens zurück, der der angegebenen Zeit entspricht	Zugang zu Zeitreihen und Indikatoren
iClose	Gibt den Close-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iHigh	Gibt den High-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iHighest	Gibt den Index des größten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
iLow	Gibt den Low-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iLowest	Gibt den Index des kleinsten gefundenen Wertes (Verschiebung relativ zum aktuellen Balken) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iOpen	Gibt den Open-Preis des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iTime	Gibt den Zeitpunkt der Eröffnung eines Balkens (der mithilfe des Parameters shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iTickVolume	Gibt den Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iRealVolume	Gibt den Wert des echten Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iVolume	Gibt den Wert des Tick-Volumens des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iSpread	Gibt den Wert des Spreads des Balkens (der durch den Parameter shift vorgegeben wurde) des entsprechenden Charts zurück	Zugang zu Zeitreihen und Indikatoren
iAD	Accumulation/Distribution	Technische Indikatoren
iADX	Average Directional Index	Technische Indikatoren
iADXWilder	Average Directional Index by Welles Wilder	Technische Indikatoren
iAlligator	Alligator	Technische Indikatoren

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
iAMA	Adaptive Moving Average	Technische Indikatoren
iAO	Awesome Oscillator	Technische Indikatoren
iATR	Average True Range	Technische Indikatoren
iBands	Bollinger Bands®	Technische Indikatoren
iBearsPower	Bears Power	Technische Indikatoren
iBullsPower	Bulls Power	Technische Indikatoren
iBWMFI	Market Facilitation Index by Bill Williams	Technische Indikatoren
iCCI	Commodity Channel Index	Technische Indikatoren
iChaikin	Chaikin Oscillator	Technische Indikatoren
iCustom	Benutzerindikator	Technische Indikatoren
iDEMA	Double Exponential Moving Average	Technische Indikatoren
iDeMarker	DeMarker	Technische Indikatoren
iEnvelopes	Envelopes	Technische Indikatoren
iForce	Force Index	Technische Indikatoren
iFractals	Fractals	Technische Indikatoren
iFrAMA	Fractal Adaptive Moving Average	Technische Indikatoren
iGator	Gator Oscillator	Technische Indikatoren
iIchimoku	Ichimoku Kinko Hyo	Technische Indikatoren
iMA	Moving Average	Technische Indikatoren
iMACD	Moving Averages Convergence-Divergence	Technische Indikatoren
iMFI	Money Flow Index	Technische Indikatoren
iMomentum	Momentum	Technische Indikatoren
IndicatorCreate	Gibt handle des angegebenen technischen Anzeigers zurück, das vom Feld des Typs Parameter MqlParam erzeugt wurde	Zugang zu Zeitreihen und Indikatoren
IndicatorParameters	Basierend auf das angegebenen Handle gibt Anzahl von Eingabeparametern des Indikators, sowie die Werte und Type der Parameter zurück	Zugang zu Zeitreihen und Indikatoren

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
IndicatorRelease	Entfernt handle des Anzeigers und befreit den Berechnungsteol des Anzeigers, wenn niemand den verwendet	Zugang zu Zeitreihen und Indikatoren
IndicatorSetDouble	Gibt den Wert des Indikators vor, der den Typ double hat	Benutzerindikatoren
IndicatorSetInteger	Gibt Eigenschaftswert des Indikators vor, der den Typ int hat	Benutzerindikatoren
IndicatorSetString	Gibt Eigenschaftswert des Indikators vor, der den Typ string hat	Benutzerindikatoren
IntegerToString	Umwandlung des ganzzahligen Wertes in die Zeile der vorgegebenen Laenge	Datenverarbeitung
iOBV	On Balance Volume	Technische Indikatoren
iOsMA	Moving Average of Oscillator (MACD histogram)	Technische Indikatoren
iRSI	Relative Strength Index	Technische Indikatoren
iRVI	Relative Vigor Index	Technische Indikatoren
iSAR	Parabolic Stop And Reverse System	Technische Indikatoren
IsStopped	Gibt true zurück, wenn mql5-Programm befielt, das Programm zu beenden	Prüfung des Status
iStdDev	Standard Deviation	Technische Indikatoren
iStochastic	Stochastic Oscillator	Technische Indikatoren
iTEMA	Triple Exponential Moving Average	Technische Indikatoren
iTriX	Triple Exponential Moving Averages Oscillator	Technische Indikatoren
iVIDyA	Variable Index Dynamic Average	Technische Indikatoren
iVolumes	Volumes	Technische Indikatoren
iWPR	Williams' Percent Range	Technische Indikatoren
log	Gibt Napierschen Logarithmus zurück	Mathematische Funktionen
log10	Gibt Dezimallogarithmus zurück	Mathematische Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
MarketBookAdd	Sorgt für Eroeffnung von DOM für das gewaehlte Symbol, und subskribiert auf Nachrichten über DOM Veränderung	Marktinformation erhalten
MarketBookGet	Gibt Feld des Typs Strukturen MqlBookInfo, das Aufzeichnungen von DOM des angegebenen Symbols enthält	Marktinformation erhalten
MarketBookRelease	Sorgt für Schliessen von DOM für das gewaehlte Symbol und annulliert Subskription auf Nachrichten über DOM Veränderung	Marktinformation erhalten
MathAbs	Gibt absoluten Wert (Modulus) der angegebenen Zahl zurück	Mathematische Funktionen
MathArccos	Gibt arccos x in Radianen zurück	Mathematische Funktionen
MathArcsin	Gibt arcsin x in Radianen zurück	Mathematische Funktionen
MathArctan	Gibt arctg x in Radianen zurück	Mathematische Funktionen
MathCeil	Gibt den ganzzahligen Wert, der der naechste von oben ist, zurück	Mathematische Funktionen
MathCos	Gibt Kosinus zurück	Mathematische Funktionen
MathExp	Gibt Exponente der Zahl zurück	Mathematische Funktionen
MathFloor	Gibt den ganzzahligen Wert, der der naechste von unten ist, zurück	Mathematische Funktionen
MathIsValidNumber	Prueft Richtigkeit der Realzahl	Mathematische Funktionen
MathLog	Gibt Napierschen Logarithmus zurück	Mathematische Funktionen
MathLog10	Gibt Dezimallogarithmus zurück	Mathematische Funktionen
MathMax	Gibt maximalen Wert der zwei numerischen Werte	Mathematische Funktionen
MathMin	Gibt minimalen Wert der zwei numerischen Werte	Mathematische Funktionen
MathMod	Gibt reellen überbleibsel der Division der zwei Zahlen	Mathematische Funktionen
MathPow	Erhebt die Basis in die angegebene Potenz	Mathematische Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
MathRand	Gibt die pseudozufällige Zahl im Bereich von 0 bis 32767 zurück	Mathematische Funktionen
MathRound	Rundet die Zahl bis die naechste Ganzzahl	Mathematische Funktionen
MathSin	Gibt sin der Zahl zurück	Mathematische Funktionen
MathSqrt	Gibt Quadratwurzel zurück	Mathematische Funktionen
MathSrand	Stellt Anfangszustand für Geberieren der pseudozufälligen Realzahlen	Mathematische Funktionen
MathTan	Gibt tg der Zahl zurück	Mathematische Funktionen
MessageBox	Erzeugt und exponiert Meldungsfenster und auch verwaltet es	Allgemeine Funktionen
MQLInfoInteger	Gibt den Wert des ganzzahligen Typs der entsprechenden Eigenschaft des eingesetzten mql5-Programms zurück	Prüfung des Status
MQLInfoString	Gibt den Wert des Typs string der entsprechenden Eigenschaft des eingesetzten mql5-Programms	Prüfung des Status
MT5Initialize	Stellt eine Verbindung mit dem MetaTrader 5 Terminal her	MetaTrader für Python
MT5Shutdown	Schließt die vorher hergestellte Verbindung zu MetaTrader 5 Terminal wieder	MetaTrader für Python
MT5TerminalInfo	Abfrage des Status' und der Parameter des verbundenen MetaTrader 5 Terminals	MetaTrader für Python
MT5Version	Rückgabe der Version des MetaTrader 5 Terminals	MetaTrader für Python
MT5CopyRatesFrom	Abrufen der Bars vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Datum	MetaTrader für Python
MT5CopyRatesFromPos	Abrufen der Bars vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Index	MetaTrader für Python
MT5CopyRatesRange	Abrufen der Bars der angegebenen Zeitspanne vom MetaTrader 5 Terminal	MetaTrader für Python

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
MT5CopyTicksFrom	Abrufen der Ticks vom MetaTrader 5 Terminal, beginnend mit dem angegebenen Datum	MetaTrader für Python
MT5CopyTicksRange	Abrufen der Ticks der angegebenen Zeitspanne vom MetaTrader 5 Terminal	MetaTrader für Python
NormalizeDouble	Runden der Zahl mit dem Fliesspunkt bis der angegebenen Genauigkeit	Datenverarbeitung
ObjectCreate	Erzeugt das Objekt des vorgegebenen Typs auf dem vorgegebenen Chart	Graphische Objekte
ObjectDelete	Entfernt das Objekt mit dem angegebenen Namen vom angegebenen Chart (vom angegebenen Subfenster des Charts)	Graphische Objekte
ObjectFind	Sucht Objekt mit dem angegebenen Identifikator nach dem Namen	Graphische Objekte
ObjectGetDouble	Gibt den Wert des Typs double der entsprechenden Eigenschaft des Objekts zurück.	Graphische Objekte
ObjectGetInteger	Gibt den ganzzahligen Wert der entsprechenden Eigenschaft des Objekts zurück.	Graphische Objekte
ObjectGetString	Gibt den Wert des Typs string der entsprechenden Eigenschaft des Objekts	Graphische Objekte
ObjectGetTimeByValue	Gibt Zeitwert für den angegebenen Preiswert des Objekts zurück	Graphische Objekte
ObjectGetValueByTime	Gibt Preiswert des Objekts für die angegebene Zeit zurück	Graphische Objekte
ObjectMove	Verändert die Koordinaten des angegebenen Bezugspunktes des Objekts.	Graphische Objekte
ObjectName	Gibt den Namen des Objekts des entsprechenden Typs im	Graphische Objekte

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	angegebenen Chart zurück (im angegebenen Fenster des Charts)	
ObjectsDeleteAll	Entfernt alle Objekte des angegebenen Typs vom angegebenen Chart (vom angegebenen Subfenster des Charts)	Graphische Objekte
ObjectSetDouble	Stellt den wert der entsprechenden Eigenschaft des Objekts ein	Graphische Objekte
ObjectSetInteger	Stellt den wert der entsprechenden Eigenschaft des Objekts ein	Graphische Objekte
ObjectSetString	Stellt den wert der entsprechenden Eigenschaft des Objekts ein	Graphische Objekte
ObjectsTotal	Gibt die Anzahl der Objekte des angegebenen Typs im angegebenen Chart (angegebenen Subfenster des Charts) zurück.	Graphische Objekte
OnStart	Die Funktion wird aufgerufen, wenn das Ereignis Start auftritt, um Aktionen auszuführen, die im Skript aufgeführt sind.	Ereignisbehandlung
OnInit	Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis Init eintritt, um ein gestartetes MQL5-Programm zu initialisieren.	Ereignisbehandlung
OnDeinit	Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis Deinit auftritt, um ein laufendes MQL5-Programm zu de-initialisieren.	Ereignisbehandlung
OnTick	Die Funktion wird von EAs aufgerufen, wenn das Ereignis NewTick eintritt, um einen neuen Tick abzuarbeiten.	Ereignisbehandlung
OnCalculate	Die Funktion wird von Indikatoren aufgerufen, wenn das Ereignis Calculate eintritt, um	Ereignisbehandlung

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Preisdatenänderungen abzuarbeiten.	
OnTimer	Die Funktion wird von Indikatoren und EAs während des periodisches Ereignis Timer aufgerufen, das vom Terminal in festen Zeitintervallen erzeugt wird.	Ereignisbehandlung
OnTrade	Die Funktion wird von EAs während des Trade Ereignisses aufgerufen, das am Ende einer Handelsoperation auf einem Handelsserver generiert wird.	Ereignisbehandlung
OnTradeTransaktion	Die Funktion wird von EAs aufgerufen, wenn das Ereignis TradeTransaction auftritt, um die Ergebnisse einer Handelsanfrage zu verarbeiten	Ereignisbehandlung
OnBookEvent	Die Funktion wird von EAs aufgerufen, wenn das Ereignis BookEvent auftritt, um Änderungen in der Markttiefe zu verarbeiten.	Ereignisbehandlung
OnChartEvent	Die Funktion wird von Indikatoren und EAs aufgerufen, wenn das Ereignis ChartEvent auftritt, um Änderungen im Chart zu verarbeiten, die von einem Benutzer oder einem MQL5-Programm vorgenommen wurden.	Ereignisbehandlung
OnTester	Die Funktion wird von EAs aufgerufen, wenn das Ereignis Tester auftritt, um notwendige Aktionen durchzuführen, nachdem ein EA mit historischen Daten getestet wurde.	Ereignisbehandlung
OnTesterInit	Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterInit auftritt, um notwendige Aktionen vor der Optimierung im Strategie-Tester durchzuführen.	Ereignisbehandlung
OnTesterDeinit	Die Funktion wird von EAs aufgerufen, wenn das Ereignis TesterDeinit nach der EA-	Ereignisbehandlung

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Optimierung im Strategie-Tester auftritt.	
OnTesterPass	Die Funktion wird von EAs aufgerufen, wenn TesterPass auftritt, um die Ankunft eines neuen Datenrahmens während der EA-Optimierung im Strategie-Tester zu handhaben.	Ereignisbehandlung
OrderCalcMargin	Berechnet die Größe der Marge, die für den angegebenen Typ der Order notwendig ist, in der Währung des Kontos	Handelsfunktionen
OrderCalcProfit	Berechnet die Größe des Gewinns aufgrund der übertragenen Parameter, in der Währung des Kontos	Handelsfunktionen
OrderCheck	Prüft, ob es genug Geld ist, um die notwendige Handelsoperation auszuführen.	Handelsfunktionen
OrderGetDouble	Bringt die angeforderte Eigenschaft der Order zurück (double)	Handelsfunktionen
OrderGetInteger	Bringt die angeforderte Eigenschaft der Order zurück (datetime oder int)	Handelsfunktionen
OrderGetString	Bringt die angeforderte Eigenschaft der Order zurück (string)	Handelsfunktionen
OrderGetTicket	Bringt Ticket der entsprechenden Order zurück	Handelsfunktionen
OrderSelect	Wählt Order für die weitere Arbeit damit	Handelsfunktionen
OrderSend	Sendet Handelsanforderungen auf das Sever	Handelsfunktionen
OrderSendAsync	Asynchron sendet Handelsanforderungen, ohne auf die Antwort des Handelsservers zu warten	Handelsfunktionen
OrdersTotal	Bringt die Zahl der Ordern zurück	Handelsfunktionen
ParameterGetRange	Empfängt die Information über den Bereich der Werte und den	Arbeit mit Ergebnisse der Optimierung

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Schritt der Veränderung für input-Variable bei der Optimierung eines Expert Advisors in Strategie-Tester	
ParameterSetRange	Erstellt Regeln für die Verwendung von input-Variable bei der Optimierung eines Expert Advisors in Strategie-Tester: Wert, Schritt der Veränderung, Anfangs-und Endwerte	Arbeit mit Ergebnisse der Optimierung
Period	Gibt den Wert von Timeframe des laufenden Charts zurück	Prüfung des Status
PeriodSeconds	Gibt Sekundenanzahl in der Periode zurück	Allgemeine Funktionen
PlaySound	Spielt Audiodatei ab	Allgemeine Funktionen
PlotIndexGetInteger	Gibt Zeilenwert des Indikators zurück, der den ganzzahligen Typ hat	Benutzerindikatoren
PlotIndexSetDouble	Gibt Zeilenwert des Indikators vor, der den Typ double hat	Benutzerindikatoren
PlotIndexSetInteger	Gibt Zeilenwert des Indikators vor,der den Typ int hat	Benutzerindikatoren
PlotIndexSetString	Gibt Zeilenwert des Indikators vor, der den Typ string hat	Benutzerindikatoren
Point	Gibt den Wert des Punktes des laufenden Instrumentes in der Quotationswaehrung zurück.	Prüfung des Status
PositionGetDouble	Bringt die angeforderte Eigenschaft der offenen Position zurück (double)	Handelsfunktionen
PositionGetInteger	Bringt die angeforderte Eigenschaft der offenen Position zurück (datetime oder int)	Handelsfunktionen
PositionGetString	Bringt die angeforderte Eigenschaft der offenen Position zurück (string)	Handelsfunktionen
PositionGetSymbol	Bringt das Symbol der entsprechenden offenen Position zurück	Handelsfunktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
PositionSelect	Wählt die offene Position für die weitere Arbeit damit	Handelsfunktionen
PositionsTotal	Bringt Anzahl der offenen Positionen zurück	Handelsfunktionen
pow	Erhebt die Basis in die angegebene Potenz	Mathematische Funktionen
Print	Gibt die Meldung ins Magazin aus	Allgemeine Funktionen
PrintFormat	Formatiert und druckt Schriftsatz und Werte in die Log-Datei aus entsprechend dem vorgegebenen Format	Allgemeine Funktionen
rand	Gibt die pseudozufällige Zahl im Bereich von 0 bis 32767 zurück	Mathematische Funktionen
ResetLastError	Stellt den Wert der vorbestimmten Variable _LastError auf Null	Allgemeine Funktionen
ResourceCreate	Erstellt eine Ressource eines Bildes basierend auf dem Datensatz	Allgemeine Funktionen
ResourceFree	Löscht dynamisch erstellte Ressource (freit den Speicher)	Allgemeine Funktionen
ResourceReadImage	Liest Daten einer Grafik-Ressource, die von der Funktion ResourceCreate() erstellt war oder in EX5-Datei beim Kompilieren gespeichert war	Allgemeine Funktionen
ResourceSave	Speichert die Ressource in eine angegebene Datei	Allgemeine Funktionen
round	Rundet die Zahl bis die naechste Ganzzahl	Mathematische Funktionen
SendFTP	Sendet Datei an die Adresse, angegeben im Fenster der Einstellungen im Registerblatt "Publikation"	Allgemeine Funktionen
SendMail	Sendet E-mail an die Adresse, angegeben im Fenster der Einstellungen im Registerblatt "Post"	Allgemeine Funktionen
SendNotification	Sendet Push-Benachrichtigungen zu mobilen Terminals, deren	Allgemeine Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	MetaQuotes-IDs in das Optionen-Fenster auf der Registerkarte "Benachrichtigungen" angegeben sind	
SeriesInfoInteger	Gibt die Information über den Zustand der historischen Daten zurück	Zugang zu Zeitreihen und Indikatoren
SetIndexBuffer	Verbindet den Indikatorpuffer mit dem eindimensionalen dynamischen Feld Feld des Typs double	Benutzerindikatoren
ShortArrayToString	Kopiert Teil des Feldes in die Zeile	Datenverarbeitung
ShortToString	Umwandlung des Symbolcodes (unicode) in die Einsymbolzeile	Datenverarbeitung
SignalBaseGetDouble	Gibt den Wert des Eigenschafts vom Typ double für das gewählte Signal zurück	Trade Signals
SignalBaseGetInteger	Gibt den Wert des Eigenschafts vom Typ integer für das gewählte Signal zurück	Trade Signals
SignalBaseGetString	Gibt den Wert des Eigenschafts vom Typ string für das gewählte Signal zurück	Trade Signals
SignalBaseSelect	Wählt ein Signal aus der Basis der Handelssignale, die im Terminal erhältlich sind	Trade Signals
SignalBaseTotal	Gibt die Anzahl der im Terminal verfügbare Signale zurück	Trade Signals
SignalInfoGetDouble	Gibt den Wert des Eigenschafts vom Typ double aus der Kopiereinstellungen des Handelssignals zurück	Trade Signals
SignalInfoGetInteger	Gibt den Wert des Eigenschafts vom Typ integer aus der Kopiereinstellungen des Handelssignals zurück	Trade Signals
SignalInfoGetString	Gibt den Wert des Eigenschafts vom Typ string aus der Kopiereinstellungen des Handelssignals zurück	Trade Signals

