QuantShare

Table of contents

1. QuantShare CSharp & JScript.Net

1.1 Advanced Money Management

1.1.1 OnEndPeriod

QuantShare API Documentation

OnEndPeriod Members

Manage your trading system using the advanced money management script.

Fields

TradingSystemSettings (of type **MMTradingSystemSettings**)	Trading system settings
Variables (of type **MMVariables**)	A class that is used to manipulate variables
Portfolio (of type **MMPortfolio**)	Porftolio Metrics
Divers (of type **MMDivers**)	Divers functions
Orders (of type **MMOrders**)	A class that is used to generate orders
Data (of type **MMData**)	A class that is used to get signals data, parse formulas...
Functions (of type **MMOnEndPeriod**)	Functions related to the OnEndPeriod event

[TOP]

1.1.2 OnEndSimulation

QuantShare API Documentation

OnEndSimulation Members

Manage your trading system using the advanced money management script.

Fields

TradingSystemSettings (of type **MMTradingSystemSettings**)	Trading system settings
Variables (of type **MMVariables**)	A class that is used to manipulate variables
Portfolio (of type **MMPortfolio**)	Porftolio Metrics
Divers (of type **MMDivers**)	Divers functions
Functions (of type **MMOnEndSimulation**)	Functions related to the OnEndSimulation event

[TOP]

1.1.3 OnStartSimulation

QuantShare API Documentation

OnStartSimulation Members

Manage your trading system using the advanced money management script.

Fields

TradingSystemSettings (of type **MMTradingSystemSettings**)	Trading system settings
Variables (of type **MMVariables**)	A class that is used to manipulate variables
Portfolio (of type **MMPortfolio**)	Porftolio Metrics
Divers (of type **MMDivers**)	Divers functions
Functions (of type **MMOnStartSimulation**)	Functions related to the OnStartSimulation event
Optimize (of type **MMOptimize**)	A class that can be used to define variables that will be used in the optimization

[TOP]

1.1.4 OnClosePosition

OnClosePosition Members

Manage your trading system using the advanced money management script.

Fields

TradingSystemSettings (of type **MMTradingSystemSettings**)	Trading system settings
Variables (of type **MMVariables**)	A class that is used to manipulate variables
Portfolio (of type **MMPortfolio**)	Porftolio Metrics
Divers (of type **MMDivers**)	Divers functions
Orders (of type **MMOrders**)	A class that is used to generate orders
Data (of type **MMData**)	A class that is used to get signals data, parse formulas...
Functions (of type **MMOnClosePosition**)	Functions related to the OnClosePosition event

[TOP]

1.1.5 OnNewPosition

QuantShare API Documentation

OnNewPosition Members

Manage your trading system using the advanced money management script.

Fields

TradingSystemSettings (of type **MMTradingSystemSettings**)	Trading system settings
Variables (of type **MMVariables**)	A class that is used to manipulate variables
Portfolio (of type **MMPortfolio**)	Porftolio Metrics
Divers (of type **MMDivers**)	Divers functions
Orders (of type **MMOrders**)	A class that is used to generate orders
Data (of type **MMData**)	A class that is used to get signals data, parse formulas...
NewPosition (of type **MMNewPosition**)	Class containing information about the position the simulator is about to buy/short
Functions (of type **MMOnNewPosition**)	Functions related to the OnNewPosition event

[TOP]

1.1.6 Members

1.1.6.1 _TradingOrder

QuantShare API Documentation

_TradingOrder Members

Parent

Properties

OrderLag (of type **Int32**)	Execute order at tomorrow plus the number of bars specified in the OrderLag variable (-1 to execute at the current bar)
Symbol (of type **String**)	Security symbol
Var1 (of type **Object**)	Custom variable
NbShares (of type **Double**)	Number of shares
PositionValidFor (of type **Int32**)	For entry orders: The number of periods to keep the order alive (in case the order isn't filled) For exit orders: The number of periods to keep the order alive before transforming it to an open market order

Methods

CancelOrder (returns **Void**)	(optional reason **String**) Cancel the order
SetCommissions (returns **Void**)	(commissions **Double**) Update commissions value

[TOP]

1.1.6.2 CustomDatabaseValues

QuantShare API Documentation

CustomDatabaseValues Members

Parent

Properties

[Int32] (of type **Object**)	Gets the element at the specified index
Length (of type **Int32**)	Gets the number of elements for the current bar period

Methods

GetValue (returns **Object**)	(index **Int32**) Gets the element at the specified index

[TOP]

1.1.6.3 MMData

QuantShare API Documentation

MMData Members

A class that is used to get signals data, parse formulas...

Parent

Methods

GetCustomDatabaseData (returns **VectorCustom**)	(database **String**, field **String**, symbol name **String**) Gets the specified database field data
ParseFormula (returns **MMParser**)	(formula. Ex: a = rsi(14); **String**) Parse a vector-based formula
IsMarketData (returns **Boolean**)	(Symbol name **String**) Returns whether there is market data or not on the current date
GetPriceSeries (returns **TimeSeries**)	(parameterless) Returns a price series
IsSymbolExists (returns **Boolean**)	(symbol name **String**) Returns TRUE if the specified symbol exists in this trading system
IsSymbolExistsInDatabase (returns **Boolean**)	(symbol name **String**) Returns TRUE if the specified symbol exists in your database
GetSymbols (returns **String[]**)	(parameterless) Gets the strategy's symbols
GetAllSymbols (returns **String[]**)	(parameterless) Gets all symbols in your database
GetSymbolInfo (returns **Symbol**)	(symbol name **String**) Gets symbol's informatiion
GetBuySignals (returns **MMEntrySignals**)	(parameterless) Gets buy signals generated at the end of this period

GetShortSignals (returns **MMEntrySignals**)	(parameterless) Gets short signals generated at the end of this period
GetSellSignals (returns **MMExitSignals**)	(parameterless) Gets sell signals generated at the end of this period
GetCoverSignals (returns **MMExitSignals**)	(parameterless) Gets cover signals generated at the end of this period

[TOP]

1.1.6.4 MMDivers

QuantShare API Documentation

MMDivers Members

Divers functions

Parent

Properties

IsFirstBar (of type **Boolean**)	Specify whether the current bar is the first one or not
CurrentDate (of type **DateTime**)	Current Date

Methods

Output (returns **Void**)	(parameterless) Set Output. Output can be found in Trading System Report, under the Details tab (Log Panel).

[TOP]

1.1.6.5 MMEntrySignals

QuantShare API Documentation

MMEntrySignals Members

Parent

Properties

Count (of type **Int32**)	Gets the number of signals

Methods

GetSymbol (returns **String**)	(Signal index **Int32**) Gets the symbol name
GetDetail (returns **Double**)	(Signal index **Int32**, Tags: limitprice, rank, stoploss, stoptrailing, stopprofit, stopbars **String**) Gets a symbol detail

[TOP]

1.1.6.6 MMExitPosition

QuantShare API Documentation

MMExitPosition Members

Parent

Properties

EntryDate (of type **DateTime**)	Current position entry date
SignalEntryDate (of type **DateTime**)	Current position signal date. Signal date is the date where the symbol buy/short value became TRUE. Entry date is the date where the symbol was bought or shorted. The Signal date could be different than the Entry date when the buy/short entry price could not be met on the signal date.
Symbol (of type **String**)	Current position symbol
PositionEquity (of type **Double**)	Current position size
Performance (of type **Double**)	Current position performance (in percentage)
Profit (of type **Double**)	Current position profit or loss
EntryPrice (of type **Double**)	Current position entry price
BarsSinceEntry (of type **Int32**)	Current position number of bars held
MAE (of type **Double**)	Current position maximum adverse excursion
MFE (of type **Double**)	Current position maximum favorable excursion
NbShares (of type **Int32**)	Current position number of shares
MaximumDrawdown (of type **Double**)	Current position maximum drawdown
NumberOfScaleIn (of type **Int32**)	Current position number of scale-in trades
NumberOfScaleOut (of type **Int32**)	Current position number of scale-out trades

LastPrice (of type **Double**)	Current position last price
Var1 (of type **Object**)	Custom variable
Var2 (of type **Double**)	Custom variable
Var3 (of type **Double**)	Custom variable
Var4 (of type **Double**)	Custom variable
StopSettings (of type **MMStopSettings**)	Stop Settings

Methods

GetValueFromFormula (returns **Double**)	(parameterless) Parse a vector-based formula and returns the vector value for the current date. Example: GetValueFromFormula('a = rsi(14);', 'a'); -> to get the RSI value of the current symbol for the current date.