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
SignalInfoSetDouble	Setzt den Wert des Eigenschafts vom Typ double in Kopiereinstellungen des Handelssignals	Trade Signals
SignalInfoSetInteger	Setzt den Wert des Eigenschafts vom Typ integer in Kopiereinstellungen des Handelssignals	Trade Signals
SignalSubscribe	Abonniert auf ein Handelssignal	Trade Signals
SignalUnsubscribe	Abbestellt Subscription auf ein Handelssignal	Trade Signals
sin	Gibt sin der Zahl zurück	Mathematische Funktionen
Sleep	Hemmt die Ausführung des laufenden Experten oder Script für bestimmte Zeit	Allgemeine Funktionen
SocketCreate	Erstellen eines Sockets mit den angegebenen Flags und der Rückgabe des Handles	Netzwerkfunktionen
SocketClose	Schließen des Sockets	Netzwerkfunktionen
SocketConnect	Verbinden mit einem Server inklusive dem Bestimmen eines Timeouts (Zeitkontrolle)	Netzwerkfunktionen
SocketIsConnected	Prüft, ob der Socket aktuell verbunden ist	Netzwerkfunktionen
SocketIsReadable	Abrufen einer Anzahl von Bytes, die vom Socket gelesen werden können	Netzwerkfunktionen
SocketIsWritable	Prüfen, ob jetzt Daten auf einen Socket geschrieben werden können	Netzwerkfunktionen
SocketTimeouts	Festlegen eines Timeouts für das Senden und Empfangen der Daten eines Sockets	Netzwerkfunktionen
SocketRead	Lesen der Daten von einem Socket	Netzwerkfunktionen
SocketSend	Schreiben der Daten auf einen Socket	Netzwerkfunktionen
SocketTlsHandshake	Initiieren einer sicheren (SSL) Verbindung mit einem	Netzwerkfunktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	angegebenen Host über das Protokoll TLS-Handshake	
SocketTlsCertificate	Abrufen der Daten eines Zertifikates, das für eine sichere Verwendung TLS-Handshake verwandt wird	Netzwerkfunktionen
SocketTlsRead	Lesen der Daten einer sicheren TLS-Verbindung	Netzwerkfunktionen
SocketTlsReadAvailable	Lesen aller verfügbaren Daten einer sicheren TLS-Verbindung	Netzwerkfunktionen
SocketTlsSend	Senden von Daten über eine sichere TLS-Verbindung	Netzwerkfunktionen
sqrt	Gibt Quadratwurzel zurück	Mathematische Funktionen
srand	Stellt Anfangszustand für Geberieren der pseudozufälligen Realzahlen	Mathematische Funktionen
StringAdd	Fuegt ein String in den Platz der angegebenen Unterzeile	Stringfunktionen
StringBufferLen	Gibt die Größe des Puffers zurück, der für den String verteilt wurde	Stringfunktionen
StringCompare	Vergleicht zwei Strings und gibt 1 zurück, wenn der erste String größer als der zweite ist, 0 - wenn sie gleich sind; -1 (minus eins) - wenn der erste String kleiner als der zweite ist	Stringfunktionen
StringConcatenate	Formiert einen String aus übertragenen Parametern	Stringfunktionen
StringFill	Fuellt den angegebenen String durch die gewaehlten Symbole aus	Stringfunktionen
StringFind	Suche des Teilstrings im String	Stringfunktionen
StringFormat	Wandelt Zahl in die Zeile entsprechend dem vorgegebenen Format um	Datenverarbeitung
StringGetCharacter	Gibt den Wert des Symboles, das in der angegebenen Position des Strings liegt	Stringfunktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
StringInit	Initialisiert den String durch die gewaehlten Symbole und stellt die angegebene Stringgroesse bereit	Stringfunktionen
StringLen	Gibt die Zahl der Symbole im String zurück	Stringfunktionen
StringReplace	Ersetzt alle Substrings im String mit einer bestimmten Abfolge von Symbole	Stringfunktionen
StringSetCharacter	Gibt die Kopie des Strings mit dem veränderten Wert des Symboles in der angegebenen Position	Stringfunktionen
StringSplit	Bekommt Teilstrings durch einen angegebenen Trennzeichen aus dem angegebenen String, gibt die Anzahl der Teilstrings zurück	Stringfunktionen
StringSubstr	Extrahiert die Substring aus dem Textstring, die mit der angegebenen Position beginnt	Stringfunktionen
StringToCharArray	Kopiert die Zeile, die aus Unicode in ANSI umgewandelt wurde, in den gewaehlten Platz des Typs uchar	Datenverarbeitung
StringToColor	Wandelt Zeile des Typs "R,G,B" oder Zeile mit Farbennamen in den Wert des Typs color um	Datenverarbeitung
StringToDouble	Umwandlung der Zeile mit der Symboldarstellung der Zahl in die Zahl des Typs double	Datenverarbeitung
StringToInteger	Umwandlung der Zeile mit der Symboldarstellung der Zahl in die Zahl des Typs int	Datenverarbeitung
StringToLower	Wandelt alle Symbole des angegebenen Strings in kleine Symbole um	Stringfunktionen
StringToShortArray	Kopiert die Zeile in den gewaehlten Platz des Feldes des Typs ushort	Datenverarbeitung
StringToTime	Umwandlung der Zeile mit Zeit und/oder Datum im Format	Datenverarbeitung

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	"yyyy.mm.dd [hh:mi]", in eine Zahl des Typs datetime	
StringToUpper	Wandelt alle Symbole des angegebenen Strings in grosse Symbole um	Stringfunktionen
StringTrimLeft	Entfernt Zeilenvorschub, Spaces und Tabs am Anfang des Strings	Stringfunktionen
StringTrimRight	Entfernt Zeilenvorschub, Spaces und Tabs am Ende des Strings	Stringfunktionen
StructToTime	Wandelt Variable des Typs der Strukturen MqlDateTime in den Wert des Typs datetime um	Datum und Zeit
Symbol	Gibt Symbolname des laufenden Charts.	Prüfung des Status
SymbolInfoDouble	Gibt den Wert des Typs double des angegebenen Symbols für die entsprechende Eigenschaft zurück	Marktinformation erhalten
SymbolInfoInteger	Gibt den Wert des ganzzahligen Typs (long, datetime, int oder bool) des angegebenen Symbols für die entsprechende Eigenschaft zurück	Marktinformation erhalten
SymbolInfoMarginRate	Gibt die Margin je nach Ordertyp und -richtung zurück	Marktinformation erhalten
SymbolInfoSessionQuote	Ermöglicht die Anfangszeit und die Abschlusszeit der angegebenen Kotierungssession für das angegebene Symbol und für den angegebenen Wochentag	Marktinformation erhalten
SymbolInfoSessionTrade	Ermöglicht die Anfangszeit und die Abschlusszeit der angegebenen Handelssession für das angegebene Symbol und für den angegebenen Wochentag	Marktinformation erhalten
SymbolInfoString	Gibt den Wert des Typs string des angegebenen Symbols für die entsprechende Eigenschaft zurück	Marktinformation erhalten
SymbolInfoTick	Gibt laufende Preise für das angegebene Symbol in der Variable des Typs MqlTick	Marktinformation erhalten

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
SymbolIsSynchronized	Überprüft ob die Daten des angegebenen Symbol mit Daten auf dem Handelsserversynchronisiert sind	Marktinformation erhalten
SymbolName	Gibt den Namen des bezeichneten Symbols zurück	Marktinformation erhalten
SymbolSelect	Waehlt das Symbol im Fenster MarketWatch oder entfernt das Symbol aus dem Fenster	Marktinformation erhalten
SymbolsTotal	Gibt die Anzahl zugaenglicher (gewaehlter in MarketWatch oder aller) Symbole zurück	Marktinformation erhalten
tan	Gibt tg der Zahl zurück	Mathematische Funktionen
TerminalClose	Sendet dem Terminal Adressbefehl, Arbeit zu beenden	Allgemeine Funktionen
TerminalInfoDouble	Gibt den Wert vom Typ double der entsprechenden Eigenschaft der Umgebung eines mql5-Programms zurück	Prüfung des Status
TerminalInfoInteger	Gibt den Wert des ganzzahligen Types der entsprechenden Eigenschaft der Umwelt des mql5-Programms zurück	Prüfung des Status
TerminalInfoString	Gibt den Wert desw Typs string der entsprechenden Eigenschaft der Umwelt des mql5-Programms zurück	Prüfung des Status
TesterHideIndicators	Aktiviert den Modus zum Anzeigen/Ausblenden von Indikatoren, die in einem Expert Advisor verwendet werden	Allgemeine Funktionen
TesterStop	Erteilt den Befehl die Programmausführung von Tests zu beenden.	Allgemeine Funktionen
TesterDeposit	Eine spezielle Funktion, die während eines Tests Einzahlungen emuliert	Allgemeine Funktionen
TesterWithdrawal	Sonderfunktion für Emulation der Geldabhebungen beim Testen	Allgemeine Funktionen
TesterStatistics	Es gibt den Wert der statistischen Parameter, der nach	Allgemeine Funktionen

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
	Testergebnisse berechnet ist, zurück	
TextGetSize	Gibt die Breite und Höhe der Zeile mit den aktuellen Schrifteinstellungen zurück	Graphische Objekte
TextOut	Gibt den Text in einem benutzerdefinierten Array (Puffer), der für die Erstellung von grafischen Ressource benutzt wird, aus	Graphische Objekte
TextSetFont	Setzt die Schrift für die Textausgabe durch Zeichnenmethoden (die Standardschriftart ist Arial 20)	Graphische Objekte
TimeCurrent	Bringt den letzten bekannten Serevernamen (Zeit der letzten Kotierung) in datetime Format zurück	Datum und Zeit
TimeDaylightSavings	Gibt Zeichen des überganges zur Sommer-/Winterzeit zurück	Datum und Zeit
TimeGMT	Bringt GMT Zeit in Format datetime mit Rücksicht auf Übergang zur Sommer-/Winterzeit für lokale Zeit des Computers mit dem Client-Terminals	Datum und Zeit
TimeGMTOffset	Gibt die laufende Differenz zwischen Zeit GMT und lokaler Zeit des Computers in Sekunden mit Rücksicht übergang zur Sommer-/Winterzeit	Datum und Zeit
TimeLocal	Bringt die lokale Computerzeit in datetime Format zurück	Datum und Zeit
TimeToString	Umwandlung des Wertes mi t der Zeit in Sekunden, die seit 01.01.1970 vergangen hat, in die Zeile mit Format "yyyy.mm.dd hh:mi"	Datenverarbeitung
TimeToStruct	Wandelt einen Wert des Typs datetime in eine Variable des Typs der Strukturen MqlDateTime um	Datum und Zeit

© 2000-2025, MetaQuotes Ltd.

Funktion	Aktion	Sektion
TimeTradeServer	Bringt den laufenden Berechnunngszeitrzum des Handelsservers zurück	Datum und Zeit
UninitializeReason	Gibt den Kode des Deinitialisierungsgrundes	Prüfung des Status
WebRequest	Sendet eine HTTP-Anfrage an den angegebenen Server	Allgemeine Funktionen
ZeroMemory	Nullt die Variable aus, die durch Referenz übertragen wird. Typ der Variable kann verschieden sein, Ausnahmen bilden nur Klassen und Strukturen mit Konstrukteuren	Allgemeine Funktionen

© 2000-2025, MetaQuotes Ltd.

MQL5 Konstantenliste

Alle MQL5 Konstanten in alphabetischer Reihenfolge.

Konstante	Beschreibung	Verwendung
__DATE__	Datum der Kompilierung einer Date ohne die Uhrzeit (Stunden, Minuten und Sekunden sind 0)	Print
__DATETIME__	Datum und Uhrzeit der Kompilierung einer Date	Print
__FILE__	Name der laufenden kompilierten Datei	Print
__FUNCSIG__	Signatur der Funktion, in deren Körper befindet sich das Makro. Protokollierung der vollständ	Print

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ige Beschreibung der Funktionen kann bei der Identifizierung von überladenen Funktionen nützlich sein	
__FUNCTION__	Funktionsname, in deren Körper sich das Makro befindet	Print
__LINE__	Nummer der Zeile im Ausgangskode, auf der sich der Makro befindet	Print
__MQLBUILD__, __MQL5BUILD__	Buildnummer des Compilers	Print
__PATH__	Absoluter Pfad zu der laufenden kompilie	Print

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	rten Datei	
ACCOUNT_ASSETS	Die aktuelle Aktiva auf dem Konto	AccountInfoDouble
ACCOUNT_BALANCE	Bilanz des Kontos in der Währung des Deposits	AccountInfoDouble
ACCOUNT_COMMISSION_BLOCKED	Die aktuelle blockierte Kommission für das Konto	AccountInfoDouble
ACCOUNT_COMPANY	Name der Gesellschaft, die das Konto bedient	AccountInfoString
ACCOUNT_CREDIT	Grösse des gegebenen Kredits in der Währung des Deposits	AccountInfoDouble
ACCOUNT_CURRENCY	Depositenwaehrung	AccountInfoString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ACCOUNT_EQUITY	Wer der Eigenmittel auf dem Konto in der Währung des Deposits	AccountInfoDouble
ACCOUNT_LEVERAGE	Größe von Leverage	AccountInfoInteger
ACCOUNT_LIABILITIES	Aktuelle Verbindlichkeiten auf dem Konto	AccountInfoDouble
ACCOUNT_LIMIT_ORDERS	Maximale zulässige Anzahl der geltenden Warteordern	AccountInfoInteger
ACCOUNT_LOGIN	Nummer des Kontos	AccountInfoInteger
ACCOUNT_MARGIN	Wert des reservierten Pfandgeldes in der Währung des Deposits	AccountInfoDouble
ACCOUNT_MARGIN_FREE	Grösse der	AccountInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	freien Mittel auf dem Konto in der Währung des Deposits, die für die Öffnung der Position zugänglich sind	
ACCOUNT_MARGIN_INITIAL	Initial Margin. Die reservierte Mittel auf dem Konto als die Garantie für alle ausstehenden Aufträge (Pending Ordern)	AccountInfoDouble
ACCOUNT_MARGIN_LEVEL	Konto-Margennniveau in Prozenten	AccountInfoDouble
ACCOUNT_MARGIN_MAINTENANCE	Maintenance Margin. Die reservierte Mittel auf dem Konto als ein	AccountInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Sicherheitsmindestbetrag für alle offenen Positionen	
ACCOUNT_MARGIN_SO_CALL	Margin Call Level. Depending on the set ACCOUNT_MARGIN_SO_MODE is expressed in percents or in the deposit currency. Abhängig vom eingestellten ACCOUNT_MARGIN_SO_MODE wird in Prozenten oder in der Währung des Deposits ausgedrückt	AccountInfoDouble
ACCOUNT_MARGIN_SO_MODE	Mode des minimal zugängli	AccountInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	chen Marge	
ACCOUNT_MARGIN_SO_SO	Margin stop out level. Depending on the set ACCOUNT_MARGIN_SO_MODE is expressed in percents or in the deposit currency. Abhängig vom eingestellten ACCOUNT_MARGIN_SO_MODE wird in Prozenten oder in der Währung des Deposits ausgedrückt	AccountInfoDouble
ACCOUNT_NAME	Name des Kunden	AccountInfoString
ACCOUNT_PROFIT	Grösse des laufenden Gewinns auf dem Konto in	AccountInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	der Währung des Deposits	
ACCOUNT_SERVER	Name des Handelsservers	AccountInfoString
ACCOUNT_STOPOUT_MODE_MONEY	Level wird in Geld vorgegeben	AccountInfoInteger
ACCOUNT_STOPOUT_MODE_PERCENT	Level wird in Prozenten vorgegeben	AccountInfoInteger
ACCOUNT_TRADE_ALLOWED	Erlaubnis zu handeln für das laufende Konto	AccountInfoInteger
ACCOUNT_TRADE_EXPERT	Erlaubnis zu handeln für den Experten	AccountInfoInteger
ACCOUNT_TRADE_MODE	Handelsweise des Kontos	AccountInfoInteger
ACCOUNT_TRADE_MODE_CONTEST	Wettbewerbskonto	AccountInfoInteger
ACCOUNT_TRADE_MODE_DEMO	Demokonto	AccountInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ACCOUNT_TRADE_MODE_REAL	Realkonto	AccountInfoInteger
ALIGN_CENTER	Zentrierte Ausrichtung (für"Eingabefeld")	ObjectSetInteger, ObjectGetInteger, ChartScreenShot
ALIGN_LEFT	Linksbündige Ausrichtung	ObjectSetInteger, ObjectGetInteger, ChartScreenShot
ALIGN_RIGHT	Rechtsbündige Ausrichtung	ObjectSetInteger, ObjectGetInteger, ChartScreenShot
ANCHOR_CENTER	Bindepunkt im Objektzentrum	ObjectSetInteger, ObjectGetInteger
ANCHOR_LEFT	Bindepunkt links zentriert	ObjectSetInteger, ObjectGetInteger
ANCHOR_LEFT_LOWER	Bindepunkt im unteren linken Winkel	ObjectSetInteger, ObjectGetInteger
ANCHOR_LEFT_UPPER	Binde im oberen linken Winkel	ObjectSetInteger, ObjectGetInteger
ANCHOR_LOWER	Bindepunkt unten zentriert	ObjectSetInteger, ObjectGetInteger
ANCHOR_RIGHT	Bindepunkt rechts	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	zentriert	
ANCHOR_RIGHT_LOWER	Bindepunkt im unteren rechten Winkel	ObjectSetInteger, ObjectGetInteger
ANCHOR_RIGHT_UPPER	Bindepunkt rechts zentriert	ObjectSetInteger, ObjectGetInteger
ANCHOR_UPPER	Bindepunkt oben zentriert	ObjectSetInteger, ObjectGetInteger
BASE_LINE	Grundlinie	Linien der Indikatoren
BOOK_TYPE_BUY	Kaufsorder	MqlBookInfo
BOOK_TYPE_BUY_MARKET	Kaufsorder mit Marktpreis	MqlBookInfo
BOOK_TYPE_SELL	Verkaufsorder	MqlBookInfo
BOOK_TYPE_SELL_MARKET	Verkaufsorder mit Marktpreis	MqlBookInfo
BORDER_FLAT	Flache Form	ObjectSetInteger, ObjectGetInteger
BORDER_RAISED	Konvexe Form	ObjectSetInteger, ObjectGetInteger
BORDER_SUNKEN	Konkave Form	ObjectSetInteger, ObjectGetInteger
CHAR_MAX	Maximaler Wert, der	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	durch den Typ char dargestellt werden kann	
CHAR_MIN	Minimaler Wert, der durch den Typ char dargestellt werden kann	Konstanten der numerischenTypen
CHART_AUTOSCROLL	Automatischer Sprung zum rechten Schlussrand des Charts	ChartSetInteger, ChartGetInteger
CHART_BARS	Darstellung in Form von Bars	ChartSetInteger
CHART_BEGIN	Anfang des Charts (die ältesten Preise)	ChartNavigate
CHART_BRING_TO_TOP	Den Chart über anderen Chart anzeigen	ChartSetInteger, ChartGetInteger
CHART_CANDLES	Darstellung in Form	ChartSetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	von japanischen Kerzen	
CHART_COLOR_ASK	Linienfarbe des Ask-Preises	ChartSetInteger, ChartGetInteger
CHART_COLOR_BACKGROUND	Hintergrundfarbe des Charts	ChartSetInteger, ChartGetInteger
CHART_COLOR_BID	Linienfarbe des Bid-Preises	ChartSetInteger, ChartGetInteger
CHART_COLOR_CANDLE_BEAR	Körperfarbe der Bärenkerze	ChartSetInteger, ChartGetInteger
CHART_COLOR_CANDLE_BULL	Körperfarbe der der Bullenkerze	ChartSetInteger, ChartGetInteger
CHART_COLOR_CHART_DOWN	Farbe des Balkens unten, des Schattens und Umrandung des Körpers der Bärenkerze	ChartSetInteger, ChartGetInteger
CHART_COLOR_CHART_LINE	Farbe der Chartlinie und der	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	japanischen Kerzen "Dodzhi"	
CHART_COLOR_CHART_UP	Farbe des Balkens oben, des Schattens und der Umrandung des Körpers der Bullenkerze	ChartSetInteger, ChartGetInteger
CHART_COLOR_FOREGROUND	Farbe von Achsen, Skale und Zeile OHLC	ChartSetInteger, ChartGetInteger
CHART_COLOR_GRID	Farbe des Zeitrasters	ChartSetInteger, ChartGetInteger
CHART_COLOR_LAST	Linienfarbe des Preises des letzten abgeschlossenen Deals (Last)	ChartSetInteger, ChartGetInteger
CHART_COLOR_STOP_LEVEL	Levelsfarbe der Stop-Ordern (Stop Loss und	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Take Profit)	
CHART_COLOR_VOLUME	Farbe der Volumen und Levels der offenen Positionen	ChartSetInteger, ChartGetInteger
CHART_COMMENT	Kommentartext auf dem Chart	ChartSetString, ChartGetString
CHART_CURRENT_POS	Laufende Position	ChartNavigate
CHART_DRAG_TRADE_LEVELS	iehen der Handelsstufen auf dem Chart mit dem Maus erlauben. Standardmäßig ist der Ziehen-Modus aktiviert (true)	ChartSetInteger, ChartGetInteger
CHART_EVENT_MOUSE_MOVE	Senden alle MQL5-Programme auf dem Chart Benachrichtigungen über	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Ereignisse der Bewegung und Klicken der Maustasten (CHART_EVENT_MOUSE_MOVE)	
CHART_EVENT_OBJECT_CREATE	Senden alle MQL5-Programme auf dem Chart eine Benachrichtigung über Ereignis des neuen Objekts Erstellung (CHART_EVENT_OBJECT_CREATE)	ChartSetInteger, ChartGetInteger
CHART_EVENT_OBJECT_DELETE	Senden alle MQL5-Programme auf dem Chart eine Benachrichtigung über Ereignis des	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Objekts Löschung (CHART_EVENT_OBJECT_DELETE)	
CHART_FIRST_VISIBLE_BAR	Nummer der ersten sichtbaren Balken auf dem Chart. Indizierung von Balken ist das gleiche wie für Zeitreihen.	ChartSetInteger, ChartGetInteger
CHART_FIXED_MAX	Festes Chartmaximum	ChartSetDouble, ChartGetDouble
CHART_FIXED_MIN	Festes Chartminimum	ChartSetDouble, ChartGetDouble
CHART_FIXED_POSITION	Position der festen Position des Chart vom linken Rand als Prozentsatz. Feste Position des Charts ist mit	ChartSetDouble, ChartGetDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	einem kleinen grauen Dreieck auf der horizontalen Achse der Zeit gezeichnet und wird nur angezeigt, wenn Autoscroll des Charts mit einem neuen Tick nach rechts deaktiviert ist (siehe Eigenschaft CHART_AUTOSCROLL). Der Bar, der in der feste Position ist, bleibt an der gleiche Stelle mit Zoom-in und Zoom-out.	

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
CHART_FOREGROUND	Preischart auf dem Hintergrund	ChartSetInteger, ChartGetInteger
CHART_HEIGHT_IN_PIXELS	Charthöhe in Pixel	ChartSetInteger, ChartGetInteger
CHART_IS_OBJECT	Attribut um Objekt "Chart" (OBJ_CHART) zu identifizieren - für ein grafisches Objekt gibt true zurück. Für dieses Chart ist die Eigenschaft false	ChartSetInteger, ChartGetInteger
CHART_LINE	Darstellung in Form einer Linie, gezogen an den Preisen Close	ChartSetInteger
CHART_MODE	Charttyp (Kerzen, Bars, oder Linie)	ChartSetInteger, ChartGetInteger
CHART_MOUSE_SCROLL	Den Chart mit der	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	linken Maustaste horizontal scrollen. Vertikales Scrollen wird verfügbar sein, wenn der Wert von einer der drei Eigenschaften true ist: CHART_SCALEFIX, CHART_SCALEFIX_11 oder CHART_SCALE_PT_PER_BAR	
CHART_POINTS_PER_BAR	Massstabwert in Punkt/Balken	ChartSetDouble, ChartGetDouble
CHART_PRICE_MAX	Chartmaximum	ChartSetDouble, ChartGetDouble
CHART_PRICE_MIN	Chartminimum	ChartSetDouble, ChartGetDouble
CHART_SCALE	Maßstab	ChartSetInteger, ChartGetInteger
CHART_SCALE_PT_PER_BAR	Masßtab in Punkt/Balken	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
CHART_SCALEFIX	Festmaßstab	ChartSetInteger, ChartGetInteger
CHART_SCALEFIX_11	Maßstab 1:1	ChartSetInteger, ChartGetInteger
CHART_SHIFT	Absatz des Preischarts rechtsbündig	ChartSetInteger, ChartGetInteger
CHART_SHIFT_SIZE	Absatzgröße der Nullbar vom rechten Schlussrand	ChartSetDouble, ChartGetDouble
CHART_SHOW_ASK_LINE	Darstellung des Wertes Ask als horizontale Linie auf dem Chart	ChartSetInteger, ChartGetInteger
CHART_SHOW_BID_LINE	Darstellung des Wertes Bid als horizontale Linie auf dem Chart	ChartSetInteger, ChartGetInteger
CHART_SHOW_DATE_SCALE	Anzeigen die Zeit-Skala auf dem Diagram	ChartSetInteger, ChartGetInteger
CHART_SHOW_GRID	Darstellung des Zeitrasters auf	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	dem Chart	
CHART_SHOW_LAST_LINE	Darstellung des Wertes Last durch horizontale Linie auf dem Chart	ChartSetInteger, ChartGetInteger
CHART_SHOW_OBJECT_DESCR	Popup-Beschreibungen der graphischen Objekte	ChartSetInteger, ChartGetInteger
CHART_SHOW_OHLC	Darstellung der Werte OHLC im oberen linken Winkel	ChartSetInteger, ChartGetInteger
CHART_SHOW_ONE_CLICK	Anzeige des Panels für schnellen Handel auf dem Chart (die Option "Ein-Klick-Handel")	ChartSetInteger, ChartGetInteger
CHART_SHOW_PERIOD_SEP	Darstellung der senkrechten Begrenzungen unter	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Nachbarperioden	
CHART_SHOW_PRICE_SCALE	Anzeigen die Preis-Skala auf dem Diagram	ChartSetInteger, ChartGetInteger
CHART_SHOW_TRADE_LEVELS	Darstellung der Handelsniveaus auf dem Chart (Niveaus der offenen Positionen, Stop Loss, Take Profit und wartender Order)	ChartSetInteger, ChartGetInteger
CHART_SHOW_VOLUMES	Darstellung von Volumen auf dem Chart	ChartSetInteger, ChartGetInteger
CHART_VISIBLE_BARS	Anzahl der Balken auf dem Chart, zugänglich für die Darstellung	ChartSetInteger, ChartGetInteger
CHART_VOLUME_HIDE	Volumen nicht gezeigt	ChartSetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
CHART_VOLUME_REAL	Handelsvolumen	ChartSetInteger
CHART_VOLUME_TICK	Tickvolumen	ChartSetInteger
CHART_WIDTH_IN_BARS	Chartbreite in Balken	ChartSetInteger, ChartGetInteger
CHART_WIDTH_IN_PIXELS	Chartbreite in Pixel	ChartSetInteger, ChartGetInteger
CHART_WINDOW_HANDLE	Handle des Charts (HWND)	ChartSetInteger, ChartGetInteger
CHART_WINDOW_IS_VISIBLE	Sichtbarkeit der Subfenster	ChartSetInteger, ChartGetInteger
CHART_WINDOW_YDISTANCE	Der Abstand zwischen dem oberen Rahmen des Unterfensters eines Indikators und dem oberen Rahmen des Hauptfensters des Charts, entlang der vertikalen Y-Achse, in	ChartSetInteger, ChartGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Pixeln. Im Falle eines Maus-Ereignisses, werden die Cursor-Koordinaten im Koordinaten des Hauptfensters übergegeben, während die Koordinaten von grafischen Objekten im Indikator-Unterfenster relativ zur oberen linken Ecke des Unterfensters eingestellt sind. Der Wert ist erforderlich, um die absoluten Koordinaten des Hauptfe	

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	nsters zu den lokalen Koordinaten eines Unterfensters für die korrekte Arbeit mit der grafischen Objekte, deren Koordinaten sind relativ zur oberen linken Ecke des Unterfensters angegeben, umzuwandeln.	
CHART_WINDOWS_TOTAL	Gesamtzahl der Chartfenster, einschließlich Subfenster der Indikatoren	ChartSetInteger, ChartGetInteger
CHARTEVENT_CHART_CHANGE	Ereignis der Veränderung der Chart-Größe oder	OnChartEvent

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Veränderung der Chart-Eigenschaften durch Dialog von Eigenschaften	
CHARTEVENT_CLICK	Mausklick auf dem Chart	OnChartEvent
CHARTEVENT_CUSTOM	Anfangsnummer des Ereignisses vom Bereich der Benutzerereignisse	OnChartEvent
CHARTEVENT_CUSTOM_LAST	Endnummer des Ereignisses vom Bereich der Benutzerereignisse.	OnChartEvent
CHARTEVENT_KEYDOWN	Tasten	OnChartEvent
CHARTEVENT_MOUSE_MOVE	Das Bewegen der Maus und Mausklicks (wenn die Eigenschaft CHART_	OnChartEvent

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	EVENT_MOUSE_MOVE=true ist für den Chart eingegeben)	
CHARTEVENT_OBJECT_CHANGE	Veranderung der Objekteigenschaften durch Dialog der Eigenschaften	OnChartEvent
CHARTEVENT_OBJECT_CLICK	Betaetigen der Taste mit der Maus auf dem graphischen Objekt	OnChartEvent
CHARTEVENT_OBJECT_CREATE	Erzeugung vom graphischen Objekt (wenn die Eigenschaft CHART_EVENT_OBJECT_CREATE =true ist für den Chart eingegeben)	OnChartEvent

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
CHARTEVENT_OBJECT_DELETE	Entfernung vom graphischen Objekt (wenn die Eigenschaft CHART_EVENT_OBJECT_DELETE=true ist für den Chart eingegeben)	OnChartEvent
CHARTEVENT_OBJECT_DRAG	Ziehen des graphischen Objektes	OnChartEvent
CHARTEVENT_OBJECT_ENDEDIT	Beenden von Textedit ieren im graphischen Objekt Edit	OnChartEvent
CHARTS_MAX	Maximal mögliche Zahl der gleichzeitig offenen Charts im Terminal	Andere Konstanten
CHINKOUSPAN_LINE	Linie Chinkou Span	Linien der Indikatoren

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
clrAliceBlue	Alice Blue	Web-Farben
clrAntiqueWhite	Antique White	Web-Farben
clrAqua	Aqua	Web-Farben
clrAquamarine	Aquamarine	Web-Farben
clrBeige	Beige	Web-Farben
clrBisque	Bisque	Web-Farben
clrBlack	Black	Web-Farben
clrBlanchedAlmond	Blanched Almond	Web-Farben
clrBlue	Blue	Web-Farben
clrBlueViolet	Blue Violet	Web-Farben
clrBrown	Brown	Web-Farben
clrBurlyWood	Burly Wood	Web-Farben
clrCadetBlue	Cadet Blue	Web-Farben
clrChartreuse	Chartreuse	Web-Farben
clrChocolate	Chocolate	Web-Farben
clrCoral	Coral	Web-Farben
clrCornflowerBlue	Cornflower Blue	Web-Farben
clrCornsilk	Cornsilk	Web-Farben
clrCrimson	Crimson	Web-Farben
clrDarkBlue	Dark Blue	Web-Farben
clrDarkGoldenrod	Dark Goldenrod	Web-Farben

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
clrDarkGray	Dark Gray	Web-Farben
clrDarkGreen	Dark Green	Web-Farben
clrDarkKhaki	Dark Khaki	Web-Farben
clrDarkOliveGreen	Dark Olive Green	Web-Farben
clrDarkOrange	Dark Orange	Web-Farben
clrDarkOrchid	Dark Orchid	Web-Farben
clrDarkSalmon	Dark Salmon	Web-Farben
clrDarkSeaGreen	Dark Sea Green	Web-Farben
clrDarkSlateBlue	Dark Slate Blue	Web-Farben
clrDarkSlateGray	Dark Slate Gray	Web-Farben
clrDarkTurquoise	Dark Turquoise	Web-Farben
clrDarkViolet	Dark Violet	Web-Farben
clrDeepPink	Deep Pink	Web-Farben
clrDeepSkyBlue	Deep Sky Blue	Web-Farben
clrDimGray	Dim Gray	Web-Farben
clrDodgerBlue	Dodger Blue	Web-Farben
clrFireBrick	Fire Brick	Web-Farben

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
clrForestGreen	Forest Green	Web-Farben
clrGainsboro	Gainsboro	Web-Farben
clrGold	Gold	Web-Farben
clrGoldenrod	Goldenrod	Web-Farben
clrGray	Gray	Web-Farben
clrGreen	Green	Web-Farben
clrGreenYellow	Green Yellow	Web-Farben
clrHoneydew	Honeydew	Web-Farben
clrHotPink	Hot Pink	Web-Farben
clrIndianRed	Indian Red	Web-Farben
clrIndigo	Indigo	Web-Farben
clrIvory	Ivory	Web-Farben
clrKhaki	Khaki	Web-Farben
clrLavender	Lavender	Web-Farben
clrLavenderBlush	Lavender Blush	Web-Farben
clrLawnGreen	Lawn Green	Web-Farben
clrLemonChiffon	Lemon Chiffon	Web-Farben
clrLightBlue	Light Blue	Web-Farben
clrLightCoral	Light Coral	Web-Farben
clrLightCyan	Light Cyan	Web-Farben
clrLightGoldenrod	Light Goldenro	Web-Farben

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	d	
clrLightGray	Light Gray	Web-Farben
clrLightGreen	Light Green	Web-Farben
clrLightPink	Light Pink	Web-Farben
clrLightSalmon	Light Salmon	Web-Farben
clrLightSeaGreen	Light Sea Green	Web-Farben
clrLightSkyBlue	Light Sky Blue	Web-Farben
clrLightSlateGray	Light Slate Gray	Web-Farben
clrLightSteelBlue	Light Steel Blue	Web-Farben
clrLightYellow	Light Yellow	Web-Farben
clrLime	Lime	Web-Farben
clrLimeGreen	Lime Green	Web-Farben
clrLinen	Linen	Web-Farben
clrMagenta	Magenta	Web-Farben
clrMaroon	Maroon	Web-Farben
clrMediumAquamarine	Medium Aquamarine	Web-Farben
clrMediumBlue	Medium Blue	Web-Farben
clrMediumOrchid	Medium Orchid	Web-Farben
clrMediumPurple	Medium Purple	Web-Farben

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
clrMediumSeaGreen	Medium Sea Green	Web-Farben
clrMediumSlateBlue	Medium Slate Blue	Web-Farben
clrMediumSpringGreen	Medium Spring Green	Web-Farben
clrMediumTurquoise	Medium Turquoise	Web-Farben
clrMediumVioletRed	Medium Violet Red	Web-Farben
clrMidnightBlue	Midnight Blue	Web-Farben
clrMintCream	Mint Cream	Web-Farben
clrMistyRose	Misty Rose	Web-Farben
clrMoccasin	Moccasin	Web-Farben
clrNavajoWhite	Navajo White	Web-Farben
clrNavy	Navy	Web-Farben
clrNONE	Abwesenheit der Farbe	Andere Konstanten
clrOldLace	Old Lace	Web-Farben
clrOlive	Olive	Web-Farben
clrOliveDrab	Olive Drab	Web-Farben
clrOrange	Orange	Web-Farben
clrOrangeRed	Orange Red	Web-Farben
clrOrchid	Orchid	Web-Farben

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
clrPaleGoldenrod	Pale Goldenrod	Web-Farben
clrPaleGreen	Pale Green	Web-Farben
clrPaleTurquoise	Pale Turquoise	Web-Farben
clrPaleVioletRed	Pale Violet Red	Web-Farben
clrPapayaWhip	Papaya Whip	Web-Farben
clrPeachPuff	Peach Puff	Web-Farben
clrPeru	Peru	Web-Farben
clrPink	Pink	Web-Farben
clrPlum	Plum	Web-Farben
clrPowderBlue	Powder Blue	Web-Farben
clrPurple	Purple	Web-Farben
clrRed	Red	Web-Farben
clrRosyBrown	Rosy Brown	Web-Farben
clrRoyalBlue	Royal Blue	Web-Farben
clrSaddleBrown	Saddle Brown	Web-Farben
clrSalmon	Salmon	Web-Farben
clrSandyBrown	Sandy Brown	Web-Farben
clrSeaGreen	Sea Green	Web-Farben
clrSeashell	Seashell	Web-Farben
clrSienna	Sienna	Web-Farben

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
clrSilver	Silver	Web-Farben
clrSkyBlue	Sky Blue	Web-Farben
clrSlateBlue	Slate Blue	Web-Farben
clrSlateGray	Slate Gray	Web-Farben
clrSnow	Snow	Web-Farben
clrSpringGreen	Spring Green	Web-Farben
clrSteelBlue	Steel Blue	Web-Farben
clrTan	Tan	Web-Farben
clrTeal	Teal	Web-Farben
clrThistle	Thistle	Web-Farben
clrTomato	Tomato	Web-Farben
clrTurquoise	Turquoise	Web-Farben
clrViolet	Violet	Web-Farben
clrWheat	Wheat	Web-Farben
clrWhite	White	Web-Farben
clrWhiteSmoke	White Smoke	Web-Farben
clrYellow	Yellow	Web-Farben
clrYellowGreen	Yellow Green	Web-Farben
CORNER_LEFT_LOWER	Mittelpunkt der Koordinaten im unteren linken Winkel des Charts	ObjectSetInteger, ObjectGetInteger
CORNER_LEFT_UPPER	Mittelpunkt der	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Koordinaten im oberen linken Winkel des Charts	
CORNER_RIGHT_LOWER	Mittelpunkt der Koordinaten im unteren rechten Winkel des Charts	ObjectSetInteger, ObjectGetInteger
CORNER_RIGHT_UPPER	Mittelpunkt der Koordinaten im oberen rechten Winkel des Charts	ObjectSetInteger, ObjectGetInteger
CP_ACP	Laufende Kodeseite ANSIim Operationssystem Windows	CharArrayToString, StringToCharArray, FileOpen
CP_MACCP	Laufende Kodeseite Macintosh. Bemerkung: Dieser Wert wird vor	CharArrayToString, StringToCharArray, FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	allem in früher erzeugten Programmkodes verwendet und ist nicht notwendig, denn moderne Computers Macintosh verwenden Unicode Kodieren.	
CP_OEMCP	Laufende Kodeseite OEM.	CharArrayToString, StringToCharArray, FileOpen
CP_SYMBOL	Kodeseite Symbol	CharArrayToString, StringToCharArray, FileOpen
CP_THREAD_ACP	Kodieren Windows ANSI für laufenden Thread.	CharArrayToString, StringToCharArray, FileOpen
CP_UTF7	Kodeseite UTF-7.	CharArrayToString, StringToCharArray, FileOpen
CP_UTF8	Kodeseite UTF-8.	CharArrayToString, StringToCharArray, FileOpen
CRYPT_AES128	AES Verschlüsselung mit einem 128-Bit-	CryptEncode, CryptDecode

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Schlüssel (16 Byte)	
CRYPT_AES256	AES Verschlüsselung mit einem 256-Bit-Schlüssel (32 Byte)	CryptEncode, CryptDecode
CRYPT_ARCH_ZIP	ZIP-Archivierung	CryptEncode, CryptDecode
CRYPT_BASE64	BASE64 Verschlüsselung (Umkodierung)	CryptEncode, CryptDecode
CRYPT_DES	DES Verschlüsselung mit einem 56-Bit-Schlüssel (7 Byte)	CryptEncode, CryptDecode
CRYPT_HASH_MD5	Berechnung HASH MD5	CryptEncode, CryptDecode
CRYPT_HASH_SHA1	Berechnung HASH SHA1	CryptEncode, CryptDecode
CRYPT_HASH_SHA256	Berechnung HASH SHA256	CryptEncode, CryptDecode
DBL_DIG	Anzahl der bedeute	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	nden Dezimalzeichen	
DBL_EPSILON	Der minimale Wert, für den Die Bedingung 1.0+DBL_EPSILON != 1.0 erfuellt wird	Konstanten der numerischenTypen
DBL_MANT_DIG	Anzahl der Bits in mantissa	Konstanten der numerischenTypen
DBL_MAX	Maximaler Wert, der durch den Typ double dargestellt werden kann	Konstanten der numerischenTypen
DBL_MAX_10_EXP	Maximaler dezimaler Wert der Exponentenstufung	Konstanten der numerischenTypen
DBL_MAX_EXP	Maximaler binaerer Wert der Exponentenstufung	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
DBL_MIN	Minimaler positiver Wert, der durch den Typ double dargestellt werden kann	Konstanten der numerischenTypen
DBL_MIN_10_EXP	Minimaler dezimaler Wert der Exponentenstufung	Konstanten der numerischenTypen
DBL_MIN_EXP	Minimaler binaerer Wert der Exponentenstufung	Konstanten der numerischenTypen
DEAL_COMMENT	Kommentar zum Deal	HistoryDealGetString
DEAL_COMMISSION	Dealkommission	HistoryDealGetDouble
DEAL_ENTRY	Dealsrichtung - Markteingang, Marktausgang oder Kehrwendung	HistoryDealGetInteger
DEAL_ENTRY_IN	Markteingang	HistoryDealGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
DEAL_ENTRY_INOUT	Kehrwendung	HistoryDealGetInteger
DEAL_ENTRY_OUT	Marktausgang	HistoryDealGetInteger
DEAL_MAGIC	Magic number für Deal (sehen Sie ORDER_MAGIC)	HistoryDealGetInteger
DEAL_ORDER	Order, auf deren Grund der Deal abgeschlossen wurde	HistoryDealGetInteger
DEAL_POSITION_ID	Indetifikator der Position, an deren Öffnung, Veränderung oder Schliessung sich der Deal teilnahm. Jede Position hat ihren unikalen Identifikator, der allen Deals zugeordnet wird, die im Instrument	HistoryDealGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	innerhalb des ganzen Lebens der Position abgeschlossen wurde.	
DEAL_PRICE	Dealpreis	HistoryDealGetDouble
DEAL_PROFIT	finanzielles Ergebnis des Deals	HistoryDealGetDouble
DEAL_SWAP	Gesamtswap beim Schliessen	HistoryDealGetDouble
DEAL_SYMBOL	Dealssymbol	HistoryDealGetString
DEAL_TIME	Zeit des Dealabschlusses	HistoryDealGetInteger
DEAL_TIME_MSC	Zeitpunkt der Transaktion in Millisekunden seit 01.01.1970	HistoryDealGetInteger
DEAL_TYPE	Typ des Deals	HistoryDealGetInteger
DEAL_TYPE_BALANCE	Bilanz	HistoryDealGetInteger
DEAL_TYPE_BONUS	Enumeration des Bonus	HistoryDealGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
DEAL_TYPE_BUY	Kauf	HistoryDealGetInteger
DEAL_TYPE_BUY_CANCELED	Abgebrochener Kaufdeal. Es kann eine Situation sein, wenn zuvor abgeschlossener Kaufdeal abgebrochen wird. In solchem Fall ändert sich der Typ des früher abgeschlossenen Deals (DEAL_TYPE_BUY) aufDEAL_TYPE_BUY_CANCELED, und seiner Gewinn/Verlust wird auf Null gesetzt. Zuvor erhaltener Gewinn/Verlust wird auf das	HistoryDealGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Konto mittels gesonderten Bilanzoperation angerechnet/abgebucht	
DEAL_TYPE_CHARGE	zusaetzliche Abzuege	HistoryDealGetInteger
DEAL_TYPE_COMMISSION	Zusätzliche Kommissionen	HistoryDealGetInteger
DEAL_TYPE_COMMISSION_AGENT_DAILY	Agentenkommission, die am Ende des Handelstages angerechnet wird	HistoryDealGetInteger
DEAL_TYPE_COMMISSION_AGENT_MONTHLY	Agentenkommission, die am Ende des Monats angerechnet wird	HistoryDealGetInteger
DEAL_TYPE_COMMISSION_DAILY	Kommission, die am Ende des Handelstages angerechnet wird	HistoryDealGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
DEAL_TYPE_COMMISSION_MONTHLY	Kommission, die am Ende des Monats angerechnet wird	HistoryDealGetInteger
DEAL_TYPE_CORRECTION	Korrektur	HistoryDealGetInteger
DEAL_TYPE_CREDIT	Kredit	HistoryDealGetInteger
DEAL_TYPE_INTEREST	Anrechnungen von Zinsen auf freie Mittel	HistoryDealGetInteger
DEAL_TYPE_SELL	Verkauf	HistoryDealGetInteger
DEAL_TYPE_SELL_CANCELED	Abgebrochener Verkaufdeal. Es kann eine Situation sein, wenn zuvor abgeschlossener Verkaufdeal abgebrochen wird. In solchem Fall ändert sich der Typ des zuvor abgeschlossenen Deals (DEAL_T	HistoryDealGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	YPE_SELL) aufDEAL_TYPE_SELL_CANCELED, und seiner Gewinn/Verlust wird auf Null gesetzt. Zuvor erhaltener Gewinn/Verlust wird auf das Konto mittels gesonderten Bilanzoperation angerechnet/abgebucht	
DEAL_VOLUME	Dealvolumen	HistoryDealGetDouble
DRAW_ARROW	Weichenaufzeichnung	Zeichnungsstile
DRAW_BARS	Darstellung als Folge von Bars	Zeichnungsstile
DRAW_CANDLES	Darstellung als Folge von Kerzen	Zeichnungsstile