[TOP]

1.1.6.7 MMExitSignals

QuantShare API Documentation

MMExitSignals Members

Parent

Properties

Count (of type **Int32**)	Gets the number of signals

Methods

GetSymbol (returns **String**)	(Signal index **Int32**) Gets the symbol name
GetPrice (returns **Double**)	(Signal index **Int32**) Gets the symbol sell/cover limit price

[TOP]

1.1.6.8 MMNewPosition

QuantShare API Documentation

MMNewPosition Members

Class containing information about the position the simulator is about to buy/short

Parent

Properties

IsLong (of type **Boolean**)	Specify whether the new position is Long or Short
Price (of type **Double**)	The new position price
NbShares (of type **Int32**)	The number of shares for the current position
Rank (of type **Double**)	The Rank of this symbol in the specified long or short ranking system
Symbol (of type **String**)	The symbol name
RemainingCash (of type **Double**)	Portfolio cash at the end of the previous period
NbPositions (of type **Int32**)	The current number of positions in the portfolio
NbLongPositions (of type **Int32**)	The current number of long positions in the portfolio
NbShortPositions (of type **Int32**)	The current number of short positions in the portfolio

[TOP]

1.1.6.9 MMOnClosePosition

QuantShare API Documentation

MMOnClosePosition Members

Functions related to the OnClosePosition event

Parent

Methods

AddTradeMetric (returns **Void**)	(Metric name **String**, Metric value **Double**) Adds a metric for this position
GetPositionDetails (returns **MMExitPosition**)	(parameterless) Gets details about the current position
DontCloseTheCurrentPosition (returns **Void**)	(parameterless) Restricts the simulator from exiting the current position
DontClosePositionsForThisPeriod (returns **Void**)	(parameterless) Restricts the simulator from exiting all the positions within this period (date)
UpdateNumberOfShares (returns **Boolean**)	(Number of shares **Int32**) Updates the number of shares to sell/cover
UpdateOrder (returns **Boolean**)	(Order - Example: To create a limit order -> Orders.LimitOrder(price) **_TradingOrder**) Updates the order used to exit this position

[TOP]

1.1.6.10 MMOnEndPeriod

QuantShare API Documentation

MMOnEndPeriod Members

Functions related to the OnEndPeriod event

Parent

Methods

AddMetric (returns **Void**)	(Metric name **String**, Metric value for the current period (date) **Double**) Adds a time-series metric to the simulation
SkipPeriods (returns **Void**)	(Number of periods **Double**) Skips money management execution for the specified number of periods (for all symbols)
RejectNewPositionsDuringTheNextPeriods (returns **Void**)	(Number of periods **Double**) Instructs the simulator to not open new positions during the specified number of periods
CreatePositionSettings (returns **MMPositionSettings**)	(Specify the category of the position. Long, Short or any custom value **String**) A position settings can be used to add additional information to a submitted order
CloseAllPositions (returns **Void**)	(Number of periods to skip executing OnEndPeriod script **Double**) Closes all portfolio's positions using market-at-open orders
CloseAllPositions (returns **Void**)	(Number of periods to skip executing OnEndPeriod script **Double**, Long for long orders - Short for short orders **String**) Closes portfolio's positions for the specified category using market-at-open orders
CloseAllPositions (returns **Void**)	(Number of periods to skip executing OnEndPeriod script **Double**, Trading order **_TradingOrder**) Closes all portfolio's positions
CloseAllPositions (returns **Void**)	(Number of periods to skip executing OnEndPeriod script **Double**, Trading order **_TradingOrder**, Long for long orders - Short for short orders **String**) Closes portfolio's positions for the specified category

GetNumberShares (returns **Double**)	(Symbol name **String**, Long or Short **Boolean**, Trading order **_TradingOrder**) Gets the approximative number of shares to be bought or shorted
GetNumberShares (returns **Double**)	(Symbol name **String**, Long or Short **Boolean**, Trading order **_TradingOrder**, Long for long orders - Short for short orders **String**) Gets the approximative number of shares to be bought or shorted
AddLongPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**) Adds a new long position
AddLongPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**) Adds a new long position - Let the simulator automatically choose the number of shares to buy
AddLongPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**, Create a position settings using the function: CreatePositionSettings **MMPositionSettings**) Adds a new long position
AddLongPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**, Create a position settings using the function: CreatePositionSettings **MMPositionSettings**) Adds a new long position - Let the simulator automatically choose the number of shares to buy
AddShortPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**) Adds a new short position
AddShortPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**) Adds a new short position - Let the simulator automatically choose the number of shares to short
AddShortPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**, Create a position settings using the function: CreatePositionSettings **MMPositionSettings**) Adds a new short position
AddShortPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**, Create a position settings using the function: CreatePositionSettings **MMPositionSettings**) Adds a new short position - Let the simulator automatically choose the number of shares to short

SellPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**) Sell a long position
SellPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**) Sell a long position - Sell all shares
SellPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**, Long, Short or any custom category **String**) Sell a long position
SellPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**, Long, Short or any custom category **String**) Sell a long position - Sell all shares
CoverPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**) Cover a short position
CoverPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**) Cover a short position - Cover all shares
CoverPosition (returns **Boolean**)	(Symbol name **String**, Number of shares **Double**, Trading order **_TradingOrder**, Long, Short or any custom category **String**) Cover a short position
CoverPosition (returns **Boolean**)	(Symbol name **String**, Trading order **_TradingOrder**, Long, Short or any custom category **String**) Cover a short position - Cover all shares

[TOP]

1.1.6.11 MMOnEndSimulation

QuantShare API Documentation

MMOnEndSimulation Members

Functions related to the OnEndSimulation event

Parent

Methods

AddMetric (returns **Void**)	(Metric name **String**, Metric value **Double**) Adds a metric to the simulator

[TOP]

1.1.6.12 MMOnNewPosition

QuantShare API Documentation

MMOnNewPosition Members

Functions related to the OnNewPosition event

Parent

Properties

StopSettings (of type **MMStopSettings**)	Stop Settings

Methods

RejectPosition (returns **Void**)	(parameterless) Reject the current position
RejectAllPositionsForThisPeriod (returns **Void**)	(parameterless) Reject all positions for the current period
UpdateNumberOfShares (returns **Boolean**)	(Number of shares **Int32**) Update the number of shares to buy/short
UpdateOrder (returns **Boolean**)	(Order - Example: To create a limit order -> Orders.LimitOrder(price) **_TradingOrder**) Update the order used to enter this position

[TOP]

1.1.6.13 MMOnStartSimulation

QuantShare API Documentation

MMOnStartSimulation Members

Functions related to the OnStartSimulation event

Parent

Methods

SetTextInput (returns **Void**)	(Variable name **String**, Default Value **String**, Description **String**) Gets a text value for the Money Management Input Form, use the Variable Class to get the value
SetTextInput (returns **Void**)	(Variable name **String**, Default Value **String**, Description **String**, Set the possible values **String[]**) Gets a text value for the Money Management Input Form, use the Variable Class to get the value
SetNumericInput (returns **Void**)	(Variable name **String**, Default Value **Double**, Description **String**) Gets a numeric value for the Money Management Input Form, use the Variable Class to get the value
AddReportMetric (returns **Void**)	(parameterless) Adds a variable to the simulator report grid - Variables can be updated later using 'Variables.SetVariable'

[TOP]

1.1.6.14 MMOptimize

QuantShare API Documentation

MMOptimize Members

A class that can be used to define variables that will be used in the optimization

Parent

Methods

OptimizeText (returns **Void**)	(variable name **String**, values **String[]**) Creates an optimizable variable, use the Variable Class to get the value
OptimizeDouble (returns **Void**)	(variable name **String**, start value **Double**, end value **Double**, step value **Double**) Creates an optimizable variable, use the Variable Class to get the value

[TOP]

1.1.6.15 MMOrders

QuantShare API Documentation

MMOrders Members

A class that is used to generate orders

Parent

Methods

OpenMarketOrder (returns **_TradingOrder**)	(parameterless) Creates an market order that is executed at the open
OpenMarketOrder (returns **_TradingOrder**)	(Execute order at the next bar plus the number of specified bars (-1 to execute at the current bar) **Int32**, Keep the order active for the specified number of bars **Int32**) Creates an market order that is executed at the open
CloseMarketOrder (returns **_TradingOrder**)	(parameterless) Creates an market order that is executed at the close
CloseMarketOrder (returns **_TradingOrder**)	(Execute order at the next bar plus the number of specified bars (-1 to execute at the current bar) **Int32**, Keep the order active for the specified number of bars **Int32**) Creates an market order that is executed at the close
LimitOrder (returns **_TradingOrder**)	(limit price **Double**) Creates an market order that is executed at the specified price
LimitOrder (returns **_TradingOrder**)	(limit price **Double**, Execute order at the next bar plus the number of specified bars (-1 to execute at the current bar) **Int32**, Keep the order active for the specified number of bars **Int32**) Creates an market order that is executed at the specified price
GetPendingBuyOrders (returns **_TradingOrder[]**)	(parameterless) Gets the list of all BUY pending orders
GetPendingSellOrders (returns **_TradingOrder[]**)	(parameterless) Gets the list of all SELL pending orders
GetPendingShortOrders (returns **_TradingOrder[]**)	(parameterless)

	Gets the list of all SHORT pending orders
GetPendingCoverOrders (returns **_TradingOrder[]**)	(parameterless) Gets the list of all COVER pending orders

[TOP]

1.1.6.16 MMParser

QuantShare API Documentation

MMParser Members

Parent

Properties

IsErrors (of type **Boolean**)	Returns whether an error occurred while parsing the formula

Methods

GetErrors (returns **String**)	(parameterless) Gets the last execution errors
GetTimeSeries (returns **TimeSeries**)	(symbol name **String**, variable within the formula **String**) Retrieves a Time-Series.