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
DRAW_COLOR_ARROW	Zeichnung mit mehrfarbigen Weichen	Zeichnungsstile
DRAW_COLOR_BARS	Mehrfarbige Bars	Zeichnungsstile
DRAW_COLOR_CANDLES	Mehrfarbige Kerzen	Zeichnungsstile
DRAW_COLOR_HISTOGRAM	Mehrfarbiges Histogramm von der Nulllinie	Zeichnungsstile
DRAW_COLOR_HISTOGRAM2	Mehrfarbiges Histogramm der zwei Indikatorpuffer	Zeichnungsstile
DRAW_COLOR_LINE	Mehrfarbige Linie	Zeichnungsstile
DRAW_COLOR_SECTION	Mehrfarbige Abschnitte	Zeichnungsstile
DRAW_COLOR_ZIGZAG	Mehrfarbiger ZigZag	Zeichnungsstile
DRAW_FILLING	Farbenfüllung zwischen zwei Levels	Zeichnungsstile

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
DRAW_HISTOGRAM	Histogramm von der Nulllinie	Zeichnungsstile
DRAW_HISTOGRAM2	Histogramm der 2 Indikatorpuffer	Zeichnungsstile
DRAW_LINE	Linie	Zeichnungsstile
DRAW_NONE	Nicht gezeichnet	Zeichnungsstile
DRAW_SECTION	Abschnitte	Zeichnungsstile
DRAW_ZIGZAG	Stil Zigzag laesst vertikale Abschnitte zu	Zeichnungsstile
ELLIOTT_CYCLE	Zyklus (Cycle)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_GRAND_SUPERCYCLE	Hauptsuperzyklus (Grand Supercycle)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_INTERMEDIATE	Mittelstelle (Intermediate)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_MINOR	Sekundaeres Zyklus (Minor)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_MINUETTE	Sekunde (Minuette)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_MINUTE	Minute (Minute)	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ELLIOTT_PRIMARY	primaeres Zyklus (Primary)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_SUBMINUETTE	Subsekunde (Subminuette)	ObjectSetInteger, ObjectGetInteger
ELLIOTT_SUPERCYCLE	Superzyklus (Supercycle)	ObjectSetInteger, ObjectGetInteger
EMPTY_VALUE	Leerwert im Indikatorpuffer	Andere Konstanten
ERR_ACCOUNT_WRONG_PROPERTY	Falscher Identifikator der Kontoeigenschaft	GetLastError
ERR_ARRAY_BAD_SIZE	Angeforderte Feldgröße ist mehr als 2 Gigabyte	GetLastError
ERR_ARRAY_RESIZE_ERROR	nicht genug Speicherplatz für Neuzuordnung des Feldes oder der Versuch der Größenveränderu	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ng des statischen Feldes	
ERR_BOOKS_CANNOT_ADD	DOM kann nicht zugefügt werden	GetLastError
ERR_BOOKS_CANNOT_DELETE	DOM kann nicht entfernt werden	GetLastError
ERR_BOOKS_CANNOT_GET	DOM Dasten können nicht erhalten werden	GetLastError
ERR_BOOKS_CANNOT_SUBSCRIBE	Fehler in Subskription neue DOM Daten zu erhalten	GetLastError
ERR_BUFFERS_NO_MEMORY	Nicht genug Speicherplatz für Neuzuordnung der Indikatorpuffer	GetLastError
ERR_BUFFERS_WRONG_INDEX	Falscher Index des Indikatorpuffers	GetLastError
ERR_CANNOT_CLEAN_DIRECTORY	Misslungen, das	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Verzeichnis zu löschen (eine oder mehrere Dateien können blockiert werden und Operation der Entfernung ist misslungen)	
ERR_CANNOT_DELETE_DIRECTORY	Verzeichnis kann nicht entfernt werden	GetLastError
ERR_CANNOT_DELETE_FILE	Fehler der Dateientfernung	GetLastError
ERR_CANNOT_OPEN_FILE	Fehler der Dateieröffnung	GetLastError
ERR_CHAR_ARRAY_ONLY	Muss das Feld des Typs char sein	GetLastError
ERR_CHART_CANNOT_CHANGE	Fehler bei Veränderung des Symbols und der Periode des Charts	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ERR_CHART_CANNOT_CREATE_TIMER	Fehler bei Timererzeugung	GetLastError
ERR_CHART_CANNOT_OPEN	Fehler bei Charteröffnung	GetLastError
ERR_CHART_INDICATOR_CANNOT_ADD	Fehler beim Hinzufügen eines Indikators zu einem Chart	GetLastError
ERR_CHART_INDICATOR_CANNOT_DEL	Fehler beim Entfernen des Indikators aus dem Chart	GetLastError
ERR_CHART_INDICATOR_NOT_FOUND	Der Indikator kann nicht auf dem angegebenen Chart gefunden werden	GetLastError
ERR_CHART_NAVIGATE_FAILED	Fehler der Navigierung durch Chart	GetLastError
ERR_CHART_NO_EXPERT	Im Chart gibt es keinen	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Expert Advisor, der das Ereignis verarbeiten kann	
ERR_CHART_NO_REPLY	Chart antwortet nicht	GetLastError
ERR_CHART_NOT_FOUND	Chart ist nicht gefunden	GetLastError
ERR_CHART_SCREENSHOT_FAILED	Fehler bei Screenshot Erzeugung	GetLastError
ERR_CHART_TEMPLATE_FAILED	Fehler der Schablonenverwendung	GetLastError
ERR_CHART_WINDOW_NOT_FOUND	Subfenster mit angegebenen Indikator nicht gefunden	GetLastError
ERR_CHART_WRONG_ID	Falscher Identifikator des Charts	GetLastError
ERR_CHART_WRONG_PARAMETER	Ein fehlerhafter Parameterwert für	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Funktionen für die Arbeit mit dem Chart	
ERR_CHART_WRONG_PROPERTY	Falsche Charteigenschaft ID	GetLastError
ERR_CUSTOM_WRONG_PROPERTY	Falscher Identifikator der Eigenschaft von Benutzerindikator	GetLastError
ERR_DIRECTORY_NOT_EXIST	Verzeichnis existiert nicht	GetLastError
ERR_DOUBLE_ARRAY_ONLY	Muss das Feld des Typs double sein	GetLastError
ERR_FILE_BINSTRINGSIZE	Es muss Zeilengröße angegeben werden, denn die Datei ist als binäre Datei eröffnet	GetLastError
ERR_FILE_CACHEBUFFER_ERROR	Nicht genug Speicherplatz,	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	um Cache zu lesen	
ERR_FILE_CANNOT_REWRITE	Datei kann nicht neugeschrieben werden	GetLastError
ERR_FILE_IS_DIRECTORY	Das ist keine Datei, sondern ein Verzeichnis	GetLastError
ERR_FILE_ISNOT_DIRECTORY	Das ist eine Datei, nicht Verzeichnis	GetLastError
ERR_FILE_NOT_EXIST	Datei existiert nicht	GetLastError
ERR_FILE_NOTBIN	Datei muss als binäre Datei eröffnet werden	GetLastError
ERR_FILE_NOTCSV	Datei muss als CSV eröffnet werden	GetLastError
ERR_FILE_NOTTOREAD	Datei muss für Lesen eröffnet werden	GetLastError
ERR_FILE_NOTTOWRITE	Datei muss für	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Schreiben eröffnet werden	
ERR_FILE_NOTTXT	Datei muss als Textdatei eröffnet werden	GetLastError
ERR_FILE_NOTTXTORCSV	Datei muss als Textdatei oder CSV eröffnet werden	GetLastError
ERR_FILE_READERROR	Fehler des Datei Lesen	GetLastError
ERR_FILE_WRITEERROR	Ressource konnte nicht in die Datei geschrieben werden	GetLastError
ERR_FLOAT_ARRAY_ONLY	Muss das Feld des Typs float sein	GetLastError
ERR_FTP_SEND_FAILED	Fehler beim Dateisenden via ftp	GetLastError
ERR_FUNCTION_NOT_ALLOWED	Systemfunktion ist für Anruf	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	nicht erlaubt	
ERR_GLOBALVARIABLE_EXISTS	Globale Variable des Client-Terminals mit demselben Namen existiert schon	GetLastError
ERR_GLOBALVARIABLE_NOT_FOUND	Globale Variable des Client-Terminals ist nicht gefunden	GetLastError
ERR_HISTORY_NOT_FOUND	Angeforderte Geschichte ist nicht gefunden	GetLastError
ERR_HISTORY_WRONG_PROPERTY	Falscher Identifikator der Geschichteeigenschaft	GetLastError
ERR_INCOMPATIBLE_ARRAYS	Kopieren der unkompatibeler Felder. Zeilenfeld kann nur in ein Zeilenfel	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	d kopiert werden, numerisches Feld- in ein numerisches Feld.	
ERR_INCOMPATIBLE_FILE	für Zeilenfelder muss Textdatei sein, für andere - binäre Datei	GetLastError
ERR_INDICATOR_CANNOT_ADD	Fehler bei Indikatorzufügung	GetLastError
ERR_INDICATOR_CANNOT_APPLY	Indikator kann nicht zum anderen Indikator angewendet werden	GetLastError
ERR_INDICATOR_CANNOT_CREATE	Indikator kann nicht erzeugt werden	GetLastError
ERR_INDICATOR_CUSTOM_NAME	Der erste Parameter im Feld	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	muss der Name des Benutzeranzeigers sein	
ERR_INDICATOR_DATA_NOT_FOUND	Angeforderte Daten nicht gefunden	GetLastError
ERR_INDICATOR_NO_MEMORY	Nicht genug Speicherplatzfür Indikatorzufügung	GetLastError
ERR_INDICATOR_PARAMETER_TYPE	Falscher Parametertyp im Feld bei Anzeiger Erzeugung	GetLastError
ERR_INDICATOR_PARAMETERS_MISSING	Keine Parameter bei Anzeiger Erzeugung	GetLastError
ERR_INDICATOR_UNKNOWN_SYMBOL	Unbekantes Symbol	GetLastError
ERR_INDICATOR_WRONG_HANDLE	Fehlerhandle des Indikators	GetLastError
ERR_INDICATOR_WRONG_INDEX	Fehlerindex des angeford	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	erten Indikatorpuffers	
ERR_INDICATOR_WRONG_PARAMETERS	Falsche Parameterzahl bei Anzeiger Erzeugung	GetLastError
ERR_INT_ARRAY_ONLY	Muss das Feld des Typs int sein	GetLastError
ERR_INTERNAL_ERROR	Unerwarteter interner Fehler	GetLastError
ERR_INVALID_ARRAY	Feld des ungültigen Typs, der ungültiger Größe oder beschädigtes Objekt des dynamischen Feldes	GetLastError
ERR_INVALID_DATETIME	Ungültiger Wert des Datums und/oder der Zeit An	GetLastError
ERR_INVALID_FILEHANDLE	Datei mit diesem handle ist schon	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	geschlossen oder war ganz nicht eröffnet	
ERR_INVALID_PARAMETER	Fehlerparameter beim Aufruf der Systemfunktion	GetLastError
ERR_INVALID_POINTER	Ungültiger Anzeiger	GetLastError
ERR_INVALID_POINTER_TYPE	Falscher Typ des Anzeigers	GetLastError
ERR_LONG_ARRAY_ONLY	Muss das Feld des Typs long sein	GetLastError
ERR_MAIL_SEND_FAILED	Fehler beim Mailsenden	GetLastError
ERR_MARKET_LASTTIME_UNKNOWN	Zeit des Letzten Ticks ist unbekannt (es gab keine Ticks)	GetLastError
ERR_MARKET_NOT_SELECTED	Symbol nicht gewählt in Market Watch	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ERR_MARKET_SELECT_ERROR	Fehler bei Hinzufügung oder Löschung eines Symbols in Market Watch	GetLastError
ERR_MARKET_UNKNOWN_SYMBOL	Unbekanntes Symbol	GetLastError
ERR_MARKET_WRONG_PROPERTY	Fehleridentifikator der Symboleigenschaft	GetLastError
ERR_MQL5_WRONG_PROPERTY	Falscher Identifikator der Programmeigenschaft	GetLastError
ERR_NO_STRING_DATE	In der Zeile gibt es kein Datum	GetLastError
ERR_NOT_ENOUGH_MEMORY	Nicht genug Speicherplatz für Ausführung der Systemfunktion	GetLastError
ERR_NOTIFICATION_SEND_FAILED	Benachrichtigung konnte nicht gesendet werden	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ERR_NOTIFICATION_TOO_FREQUENT	Zu häufige Zusendung von Benachrichtigungen	GetLastError
ERR_NOTIFICATION_WRONG_PARAMETER	Ungültige Parameter, um Benachrichtigung zu senden - ein leerer String oder NULL war in die Funktion SendNotification() übergeben	GetLastError
ERR_NOTIFICATION_WRONG_SETTINGS	Falsche Einstellung von Benachrichtigungen im Terminal (ID ist nicht eingegeben oder keine Berechtigung ist aufgesetzt)	GetLastError
ERR_NOTINITIALIZED_STRING	nicht initialisi	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	erte Zeile	
ERR_NUMBER_ARRAYS_ONLY	Muss numerisches Feld sein	GetLastError
ERR_OBJECT_ERROR	Fehler bei Arbeit mit dem graphischen Objekt	GetLastError
ERR_OBJECT_GETDATE_FAILED	Datum entsprechend dem Wert kann nicht erhalten werden	GetLastError
ERR_OBJECT_GETVALUE_FAILED	Wert entsprechend dem Datum kann nicht erhalten werden	GetLastError
ERR_OBJECT_NOT_FOUND	Graphisches Objekt ist nicht gefunden	GetLastError
ERR_OBJECT_WRONG_PROPERTY	Fehleridentifikator der Eigenschaft des graphischen	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Objekte s	
ERR_ONEDIM_ARRAYS_ONLY	Muss eindimensionales s Feld sein	GetLastError
ERR_OPENCL_BUFFER_CREATE	Fehler beim Erstellen des OpenCL-Puffers	GetLastError
ERR_OPENCL_CONTEXT_CREATE	Fehler beim Erstellen von OpenCL-Kontext	GetLastError
ERR_OPENCL_EXECUTE	Laufzeitfehler vom OpenCL-Programm	GetLastError
ERR_OPENCL_INTERNAL	Interner Fehler bei der Ausführung von OpenCL	GetLastError
ERR_OPENCL_INVALID_HANDLE	Ungültiges OpenCL-Handle	GetLastError
ERR_OPENCL_KERNEL_CREATE	Fehler beim Erstellen vom OpenCL-Kernel	GetLastError
ERR_OPENCL_NOT_SUPPORTED	OpenCL-Funktion	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	en werden auf diesem Computer nicht unterstützt	
ERR_OPENCL_PROGRAM_CREATE	Fehler beim Kompilieren eines OpenCL-Programms	GetLastError
ERR_OPENCL_QUEUE_CREATE	Es konnte kein Lauf-Warteschlange in OpenCL erstellen	GetLastError
ERR_OPENCL_SET_KERNEL_PARAMETER	Fehler bei Eingabe von Parametern für den OpenCL-Kernel	GetLastError
ERR_OPENCL_TOO_LONG_KERNEL_NAME	Zu lange Kernel-Name(OpenCL-Kernel)	GetLastError
ERR_OPENCL_WRONG_BUFFER_OFFSET	Ungültiges Offset im OpenCL-Puffer	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ERR_OPENCL_WRONG_BUFFER_SIZE	Ungültige Größe des OpenCL-Puffers	GetLastError
ERR_PLAY_SOUND_FAILED	Fehler beim Lautabspielen	GetLastError
ERR_RESOURCE_NAME_DUPLICATED	Die Namen der dynamischen und statischen Ressource sind gleich	GetLastError
ERR_RESOURCE_NAME_IS_TOO_LONG	Der Name der Ressource ist mehr als 63 Zeichen	GetLastError
ERR_RESOURCE_NOT_FOUND	Ressource mit diesem Namen ist nicht in EX5 gefunden	GetLastError
ERR_RESOURCE_UNSUPPORTED_TYPE	Nicht unterstützte Ressource-Typ oder Größe von mehr als 16 MB	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ERR_SERIES_ARRAY	Timeserie kann nicht verwendet werden	GetLastError
ERR_SHORT_ARRAY_ONLY	Muss das Feld des Typs short sein	GetLastError
ERR_SMALL_ARRAY	Zu kleines Feld, Startposition ausserhalb des Feldes	GetLastError
ERR_SMALL_ASSERIES_ARRAY	Aufnahmefeld ist als AS_SERIES erklärt, und es ist von der unzureichender Größe	GetLastError
ERR_STRING_OUT_OF_MEMORY	Nicht genug Speicherplatz für Zeile	GetLastError
ERR_STRING_RESIZE_ERROR	Nicht genug Speicherplatz für Neuzuordnung der Zeile	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ERR_STRING_SMALL_LEN	Zeilenlänge ist kleiner als erwartet	GetLastError
ERR_STRING_TIME_ERROR	Fehler der Umwandlung der Zeile in Datum	GetLastError
ERR_STRING_TOO_BIGNUMBER	Zu lange Zahl, länger als ULONG_MAX	GetLastError
ERR_STRING_UNKNOWNTYPE	Unbekannter Datentyp bei Umwandlung in die Zeile	GetLastError
ERR_STRING_ZEROADDED	Zum Zeilenende ist 0 zugefuegt, sinnlose Operation	GetLastError
ERR_STRINGPOS_OUTOFRANGE	Position ausserhalb der Zeile	GetLastError
ERR_STRUCT_WITHOBJECTS_ORCLASS	Struktur hat Zeilenobjekte und/oder Objekte	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	der dynamischen Felder und/oder Strukturen mit solchen Objekten und/oder Klassen	
ERR_SUCCESS	Operation erfolgreich ausgeführt	GetLastError
ERR_TERMINAL_WRONG_PROPERTY	Falscher Identifikator der Terminaleigenschaft	GetLastError
ERR_TOO_LONG_FILENAME	Zu langer Dateiname	GetLastError
ERR_TOO_MANY_FILES	Mehr als 64 Dateien können nicht gleichzeitig eröffnet werden	GetLastError
ERR_TOO_MANY_FORMATTERS	Zahl der Formatspezifikatoren ist mehr, als	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Parametererzahl	
ERR_TOO_MANY_PARAMETERS	Parametererzahl ist mehr als Zahl der Formatspezifikatoren	GetLastError
ERR_TRADE_DEAL_NOT_FOUND	Deal ist nicht gefunden	GetLastError
ERR_TRADE_DISABLED	Handel für Expert ist verboten	GetLastError
ERR_TRADE_ORDER_NOT_FOUND	Order ist nicht gefunden	GetLastError
ERR_TRADE_POSITION_NOT_FOUND	Position ist nicht gefunden	GetLastError
ERR_TRADE_SEND_FAILED	Fehler beim Senden der Handelsanforderung	GetLastError
ERR_TRADE_WRONG_PROPERTY	Falscher Identifikator der Handelseigenschaft	GetLastError
ERR_USER_ERROR_FIRST	Mit diesem	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Kode fangen benutzerdefinierte Fehler an	
ERR_WEBREQUEST_CONNECT_FAILED	Verbindung mit der angegebenen URL fehlgeschlagen	GetLastError
ERR_WEBREQUEST_INVALID_ADDRESS	Ungültige URL	GetLastError
ERR_WEBREQUEST_REQUEST_FAILED	Bei der Ausführung der HTTP-Anfrage ist ein Fehler aufgetreten	GetLastError
ERR_WEBREQUEST_TIMEOUT	Zeitliches Limit für das Erhalten von Daten überschritten	GetLastError
ERR_WRONG_DIRECTORYNAME	Falscher Verzeichnisname	GetLastError
ERR_WRONG_FILEHANDLE	Falsches handle der Datei	GetLastError
ERR_WRONG_FILENAME	Ungültiger	GetLastError

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Dateiname	
ERR_WRONG_FORMATSTRING	Falsche Formatzeile	GetLastError
ERR_WRONG_INTERNAL_PARAMETER	Fehlerparameter beim internen Aufruf des Client-Terminals	GetLastError
ERR_WRONG_STRING_DATE	Falsches Datum in der Zeile	GetLastError
ERR_WRONG_STRING_OBJECT	Beschädigtes Zeilenobjekt	GetLastError
ERR_WRONG_STRING_PARAMETER	Beschädigter Parameter des Typs	GetLastError
ERR_WRONG_STRING_TIME	Falsche Zeit in der Zeile	GetLastError
ERR_ZEROSIZE_ARRAY	Feld der Nullgröße	GetLastError
FILE_ACCESS_DATE	Datum der letzten Zugriff zur Datei	FileGetInteger
FILE_ANSI	Zeilen des Typs	FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ANSI (1-Byte Symbole). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet	
FILE_BIN	Binares Regime von Lesen-Shreiben (ohne Umwandlung von Zeile in die Zeile). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet	FileOpen
FILE_COMMON	Datei /// im Verzeichnis aller Client-Terminals \Terminal\Common\Files. Flagge wird	FileOpen, FileCopy, FileMove, FileIsExist

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	beim Oeffnen von Dateien (FileOpen()), Dateikopieren (FileCopy(), FileMove()) und Pruefung von Dateianesenheit (FileIsExist()) verwendet	
FILE_CREATE_DATE	Erstellungsdatum	FileGetInteger
FILE_CSV	Datei des Typs csv (alle geschriebenen Elemente werden in die Zeilen des entsprechenden Typs verwendet, unicode oder ansi, und durch Begrenzungszeichen	FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	getrennt). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet	
FILE_END	Zeichen der Dateiende erhalten	FileGetInteger
FILE_EXISTS	Prüfung der Existenz	FileGetInteger
FILE_IS_ANSI	Die Datei ist als ANSI geöffnet (Sehen Sie FILE_ANSI)	FileGetInteger
FILE_IS_BINARY	Die Datei ist als eine binäre Datei geöffnet (Sehen Sie FILE_BIN)	FileGetInteger
FILE_IS_COMMON	Die Datei wird in einem freigegebenen Ordner für allen	FileGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Terminals geöffnet (Sehen Sie FILE_COMMON)	
FILE_IS_CSV	Die Datei ist als CSV geöffnet (Sehen Sie FILE_CSV)	FileGetInteger
FILE_IS_READABLE	Die geöffnete Datei ist lesbar (Sehen Sie FILE_READ)	FileGetInteger
FILE_IS_TEXT	Die Datei ist als eine Textdatei geöffnet (Sehen Sie FILE_TXT)	FileGetInteger
FILE_IS_WRITABLE	Die geöffnete Datei ist beschreibbar (Sehen Sie FILE_WRITE)	FileGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
FILE_LINE_END	Zeichen der Zeilenende erhalten	FileGetInteger
FILE_MODIFY_DATE	Datum der letzten Überarbeitung	FileGetInteger
FILE_POSITION	Position eines Zeigers in der Datei	FileGetInteger
FILE_READ	Datei wird für Lesen geoeffnet. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet. Bei der Öffnung der Datei muss unbedingt die Flagge FILE_WRITE und/oder die Flagge FILE_READ angegeben wird.	FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
FILE_REWRITE	Moeglichket der Dateiumspeicherung durch die Funktionen FileCopy() undnden sein oder für FileMove(). Datei muss vorhanden sein oder für Schreiben geoeffnet werden. Anders wird die Datei nicht geoeffnet werden	FileCopy, FileMove
FILE_SHARE_READ	Gemeinsamer Lesezugang von mehreren Progarmmen. Flagge wird beim Oeffnen von Dateien (FileOpen())	FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	verwendet, aber ersetzt es nicht die Notwendigkeit, beim Dateieröffneng FILE_WRITE und/oder FILE_READ anzugeben	
FILE_SHARE_WRITE	Gemeinsamer Schreibezugang von mehreren Programmen. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet, aber ersetzt es nicht die Notwendigkeit, beim Dateieröffneng FILE_WRITE	FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	und/oder FILE_READ anzugeben	
FILE_SIZE	Dateigröße in Byte	FileGetInteger
FILE_TXT	Einfache Textdatei (ebensolcher wie csv, aber der Begrenzungszeichen wird nicht beachtet). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet	FileOpen
FILE_UNICODE	Zeilen des Typs UNICODE (2-Byte-Symbole). Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet	FileOpen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
FILE_WRITE	Dateiwird für Schreiben verwendet. Flagge wird beim Oeffnen von Dateien (FileOpen()) verwendet. Bei der Öffnung der Datei muss unbedingt die Flagge FILE_WRITE und/oder die Flagge FILE_READ angegeben wird.	FileOpen
FLT_DIG	Anzahl der bedeutenden Dezimalzeichen	Konstanten der numerischenTypen
FLT_EPSILON	Der minimale Wert, für den die Bedingung	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	1.0+FLT_EPSILON != 1.0 erfuellt wird	
FLT_MANT_DIG	Anzahl der Bits in mantissa	Konstanten der numerischenTypen
FLT_MAX	Maximaler Wert, der durch den Typ float repraesentiert werden kann	Konstanten der numerischenTypen
FLT_MAX_10_EXP	Maximaler dezimaler Wert der Exponentenstufung	Konstanten der numerischenTypen
FLT_MAX_EXP	Maximaler binaerer Wert der Exponentenstufung	Konstanten der numerischenTypen
FLT_MIN	Minimaler positiver Wert, der durch den Typ float repraesentiert	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	werden kann	
FLT_MIN_10_EXP	Minimaler dezimaler Wert der Exponentenstufung	Konstanten der numerischenTypen
FLT_MIN_EXP	Minimaler binaerer Wert der Exponentenstufung	Konstanten der numerischenTypen
FRIDAY	Freitag	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
GANN_DOWN_TREND	Linie entspricht dem Fallenden Trend	ObjectSetInteger, ObjectGetInteger
GANN_UP_TREND	Linie entspricht dem steidenden en Trend	ObjectSetInteger, ObjectGetInteger
GATORJAW_LINE	Linie der Kiefer	Linien der Indikatoren
GATORLIPS_LINE	Linie der Lippen	Linien der Indikatoren
GATORTEETH_LINE	Linie der Zaehne	Linien der Indikatoren
IDABORT	Schaltfläche Abort wird	MessageBox

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ausgewählt	
IDCANCEL	Schaltfläche Cancel wird ausgewählt	MessageBox
IDCONTINUE	Schaltfläche Continue wird ausgewählt	MessageBox
IDIGNORE	Schaltfläche Ignore wird ausgewählt	MessageBox
IDNO	Schaltfläche No wird ausgewählt	MessageBox
IDOK	Schaltfläche OK wird ausgewählt	MessageBox
IDRETRY	Schaltfläche Retry wird ausgewählt	MessageBox
IDTRYAGAIN	Schaltfläche Try Again wird ausgewählt	MessageBox

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
IDYES	Schaltfläche Yes wird ausgewählt	MessageBox
IND_AC	Accelerator Oscillator	IndicatorCreate, IndicatorParameters
IND_AD	Accumulation/Distribution	IndicatorCreate, IndicatorParameters
IND_ADX	Average Directional Index	IndicatorCreate, IndicatorParameters
IND_ADXW	ADX by Welles Wilder	IndicatorCreate, IndicatorParameters
IND_ALLIGATOR	Alligator	IndicatorCreate, IndicatorParameters
IND_AMA	Adaptive Moving Average	IndicatorCreate, IndicatorParameters
IND_AO	Awesome Oscillator	IndicatorCreate, IndicatorParameters
IND_ATR	Average True Range	IndicatorCreate, IndicatorParameters
IND_BANDS	Bollinger Bands®	IndicatorCreate, IndicatorParameters
IND_BEARS	Bears Power	IndicatorCreate, IndicatorParameters
IND_BULLS	Bulls Power	IndicatorCreate, IndicatorParameters
IND_BWMFI	Market Facilitation Index	IndicatorCreate, IndicatorParameters

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
IND_CCI	Commodity Channel Index	IndicatorCreate, IndicatorParameters
IND_CHAIKIN	Chaikin Oscillator	IndicatorCreate, IndicatorParameters
IND_CUSTOM	Custom indicator	IndicatorCreate, IndicatorParameters
IND_DEMA	Double Exponential Moving Average	IndicatorCreate, IndicatorParameters
IND_DEMARKER	DeMarker	IndicatorCreate, IndicatorParameters
IND_ENVELOPES	Envelopes	IndicatorCreate, IndicatorParameters
IND_FORCE	Force Index	IndicatorCreate, IndicatorParameters
IND_FRACTALS	Fractals	IndicatorCreate, IndicatorParameters
IND_FRAMA	Fractal Adaptive Moving Average	IndicatorCreate, IndicatorParameters
IND_GATOR	Gator Oscillator	IndicatorCreate, IndicatorParameters
IND_ICHIMOKU	Ichimoku Kinko Hyo	IndicatorCreate, IndicatorParameters
IND_MA	Moving Average	IndicatorCreate, IndicatorParameters
IND_MACD	MACD	IndicatorCreate, IndicatorParameters
IND_MFI	Money Flow Index	IndicatorCreate, IndicatorParameters
IND_MOMENTUM	Momentum	IndicatorCreate, IndicatorParameters

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
IND_OBV	On Balance Volume	IndicatorCreate, IndicatorParameters
IND_OSMA	OsMA	IndicatorCreate, IndicatorParameters
IND_RSI	Relative Strength Index	IndicatorCreate, IndicatorParameters
IND_RVI	Relative Vigor Index	IndicatorCreate, IndicatorParameters
IND_SAR	Parabolic SAR	IndicatorCreate, IndicatorParameters
IND_STDDEV	Standard Deviation	IndicatorCreate, IndicatorParameters
IND_STOCHASTIC	Stochastic Oscillator	IndicatorCreate, IndicatorParameters
IND_TEMA	Triple Exponential Moving Average	IndicatorCreate, IndicatorParameters
IND_TRIX	Triple Exponential Moving Averages Oscillator	IndicatorCreate, IndicatorParameters
IND_VIDYA	Variable Index Dynamic Average	IndicatorCreate, IndicatorParameters
IND_VOLUMES	Volumes	IndicatorCreate, IndicatorParameters
IND_WPR	Williams' Percent Range	IndicatorCreate, IndicatorParameters

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
INDICATOR_CALCULATIONS	Hilfspuffer für Zwischenberechnungen	SetIndexBuffer
INDICATOR_COLOR_INDEX	Farbe	SetIndexBuffer
INDICATOR_DATA	Angaben für Zeichnen	SetIndexBuffer
INDICATOR_DIGITS	Genauigkeit der Darstellung von Indikatorwerten	IndicatorSetInteger
INDICATOR_HEIGHT	Feste Höhe von Indikators eigenen Fenster (Befehl von Preprozessor #property indicator_height)	IndicatorSetInteger
INDICATOR_LEVELCOLOR	Farbe der Levellinie	IndicatorSetInteger
INDICATOR_LEVELS	Anzahl der Levels im Indikatorfenster	IndicatorSetInteger
INDICATOR_LEVELSTYLE	Stil der Levellini	IndicatorSetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	e	
INDICATOR_LEVELTEXT	Levelbeschreibung	IndicatorSetString
INDICATOR_LEVELVALUE	Levelwert	IndicatorSetDouble
INDICATOR_LEVELWIDTH	Breite der Levellinie	IndicatorSetInteger
INDICATOR_MAXIMUM	Maximum des Indikatorfensters	IndicatorSetDouble
INDICATOR_MINIMUM	Minimum des Indikatorfensters	IndicatorSetDouble
INDICATOR_SHORTNAME	Kurzer Indikatorname	IndicatorSetString
INT_MAX	Maximaler Wert, der durch den Typ int dargestellt werden kann	Konstanten der numerischenTypen
INT_MIN	Minimaler Wert, der durch den Typ int dargestellt werden kann int	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
INVALID_HANDLE	unkorrektes handle	Andere Konstanten
IS_DEBUG_MODE	Flagge der Arbeit des mq5-Programm in Debugging-Mode	Andere Konstanten
IS_PROFILE_MODE	Flagge der Arbeit des mq5-Programms im Modus von Code Profiling	Andere Konstanten
KIJUNSEN_LINE	Linie Kijun-sen	Linien der Indikatoren
LICENSE_DEMO	Eine Testversion von einem bezahlten Produkt vom Markt. Es funktioniert nur im Strategie-Tester	MQLInfoInteger
LICENSE_FREE	Eine Kostenlose	MQLInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Vollversion	
LICENSE_FULL	Eine gekaufte lizenzierte Version erlaubt mindestens fünf Aktivierungen. Der Verkäufer kann die Anzahl der zulässigen Aktivierungen erhöhen	MQLInfoInteger
LICENSE_TIME	Eine Version mit der zeitlich begrenzten Lizenz	MQLInfoInteger
LONG_MAX	Maximaler Wert, der durch den Typ long dargestellt werden kann	Konstanten der numerischenTypen
LONG_MIN	Minimaler Wert, der durch den Typ	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	long dargestellt werden kann	
LOWER_BAND	Untere Grenze	Linien der Indikatoren
LOWER_HISTOGRAM	Unteres Histogramm	Linien der Indikatoren
LOWER_LINE	Untere Linie	Linien der Indikatoren
M_1_PI	1/pi	Mathematische Konstanten
M_2_PI	2/pi	Mathematische Konstanten
M_2_SQRTPI	2/sqrt(pi)	Mathematische Konstanten
M_E	e	Mathematische Konstanten
M_LN10	ln(10)	Mathematische Konstanten
M_LN2	ln(2)	Mathematische Konstanten
M_LOG10E	log10(e)	Mathematische Konstanten
M_LOG2E	log2(e)	Mathematische Konstanten
M_PI	pi	Mathematische Konstanten
M_PI_2	pi/2	Mathematische Konstanten
M_PI_4	pi/4	Mathematische Konstanten
M_SQRT1_2	1/sqrt(2)	Mathematische Konstanten
M_SQRT2	sqrt(2)	Mathematische Konstanten
MAIN_LINE	Grundlinie	Linien der Indikatoren
MB_ABORTRETRYIGNORE	Message Window hat drei Schaltflächen: Abort, Retry	MessageBox

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	und Ignore	
MB_CANCELTRYCONTINUE	Message Window hat drei Schaltflächen: Cancel, Try Again, Continue	MessageBox
MB_DEFBUTTON1	Die erste Schaltfläche MB_DEFBUTTON1 - ist Default, wenn MB_DEFBUTTON2, MB_DEFBUTTON3, oder MB_DEFBUTTON4 nicht spezifiziert werden	MessageBox
MB_DEFBUTTON2	zweite Schaltfläche - Default-Wert	MessageBox
MB_DEFBUTTON3	dritte Schaltfläche - Default-Wert	MessageBox
MB_DEFBUTTON4	Vierte Schaltfläche -	MessageBox

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Default-Wert	
MB_ICONEXCLAMATION, MB_ICONWARNING	Darstellung von Aufrufzeichen	MessageBox
MB_ICONINFORMATION, MB_ICONASTERISK	Darstellung aus dem eingekreisten Zeichen i	MessageBox
MB_ICONQUESTION	Darstellung von Fragezeichen	MessageBox
MB_ICONSTOP, MB_ICONERROR, MB_ICONHAND	Darstellung von STOP Zeichen	MessageBox
MB_OK	Message Window hat eine Schaltfläche: OK. Default	MessageBox
MB_OKCANCEL	Message Window hat zwei Schaltflächen: OK und Cancel	MessageBox
MB_RETRYCANCEL	Message Window hat zwei Schaltflächen: Retry und Cancel	MessageBox