[TOP]

1.1.6.17 MMPortfolio

QuantShare API Documentation

MMPortfolio Members

Porftolio Metrics

Parent

Properties

Equity (of type **Double**)	Current portfolio equity
Drawdown (of type **Double**)	Current portfolio drawdown
Return (of type **Double**)	Current portfolio return in percentage
PercentageInvested (of type **Double**)	Current portfolio percentage of invested money
NbPositions (of type **Int32**)	The current number of positions in the portfolio
NbLongPositions (of type **Int32**)	The current number of long positions in the portfolio
NbShortPositions (of type **Int32**)	The current number of short positions in the portfolio

Methods

IsInPortfolio (returns **Boolean**)	(symbol name **String**, Long or Short position **Boolean**) Returns whether a security symbol is in the portfolio
GetReturn (returns **Double**)	(Number of days/minutes **Int32**) Portfolio return over the specified number of days(Historical)/minutes(Intraday)
GetReturn (returns **Double**)	(Number of days/minutes **Int32**, Long, Short or any custom category **String**) Category return over the specified number of days(Historical)/minutes(Intraday)
IsInPortfolio (returns **Boolean**)	(symbol name **String**, Long or Short position **Boolean**, Long, Short or any custom category **String**) Returns TRUE if a security is in the portfolio

IsInPendingOrders (returns **Boolean**)	(symbol name **String**, Specify the order type: Buy, Sell, Short or Cover **String**) Returns whether a security symbol is in the pending orders list
IsInPendingOrders (returns **Boolean**)	(symbol name **String**, Specify the order type: Buy, Sell, Short or Cover **String**, Long, Short or any custom category **String**) Returns whether a security symbol is in the pending orders list
GetPosition (returns **MMPosition**)	(symbol name **String**) Gets a position from the portfolio
GetPosition (returns **MMPosition**)	(symbol name **String**, Long, Short or any custom category **String**) Gets a position from the portfolio
GetAllPositions (returns **MMPosition[]**)	(parameterless) Gets all positions generated by the simulator - open and closed positions
GetLastClosedPositions (returns **MMPosition[]**)	(parameterless) Gets all previously closed positions
GetClosedPositions (returns **MMPosition[]**)	(DateTime **DateTime**) Gets all positions that were closed in or after the specified date
GetOpenPositions (returns **MMPosition[]**)	(parameterless) Gets the current portfolio positions
GetCategorySettings (returns **Double**)	(Long, Short or a custom category **String**, Returns PercentToInvest when true otherwise it returns the number of positions **Boolean**) Gets the percent of equity to invest and the maximum number of positions of a category
UpdateCategorySettings (returns **Void**)	(Long, Short or a custom category **String**, Percent of equity to invest - Example: 50 to assign 50% of the equity to the current category **Double**, Number of positions **Int32**, Order type to scale-in or scale-out positions. Set null to ignore scaling. Scaling is automatically ignored if the number of positions is changed. **_TradingOrder**) Updates the percent of equity to invest and the number of positions to allow for a category
GetAvailableCash (returns **Double**)	(Long for long category - Short for short category **String**)

	Gets the cash available for the specified category
GetAvailableNumberOfPositions (returns **Double**)	(Long for long orders - Short for short orders **String**) Gets the number of positions available for the specified category (Maximum positions - open positions - entry orders + exit orders)
GetNumberOfPositions (returns **Double**)	(Long for long category - Short for short category **String**) Gets the number of open positions for the specified category
GetAllowedNumberOfPositions (returns **Double**)	(Long for long category - Short for short category **String**) Gets the number of allowed positions for the specified category
GetTotalEquity (returns **Double**)	(Long for long category - Short for short category **String**) Gets the positions value plus cash available for the specified category
GetPositionsSize (returns **Double**)	(Long for long category - Short for short category **String**) Gets the positions value for the specified category
UpdateMarginFactor (returns **Int32**)	(margin factor **Double**) Updates the margin factor
UpdateMarginFactor (returns **Int32**)	(margin factor (higher or equal to 1) **Double**, order type used to scale-in or scale-out positions **_TradingOrder**) Updates the margin factor and scale-in or scale-out existing positions to meet the new requirements
UpdatePercentInvested (returns **Int32**)	(percent invested (between 0 and 100) **Double**) Updates the percentage invested value
UpdatePercentInvested (returns **Int32**)	(percent invested (between 0 and 100) **Double**, order type used to scale-in or scale-out positions **_TradingOrder**) Updates the percentage invested value and scale-in or scale-out existing positions to meet the new requirements

[TOP]

1.1.6.18 MMPosition

MMPosition Members

Parent

Properties

EntryDate (of type **DateTime**)	Current position entry date
SignalEntryDate (of type **DateTime**)	Current position signal date. Signal date is the date where the symbol buy/short value became TRUE. Entry date is the date where the symbol was bought or shorted. The Signal date could be different than the Entry date when the buy/short entry price could not be met on the signal date.
Symbol (of type **String**)	Current position symbol
Category (of type **String**)	Gets position category
PositionEquity (of type **Double**)	Current position size
Performance (of type **Double**)	Current position performance (in percentage)
Profit (of type **Double**)	Current position profit or loss
CloseReason (of type **String**)	Returns the close reason
EntryPrice (of type **Double**)	Current position entry price
BarsSinceEntry (of type **Int32**)	Current position number of bars held
MAE (of type **Double**)	Current position maximum adverse excursion
MFE (of type **Double**)	Current position maximum favorable excursion
NbShares (of type **Int32**)	Current position number of shares
MaximumDrawdown (of type **Double**)	Current position maximum drawdown

NumberOfScaleIn (of type **Int32**)	Current position number of scale-in trades
NumberOfScaleOut (of type **Int32**)	Current position number of scale-out trades
LastPrice (of type **Double**)	Current position last price
IsLong (of type **Boolean**)	Specifies whether this is a long or short position
Rank (of type **Double**)	The rank of this symbol in the specified long or short ranking system
Var1 (of type **Object**)	Custom variable
Var2 (of type **Double**)	Custom variable
Var3 (of type **Double**)	Custom variable
Var4 (of type **Double**)	Custom variable
StopSettings (of type **MMStopSettings**)	Stop Settings

Methods

ScaleIn (returns **Void**)	(Number of shares **Int32**) Scale-in position
ScaleIn (returns **Void**)	(Number of shares **Int32**, Trading order **_TradingOrder**) Scale-in position
ScaleOut (returns **Void**)	(Number of shares **Int32**) Scale-out position
ScaleOut (returns **Void**)	(Number of shares **Int32**, Trading order **_TradingOrder**) Scale-out position
ClosePosition (returns **Void**)	(parameterless) Closes the current position
ClosePosition (returns **Boolean**)	(Order - Example: To create a limit order -> Orders.LimitOrder(price) **_TradingOrder**) Closes the current position

GetValueFromFormula (returns **Double**)	(parameterless) Parses a vector-based formula and returns the vector value for the current date. Example: GetValueFromFormula('a = rsi(14);', 'a'); -> to get the RSI value of the current symbol for the current date.
AddTradeMetric (returns **Void**)	(Metric name **String**, Metric value **Double**) Adds a metric for this position

[TOP]

1.1.6.19 MMPositionSettings

QuantShare API Documentation

MMPositionSettings Members

Parent

Methods

AddTrailingStop (returns **Void**)	(Example: 10 (% or Points depending on the trading system settings) **Double**) Adds a trailing stop
AddStopLoss (returns **Void**)	(Example: 10 (% or Points depending on the trading system settings) **Double**) Adds a stop loss
AddNBarsStop (returns **Void**)	(Example: 10 (% or Points depending on the trading system settings) **Double**) Adds an Nbars stop
AddProfitStop (returns **Void**)	(Example: 10 (% or Points depending on the trading system settings) **Double**) Adds a profit stop

[TOP]