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
MB_YESNO	Message Window hat zwei Schaltflächen: Yes und No	MessageBox
MB_YESNOCANCEL	Message Window hat drei Schaltflächen: Yes, No und Cancel	MessageBox
MINUSDI_LINE	Linie -DI	Linien der Indikatoren
MODE_EMA	exponentielle Mittelung	Glaettungsmethoden
MODE_LWMA	lineal ausgewogene Mittelung	Glaettungsmethoden
MODE_SMA	einfache Mittelung	Glaettungsmethoden
MODE_SMMA	geglaettete Mittelung	Glaettungsmethoden
MONDAY	Montag	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
MQL_DEBUG	Flagge, das Debugging Mode des Programms angibt	MQLInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
MQL_DLLS_ALLOWED	Erlaubnis, DLL für das vorgegebene ausgeführte Programm zu verwenden	MQLInfoInteger
MQL_FRAME_MODE	Zeichen, dass der auf dem Chart laufende Experte in Frames von Optimierung-Ergebnisse sammelt	MQLInfoInteger
MQL_LICENSE_TYPE	Art der Lizenz des EX5-Moduls. Die Lizenz bezieht sich auf den EX5-Modul, aus denen ein Antrag mit MQLInfoInteger (MQL_LICENSE_TYPE)	MQLInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	gestellt wird	
MQL_MEMORY_LIMIT	Die maximale Größe der dynamischen Speicher für eine MQL5-Programme in MB	MQLInfoInteger
MQL_MEMORY_USED	Die Größe der von MQL5-Programme verwendeten Speicher in MB	MQLInfoInteger
MQL_OPTIMIZATION	Flagge der Arbeit des ausgeführten Programms im Prozess der Optimierung	MQLInfoInteger
MQL_PROFILER	Flagge der Arbeit des ausgeführten Programms im Modus	MQLInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	von Code Profiling	
MQL_PROGRAM_NAME	Name des ausgeführten MQL5-Programms	MQLInfoString
MQL_PROGRAM_PATH	Pfad für das vorgegebene ausgeführte Programm	MQLInfoString
MQL_PROGRAM_TYPE	Typ des MQL5-Programms	MQLInfoInteger
MQL_SIGNALS_ALLOWED	Erlaubnis für dieses laufenden Programm mit Signale zu arbeiten	MQLInfoInteger
MQL_TESTER	Flagge der Arbeit des ausgeführten Programms im Tester	MQLInfoInteger
MQL_TRADE_ALLOWED	Erlaubnis für das vorgege	MQLInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	bene ausgeführte Programm zu handeln	
MQL_VISUAL_MODE	Flagge der Arbeit des ausgeführten Programms im visuellen Testenmode	MQLInfoInteger
NULL	Null des jeden Typs	Andere Konstanten
OBJ_ALL_PERIODS	Objekt wird auf allen Timeframes gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_ARROW	Objekt "Weiche"	Objekttypen
OBJ_ARROW_BUY	Zeichen "Buy"	Objekttypen
OBJ_ARROW_CHECK	Zeichen "Kontrollmarke"	Objekttypen
OBJ_ARROW_DOWN	Zeichen "Weich unten"	Objekttypen
OBJ_ARROW_LEFT_PRICE	linke Preismarke	Objekttypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJ_ARROW_RIGHT_PRICE	Rechte Preismarke	Objekttypen
OBJ_ARROW_SELL	Zeichen "Sell"	Objekttypen
OBJ_ARROW_STOP	Zeichen "Stop"	Objekttypen
OBJ_ARROW_THUMB_DOWN	Zeichen "Schlecht" (Daumen unten)	Objekttypen
OBJ_ARROW_THUMB_UP	Zeichen "Gut" (Daumen oben)	Objekttypen
OBJ_ARROW_UP	Zeichen "Weiche oben"	Objekttypen
OBJ_ARROWED_LINE	Objekt "Linie mit einem Pfeil"	Objekttypen
OBJ_BITMAP	Objekt "Bild"	Objekttypen
OBJ_BITMAP_LABEL	Objekt "graphische Marke"	Objekttypen
OBJ_BUTTON	Objekt "Schaltfläche"	Objekttypen
OBJ_CHANNEL	Abstandsgleicher Kanal	Objekttypen
OBJ_CHART	Objekt "Chart"	Objekttypen
OBJ_CYCLES	Zykluslinien	Objekttypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJ_EDIT	Objekt "Eingabefeld"	Objekttypen
OBJ_ELLIOTWAVE3	Elliott Correction Wave	Objekttypen
OBJ_ELLIOTWAVE5	Elliott Motive Wave	Objekttypen
OBJ_ELLIPSE	Ellipse	Objekttypen
OBJ_EVENT	Objekt "Ereignis", das einem Ereignis in der wirtschaftlichen Kalender entspricht	Objekttypen
OBJ_EXPANSION	Erweiterung Fibonacci	Objekttypen
OBJ_FIBO	Fibonacci Retracement	Objekttypen
OBJ_FIBOARC	Fibonacci Arcs	Objekttypen
OBJ_FIBOCHANNEL	Kanal Fibonacci	Objekttypen
OBJ_FIBOFAN	Fibonacci Fan	Objekttypen
OBJ_FIBOTIMES	Zeitzonen Fibonacci	Objekttypen
OBJ_GANNFAN	Gannfan	Objekttypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJ_GANNGRID	Ganngrid	Objekttypen
OBJ_GANNLINE	Gannlinie	Objekttypen
OBJ_HLINE	horizontale Linie	Objekttypen
OBJ_LABEL	Objekt "Textmarke"	Objekttypen
OBJ_NO_PERIODS	Objekt wird auf keiner Timeframe angezeigt	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_D1	Objekt wird auf Tageschart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_H1	Objekt wird auf 1-Stunde Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_H12	Objekt wird auf 12-Stunden Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_H2	Objekt wird auf 2-Stunden Chart gezeichnet	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJ_PERIOD_H3	Objekt wird auf 3-Stunden Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_H4	Objekt wird auf 4-Stunden Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_H6	Objekt wird auf 6-Stunden Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_H8	Objekt wird auf 8-Stunden Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M1	Objekt wird auf 1-Minute Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M10	Objekt wird auf 10-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M12	Objekt wird auf 12-	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Minuten Chart gezeichnet	
OBJ_PERIOD_M15	Objekt wird auf 15-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M2	Objekt wird auf 2-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M20	Objekt wird auf 20-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M3	Objekt wird auf 3-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M30	Objekt wird auf 30-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M4	Objekt wird auf 4-Minuten Chart	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	gezeichnet	
OBJ_PERIOD_M5	Objekt wird auf 5-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_M6	Objekt wird auf 6-Minuten Chart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_MN1	Objekt wird auf Monatschart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PERIOD_W1	Objekt wird auf Wochenchart gezeichnet	ObjectSetInteger, ObjectGetInteger
OBJ_PITCHFORK	Andrews' Pitchfork	Objekttypen
OBJ_RECTANGLE	Rechteck	Objekttypen
OBJ_RECTANGLE_LABEL	Objekt "Rechteckige Marke" um eigene grafische Benutzer	Objekttypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	oberfläche zu entwerten.	
OBJ_REGRESSION	Kanal der linearen Regression	Objekttypen
OBJ_STDDEVCHANNEL	Kanal der Standardabweichung	Objekttypen
OBJ_TEXT	Objekt "Text"	Objekttypen
OBJ_TREND	Trendlinie	Objekttypen
OBJ_TRENDBYANGLE	Trendlinie nach Winkel	Objekttypen
OBJ_TRIANGLE	Dreieck	Objekttypen
OBJ_VLINE	Senkrechte Linie	Objekttypen
OBJPROP_ALIGN	Horizontale Textausrichtung in der "Edit"-Objekt (OBJ_EDIT)	ObjectSetInteger, ObjectGetInteger
OBJPROP_ANCHOR	Stellung des Bezugspunktes des graphischen Objekts	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJPROP_ANGLE	Winkel. Für die aus einem Programm erstellten Objekte, für die kein Winkel angegeben ist, ist der Wert gleich EMPTY_VALUE	ObjectSetDouble, ObjectGetDouble
OBJPROP_ARROWCODE	Weichenkode für Objekt "Weiche"	ObjectSetInteger, ObjectGetInteger
OBJPROP_BACK	Objekt auf dem Hintergrund	ObjectSetInteger, ObjectGetInteger
OBJPROP_BGCOLOR	Hintergrundfarbe für OBJ_EDIT, OBJ_BUTTON, OBJ_RECTANGLE_LABEL	ObjectSetInteger, ObjectGetInteger
OBJPROP_BMPFILE	Name der BMP-Datei für Objekt "Graphische Marke".	ObjectSetString, ObjectGetString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Sehen Sie auch Ressourcen	
OBJPROP_BORDER_COLOR	Die Rahmenfarbe für die Objekte OBJ_EDIT und OBJ_BUTTON	ObjectSetInteger, ObjectGetInteger
OBJPROP_BORDER_TYPE	Frame-Typ für Object "Rechteckige Marke"	ObjectSetInteger, ObjectGetInteger
OBJPROP_CHART_ID	ID von Objekt "Chart" (OBJ_CHART). Es erlaubt mit den Eigenschaften dieses Objektes als einer regelmäßigen Diagramm mit den Funktionen der Sektion Operationen mit Charts arbeiten, aber es gibt einige	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Ausnahmen.	
OBJPROP_CHART_SCALE	Massstab für Objekt "Chart"	ObjectSetInteger, ObjectGetInteger
OBJPROP_COLOR	Farbe	ObjectSetInteger, ObjectGetInteger
OBJPROP_CORNER	Winkel des Charts für Snap des Graphischen Objektes	ObjectSetInteger, ObjectGetInteger
OBJPROP_CREATETIME	Erzeugungszeit des Objekts	ObjectSetInteger, ObjectGetInteger
OBJPROP_DATE_SCALE	Merkmal der Darstellung von Preisskala für Objekt "Chart"	ObjectSetInteger, ObjectGetInteger
OBJPROP_DEGREE	Level von Elliott Wave Markierung	ObjectSetInteger, ObjectGetInteger
OBJPROP_DEVIATION	Abweichung für Kanal der Standardabweichung	ObjectSetDouble, ObjectGetDouble
OBJPROP_DIRECTION	Trend des	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Gann Objekts	
OBJPROP_DRAWLINES	Liniendarstellung für Elliott Wave Markierung	ObjectSetInteger, ObjectGetInteger
OBJPROP_ELLIPSE	Anzeigen die ganzen Ellipse für das Objekt "Fibonacci Arcs" (OBJ_FIBOARC)	ObjectSetInteger, ObjectGetInteger
OBJPROP_FILL	Füllen ein Objekt mit Farbe (für OBJ_RECTANGLE, OBJ_TRIANGLE, OBJ_ELLIPSE, OBJ_CHANNEL, OBJ_STDDEVCHANNEL, OBJ_REGRESSION)	ObjectSetInteger, ObjectGetInteger
OBJPROP_FONT	Schriftart	ObjectSetString, ObjectGetString
OBJPROP_FONTSIZE	Schriftgröße	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJPROP_HIDDEN	Anzeige des Namens von graphischem Objekt in der Objektliste aus dem Terminal-Menü "Charts" - "Objekte" - "Objektliste". Wert true ermöglicht es,ein Objekt aus der Liste zu verstecken. Standardmäßig wird true für die Objekte, die Kalenderereignisse und Handelsgeschichte zeigen, und auch für die Objekte, die aus MQL5-Program	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	men erstellt werden, angegeben. Um solche graphischen Objekte graphischen Objekte zu sehen und Zugang zu ihren Eigenschaften zu haben, klicken Sie auf "Alle" im Fenster "Objektliste".	
OBJPROP_LEVELCOLOR	Farbe von Linie-Level	ObjectSetInteger, ObjectGetInteger
OBJPROP_LEVELS	Levelszahl	ObjectSetInteger, ObjectGetInteger
OBJPROP_LEVELSTYLE	Stil der Linie-Level	ObjectSetInteger, ObjectGetInteger
OBJPROP_LEVELTEXT	Levelbeschreibung	ObjectSetString, ObjectGetString
OBJPROP_LEVELVALUE	Levelwert	ObjectSetDouble, ObjectGetDouble
OBJPROP_LEVELWIDTH	Dicke der Linie-Level	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJPROP_NAME	Objektname	ObjectSetString, ObjectGetString
OBJPROP_PERIOD	Periode für Objekt "Chart"	ObjectSetInteger, ObjectGetInteger
OBJPROP_PRICE	Koordinate des Preises	ObjectSetDouble, ObjectGetDouble
OBJPROP_PRICE_SCALE	Merkmal der Darstellung von Preisskala für Objekt "Chart"	ObjectSetInteger, ObjectGetInteger
OBJPROP_RAY	Eine vertikale Linie erstreckt sich auf alle der Chart-Fenster	ObjectSetInteger, ObjectGetInteger
OBJPROP_RAY_LEFT	Strahl setzt sich nach links fort	ObjectSetInteger, ObjectGetInteger
OBJPROP_RAY_RIGHT	Strahl setzt sich nach rechts fort	ObjectSetInteger, ObjectGetInteger
OBJPROP_READONLY	Möglichkeit von Textedit ieren im	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Objekt Edit	
OBJPROP_SCALE	Masssta b (Eigensc haft der Objekte des Objekte s "Fibinac ci Arcs")	ObjectSetDouble, ObjectGetDouble
OBJPROP_SELECTABLE	Zugängli chkeit des Objekts	ObjectSetInteger, ObjectGetInteger
OBJPROP_SELECTED	Hervorh eben des Objekte s	ObjectSetInteger, ObjectGetInteger
OBJPROP_STATE	Schaltflä chenstell ung (Gedrüc kt/nicht gedrückt)	ObjectSetInteger, ObjectGetInteger
OBJPROP_STYLE	Stil	ObjectSetInteger, ObjectGetInteger
OBJPROP_SYMBOL	Symbol für Objekt "Chart"	ObjectSetString, ObjectGetString
OBJPROP_TEXT	Objektb eschreib ung (Text, der im Objekt gibt)	ObjectSetString, ObjectGetString
OBJPROP_TIME	Koordina te der Zeit	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
OBJPROP_TIMEFRAMES	Sichtbarkeit des Objekts auf Timeframes	ObjectSetInteger, ObjectGetInteger
OBJPROP_TOOLTIP	Der Text eines Tooltip. Wenn die Eigenschaft nicht angegeben ist, wird der Tooltip automatisch vom Terminal generiert. Ein Tooltip kann durch Setzen des "\n" (Zeilenvorschub) Wertes deaktiviert werden	ObjectSetString, ObjectGetString
OBJPROP_TYPE	Objekttyp	ObjectSetInteger, ObjectGetInteger
OBJPROP_WIDTH	Weite von Linie	ObjectSetInteger, ObjectGetInteger
OBJPROP_XDISTANCE	Distanz in Pixel X-Achse vom Bezugsw	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	inkel (s. Hinweis)	
OBJPROP_XOFFSET	Die X Koordinate von der linken oberen Ecke des rechteckigen sichtbaren Bereich in der grafischen Objekte "Bitmap Label" und "Bitmap" (OBJ_BITMAP_LABEL und OBJ_BITMAP). Der Wert wird in Pixel relativ zur oberen linken Ecke des ursprünglichen Bildes gesetzt.	ObjectSetInteger, ObjectGetInteger
OBJPROP_XSIZE	Die Breite des Objekts entlang	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	der X-Achse in Pixeln. Wird für OBJ_LABEL (read only), OBJ_BUTTON, OBJ_CHART, OBJ_BITMAP, OBJ_BITMAP_LABEL, OBJ_EDIT, OBJ_RECTANGLE_LABEL angegeben.	
OBJPROP_YDISTANCE	Distanz in Pixel Y-Achse vom Bezugswinkel (s. Hinweis)	ObjectSetInteger, ObjectGetInteger
OBJPROP_YOFFSET	Die Y Koordinate von der linken oberen Ecke des rechteckigen sichtbaren Bereich in der grafischen Objekte	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	"Bitmap Label" und "Bitmap" (OBJ_BITMAP_LABEL und OBJ_BITMAP). Der Wert wird in Pixel relativ zur oberen linken Ecke des ursprünglichen Bildes gesetzt.	
OBJPROP_YSIZE	Die Höhe des Objekts entlang der Y-Achse in Pixeln. Wird für OBJ_LABEL (read only), OBJ_BUTTON, OBJ_CHART, OBJ_BITMAP, OBJ_BITMAP_LABEL, OBJ_EDIT, OBJ_REC	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	TANGLE_LABEL angegeben.	
OBJPROP_ZORDER	Priorität eines graphischen Objects für Empfang des Ereignisses einen Mausklick auf dem Chart (CHART_EVENT_CLICK). Standardmäßig ist der Wert bei Erstellen auf Null gesetzt, aber Sie können die Priorität erhöhen. Bei der Anwendung der Objekte auf jeder anderen, nur ein Objekt mit der höchsten Priorität bekomm	ObjectSetInteger, ObjectGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	t das Ereignis CHARTEVENT_CLICK.	
ORDER_COMMENT	Orderkommentar	OrderGetString, HistoryOrderGetString
ORDER_FILLING_FOK	Diese Ausfüllungspolitik bedeutet, dass eine Order nur in der angegebenen Menge gefüllt werden kann. Wenn die notwendige Menge eines Finanzinstruments ist auf dem Markt derzeit nicht verfügbar, wird die Order nicht ausgefüllt werden. Das	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	erforderliche Volumen kann mit mehreren Angeboten, die im Moment auf dem Markt verfügbar sind, gefüllt werden.	
ORDER_FILLING_IOC	Dieser Modus bedeutet, dass der Trader mit einem Deal mit dem maximalen auf dem Markt verfügbaren Volumen innerhalb der Order einverstanden ist. Falls das gesamte Volumen einer Order nicht gefüllt werden	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	kann, wird das verfügbare Volumen davon gefüllt werden und das restliche Volumen wird abgebrochen.	
ORDER_FILLING_RETURN	Dieses Modus wird für die Markt- (ORDER_TYPE_BUY undORDER_TYPE_SELL), Limit- und Stop-Limit-Orders (ORDER_TYPE_BUY_LIMIT, ORDER_TYPE_SELL_LIMIT, ORDER_TYPE_BUY_STOP_LIMIT und ORDER_TYPE_SELL_STOP_LIMIT) und nur	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	in Modi "Markt Ausführung " und "Börse Ausführung" verwendet. Im Falle einer teilweisen Ausführung wird die Markt- oder Limit-Order mit verbleibende Volumen nicht annulliert, sondern weiter verarbeitet. Für die Aktivierung der Ordern ORDER_TYPE_BUY_STOP_LIMIT und ORDER_TYPE_SELL_STOP_LIMIT wird eine entsprechende	

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Limit-Order ORDER_TYPE_BUY_LIMIT /ORDER_TYPE_SELL_LIMIT mit Ausfüllungstyp ORDER_FILLING_RETURN erstellt werden.	
ORDER_MAGIC	Identifikator des Experten, der Order gestellt hat (bestimmt dafür, dass jeder Expert seine eigene unikale Nummer stellt)	OrderGetInteger, HistoryOrderGetInteger
ORDER_POSITION_ID	Identifikator der Position, den Order nach seine Durchführung erhält. Jede	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	durchgeführte Order erzeugt einen Deal, das eine neue Position eroeffnet oder eine schon existierende Position verändert. Durchgeführte Order erhält eben diesen Identifikator dieser Position.	
ORDER_PRICE_CURRENT	Laufender Preis des Ordersymbols	OrderGetDouble, HistoryOrderGetDouble
ORDER_PRICE_OPEN	Preis angegeben in Order	OrderGetDouble, HistoryOrderGetDouble
ORDER_PRICE_STOPLIMIT	Preis der Aufstellung der Limit Order bei Auslösung von	OrderGetDouble, HistoryOrderGetDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	StopLimit Order	
ORDER_SL	Level Stop Loss	OrderGetDouble, HistoryOrderGetDouble
ORDER_STATE	Orderstatus	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_CANCELED	Order ist vom Kunden abgelehnt	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_EXPIRED	Order ist nach Ablauffrist abgelehnt	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_FILLED	Order ist vollständig durchgeführt	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_PARTIAL	Order ist teilweise durchgeführt	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_PLACED	Order angenommen	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_REJECTED	Order abgelehnt	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_REQUEST_ADD	Order ist im Registrierungsstatus (Aussetzung in Handelssystem)	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ORDER_STATE_REQUEST_CANCEL	Order ist im Löschungsstatus (Löschung aus dem Handelssystem)	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_REQUEST_MODIFY	Order ist im Änderungsstatus (Verändern von Parametern)	OrderGetInteger, HistoryOrderGetInteger
ORDER_STATE_STARTED	Order geprüft, aber vom Broker noch nicht angenommen	OrderGetInteger, HistoryOrderGetInteger
ORDER_SYMBOL	Symbol der Order	OrderGetString, HistoryOrderGetString
ORDER_TIME_DAY	Order wird nur innerhalb des Handelstages gültig	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_DONE	Durchführungszeit oder Annullieren einer Order	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_DONE_MSC	Zeit der Ausführu	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ng/Abschiebung des Orders in Millisekunden seit 01.01.1970	
ORDER_TIME_EXPIRATION	Ablauffrist einer Order	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_GTC	Warteorder vor der Ablehnung	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_SETUP	Set-up Zeit des Orders	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_SETUP_MSC	Zeitpunkt der Erstellung von Order in Millisekunden seit 01.01.1970	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_SPECIFIED	Order wird bis zum Ablauffrist gültig	OrderGetInteger, HistoryOrderGetInteger
ORDER_TIME_SPECIFIED_DAY	Order wird bis 23:59:59 Uhr am angegebenen Datum gültig.	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Wird diese Zeit nicht auf dem Handelszeit fallen, wird der Ablauf in naher den Handelszeit auftreten.	
ORDER_TP	Level Take Profit	OrderGetDouble, HistoryOrderGetDouble
ORDER_TYPE	Ordertyp	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_BUY	Marktkauforder	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_BUY_LIMIT	Schwebende Order Buy Limit	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_BUY_STOP	Schwebende Order Buy Stop	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_BUY_STOP_LIMIT	Beim Erreichen des Orderpreises wird eine Warteorder Buy Limit zum Preis StopLimit gestellt	OrderGetInteger, HistoryOrderGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
ORDER_TYPE_FILLING	Durchführungstyp nach dem Rest	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_SELL	Martverkauforder	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_SELL_LIMIT	Schwebende Order Sell Limit	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_SELL_STOP	Schwebende Order Sell Stop	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_SELL_STOP_LIMIT	Beim Erreichen des Orderpreises wird eine Pending Sell Limit Order zu StopLimit Preis platziert	OrderGetInteger, HistoryOrderGetInteger
ORDER_TYPE_TIME	Lebenszeit des Orders	OrderGetInteger, HistoryOrderGetInteger
ORDER_VOLUME_CURRENT	eine nicht durchgeführte Order	OrderGetDouble, HistoryOrderGetDouble
ORDER_VOLUME_INITIAL	Initialvolumen bei der Orderstellung	OrderGetDouble, HistoryOrderGetDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
PERIOD_CURRENT	laufende Periode	Chartperioden
PERIOD_D1	1 Tag	Chartperioden
PERIOD_H1	1 Stunde	Chartperioden
PERIOD_H12	12 Stunden	Chartperioden
PERIOD_H2	2 Stunden	Chartperioden
PERIOD_H3	3 Stunden	Chartperioden
PERIOD_H4	4 Stunden	Chartperioden
PERIOD_H6	6 Stunden	Chartperioden
PERIOD_H8	8 Stunden	Chartperioden
PERIOD_M1	1 Minute	Chartperioden
PERIOD_M10	10 Minuten	Chartperioden
PERIOD_M12	12 Minuten	Chartperioden
PERIOD_M15	15 Minuten	Chartperioden
PERIOD_M2	2 Minuten	Chartperioden
PERIOD_M20	20 Minuten	Chartperioden
PERIOD_M3	3 Minuten	Chartperioden
PERIOD_M30	30 Minuten	Chartperioden
PERIOD_M4	4 Minuten	Chartperioden
PERIOD_M5	5 Minuten	Chartperioden

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
PERIOD_M6	6 Minuten	Chartperioden
PERIOD_MN1	1 Monat	Chartperioden
PERIOD_W1	1 Woche	Chartperioden
PLOT_ARROW	Weichenkode für Stil DRAW_ARROW	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_ARROW_SHIFT	Vertikale Verschiebung der Weichen für Stil DRAW_ARROW	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_COLOR_INDEXES	Anzahl der Farben	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_DRAW_BEGIN	Zahl der Anfangsbars ohne Zeichnung und Werte in DataWindow	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_DRAW_TYPE	Typ der graphischen Konstruktion	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_EMPTY_VALUE	Leerwert für Konstruktion, für die es keine Zeichnung gibt	PlotIndexSetDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
PLOT_LABEL	Name der graphischen Indikatorserie für Darstellung im Fenster DataWindow. Für komplexe grafische Stile, die mehrere Indikatorpuffer für Anzeige brauchen, können die Namen von jedem Puffer mit Trennzeichen ";" angegeben werden. Beispiel-Code wird in DRAW_CANDLES gezeigt	PlotIndexSetString
PLOT_LINE_COLOR	Pufferindex mit zeichnerischer Farbe	PlotIndexSetInteger, PlotIndexGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
PLOT_LINE_STYLE	Stil der zeichnerischen Linie	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_LINE_WIDTH	Breite der zeichnerischen Linie	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_SHIFT	Verschiebung der graphischen Konstruktion des Indikators Zeit-Achse in Bars	PlotIndexSetInteger, PlotIndexGetInteger
PLOT_SHOW_DATA	Merkmal der Darstellung von Konstruktionswerte im Fenster DataWindow	PlotIndexSetInteger, PlotIndexGetInteger
PLUSDI_LINE	Linie +DI	Linien der Indikatoren
POINTER_AUTOMATIC	Anzeiger jedes Objektes, erzeugt automatisch (ohne new())	CheckPointer
POINTER_DYNAMIC	Anzeiger des Objektes, erzeugt	CheckPointer

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	von der Anweisung new()	
POINTER_INVALID	Unkorrekter Anzeiger	CheckPointer
POSITION_COMMENT	Kommentar zur Position	PositionGetString
POSITION_COMMISSION	Kommission	PositionGetDouble
POSITION_IDENTIFIER	Identifikator einer Position ist eine unikale Zahl, die jeder neu geoeffneten Position zugeordnet wird und innerhalb ihres ganzen Leben nicht verändert wird. Umlauf der Position verändert den Identifikator der Position nicht.	PositionGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
POSITION_MAGIC	Magic number für die Position (sehen Sie ORDER_MAGIC)	PositionGetInteger
POSITION_PRICE_CURRENT	Laufender Preis des Symbols	PositionGetDouble
POSITION_PRICE_OPEN	Position spreis	PositionGetDouble
POSITION_PROFIT	Laufender Gewinn	PositionGetDouble
POSITION_SL	Level Stop Loss für die offene Position	PositionGetDouble
POSITION_SWAP	Gesamtswap	PositionGetDouble
POSITION_SYMBOL	Symbol der Position	PositionGetString
POSITION_TIME	Eröffnungszeit der Position	PositionGetInteger
POSITION_TIME_MSC	Eröffnungszeit der Position in Millisekunden seit 01.01.1970	PositionGetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
POSITION_TIME_UPDATE	Positionsänderungszeit in Sekunden seit 01.01.1970	PositionGetInteger
POSITION_TIME_UPDATE_MSC	Positionsänderungszeit in Millisekunden seit 01.01.1970	PositionGetInteger
POSITION_TP	Level Take Profit für die offene Position	PositionGetDouble
POSITION_TYPE	Typ der Position	PositionGetInteger
POSITION_TYPE_BUY	Kauf	PositionGetInteger
POSITION_TYPE_SELL	Verkauf	PositionGetInteger
POSITION_VOLUME	Volumen der Position	PositionGetDouble
PRICE_CLOSE	Eroeffnungspreis	Preiskonstanten
PRICE_HIGH	Maximaler Preis für die Periode	Preiskonstanten
PRICE_LOW	Minimaler Preis für die Periode	Preiskonstanten

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
PRICE_MEDIAN	Medianpreis, (high+low)/2	Preiskonstanten
PRICE_OPEN	Schlusspreis	Preiskonstanten
PRICE_TYPICAL	Typischer Preis, (high+low+close)/3	Preiskonstanten
PRICE_WEIGHTED	Durchschnittlich gewogener Preis, (high+low+close+close)/4	Preiskonstanten
PROGRAM_EXPERT	Expert	MQLInfoInteger
PROGRAM_INDICATOR	Anzeiger	MQLInfoInteger
PROGRAM_SCRIPT	Skript	MQLInfoInteger
REASON_ACCOUNT	Ein anderes Konto wurde aktiviert oder das Konto wurde mit dem Handelsserver neu verbunden, nachdem die Kontoeinstellungen geändert worden	UninitializeReason, OnDeinit

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
REASON_CHARTCHANGE	Symbol oder Chartperiode wurde verändert	UninitializeReason, OnDeinit
REASON_CHARTCLOSE	Das Chart wurde abgeschlossen	UninitializeReason, OnDeinit
REASON_CLOSE	Das Terminal wurde geschlossen	UninitializeReason, OnDeinit
REASON_INITFAILED	Der Wert bedeutet, dass Bearbeiter OnInit() Nichtnullwert zurückgegeben hat	UninitializeReason, OnDeinit
REASON_PARAMETERS	Eingabeparameter wurden vom Benutzer verändert	UninitializeReason, OnDeinit
REASON_PROGRAM	Expert beendet seine Operation durch Aufruf der Funktion	UninitializeReason, OnDeinit