1.1.6.20 MMStopSettings

QuantShare API Documentation

MMStopSettings Members

Parent

Properties

StopLossEnabled (of type **Boolean**)	[Deprecated]
TrailingStopEnabled (of type **Boolean**)	[Deprecated]
ProfitStopEnabled (of type **Boolean**)	[Deprecated]
NBarsStopEnabled (of type **Boolean**)	[Deprecated]
StopLossOption (of type **Double**)	Stop Loss Option (0=Disable, 1=Percent, 2=Point) - This is a global setting for all trades
TrailingStopOption (of type **Double**)	Stop Loss Option (0=Disable, 1=Percent, 2=Point) - This is a global setting for all trades
ProfitStopOption (of type **Double**)	Stop Loss Option (0=Disable, 1=Percent, 2=Point) - This is a global setting for all trades
NBarsStopOption (of type **Double**)	Stop Loss Option (0=Disable, 1=Percent, 2=Point) - This is a global setting for all trades
StopLoss (of type **Double**)	Stop Loss - Set (double.NaN) to disable
TrailingStop (of type **Double**)	Trailing Stop - Set (double.NaN) to disable
ProfitStop (of type **Double**)	Profit Stop - Set (double.NaN) to disable
NBarsStop (of type **Double**)	N-Bars Stop - Set (double.NaN) to disable

[TOP]

1.1.6.21 MMTradingSystemSettings

QuantShare API Documentation

MMTradingSystemSettings Members

Trading system settings

Parent

Properties

InitialEquity (of type **Double**)	Portfolio initial equity
StartDate (of type **DateTime**)	Simulation start date
EndDate (of type **DateTime**)	Simulation end date
NbPositions (of type **Int32**)	Portfolio maximum number of positions
PercentToInvest (of type **Double**)	Portfolio percent of capital to invest
Formula (of type **String**)	Trading system's formula
IsLong (of type **Boolean**)	Returns true if the trading system supports long positions
IsShort (of type **Boolean**)	Returns true if the trading system supports short positions
Symbols (of type **String[]**)	Trading system's symbols
MaxWeight (of type **Double**)	Maximum Position Weight - Example: 2 - New Positions can have a maximum size of two times the average expected position size Set 0 to disable this setting.
MinWeight (of type **Double**)	Minimum Position Weight - Example: 0.4 - New Positions can have a minimum size of 40% the average expected position size Set 0 to disable this setting.
IsAutoAdjustCashNewPositions (of type **Boolean**)	Specify whether to adjust the amount of cash available for new positions after a position is entered or at the end of a period. By default, the amount of cash is updated after each position.

[TOP]

1.1.6.22 MMVariables

QuantShare API Documentation

MMVariables Members

A class that is used to manipulate variables

Parent

Properties

[String] (of type **Object**)	Gets a variable value

Methods

SetVariable (returns **Void**)	(Variable Name **String**, Value **Object**) Assigns a value to a variable
GetVariable (returns **Object**)	(Variable Name **String**) Gets a variable value
RemoveVariable (returns **Boolean**)	(Variable Name **String**) Removes a variable
GetVariables (returns **String[]**)	(parameterless) Gets a list of all the declared variables
IsVariableExists (returns **Boolean**)	(Variable Name **String**) Returns true if the variable exist, otherwise returns false

[TOP]

1.1.6.23 Symbol

Symbol Members

Parent

Properties

Name (of type **String**)	Security name
Name1 (of type **String**)	Security name1
Name2 (of type **String**)	Security name2
Name3 (of type **String**)	Security name3
FullName (of type **String**)	Security full name
Group (of type **String**)	Security group
Market (of type **String**)	Security market
Industry (of type **String**)	Security industry
Sector (of type **String**)	Security sector
Description (of type **String**)	Security description
Country (of type **String**)	Security country
Address (of type **String**)	Company address
Website (of type **String**)	Company website
IsActive (of type **Boolean**)	Specify whether the current security is active or not
IsTradable (of type **Boolean**)	Specify whether the current security is tradable or not

Currency (of type **String**)	Security currency
TickSize (of type **Double**)	Security Tick Size
RoundLotSize (of type **Double**)	Security Round Lot Size
PointValue (of type **Double**)	Security Point Value
MarginDeposit (of type **Double**)	Security Margin Deposit
Multiplier (of type **Int32**)	Security Multiplier
Strike (of type **Double**)	Security strike price
ExpirationDate (of type **String**)	Security expiration date

Methods

GetIndexesFromList (returns **String[]**)	(parameterless) Get a list of all the indexes where the current security belongs to
AddIndex (returns **Void**)	(Index **String**) Add the current symbol to an index
RemoveIndex (returns **Void**)	(Index **String**) Remove the current symbol from an index

[TOP]

1.1.6.24 TimeSeries

TimeSeries Members

Parent

Properties

Count (of type **Int32**)	Gets the time-series number of elements
[Int32] (of type **Double**)	Gets a time-series value

Methods

GetValue (returns **Double**)	(lag **Int32**) Gets a time-series value

[TOP]

1.1.6.25 VectorCustom

VectorCustom Members

Parent

Properties

Count (of type **Int32**)	Gets the number of elements
[Int32] (of type **CustomDatabaseValues**)	Gets the elements at the specified bar index

Methods

GetValue (returns **CustomDatabaseValues**)	(lag **Int32**) Gets the elements at the specified bar index

[TOP]

1.2 Analyze Ranking System Metrics

1.2.1 Analyze Ranking System Metrics

QuantShare API Documentation

Analyze Ranking System Metrics Members

Fields

Buckets (of type **RankingSystemPerformanceDetails**)	Class containing information about the ranking system buckets
Fitness (of type **Double**)	Fitness value

[TOP]

1.2.2 Members

1.2.2.1 BucketDetails

QuantShare API Documentation

BucketDetails Members

Parent

Properties

AnnualReturn (of type **Double**)	Annual return
NbPositions (of type **Int32**)	Number of positions taken within this bucket
Label (of type **String**)	Bucket label
Score (of type **Double**)	Bucket score
DetailsDate (of type **BucketDetailsDate[]**)	Bucket's positions details for rebalance dates. Each array element corresponds to a rebalance date.
RebalancePeriod (of type **Int32**)	The ranking system rebalance period

[TOP]

1.2.2.2 BucketDetailsDate

BucketDetailsDate Members

Parent

Properties

AvgPerf (of type **Double**)	Average performance
AvgBarsHeld (of type **Double**)	Average bars held
Equity (of type **Double**)	Equity value
NbPositions (of type **Int32**)	Number of positions
Date (of type **DateTime**)	Rebalance date
Score (of type **Double**)	Score value for the current bucket and the current rebalance date

[TOP]

1.2.2.3 RankingSystemPerformanceDetails

QuantShare API Documentation

RankingSystemPerformanceDetails Members

Class containing information about the ranking system buckets

Parent

Properties

Buckets (of type **BucketDetails[]**)	Array containing the buckets details, the first array element contains the data of the first bucket...
LastBucket (of type **BucketDetails**)	Last bucket details

[TOP]

1.3 Analyze Rules Metrics

1.3.1 Analyze Rules Metrics

QuantShare API Documentation

Analyze Rules Metrics Members

Fields

Output (of type **Double**)	Average output per position
OutputPerBar (of type **Double**)	Average output per position and per bar
PercentPositive (of type **Double**)	Percent of positive positions
BestPosition (of type **Double**)	Best position output
WorstPosition (of type **Double**)	Worst position output
AvgBarsHeld (of type **Double**)	Average bars held per position
NbPositions (of type **Double**)	Number of positions
StandardDeviation (of type **Double**)	Standard deviation of the position outputs
Positions (of type **SymbolPositionDetails**)	Details of all positions
Fitness (of type **Double**)	Fitness value
Functions (of type **RulesFunctions**)	Function class that lets you add additional metrics

[TOP]

1.3.2 Members

1.3.2.1 PositionDetails

QuantShare API Documentation

PositionDetails Members

Parent

Properties

Output (of type **Double**)	The position output
NbBars (of type **Int32**)	The number of bars the current position was held
Date (of type **DateTime**)	Position entry date

[TOP]

1.3.2.2 SymbolPositionDetails

SymbolPositionDetails Members

Details of all positions

Parent

Properties

Symbol (of type **String**)	Symbol name
Positions (of type **PositionDetails[]**)	Positions details for the current symbol

[TOP]

1.3.2.3 RulesFunctions

QuantShare API Documentation

RulesFunctions Members

Function class that lets you add additional metrics

Parent

Methods

AddMetric (returns **Void**)	(metric name **String**, metric value **Double**) Adds a new metric
GetOutputName (returns **String**)	(parameterless) Returns the output name
GetRuleFormula (returns **String**)	(parameterless) Returns the rule formula

[TOP]

1.4 Downloader URL-Script

1.4.1 Downloader URL-Script

QuantShare API Documentation

Downloader URL-Script Members

This script is used to generated custom URLs. These URLs will be used to download data.

Fields

Functions (of type **URLScriptFunctions**)	Contains a list of methods

[TOP]

1.4.2 Members

1.4.2.1 BrowserActions

QuantShare API Documentation

BrowserActions Members

Parent

Properties

Visible (of type **Boolean**)	Select whether to display the browser or not

Methods

Sleep (returns **Void**)	(number of milliseconds **Int32**) Sleep during the specified period
Browse (returns **Void**)	(URL **String**) Browse to a specific URL
ClickLink (returns **Void**)	(Anchor name - Text that describes the link **String**) Click on a link
ClickLink1 (returns **Void**)	(Anchor name **String**) Click on a link that contains the specified anchor name
ClickButton (returns **Void**)	(Button name **String**) Click on a button
SubmitForm (returns **Void**)	(Form name **String**) Submit a form
AddContentToParser (returns **Void**)	(parameterless) Add current html code to parser
SetField (returns **Void**)	(Field name **String**, Field value **String**) Update a specific form field

[TOP]

1.4.2.2 HTMLElement

QuantShare API Documentation

HTMLElement Members

Parent

Properties

TagName (of type **String**)	Get the tag name
InnerText (of type **String**)	Get the inner text
InnerHTML (of type **String**)	Get the inner html

Methods

GetAttribute (returns **String**)	(attribute name **String**) Get an attribute value

[TOP]

1.4.2.3 Net

QuantShare API Documentation

Net Members

Parent

Methods

DownloadString (returns **String**)	(URL **String**) Download content from the specified URL
GetLinks (returns **String[]**)	(html content **String**) Get all links from the provided content
GetLinksByAnchor (returns **String[]**)	(html content **String**, anchor text **String**) Get links from the specified content that contains the provided anchor text
GetHTMLElements (returns **HTMLElement[]**)	(html content **String**) Get HTML elements by tag name
GetHTMLElementsByTag (returns **HTMLElement[]**)	(html content **String**, tag name **String**) Get HTML elements by tag name

[TOP]

1.4.2.4 Symbol

QuantShare API Documentation

Symbol Members

Parent

Properties

Name (of type **String**)	Security name
Name1 (of type **String**)	Security name1
Name2 (of type **String**)	Security name2
Name3 (of type **String**)	Security name3
FullName (of type **String**)	Security full name
Group (of type **String**)	Security group
Market (of type **String**)	Security market
Industry (of type **String**)	Security industry
Sector (of type **String**)	Security sector
Description (of type **String**)	Security description
Country (of type **String**)	Security country
Address (of type **String**)	Company address
Website (of type **String**)	Company website
IsActive (of type **Boolean**)	Specify whether the current security is active or not
IsTradable (of type **Boolean**)	Specify whether the current security is tradable or not

Currency (of type **String**)	Security currency
TickSize (of type **Double**)	Security Tick Size
RoundLotSize (of type **Double**)	Security Round Lot Size
PointValue (of type **Double**)	Security Point Value
MarginDeposit (of type **Double**)	Security Margin Deposit
Multiplier (of type **Int32**)	Security Multiplier
Strike (of type **Double**)	Security strike price
ExpirationDate (of type **String**)	Security expiration date

Methods

GetIndexesFromList (returns **String[]**)	(parameterless) Get a list of all the indexes where the current security belongs to
AddIndex (returns **Void**)	(Index **String**) Add the current symbol to an index
RemoveIndex (returns **Void**)	(Index **String**) Remove the current symbol from an index

[TOP]

1.4.2.5 URLPath

QuantShare API Documentation

URLPath Members

Parent

Methods

AddURL (returns **Void**)	(URL **String**, Message to display **String**) Add a URL to the current Path
AddURL (returns **Void**)	(URL **String**, Message to display **String**, Referrer URL **String**) Add a URL to the current Path
AddURL (returns **Void**)	(URL **String**, Message to display **String**, Referrer URL **String**, List of Fields **String[]**, List of Values **String[]**) Add a URL to the current Path using a POST method

[TOP]

1.4.2.6 URLScriptFunctions

QuantShare API Documentation

URLScriptFunctions Members

Contains a list of methods

Parent

Properties

Net (of type **Net**)	Functions to download, parse... HTML content
Browser (of type **BrowserActions**)	Use a browser to connect and download data

Methods

GetDefaultURL (returns **String**)	(parameterless) Returns the default URL
GetStartDate (returns **DateTime**)	(parameterless) Get the start date
GetEndDate (returns **DateTime**)	(parameterless) Get the end date
CreateURLPath (returns **URLPath**)	(parameterless) Create a URL Path class, the downloader must load all the URLs specified in this class in order to download your content
AddURL (returns **Void**)	(URL **String**, Message to display **String**) Add a URL to the download list
AddURLPath (returns **Void**)	(URL Path **URLPath**) Add a URL Path, the downloader must load all the specified URLs in order to download your content
GetValues (returns **String**)	(Parameter name **String**) Get a URL parameter value. Example: The current symbol
SetValues (returns **Void**)	(Parameter name **String**, Parameter value **String**) Set a URL parameter value. Example: The current symbol

GetAllValues (returns **String[]**)	(Parameter name **String**) Get all the URL parameter values. Example: All the selected symbols
GetSymbolInfo (returns **Symbol**)	(parameterless) Get Symbol information

[TOP]

1.5 Indicators

1.5.1 Indicators

QuantShare API Documentation

Indicators Members

Create an indicator using JScript.Net.

Fields

result (of type **VectorD**)	You have to assign values to this array
cFunctions (of type **CFunctions**)	Divers functions
TA (of type **QSFunctions**)	Trading Indicators
Chart (of type **Charts**)	Plot data on a chart

[TOP]

1.5.2 Members

1.5.2.1 CFunctions

QuantShare API Documentation

CFunctions Members

Divers functions

Parent

Properties

Open (of type **VectorD**)	Open price vector
Close (of type **VectorD**)	Close price vector
SymbolInfo (of type **Symbol**)	Symbol information
Timeframe (of type **Int32**)	Current Time-frame
IsIntraday (of type **Boolean**)	Returns whether the formula is applied to Intraday data or not
Symbol (of type **String**)	Symbol name
Date (of type **VectorDate**)	Date vector
High (of type **VectorD**)	High price vector
Low (of type **VectorD**)	Low price vector
Volume (of type **VectorD**)	Volume price vector
OpenInt (of type **VectorD**)	Open Interest price vector

Methods

SetForwardAndBackwardBars (returns **Void**)	(backward bars **Int32**, forward bars **Int32**) Set the number of forward and backward bars used by this function□Theses values are used by the application to optimize the parsing process Example: sma(close, 10), uses 10 backward bars and 0 forward bars

CompileFormula (returns **QSFormula**)	(formula **String**) Compile a vector-based formula
CreateNumericVector (returns **VectorD**)	(parameterless) Returns an empty numeric vector
CreateTextVector (returns **VectorS**)	(parameterless) Returns an empty text vector
GetVectorDouble (returns **VectorD**)	(variable name **String**) Get a numeric vector from the current formula
GetVectorString (returns **VectorS**)	(variable name **String**) Get a text vector from the current formula
CreateFormula (returns **QSFormula**)	(formula **String**) Create a vector-based formula
GetTimeframeData (returns **VectorCustomDouble**)	(timeframe **Int32**, close, open, high, low, volume or openint **String**) For each trading bar get the quotes for the specified timeframe that occurred during this bar interval
GetTimeframeData (returns **VectorCustomDouble**)	(symbol **String**, timeframe **Int32**, close, open, high, low, volume or openint **String**) For each trading bar get the quotes of the specific symbol and for the specified timeframe that occurred during this bar interval
GetTimeframeData (returns **VectorCustomDouble**)	(symbol **String**, timeframe **Int32**, close, open, high, low, volume or openint **String**, Specify whether to get historical or intraday data **Boolean**) For each trading bar get the quotes of the specific symbol and for the specified timeframe that occurred during this bar interval
GetTimeframeRawData (returns **VectorR**)	(symbol **String**, timeframe **Int32**, close, open, high, low, volume or openint **String**) Get all data for the specific symbol and timeframe - not adjusted to the current symbol's data
GetTimeframeRawData (returns **VectorR**)	(symbol **String**, timeframe **Int32**, close, open, high, low, volume or openint **String**, Specify whether to get historical or intraday data **Boolean**) Get all data for the specific symbol and timeframe - not adjusted to the current symbol's data