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ExpertRemove()	
REASON_RECOMPILE	Das Programm ist umcompiliert	UninitializeReason, OnDeinit
REASON_REMOVE	Das Programm ist entfernt vom Chart	UninitializeReason, OnDeinit
REASON_TEMPLATE	andere Chartschablone wurde angewendet	UninitializeReason, OnDeinit
SATURDAY	Samstag	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
SEEK_CUR	Laufende Position des Dateianzeigers	FileSeek
SEEK_END	Dateiende	FileSeek
SEEK_SET	Dateianfang	FileSeek
SENKOUSPANA_LINE	Linie Senkou Span A	Linien der Indikatoren
SENKOUSPANB_LINE	Linie Senkou Span B	Linien der Indikatoren
SERIES_BARS_COUNT	Anzahl der Bars für	SeriesInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Symbol-Periode zum jetzigen Zeitpunkt	
SERIES_FIRSTDATE	Das erste Datum für Symbol-Periode zum jetzigen Zeitpunkt	SeriesInfoInteger
SERIES_LASTBAR_DATE	Öffnungszeit der letzten Bar für Symbol-Periode	SeriesInfoInteger
SERIES_SERVER_FIRSTDATE	Das erste Datum des Symbols auf dem Server unabhängig von der Periode	SeriesInfoInteger
SERIES_SYNCHRONIZED	Flagge der Synchronisierung von Symbol/Periode zum jetzigen Zeitpunkt	SeriesInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SERIES_TERMINAL_FIRSTDATE	Das erste Datum in der Geschichte des Symbols im Client-Terminal unabhängig von der Periode	SeriesInfoInteger
SHORT_MAX	Maximaler Wert, der durch den Typ short dargestellt werden kann	Konstanten der numerischenTypen
SHORT_MIN	Minimaler Wert, der durch den Typ short dargestellt werden kann	Konstanten der numerischenTypen
SIGNAL_BASE_AUTHOR_LOGIN	Benutzername des Signalgebers	SignalBaseGetString
SIGNAL_BASE_BALANCE	Kontostand	SignalBaseGetDouble
SIGNAL_BASE_BROKER	Name des Brokers (des	SignalBaseGetString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Unternehmens)	
SIGNAL_BASE_BROKER_SERVER	Server des Brokers	SignalBaseGetString
SIGNAL_BASE_CURRENCY	Währung des Signalkontos	SignalBaseGetString
SIGNAL_BASE_DATE_PUBLISHED	Veröffentlichungsdatum des Signals (wann wurde es zum Abonnieren freigegeben)	SignalBaseGetInteger
SIGNAL_BASE_DATE_STARTED	Anfangsdatum der Beobachtung des Signals	SignalBaseGetInteger
SIGNAL_BASE_EQUITY	Equity	SignalBaseGetDouble
SIGNAL_BASE_GAIN	Wachstum in Prozent	SignalBaseGetDouble
SIGNAL_BASE_ID	Signal-ID	SignalBaseGetInteger
SIGNAL_BASE_LEVERAGE	Hebel	SignalBaseGetInteger
SIGNAL_BASE_MAX_DRAWDOWN	Maximaler Drawdown	SignalBaseGetDouble
SIGNAL_BASE_NAME	Name des Signals	SignalBaseGetString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SIGNAL_BASE_PIPS	Handelsergebnis in Pips	SignalBaseGetInteger
SIGNAL_BASE_PRICE	Abo-Preis	SignalBaseGetDouble
SIGNAL_BASE_RATING	Position im Ranking der Signale	SignalBaseGetInteger
SIGNAL_BASE_ROI	ROI (Return on Investment) des Signals in %	SignalBaseGetDouble
SIGNAL_BASE_SUBSCRIBERS	Abonnentenzahl	SignalBaseGetInteger
SIGNAL_BASE_TRADE_MODE	Kontotyp (0-Realkonto, 1-Demokonto, 2-Wettbewerbskonto)	SignalBaseGetInteger
SIGNAL_BASE_TRADES	Anzahl von Trades	SignalBaseGetInteger
SIGNAL_INFO_CONFIRMATIONS_DISABLED	Flag der Erlaubnis der Synchronisierung ohne Anzeige des Bestätigungsdialogs	SignalInfoGetInteger, SignalInfoSetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SIGNAL_INFO_COPY_SLTP	Flag des Kopierens von Stop Loss und Take Profit	SignalInfoGetInteger, SignalInfoSetInteger
SIGNAL_INFO_DEPOSIT_PERCENT	Begrenzung nach der Einlage (in %)	SignalInfoGetInteger, SignalInfoSetInteger
SIGNAL_INFO_EQUITY_LIMIT	Equity Limit	SignalInfoGetDouble, SignalInfoSetDouble
SIGNAL_INFO_ID	Signal-ID, r/o	SignalInfoGetInteger, SignalInfoSetInteger
SIGNAL_INFO_NAME	Name des Signals, r/o	SignalInfoGetString
SIGNAL_INFO_SLIPPAGE	Slippage, mit welchem Marktorders bei der Synchronisierung von Positionen und beim Kopieren von Trades platziert werden	SignalInfoGetDouble, SignalInfoSetDouble
SIGNAL_INFO_SUBSCRIPTION_ENABLED	Flag der Erlaubnis für das Kopieren von Trades nach	SignalInfoGetInteger, SignalInfoSetInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	dem Abonnement	
SIGNAL_INFO_TERMS_AGREE	Flag der Akzeptierung der Nutzungsbedingungen des Signal-Services, r/o	SignalInfoGetInteger, SignalInfoSetInteger
SIGNAL_INFO_VOLUME_PERCENT	Begrenzung von Equity für Signale, r/o	SignalInfoGetDouble, SignalInfoSetDouble
SIGNAL_LINE	Sygnallinie	Linien der Indikatoren
STAT_BALANCE_DD	Maximaler Drawdown der Bilanz in Geld. Im Prozeß der Trading kann ein Bilanz zahlreiche Drawdowns haben, hier nehmen wir den größte Wert	TesterStatistics
STAT_BALANCE_DD_RELATIVE	Drawdown der Bilanz in Geld,	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	der zum Zeitpunkt von maximalem Bilanz-Drawdown in Prozent aufgenommen wurde (STAT_BALANCE_DDREL_PERCENT)	
STAT_BALANCE_DDREL_PERCENT	Maximaler Drawdown der Bilanz als Prozentsatz. Im Prozeß der Trading kann ein Bilanz zahlreiche Drawdowns haben, für jeden Drawdown wird sein Wert als Prozentsatz berechnet. Der größte Wert	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	wird zurückgegeben	
STAT_BALANCEDD_PERCENT	Drawdown der Bilanz als Prozentsatz, der zum Zeitpunkt von maximalem Bilanz-Drawdown in Geld aufgenommen wurde (STAT_BALANCE_DD)	TesterStatistics
STAT_BALANCEMIN	Der minimale Wert der Bilanz	TesterStatistics
STAT_CONLOSSMAX	Maximaler Verlust unter konsequent profitable Trades. Der Wert ist kleiner oder gleich Null	TesterStatistics
STAT_CONLOSSMAX_TRADES	Anzahl der	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Trades, die STAT_CONLOSSMAX geformt haben (maximaler Verlust unter konsequent verlustbringende Trades)	
STAT_CONPROFITMAX	Maximaler Gewinn unter konsequent profitable Trades. Der Wert ist größer oder gleich Null	TesterStatistics
STAT_CONPROFITMAX_TRADES	Anzahl der Trades, die STAT_CONPROFITMAX geformt haben (maximaler Gewinn unter konsequent	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	profitable Trades)	
STAT_CUSTOM_ONTESTER	Der Wert des berechneten Benutzer-Optimierungskriterium, der OnTester()-Funktion zurückgegeben wird	TesterStatistics
STAT_DEALS	Anzahl der Deals	TesterStatistics
STAT_EQUITY_DD	Maximaler Drawdown des Eigenkapitals in Geld. Im Prozeß der Trading kann ein Eigenkapital zahlreiche Drawdowns haben, hier nehmen wir den größte Wert	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
STAT_EQUITY_DD_RELATIVE	Drawdown des Eigenkapitals in Geld, der zum Zeitpunkt von maximalem Eigenkapitalsdrawdown in Prozent aufgenommen wurde (STAT_EQUITY_DDREL_PERCENT).	TesterStatistics
STAT_EQUITY_DDREL_PERCENT	Maximaler Drawdown des Eigenkapitals als Prozentsatz. Im Prozeß der Trading kann ein Eigenkapital zahlreiche Drawdowns haben, für jeden Drawdown wird sein Wert als	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Prozentsatz berechnet. Der größte Wert wird zurückgegeben	
STAT_EQUITYDD_PERCENT	Drawdown des Eigenkapitals in Prozent, der zum Zeitpunkt von maximalem Eigenkapitasdrawdown in Geld aufgenommen wurde (STAT_EQUITY_DD)	TesterStatistics
STAT_EQUITYMIN	Der minimale Wert des Eigenkapitals	TesterStatistics
STAT_EXPECTED_PAYOFF	Erwartete Auszahlung	TesterStatistics
STAT_GROSS_LOSS	Total Verlust, Betrag aller verlustbringende	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	n (negativen) Trades. Der Wert ist kleiner oder gleich Null	
STAT_GROSS_PROFIT	Total Gewinn, Betrag aller profitablen (positiven) Trades. Der Wert ist größer oder gleich Null	TesterStatistics
STAT_INITIAL_DEPOSIT	Der Wert des Ersteinzahlung	TesterStatistics
STAT_LONG_TRADES	Lange Trades	TesterStatistics
STAT_LOSS_TRADES	Verlust-Trades	TesterStatistics
STAT_LOSSTRADES_AVGCON	Die durchschnittliche Länge einer Reihe von Verlust-Trades	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
STAT_MAX_CONLOSS_TRADES	Anzahl der Trades in der längsten Reihe von verlustbringenden Trades STAT_MAX_CONLOSSES	TesterStatistics
STAT_MAX_CONLOSSES	Total Verlust in der längsten Reihe von verlustbringenden Trades	TesterStatistics
STAT_MAX_CONPROFIT_TRADES	Anzahl der Trades in der längsten Reihe von profitablen Trades STAT_MAX_CONWINS	TesterStatistics
STAT_MAX_CONWINS	Total Gewinn in der längsten Reihe von profitablen Trades	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
STAT_MAX_LOSSTRADE	Maximaler Verlust - der kleinste Wert unter allen verlustbringenden Trades. Der Wert ist kleiner oder gleich Null	TesterStatistics
STAT_MAX_PROFITTRADE	Maximaler Gewinn - der höchste Wert unter allen profitablen Trades. Der Wert ist größer oder gleich Null	TesterStatistics
STAT_MIN_MARGINLEVEL	Der minimale Wert der Margeebene	TesterStatistics
STAT_PROFIT	Nettogewinn nach Test, Betrag	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	von STAT_GROSS_PROFIT und STAT_GROSS_LOSS (STAT_GROSS_LOSS ist immer kleiner oder gleich Null)	
STAT_PROFIT_FACTOR	Profit-Faktor, gleich dem Verhältnis von STAT_GROSS_PROFIT/STAT_GROSS_LOSS. Wenn STAT_GROSS_LOSS=0, ist Profit-Faktor gleich DBL_MAX	TesterStatistics
STAT_PROFIT_LONGTRADES	Lange Gewinn-Trades	TesterStatistics
STAT_PROFIT_SHORTTRADES	Kurze Gewinn-Trades	TesterStatistics

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
STAT_PROFIT_TRADES	Gewinn-Trades	TesterStatistics
STAT_PROFITTRADES_AVGCON	Die durchschnittliche Länge einer Reihe von Gewinn-Trades	TesterStatistics
STAT_RECOVERY_FACTOR	Recovery-Faktor, gleich dem Verhältnis von STAT_PROFIT/STAT_BALANCE_DD	TesterStatistics
STAT_SHARPE_RATIO	Sharpe Ratio	TesterStatistics
STAT_SHORT_TRADES	Kurze Trades	TesterStatistics
STAT_TRADES	Anzahl der Trades	TesterStatistics
STAT_WITHDRAWAL	Von einem Konto abgezogene Geld	TesterStatistics
STO_CLOSECLOSE	Berechnung ist nach den Preisen Close/Close	Preiskonstanten
STO_LOWHIGH	Berechnung ist	Preiskonstanten

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	nach den Preisen Low/High	
STYLE_DASH	diskontinierliche Linie	Zeichnungsstile
STYLE_DASHDOT	Strichpunktlinie	Zeichnungsstile
STYLE_DASHDOTDOT	Strich-zwei Punkte	Zeichnungsstile
STYLE_DOT	Punktierte Linie	Zeichnungsstile
STYLE_SOLID	Ausgezogene Linie	Zeichnungsstile
SUNDAY	Sonntag	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
SYMBOL_ASK	Ask - das beste Kaufsangebot	SymbolInfoDouble
SYMBOL_ASKHIGH	Maximales Ask pro Tag	SymbolInfoDouble
SYMBOL_ASKLOW	Minimales Ask pro Tag	SymbolInfoDouble
SYMBOL_BANK	Quelle der laufenden Quotation	SymbolInfoString
SYMBOL_BASIS	Name der Basiswährung	SymbolInfoString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	für das abgeleitete Symbol	
SYMBOL_BID	Bid - das beste Verkaufsangebot	SymbolInfoDouble
SYMBOL_BIDHIGH	maximales Bid pro Tag	SymbolInfoDouble
SYMBOL_BIDLOW	Minimales Bid pro Tag	SymbolInfoDouble
SYMBOL_CALC_MODE_CFD	CFD mode - Berechnung von Marge und Gewinn für CFD	SymbolInfoInteger
SYMBOL_CALC_MODE_CFDINDEX	CFD index mode - Berechnung von Marge und Gewinn für CFD durch Indexe	SymbolInfoInteger
SYMBOL_CALC_MODE_CFDLEVERAGE	CFD Leverage mode - Berechnung von Marge und Gewinn für CFD beim Leverag	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	e-Handel	
SYMBOL_CALC_MODE_EXCH_FUTURES	Futures mode - Berechnung von Marge und Gewinn für den Handel von Futures-Kontrakte an der Börse	SymbolInfoInteger
SYMBOL_CALC_MODE_EXCH_FUTURES_FORTS	FORTS Futures mode - Berechnung von Marge und Gewinn für den Handel von Futures-Kontrakte an FORTS.	SymbolInfoInteger
SYMBOL_CALC_MODE_EXCH_STOCKS	Exchange mode - Berechnung von Marge und Gewinn für den Handel von Wertpapieren an der Börse	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_CALC_MODE_FOREX	Forex mode - Berechnung von Gewinn und Marge für Forex	SymbolInfoInteger
SYMBOL_CALC_MODE_FUTURES	Futures mode - Berechnung von Marge und Gewinn für futures	SymbolInfoInteger
SYMBOL_CALC_MODE_SERV_COLLATERAL	Collateral mode - das Symbol ist als nicht-handelbare Vermögen auf einem Handelskonto verwendet	SymbolInfoInteger
SYMBOL_CURRENCY_BASE	Grundwährung des Instrumentes	SymbolInfoString
SYMBOL_CURRENCY_MARGIN	Währung der Marge	SymbolInfoString
SYMBOL_CURRENCY_PROFIT	Gewinnwährung	SymbolInfoString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_DESCRIPTION	Zeilenbeschreibung des Symbols	SymbolInfoString
SYMBOL_DIGITS	Anzahl der Dezimalzeichen	SymbolInfoInteger
SYMBOL_EXPIRATION_DAY	Order ist bis zum Tagesende gültig	SymbolInfoInteger
SYMBOL_EXPIRATION_GTC	Order ist zeitlich uneigeschränkt gültig, bis zu seiner expliziten Annullierung	SymbolInfoInteger
SYMBOL_EXPIRATION_MODE	Flags der erlaubten Order-Ablaufmodi	SymbolInfoInteger
SYMBOL_EXPIRATION_SPECIFIED	Ablauffrist wird in der Order angegeben werden	SymbolInfoInteger
SYMBOL_EXPIRATION_SPECIFIED_DAY	Ablaufdatum wird in der Order angegeben werden	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_EXPIRATION_TIME	Abschlussdatum der Versteigerung nach dem Instrument (gewöhnlich wird für Futures verwendet)	SymbolInfoInteger
SYMBOL_FILLING_FOK	Diese Ausfüllungspolitik bedeutet, dass der Order nur in der angegebenen Volumen ausgefüllt werden kann. Wenn die ausreichende Menge eines Finanzinstruments sich auf dem Markt derzeit nicht präsentieren, wird der	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Order nicht ausgeführt werden. Das erforderliche Volumen kann aus mehreren Vorschlägen, die derzeit auf dem Markt erhältlich sind, bestehen.	
SYMBOL_FILLING_IOC	In diesem Fall verpflichtet sich der Händler, einen Deal mit dem maximalen Volumen, das auf dem Markt erhältlich ist, im Bereich in der angegebenen Reihenfolge zu machen. Wenn	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	die volle Ausführung unmöglich ist, wird der Order auf das verfügbare Volumen ausgeführt werden und das übrige Volumen wird storniert. Erlaubnis zur Nutzung von IOC Aufträge wird auf dem Trading-Server bestimmt.	
SYMBOL_FILLING_MODE	Flags der erlaubten Order-Füllmodi	SymbolInfoInteger
SYMBOL_ISIN	Der Name eines Symbols im System ISIN (International Securitie	SymbolInfoString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	s Identification Number). International Securities Identification Number ist ein 12-stelliger alphanumerischer Code zur eindeutigen Identifizierung eines Wertpapiers. Das Vorhandensein dieser Einstellung des Symbols wird auf der Seite eines Handelsservers ermittelt.	
SYMBOL_LAST	Preis des letzten Deals	SymbolInfoDouble
SYMBOL_LASTHIGH	Maximales Last	SymbolInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	pro Tag	
SYMBOL_LASTLOW	Minimales Last pro Tag	SymbolInfoDouble
SYMBOL_MARGIN_INITIAL	Initiale Marge bezeichnet die Größe der erforderlichen Kautionssumme in Marge-Währung für Eröffnung einer Position im Volumen von einem Lot. Wird verwendet bei der Überprüfung von Geld eines des Kunden beim Eintritt in den Markt.	SymbolInfoDouble
SYMBOL_MARGIN_MAINTENANCE	Maintenance-Marge. Wenn sie eingege	SymbolInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ben ist, setzt sie die Marge in der Marge-Währung des Symbols, die von einem Lot abgezogen wird. Wird verwendet bei der Überprüfung von Geld eines des Kunden, wenn sein/ihr Kontozustand ändert. Wenn die Maintenance-Marge gleich 0 ist, wird die Initial Marge verwendet.	
SYMBOL_OPTION_MODE	Optionstyp	SymbolInfoInteger
SYMBOL_OPTION_MODE_AMERICAN	Amerikanische Option:	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	die Option kann an jedem Handelstag vor der Fälligkeit ausgeübt werden	
SYMBOL_OPTION_MODE_EUROPEAN	Europäische Option: die Option kann nur am Fälligkeitsdatum ausgeübt werden	SymbolInfoInteger
SYMBOL_OPTION_RIGHT	Optionsrecht (Call/Put)	SymbolInfoInteger
SYMBOL_OPTION_RIGHT_CALL	Eine Call-Option bezeichnet ein Recht, eine Vermögenswert zu einem Festpreis zu kaufen	SymbolInfoInteger
SYMBOL_OPTION_RIGHT_PUT	Eine Put-Option bezeichnet ein	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Recht, eine Vermögenswert zu einem Festpreis zu verkaufen	
SYMBOL_OPTION_STRIKE		SymbolInfoDouble
SYMBOL_ORDER_LIMIT	Limit-Orders sind erlaubt (Buy Limit und Sell Limit)	SymbolInfoInteger
SYMBOL_ORDER_MARKET	Markt-Orders sind erlaubt (Buy und Sell)	SymbolInfoInteger
SYMBOL_ORDER_MODE	Flags der erlaubten Ordertypen	SymbolInfoInteger
SYMBOL_ORDER_SL	Stop Loss ist erlaubt	SymbolInfoInteger
SYMBOL_ORDER_STOP	Stop-Orders sind erlaubt (Buy Stop und Sell Stop)	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_ORDER_STOP_LIMIT	Stop-Limit-Orders sind erlaubt (Buy Stop Limit und Sell Stop Limit)	SymbolInfoInteger
SYMBOL_ORDER_TP	Take Profit ist erlaubt	SymbolInfoInteger
SYMBOL_PATH	Pfad im Symbolbaum	SymbolInfoString
SYMBOL_POINT	Wert eines Punktes	SymbolInfoDouble
SYMBOL_SELECT	Symbol ist in Market Watch gewählt	SymbolInfoInteger
SYMBOL_SESSION_AW	Der durchschnittliche Preis für die Session	SymbolInfoDouble
SYMBOL_SESSION_BUY_ORDERS	Die Gesamtzahl der Buy-Ordern im Moment	SymbolInfoInteger
SYMBOL_SESSION_BUY_ORDERS_VOLUME	Das Gesamtvolumen der Buy-Ordern	SymbolInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	im Moment	
SYMBOL_SESSION_CLOSE	Der Schlusskurs der Session	SymbolInfoDouble
SYMBOL_SESSION_DEALS	Anzahl der Deals in der aktuellen Session	SymbolInfoInteger
SYMBOL_SESSION_INTEREST	Der Gesamtumsatz der offenen Positionen	SymbolInfoDouble
SYMBOL_SESSION_OPEN	Der Eröffnungskurs der Session	SymbolInfoDouble
SYMBOL_SESSION_PRICE_LIMIT_MAX	Der maximale Preiswert für die Session	SymbolInfoDouble
SYMBOL_SESSION_PRICE_LIMIT_MIN	Der minimale Preiswert für die Session	SymbolInfoDouble
SYMBOL_SESSION_PRICE_SETTLEMENT	Abrechnungspreis der aktuellen Session	SymbolInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_SESSION_SELL_ORDERS	Die Gesamtzahl der Sell-Ordern im Moment	SymbolInfoInteger
SYMBOL_SESSION_SELL_ORDERS_VOLUME	Das Gesamtvolumen der Sell-Ordern im Moment	SymbolInfoDouble
SYMBOL_SESSION_TURNOVER	Der Gesamtumsatz in der aktuellen Session	SymbolInfoDouble
SYMBOL_SESSION_VOLUME	Das Gesamtvolumen der Deals in der aktuellen Session	SymbolInfoDouble
SYMBOL_SPREAD	Spreadgröße in Punkten	SymbolInfoInteger
SYMBOL_SPREAD_FLOAT	Symbol des fließenden Spreads	SymbolInfoInteger
SYMBOL_START_TIME	Anfangsdatum der Versteigerung nach	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	dem Instrument(gewöhnlich wird für Futures verwendet)	
SYMBOL_SWAP_LONG	Swapwert beim Kauf	SymbolInfoDouble
SYMBOL_SWAP_MODE	Modell der Swapberechnung	SymbolInfoInteger
SYMBOL_SWAP_MODE_CURRENCY_DEPOSIT	Swaps werden in Geld in der Einzahlung-Währung des Kunden angerechnet	SymbolInfoInteger
SYMBOL_SWAP_MODE_CURRENCY_MARGIN	Swaps werden in Geld in der Margin-Währung des Symbols angerechnet.	SymbolInfoInteger
SYMBOL_SWAP_MODE_CURRENCY_SYMBOL	Swaps werden in Geld in der Basiswährung des Symbols	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	angerechnet	
SYMBOL_SWAP_MODE_DISABLED	Keine Swaps	SymbolInfoInteger
SYMBOL_SWAP_MODE_INTEREST_CURRENT	Swaps werden in den jährlichen Zinsen vom Instrumentpreis zum Zeitpunkt der Berechnung des Swaps (Das Bankjahr wird mit 360 Tagen gerechnet) angerechnet	SymbolInfoInteger
SYMBOL_SWAP_MODE_INTEREST_OPEN	Swaps werden in den jährlichen Zinsen vom Preis der Eröffnung der Position nach Symbol (Das Bankjahr wird mit 360 Tagen gerechnet)	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	angerechnet	
SYMBOL_SWAP_MODE_POINTS	Swaps werden in Punkten angerechnet	SymbolInfoInteger
SYMBOL_SWAP_MODE_REOPEN_BID	Swaps werden durch die Wiedereröffnung der Position angerechnet. Am Ende eines Handelstages wird die Position zwangsmäßig geschlossen. Am nächsten Tag wird die Position nach dem aktuellen Geldkurs Bid +/- die angegebene Anzahl von Punkten (in den Paramet	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ern SYMBOL_SWAP_LONG und SYMBOL_SWAP_SHORT) wieder geöffnet	
SYMBOL_SWAP_MODE_REOPEN_CURRENT	Swaps werden durch die Wiedereröffnung der Position angerechnet. Am Ende eines Handelstages wird die Position zwangsmäßig geschlossen. Am nächsten Tag wird die Position nach einem Schlusskurs +/- die angegebene Anzahl von Punkten (in den Parametern	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	SYMBOL_SWAP_LONG und SYMBOL_SWAP_SHORT) wieder geöffnet	
SYMBOL_SWAP_ROLLOVER3DAYS	Wochentag für Anrechnung des dreifachen Swaps	SymbolInfoInteger
SYMBOL_SWAP_SHORT	Swapwert beim Verkauf	SymbolInfoDouble
SYMBOL_TICKS_BOOKDEPTH	Maximale Anzahl der in DOM gezeigten Anforderungen. Für Instrumente, die keine Quelle der Anforderungen haben, ist der Wert 0	SymbolInfoInteger
SYMBOL_TIME	Zeit der letzten Quotation	SymbolInfoInteger
SYMBOL_TRADE_CALC_MODE	Mode der Berechn	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ung von Vertragskosten	
SYMBOL_TRADE_CONTRACT_SIZE	Größe des Handelsvertrags	SymbolInfoDouble
SYMBOL_TRADE_EXECUTION_EXCHANGE	Börslicher Ausführung	SymbolInfoInteger
SYMBOL_TRADE_EXECUTION_INSTANT	Handel mit laufenden Preisen	SymbolInfoInteger
SYMBOL_TRADE_EXECUTION_MARKET	Orderdurchführung auf dem Markt	SymbolInfoInteger
SYMBOL_TRADE_EXECUTION_REQUEST	Handel auf Anfrage	SymbolInfoInteger
SYMBOL_TRADE_EXEMODE	Mode des Dealsabschlusses	SymbolInfoInteger
SYMBOL_TRADE_FREEZE_LEVEL	Distanz der Einfrierung der Handelsoperationen (in Punkten)	SymbolInfoInteger
SYMBOL_TRADE_MODE	Typ der Orderdurchführung	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_TRADE_MODE_CLOSEONLY	Erlaubt werden die Operationen der Schliessung von Positionen	SymbolInfoInteger
SYMBOL_TRADE_MODE_DISABLED	Symbolhandel ist verboten	SymbolInfoInteger
SYMBOL_TRADE_MODE_FULL	Keine Einschränkungen für Handelsoperationen	SymbolInfoInteger
SYMBOL_TRADE_MODE_LONGONLY	Erlaubt werden nur Käufe	SymbolInfoInteger
SYMBOL_TRADE_MODE_SHORTONLY	Erlaubt werden nur Verkäufe	SymbolInfoInteger
SYMBOL_TRADE_STOPS_LEVEL	Minimale Einrückung in Punkten vom laufenden Schlusspreis für Einstellung der Stop Order	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
SYMBOL_TRADE_TICK_SIZE	Minimale Preisveränderung	SymbolInfoDouble
SYMBOL_TRADE_TICK_VALUE	Wert SYMBOL_TRADE_TICK_VALUE_PROFIT	SymbolInfoDouble
SYMBOL_TRADE_TICK_VALUE_LOSS	Berechnete Kosten des Ticks für unwirtschaftliche Position	SymbolInfoDouble
SYMBOL_TRADE_TICK_VALUE_PROFIT	Berechnete Kosten des Ticks für wirtschaftliche Position	SymbolInfoDouble
SYMBOL_VOLUME	Volume - Volumen im letzten Deal	SymbolInfoInteger
SYMBOL_VOLUME_LIMIT	Die maximale zulässige gesamte Volumen von einer offenen Position und schwebe	SymbolInfoDouble