GetCustomDatabaseData (returns **VectorCustom**)	(database **String**, field **String**) For each trading bar get the elements of the database field data that occurred during this bar interval
GetCustomDatabaseData (returns **VectorCustom**)	(database **String**, field **String**, symbol name **String**) For each trading bar get the elements of the database field data that occurred during this bar interval
GetCustomDatabaseRawData (returns **VectorR**)	(database **String**, field **String**, symbol name **String**) Get all the elements of the database field - not adjusted to the current symbol's data

[TOP]

1.5.2.2 Charts

QuantShare API Documentation

Charts Members

Plot data on a chart

Parent

Methods

Plot (returns **Void**)	(array **VectorD**, description **VectorS**) Plot a time-series
Plot (returns **Void**)	(array **VectorD**, description **VectorS**, color **VectorColor**) Plot a time-series
Plot (returns **Void**)	(array **VectorD**, description **VectorS**, color **VectorColor**, chart type **ChartType**, style **Style**) Plot a time-series
Plot1 (returns **Void**)	(first time-series **VectorD**, second time-series **VectorD**, description1 **VectorS**, description2 **VectorS**, Fill color **VectorColor**, Color1 **VectorColor**, Color2 **VectorColor**, Chart **ChartType**, Style **Style**) Plot two time-series
PlotCandleStick (returns **Void**)	(parameterless) Plot an OHLC Chart
PlotCandleStick1 (returns **Void**)	(parameterless) Plot an OHLC Chart
PlotSymbol (returns **Void**)	(parameterless) Plot Symbols on a Chart
PlotArrow (returns **Void**)	(parameterless) Plot Arrows
PrintChart (returns **Void**)	(parameterless) Plot Text on a Chart
Print (returns **Void**)	(parameterless)

	Print Text (View -> Output)
UpdateSettings (returns **Void**)	(parameterless) Update Chart Settings
UpdatePrintSettings (returns **Void**)	(parameterless) Update Print Settings
UpdateColor (returns **Void**)	(parameterless) Update color of the last plot using the specified conditions
SetMinScale (returns **Void**)	(parameterless) Set the minimum chart scale
SetMaxScale (returns **Void**)	(parameterless) Set the maximum chart scale
SetScale (returns **Void**)	(Min Value **Double**, Max Value **Double**) Set the chart scale
SetPercentScale (returns **Void**)	(parameterless) Update chart scale by a percentage
SetHatchBrush (returns **Void**)	(parameterless) Set a hatch brush for the last plot

[TOP]

1.5.2.3 CustomDatabaseValues

QuantShare API Documentation

CustomDatabaseValues Members

Parent

Properties

[Int32] (of type **Object**)	Get the element at the specified index
Length (of type **Int32**)	Get the number of elements for the current bar period

Methods

GetValue (returns **Object**)	(index **Int32**) Get the element at the specified index

1.5.2.4 CustomDatabaseValuesDouble

QuantShare API Documentation

CustomDatabaseValuesDouble Members

Parent

Properties

[Int32] (of type **Double**)	Get the element at the specified index
Length (of type **Int32**)	Get the number of elements for the current bar period

Methods

GetValue (returns **Double**)	(index **Int32**) Get the element at the specified index

[TOP]

1.5.2.5 QSFormula

QSFormula Members

Parent

Methods

GetVectorDouble (returns **VectorD**)	(variable name **String**) Get a double variable from the compiled formula
GetVectorString (returns **VectorS**)	(variable name **String**) Get a string variable from the compiled formula
SetParameter (returns **Void**)	(variable name **String**, vector **VectorD**) Set a formula variable
Compile (returns **Boolean**)	(parameterless) Compile the formula. Returns true if the compilation succeeds

[TOP]

1.5.2.6 QSFunctions

QuantShare API Documentation

QSFunctions Members

Trading Indicators

Parent

Methods

NbInIndex (returns **VectorD**)	(Index name **VectorS**) Returns the number of symbols that are included in the specified index
BSignalLong (returns **VectorD**)	(Min profit **VectorD**, Max drawdown **VectorD**, Minimum days **VectorD**, Maximum days **VectorD**) Gives you the best long entries and exits depending on the parameters you choose|Example if you choose a minimum profit of 20 percent, a maximum drawdown of 10 percent,|a minimum number of bars equal to 10 and a maximum number of bars equal to 100|The system will give you the most profitable trades that meet these criteria|Plot the returned array in a graph that contains CandleStick data to see entries and exits arrows|Click on those arrows to see additional information
BSignalShort (returns **VectorD**)	(Min profit **VectorD**, Max drawdown **VectorD**, Minimum days **VectorD**, Maximum days **VectorD**) Gives you the best short entries and exits depending on the parameters you choose|Example if you choose a minimum profit of 20 percent, a maximum drawdown of 10 percent,|a minimum number of bars equal to 10 and a maximum number of bars equal to 100|The system will give you the most profitable trades that meet these criteria|Plot the returned array in a graph that contains CandleStick data to see entries and exits arrows|Click on those arrows to see additional information
BSignal (returns **VectorD**)	(Min profit **VectorD**, Max drawdown **VectorD**, Minimum days **VectorD**, Maximum days **VectorD**) Gives you the best long and short entries and exits depending on the parameters you choose|Example if you choose a minimum profit of 20 percent, a maximum drawdown of 10 percent,|a minimum number of bars equal to 10 and a maximum number of bars equal to 100|The system will give you the most profitable trades that meet these criteria|Plot the returned array in a graph that contains CandleStick data to see entries and exits arrows|Click on those arrows to see additional information
EntryLongProfit (returns **VectorD**)	(Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance

	of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached
EntryShortProfit (returns **VectorD**)	(Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached
EntryLongProfitPerBar (returns **VectorD**)	(Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance PER BAR of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached
EntryShortProfitPerBar (returns **VectorD**)	(Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance PER BAR of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached
EntryLongProfitCond (returns **VectorD**)	(Condition **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached , |then move to the bar number (next bar number + last trade holding period)
EntryShortProfitCond (returns **VectorD**)	(Condition **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached , |then move to the bar number (next bar number + last trade holding period)
EntryLongProfitPerBarCond (returns **VectorD**)	(Condition **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance PER BAR of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached , |then move to the bar number (next bar number + last trade holding period)
EntryShortProfitPerBarCond (returns **VectorD**)	(Condition **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance PER BAR of a system that short the current symbol at the next open bar and cover it when

	the system drawdown becomes|higher than the specified number or the maximum number of bar is reached , |then move to the bar number (next bar number + last trade holding period)
EntryLongProfitExitRule (returns **VectorD**)	(Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE
EntryShortProfitExitRule (returns **VectorD**)	(Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE
EntryLongProfitPerBarExitRule (returns **VectorD**)	(Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance PER BAR of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE
EntryShortProfitPerBarExitRule (returns **VectorD**)	(Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) For each bar in the 'close' array, calculates the performance PER BAR of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE
EntryLongProfitCondExitRule (returns **VectorD**)	(Condition **VectorD**, Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE, |then move to the bar number (next bar number + last trade holding period)
EntryShortProfitCondExitRule (returns **VectorD**)	(Condition **VectorD**, Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE, |then move to the bar number (next bar number + last trade holding period)
EntryLongProfitPerBarCondExitRule (returns **VectorD**)	(Condition **VectorD**, Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**)

	Calculates the performance PER BAR of a system that buy the current symbol at the next open bar and sell it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE, |then move to the bar number (next bar number + last trade holding period)
EntryShortProfitPerBarCondExitRule (returns **VectorD**)	(Condition **VectorD**, Exit **VectorD**, Max drawdown **VectorD**, Maximum days **VectorD**) Calculates the performance PER BAR of a system that short the current symbol at the next open bar and cover it when the system drawdown becomes|higher than the specified number or the maximum number of bar is reached or the exit rule is TRUE, |then move to the bar number (next bar number + last trade holding period)
CdlLadderbottom (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Ladder Bottom, The downtrend is finishing with four consecutive black candles, each closing lower than the previous day.
CdlLongleggeddoji (returns **VectorD**)	(parameterless) Long Legged Doji, The Long-legged Doji is composed of long upper and lower shadows.
CdlLongleggeddoji (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Long Legged Doji, The Long-legged Doji is composed of long upper and lower shadows.
CdlLongline (returns **VectorD**)	(parameterless) Long Line Candle, A long day (LONGLINE) represents a large price move from open to close.
CdlLongline (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Long Line Candle, A long day (LONGLINE) represents a large price move from open to close.
CdlMarubozu (returns **VectorD**)	(parameterless) Marubozu, in Japanese, Marubozu means close cropped or close-cut.
CdlMarubozu (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Marubozu, in Japanese, Marubozu means close cropped or close-cut.
CdlMatchinglow (returns **VectorD**)	(parameterless) Matching Low, The Matching Low pattern is similar to the

	Homing Pigeon patter, the exception being that the two days of the pattern close on their lows, at the same level.
CdlMatchinglow (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Matching Low, The Matching Low pattern is similar to the Homing Pigeon patter, the exception being that the two days of the pattern close on their lows, at the same level.
CdlMathold (returns **VectorD**)	(penetration **VectorD**) Mat Hold, The pattern appears during an uptrend, which is further confirmed by the first long white candlestick.
CdlMathold (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Mat Hold, The pattern appears during an uptrend, which is further confirmed by the first long white candlestick.
CdlMorningdojistar (returns **VectorD**)	(penetration **VectorD**) Morning Doji Star, A three day bullish reversal pattern that is very similar to the Morning Star.
CdlMorningdojistar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Morning Doji Star, A three day bullish reversal pattern that is very similar to the Morning Star.
CdlMorningstar (returns **VectorD**)	(penetration **VectorD**) Morning Star, The Morning Star is a bottom reversal signal.
CdlMorningstar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Morning Star, The Morning Star is a bottom reversal signal.
CdlOnneck (returns **VectorD**)	(parameterless) On-Neck Pattern, The On Neck Line pattern is almost a 'meeting line pattern', but the critical term is 'almost'.
CdlOnneck (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) On-Neck Pattern, The On Neck Line pattern is almost a 'meeting line pattern', but the critical term is 'almost'.
CdlPiercing (returns **VectorD**)	(parameterless) Piercing Pattern, The Piercing Pattern is composed of a two-candle formation in a down trending market.
CdlPiercing (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**)

	Piercing Pattern, The Piercing Pattern is composed of a two-candle formation in a down trending market.
CdlRickshawman (returns **VectorD**)	(parameterless) Rickshaw Man, Rickshaw man is a specific type of long-legged doji where the open and close are in the middle of the price range.
CdlRickshawman (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Rickshaw Man, Rickshaw man is a specific type of long-legged doji where the open and close are in the middle of the price range.
CdlRisefall3methods (returns **VectorD**)	(parameterless) Rising/Falling Three Methods, The Falling Three Method is basically the opposite of the Rising Three Method, The market has been in a downtrend.
CdlRisefall3methods (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Rising/Falling Three Methods, The Falling Three Method is basically the opposite of the Rising Three Method, The market has been in a downtrend.
CdlSeparatinglines (returns **VectorD**)	(parameterless) Separating Lines, You can identify it from the following points: The first day is a long white candle.
CdlSeparatinglines (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Separating Lines, You can identify it from the following points: The first day is a long white candle.
CdlShootingstar (returns **VectorD**)	(parameterless) Shooting Star, The Shooting Star is comprised of one candle.
CdlShootingstar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Shooting Star, The Shooting Star is comprised of one candle.
CdlShortline (returns **VectorD**)	(parameterless) Short Line Candle, Short days (SHORTLINES) can be interpreted by the same analytical process of the long candles.
CdlShortline (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**)

	Short Line Candle, Short days (SHORTLINES) can be interpreted by the same analytical process of the long candles.
CdlSpinningtop (returns **VectorD**)	(parameterless) Spinning Top, spinning Tops are depicted with small bodies relative to the shadows.
CdlSpinningtop (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Spinning Top, spinning Tops are depicted with small bodies relative to the shadows.
CdlStalledpattern (returns **VectorD**)	(parameterless) Stalled Pattern, Another pattern close to the Three White Soldiers pattern is the Stalled Pattern (commonly known as the Deliberation pattern).
CdlStalledpattern (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Stalled Pattern, Another pattern close to the Three White Soldiers pattern is the Stalled Pattern (commonly known as the Deliberation pattern).
CdlSticksandwich (returns **VectorD**)	(parameterless) Stick Sandwich, The Stick Sandwich looks somewhat like an ice cream sandwich.
CdlSticksandwich (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Stick Sandwich, The Stick Sandwich looks somewhat like an ice cream sandwich.
CdlTakuri (returns **VectorD**)	(parameterless) Takuri (Dragonfly Doji with very long lower shadow) : This function is contained within the Pattern Recognition set of indicators.
CdlTakuri (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Takuri (Dragonfly Doji with very long lower shadow) : This function is contained within the Pattern Recognition set of indicators.
CdlTasukigap (returns **VectorD**)	(parameterless) Tasuki Gap, The Upside Tasuki Gap is found in a rising trend.
CdlTasukigap (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**)

	Tasuki Gap, The Upside Tasuki Gap is found in a rising trend.
CdlThrusting (returns **VectorD**)	(parameterless) Thrusting Pattern, The Thrusting pattern is almost an 'On Neck' or an 'In Neck' pattern and resembles the Meeting Line pattern, also.
CdlThrusting (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Thrusting Pattern, The Thrusting pattern is almost an 'On Neck' or an 'In Neck' pattern and resembles the Meeting Line pattern, also.
CdlTristar (returns **VectorD**)	(parameterless) Tristar Pattern, The Tri Star pattern is relatively rare.
CdlTristar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Tristar Pattern, The Tri Star pattern is relatively rare.
CdlUnique3river (returns **VectorD**)	(parameterless) Unique 3 River, The Unique Three River Bottom is a bullish pattern, somewhat characteristic of the Morning Star Pattern.
CdlUnique3river (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Unique 3 River, The Unique Three River Bottom is a bullish pattern, somewhat characteristic of the Morning Star Pattern.
CdlUpsidegap2crows (returns **VectorD**)	(parameterless) Upside Gap Two Crows, The Upside Gap Two Crows is a three-day pattern.
CdlUpsidegap2crows (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Upside Gap Two Crows, The Upside Gap Two Crows is a three-day pattern.
CdlXsidegap3methods (returns **VectorD**)	(parameterless) Upside/Downside Gap Three Methods, the Bullish Upside Gap Three Methods Pattern, the market is in a strong bullish mood.
CdlXsidegap3methods (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Upside/Downside Gap Three Methods, the Bullish Upside

	Gap Three Methods Pattern, the market is in a strong bullish mood.
Absolute (returns **VectorD**)	(array **VectorD**) Calculate the absolute value of each element in the array
Atan (returns **VectorD**)	(array **VectorD**) Calculate the arc tangent of each element in the array
Ceil (returns **VectorD**)	(array **VectorD**) For each element in the array, calculate the lowest integer greater than or equal to the element value
Cos (returns **VectorD**)	(array **VectorD**) Calculate the cosinus of each element in the array
Sin (returns **VectorD**)	(array **VectorD**) Calculate the sinus of each element in the array
Tan (returns **VectorD**)	(array **VectorD**) Calculate the tangent of each element in the array
Cosh (returns **VectorD**)	(array **VectorD**) Calculate the hyperbolic cosine of each element in the array
Sinh (returns **VectorD**)	(array **VectorD**) Calculate the hyperbolic sine of each element in the array
Tanh (returns **VectorD**)	(array **VectorD**) Calculate the hyperbolic tangent of each element in the array
Exp (returns **VectorD**)	(array **VectorD**) For each element in the array, calculate 'e' raised to the element value
Floor (returns **VectorD**)	(array **VectorD**) For each element in the array, calculate the largest integer less than or equal to the element value
Frac (returns **VectorD**)	(array **VectorD**) Calculate the fractional part of each element in the array
Int (returns **VectorD**)	(array **VectorD**)

	Calculate the integer part of each element in the array
Log (returns **VectorD**)	(array **VectorD**) Calculate the natural logarithm (ln) of each element in the array
Log10 (returns **VectorD**)	(array **VectorD**) Calculate the base 10 logarithm of each element in the array
Round (returns **VectorD**)	(array **VectorD**, decimals **VectorD**) Calculate the rounded value of each element in the array
Sign (returns **VectorD**)	(array **VectorD**) Calculate the sign of each element in the array
Sqrt (returns **VectorD**)	(array **VectorD**) Calculate the square root of each element in the array
Pow (returns **VectorD**)	(array **VectorD**, power **VectorD**) For each element in the array, raise the element value to the specified power
IeeeRemainder (returns **VectorD**)	(array **VectorD**, array **VectorD**) For each element in the array, calculate the remainder resulting from the division of the element value by the specified number
DivRem (returns **VectorD**)	(array **VectorD**, array **VectorD**) For each element in the array, calculate the quotion
IsNull (returns **VectorD**)	(array **VectorD**) Returns 1 if all elements in the array equal to 0
IsNaN (returns **VectorD**)	(array **VectorD**) Returns 1 if all elements in the array equal to NaN
IsNoZero (returns **VectorD**)	(array **VectorD**) Returns 0 if at least one elements in the array is equal to 0
IsNoNaN (returns **VectorD**)	(array **VectorD**) Returns 0 if at least one elements in the array is equal to NaN
NaNtoZero (returns **VectorD**)	(Array **VectorD**, type **VectorD**)