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	nde Ordern in einer Richtung (Kauf oder Verkauf) für das Symbol. Zum Beispiel mit der Begrenzung von 5 Lots, können Sie eine offene Kaufposition mit dem Volumen von 5 Lots haben und eine schwebende Order Sell Limit mit dem Volumen von 5 Lots stellen. Aber in diesem Fall können Sie nicht Buy Limit (seit dem Gesamtvolumen in eine	

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Richtung wird die Begrenzung überschreiten) oder Sell Limit mit dem Volumen von mehr als 5 Lots stellen.	
SYMBOL_VOLUME_MAX	Maximales Volumen für Dealsabschluss	SymbolInfoDouble
SYMBOL_VOLUME_MIN	Minimale Volumen für Dealsabschluss	SymbolInfoDouble
SYMBOL_VOLUME_STEP	Minimaler Schritt der Volumenveränderung für Dealsabschluss	SymbolInfoDouble
SYMBOL_VOLUMEHIGH	Maximales Volumen pro Tag	SymbolInfoInteger
SYMBOL_VOLUMELOW	Minimales Volumen pro Tag	SymbolInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TENKANSEN_LINE	Linie Tenkan-sen	Linien der Indikatoren
TERMINAL_BUILD	Buildnummer des abgelaufenen Terminals	TerminalInfoInteger
TERMINAL_CODEPAGE	Nummer der Kodeseite der Sprache, die im Client-Terminal installiert wird	TerminalInfoInteger
TERMINAL_COMMONDATA_PATH	Gesamtordner aller Client-Terminals, die im Computer installiert werden	TerminalInfoString
TERMINAL_COMMUNITY_ACCOUNT	Flagge der Verfügbarkeit von MQL5.community-Autorisierungsdaten im Terminal	TerminalInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TERMINAL_COMMUNITY_BALANCE	Balance des Benutzers auf MQL5.community	TerminalInfoDouble
TERMINAL_COMMUNITY_CONNECTION	Verbindung zum MQL5.community	TerminalInfoInteger
TERMINAL_COMPANY	Name der Gesellschaft	TerminalInfoString
TERMINAL_CONNECTED	Verbindung mit dem Handelsserver	TerminalInfoInteger
TERMINAL_CPU_CORES	Die Anzahl der CPU-Kerne im System	TerminalInfoInteger
TERMINAL_DATA_PATH	Ordner, wo Terminladaten aufbewahren werden	TerminalInfoString
TERMINAL_DISK_SPACE	Freier Festplattenspeicher für den Ordner MQL5\Files des Terminals (des Agenten), in MB	TerminalInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TERMINAL_DLLS_ALLOWED	Erlaubnis, DLL zu verwenden	TerminalInfoInteger
TERMINAL_EMAIL_ENABLED	Erlaubnis, Briefe zu senden mit der Verwendung von SMTP-Server und Login, die in Einstellungen des Terminals angegeben werden	TerminalInfoInteger
TERMINAL_FTP_ENABLED	Erlaubnis, Berichte via FTP zu senden an das angegebene Server für das in Terminal einstellungen angegebene Handelskonto	TerminalInfoInteger
TERMINAL_LANGUAGE	Sprache des	TerminalInfoString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Terminals	
TERMINAL_MAXBARS	Maximale Anzahl der Bars auf dem Chart	TerminalInfoInteger
TERMINAL_MEMORY_AVAILABLE	Freier Speicher vom verfahren des Terminals (den Agenten), in MB	TerminalInfoInteger
TERMINAL_MEMORY_PHYSICAL	Physikalischer Speicher im System, in Mb	TerminalInfoInteger
TERMINAL_MEMORY_TOTAL	Speicher der dem Verfahren des Terminals (des Agenten) verfügbar ist, in MB	TerminalInfoInteger
TERMINAL_MEMORY_USED	Die Größe des durch das Terminal (den Agent) verwendeten Speichers, in MB	TerminalInfoInteger

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TERMINAL_MQID	Flagge der Verfügbarkeit von MetaQuotes ID um Push-Benachrichtigungen zu senden	TerminalInfoInteger
TERMINAL_NAME	Terminal name	TerminalInfoString
TERMINAL_NOTIFICATIONS_ENABLED	Erlaubnis die Benachrichtigungen auf Smartphone zu senden	TerminalInfoInteger
TERMINAL_OPENCL_SUPPORT	Die version von unterstützten OpenCL in Form von 0x00010002 = 1.2. "0" bedeutet, dass OpenCL nicht unterstützt wird	TerminalInfoInteger
TERMINAL_PATH	Ordner aus dem Terminal gestartet wird	TerminalInfoString

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TERMINAL_PING_LAST	Der letzte bekannte Wert des Pings zum Handelsserver in Mikrosekunden. Eine Mikrosekunde ist eine millionstel Sekunde	TerminalInfoInteger
TERMINAL_SCREEN_DPI	Die Bildschirmauflösung wird in DPI (Dots per Inch) angegeben	TerminalInfoInteger
TERMINAL_TRADE_ALLOWED	Erlaubnis zu handeln	TerminalInfoInteger
TERMINAL_X64	Die Anzeige von "64-Bit-Terminal"	TerminalInfoInteger
THURSDAY	Donnerstag	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
TRADE_ACTION_DEAL	Handelsorder auf den sofortigen Dealabsc	MqlTradeRequest

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	hluss mit den angegebenen Parametern stellen (Marktorder stellen)	
TRADE_ACTION_MODIFY	Parameter der früher eingestellten Order verändern	MqlTradeRequest
TRADE_ACTION_PENDING	Handelsorder für Dealabschluss bei den angegebenen Bedingungen (wartende Order) stellen	MqlTradeRequest
TRADE_ACTION_REMOVE	Wartende Handelsserver entfernen	MqlTradeRequest
TRADE_ACTION_SLTP	die Werte Stop Loss und Take Profit der offenen Position	MqlTradeRequest

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	verändern	
TRADE_RETCODE_CANCEL	Anforderung ist vom Trader aufgehoben	MqlTradeResult
TRADE_RETCODE_CLIENT_DISABLES_AT	Autotrading ist vom Client-Terminal untersagt	MqlTradeResult
TRADE_RETCODE_CONNECTION	keine Verbindung mit dem Handelsserver	MqlTradeResult
TRADE_RETCODE_DONE	Anforderung ist erfüllt	MqlTradeResult
TRADE_RETCODE_DONE_PARTIAL	Anforderung ist partiell erfüllt	MqlTradeResult
TRADE_RETCODE_ERROR	Fehler der Anforderung Verarbeitung	MqlTradeResult
TRADE_RETCODE_FROZEN	Order oder Position sind eingefroren	MqlTradeResult
TRADE_RETCODE_INVALID	Ungültige	MqlTradeResult

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Anforderung	
TRADE_RETCODE_INVALID_EXPIRATION	Ungültiges Datum des Orderablaufs in der Anforderung	MqlTradeResult
TRADE_RETCODE_INVALID_FILL	Nicht unterstützter Durchführungstyp der Order nach dem Rest	MqlTradeResult
TRADE_RETCODE_INVALID_ORDER	Ungültiger oder verbotener Ordertyp	MqlTradeResult
TRADE_RETCODE_INVALID_PRICE	Ungültiger Preis in der Anforderung	MqlTradeResult
TRADE_RETCODE_INVALID_STOPS	Ungültige Stops in der Anforderung	MqlTradeResult
TRADE_RETCODE_INVALID_VOLUME	Ungültiges Volumen in der Anforderung	MqlTradeResult

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TRADE_RETCODE_LIMIT_ORDERS	Limit für Anzahl der Warteordern ist erreicht	MqlTradeResult
TRADE_RETCODE_LIMIT_VOLUME	Limit für Volumen der Ordern und Positionen für dieses Symbol ist erreicht	MqlTradeResult
TRADE_RETCODE_LOCKED	Anforderung ist für Verarbeitung blockiert	MqlTradeResult
TRADE_RETCODE_MARKET_CLOSED	Markt ist geschlossen	MqlTradeResult
TRADE_RETCODE_NO_CHANGES	Keine Veränderungen in der Anforderung	MqlTradeResult
TRADE_RETCODE_NO_MONEY	Nicht genug Geld, um Anforderung zu erfüllen	MqlTradeResult
TRADE_RETCODE_ONLY_REAL	Operation ist nur für	MqlTradeResult

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	reelle Kontos erlaubt	
TRADE_RETCODE_ORDER_CHANGED	Order Status hat sich verändert	MqlTradeResult
TRADE_RETCODE_PLACED	Order ist platziert	MqlTradeResult
TRADE_RETCODE_POSITION_CLOSED	Die Position mit dem angegebenen POSITION_IDENTIFIER wurde bereits geschlossen	MqlTradeResult
TRADE_RETCODE_PRICE_CHANGED	Preise haben sich verändert	MqlTradeResult
TRADE_RETCODE_PRICE_OFF	Es gibt keine Quotationen für die Verarbeitung der Anforderung	MqlTradeResult
TRADE_RETCODE_REJECT	Anforderung ist abgelehnt	MqlTradeResult
TRADE_RETCODE_REQUOTE	Requote	MqlTradeResult

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TRADE_RETCODE_SERVER_DISABLES_AT	Autotrading ist vom Server untersagt	MqlTradeResult
TRADE_RETCODE_TIMEOUT	Anforderung abgelehnt nach Zeitablaufs	MqlTradeResult
TRADE_RETCODE_TOO_MANY_REQUESTS	Sehr offene Anforderungen	MqlTradeResult
TRADE_RETCODE_TRADE_DISABLED	Handel ist verboten	MqlTradeResult
TRADE_TRANSACTION_DEAL_ADD	Hinzufügen des Deals zur Geschichte. Es erfolgt als Ergebnis der Orderausfüllung oder Durchführung der Operationen mit einem Kontostand.	MqlTradeTransaction
TRADE_TRANSACTION_DEAL_DELETE	Entfernung der Order aus der Geschichte. Es	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	kann eine Situation sein, wenn zuvor ausgeführter Deal vom Server entfernt wird. Zum Beispiel, ein Deal war vom externen Handelssystem (die Börse) entfernt, wohin er vom Broker übertragen wurde.	
TRADE_TRANSACTION_DEAL_UPDATE	Änderung des Deals in der Geschichte. Es kann eine Situation sein, wenn zuvor ausgeführter Deal auf dem Server geändert	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	wird. Zum Beispiel, ein Deal war im externen Handelssystem (die Börse) geändert, wohin er vom Broker übertragen wurde.	
TRADE_TRANSACTION_HISTORY_ADD	Hinzufügen der Order zur Geschichte als Ergebnis der Ausführung oder Stornierung.	MqlTradeTransaction
TRADE_TRANSACTION_HISTORY_DELETE	Entfernung der Order aus der Ordergeschichte. Dieser Typ ist für die Erweiterung der Funktionalität auf der Seite des Handel-	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Servers vorgesehen.	
TRADE_TRANSACTION_HISTORY_UPDATE	Änderung der Order, die sich in der Ordergeschichte befindet. Dieser Typ ist für die Erweiterung der Funktionalität auf der Seite des Handel-Servers vorgesehen.	MqlTradeTransaction
TRADE_TRANSACTION_ORDER_ADD	Hinzufügen von einer neuen offenen Order.	MqlTradeTransaction
TRADE_TRANSACTION_ORDER_DELETE	Entfernung der Order aus der offenen Auftragsliste. Eine Order kann aus der offenen Auftragsliste als Ergebnis	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	der Stellung der entsprechenden Anfrage oder als Ergebnis der Durchführung (Füllung) und Übertragung in die Geschichte entfernt werden.	
TRADE_TRANSACTION_ORDER_UPDATE	Änderung der offenen Order. Zu solchen Änderungen gehören nicht nur die offensichtlichen Änderungen seitens des Client-Terminals oder Handel-servers, sondern auch die Änderung ihres Zustand	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	es bei der Ausstellung (zum Beispeil, Übergang vom Zustand ORDER_STATE_STARTED im ORDER_STATE_PLACED oder vom ORDER_STATE_PLACED im ORDER_STATE_PARTIAL usw.).	
TRADE_TRANSACTION_POSITION	Änderung der Position, die mit der Ausführung des Deals nicht verbunden ist. Dieser Typ der Transaktion zeigt, dass die Position auf der Seite des Handel-	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Servers geändert wurde. Volumen, Eröffnungspreis sowie Stop Loss und Take Profit Ebenen von Position können geändert werden. Informationen über Änderungen werden in der Struktur MqlTradeTransaction durch den Handler OnTradeTransaction übergeben. Änderung der Position (Hinzufügen, Änderung oder Liquidation) als Ergebnis des	

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	Abschlusses des Deals zieht kein Erscheinen der Transaktion TRADE_TRANSACTION_POSITION nach sich.	
TRADE_TRANSACTION_REQUEST	Benachrichtigung, dass die Handelsanfrage vom Server bearbeitet ist und das Ergebnis seiner Bearbeitung empfangen ist. Für Transaktionen dieses Typs in der Struktur MqlTradeTransaction muss nur type Feld (Typ der Transakt	MqlTradeTransaction

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	ion) analysiert werden. Für zusätzliche Informationen muss man die zweiten und dritten Parameter der Funktion OnTradeTransaction (request und result) analysieren.	
TUESDAY	Dienstag	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
TYPE_BOOL	bool	MqlParam
TYPE_CHAR	char	MqlParam
TYPE_COLOR	color	MqlParam
TYPE_DATETIME	datetime	MqlParam
TYPE_DOUBLE	double	MqlParam
TYPE_FLOAT	float	MqlParam
TYPE_INT	int	MqlParam
TYPE_LONG	long	MqlParam
TYPE_SHORT	short	MqlParam
TYPE_STRING	string	MqlParam
TYPE_UCHAR	uchar	MqlParam

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
TYPE_UINT	uint	MqlParam
TYPE_ULONG	ulong	MqlParam
TYPE_USHORT	ushort	MqlParam
UCHAR_MAX	Maximaler Wert, der durch den Typ uchar dargestellt werden kann	Konstanten der numerischenTypen
UINT_MAX	Maximaler Wert, der durch den Typ uint dargestellt werden kann	Konstanten der numerischenTypen
ULONG_MAX	Maximaler Wert, der durch den Typ ulong dargestellt werden kann	Konstanten der numerischenTypen
UPPER_BAND	Obere Grenze	Linien der Indikatoren
UPPER_HISTOGRAM	Oberes Histogramm	Linien der Indikatoren
UPPER_LINE	Obere Linie	Linien der Indikatoren
USHORT_MAX	Maximaler Wert, der	Konstanten der numerischenTypen

© 2000-2025, MetaQuotes Ltd.

Konstante	Beschreibung	Verwendung
	durch den Typ ushort dargestellt werden kann	
VOLUME_REAL	Handelsvolumen	Preiskonstanten
VOLUME_TICK	Tickvolumen	Preiskonstanten
WEDNESDAY	Mittwoch	SymbolInfoInteger, SymbolInfoSessionQuote, SymbolInfoSessionTrade
WHOLE_ARRAY	Bedeutet die Anzahl von Elementen, die bis zum Feldende bleiben, bis das ganze Feld verarbeitet wird	Andere Konstanten
WRONG_VALUE	Konstante kann implizit reduziert werden zum Typ jeder Enumeration.	Andere Konstanten

© 2000-2025, MetaQuotes Ltd.