	Change NaN value to Zero in the specified array.|Set zero to the parameter 'type' to update only the firsts NaN, Set one to update all NaN values
NaNtoLast (returns **VectorD**)	(array **VectorD**) Update the NA values with the last non NaN values
NbMarkets (returns **VectorD**)	(parameterless) Returns the number of markets
NbGroups (returns **VectorD**)	(parameterless) Returns the number of groups
NbSectors (returns **VectorD**)	(parameterless) Returns the number of sectors
NbIndustries (returns **VectorD**)	(parameterless) Returns the number of industries
NbIndexes (returns **VectorD**)	(parameterless) Returns the number of indexes
NbInMarket (returns **VectorD**)	(Market name **VectorS**) Returns the number of symbols that are included in the specified market
NbInGroup (returns **VectorD**)	(Group name **VectorS**) Returns the number of symbols that are included in the specified group
NbInSector (returns **VectorD**)	(Sector name **VectorS**) Returns the number of symbols that are included in the specified sector
NbInIndustry (returns **VectorD**)	(Industry name **VectorS**) Returns the number of symbols that are included in the specified industry
Minute (returns **VectorD**)	(parameterless) Returns the current bar's minute
Second (returns **VectorD**)	(parameterless) Returns the current bar's second
Interval (returns **VectorD**)	(parameterless)

	Returns the number of ticks between the current and previous bar date
NbDays (returns **VectorD**)	(period **VectorD**) Returns the number of days between the current bar and the bar referenced by the period value
IsTrue (returns **VectorD**)	(array **VectorD**, lag **VectorD**) Returns 1 if all the values between the current bar and the lag bar are superior to 0
IsFalse (returns **VectorD**)	(array **VectorD**, lag **VectorD**) Returns 1 if all the values between the current bar and the lag bar are inferior or equal to 0
TotalBars (returns **VectorD**)	(parameterless) Returns the total number of bars
IsLastBar (returns **VectorD**)	(parameterless) Returns 1 if the the current bar is the last one
LSwitch (returns **VectorD**)	(array **VectorD**, array **VectorD**) Returns 1 for all bars after the first array is 'true' until a 'true' occurs in the second array. In this case, next bars will get 0 until the first array is 'true' again.
AMA (returns **VectorD**)	(array **VectorD**, smoothing factor **VectorD**) Returns the adaptive moving average, which is a moving average with a time-variant smoothing factor.
HistoPrice (returns **VectorD**)	(System.Collections.Generic.List`1[System.String] **String**, day lag (O to get the current day) **VectorD**) Returns the historical close, open, high, low, volume or open interest of past days
HistoPrice (returns **VectorD**)	(System.Collections.Generic.List`1[System.String] **String**, day lag (O to get the current day) **VectorD**, Use Open Session Time. 1 for TRUE. **VectorD**) Returns the historical close, open, high, low, volume or open interest of past days. Ability to select whether to use session open time or not.
Cdl2crows (returns **VectorD**)	(parameterless) Two Crows, The Two Crows Pattern is a 3-day pattern.
Cdl2crows (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Two Crows, The Two Crows Pattern is a 3-day pattern.

Cdl3blackcrows (returns **VectorD**)	(parameterless) Three Black Crows, The Three Black Crows got their name from the resemblance of three crows looking down from their perch from a tree.
Cdl3blackcrows (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Three Black Crows, The Three Black Crows got their name from the resemblance of three crows looking down from their perch from a tree.
Cdl3inside (returns **VectorD**)	(parameterless) Three Inside Up/Down, Note that after the long candle day that is in the same direction of the trend that the Harami pattern occurs.
Cdl3inside (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Three Inside Up/Down, Note that after the long candle day that is in the same direction of the trend that the Harami pattern occurs.
Cdl3linestrike (returns **VectorD**)	(parameterless) Three-Line Strike : This function is contained within the Pattern Recognition set of indicators.
Cdl3linestrike (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Three-Line Strike : This function is contained within the Pattern Recognition set of indicators.
Cdl3outside (returns **VectorD**)	(parameterless) Three Outside Up/Down : This function is contained within the Pattern Recognition set of indicators.
Cdl3outside (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Three Outside Up/Down : This function is contained within the Pattern Recognition set of indicators.
Cdl3staRsinsouth (returns **VectorD**)	(parameterless) Three Stars In The South, The slow down of the trend is visually obvious.
Cdl3staRsinsouth (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Three Stars In The South, The slow down of the trend is visually obvious.

Cdl3whitesoldiers (returns **VectorD**)	(parameterless) Three Advancing White Soldiers, The Three White Soldiers (also known as The Advancing Three White Soldiers) is a healthy market reversal pattern.
Cdl3whitesoldiers (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Three Advancing White Soldiers, The Three White Soldiers (also known as The Advancing Three White Soldiers) is a healthy market reversal pattern.
CdlAbandonedbaby (returns **VectorD**)	(penetration **VectorD**) Abandoned Baby, A rare reversal pattern characterized by a gap followed by a Doji, which is then followed by another gap in the opposite direction.
CdlAbandonedbaby (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Abandoned Baby, A rare reversal pattern characterized by a gap followed by a Doji, which is then followed by another gap in the opposite direction.
CdlAdvanceblock (returns **VectorD**)	(parameterless) Advance Block, The Advance Block is somewhat indicative as the Three White Soldiers but it is a bearish signal.
CdlAdvanceblock (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Advance Block, The Advance Block is somewhat indicative as the Three White Soldiers but it is a bearish signal.
CdlBelthold (returns **VectorD**)	(parameterless) Belt-hold, The Belt Hold lines are formed by single candlesticks.
CdlBelthold (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Belt-hold, The Belt Hold lines are formed by single candlesticks.
CdlBreakaway (returns **VectorD**)	(parameterless) Breakaway, If a trend has been evident, the breakaway pattern, whether bullish or bearish initially indicates the acceleration of that trend.
CdlBreakaway (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Breakaway, If a trend has been evident, the breakaway pattern, whether bullish or bearish initially indicates the acceleration of that trend.

CdlClosingmarubozu (returns **VectorD**)	(parameterless) Closing Marubozu, a Closing Marubozu has no shadow at it's closing end.
CdlClosingmarubozu (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Closing Marubozu, a Closing Marubozu has no shadow at it's closing end.
CdlConcealbabyswall (returns **VectorD**)	(parameterless) Concealing Baby Swallow, The first two days of the signal, two Black Marubozus, demonstrate the continuation of the downtrend.
CdlConcealbabyswall (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Concealing Baby Swallow, The first two days of the signal, two Black Marubozus, demonstrate the continuation of the downtrend.
CdlCounterattack (returns **VectorD**)	(parameterless) Counterattack, Meeting Lines (or Counterattack Lines) are formed when opposite coloured bodies have the same closing price.
CdlCounterattack (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Counterattack, Meeting Lines (or Counterattack Lines) are formed when opposite coloured bodies have the same closing price.
CdlDarkcloudcover (returns **VectorD**)	(penetration **VectorD**) Dark Cloud Cover, The dark Cloud Cover is the bearish counterpart to the Piercing pattern.
CdlDarkcloudcover (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Dark Cloud Cover, The dark Cloud Cover is the bearish counterpart to the Piercing pattern.
CdlDoji (returns **VectorD**)	(parameterless) Doji, the Doji is one of the most important signals in candlestick analysis.
CdlDoji (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Doji, the Doji is one of the most important signals in candlestick analysis.

CdlDojistar (returns **VectorD**)	(parameterless) Doji Star, Upon seeing a Doji in an overbought or oversold condition, an extremely high probability reversal situation becomes evident.
CdlDojistar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Doji Star, Upon seeing a Doji in an overbought or oversold condition, an extremely high probability reversal situation becomes evident.
CdlDragonflydoji (returns **VectorD**)	(parameterless) Dragonfly Doji, The Dragonfly Doji occurs when trading opens, trades lower, then closes at the open price which is the high of the day.
CdlDragonflydoji (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Dragonfly Doji, The Dragonfly Doji occurs when trading opens, trades lower, then closes at the open price which is the high of the day.
CdlEngulfing (returns **VectorD**)	(parameterless) Engulfing Pattern, Two of the most compelling candlestick signals are the Bullish Engulfing Pattern and Bearish Engulfing Pattern.
CdlEngulfing (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Engulfing Pattern, Two of the most compelling candlestick signals are the Bullish Engulfing Pattern and Bearish Engulfing Pattern.
CdlEveningdojistar (returns **VectorD**)	(penetration **VectorD**) Evening Doji Star, A three day bearish reversal pattern similar to the Evening Star.
CdlEveningdojistar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Evening Doji Star, A three day bearish reversal pattern similar to the Evening Star.
CdlEveningstar (returns **VectorD**)	(penetration **VectorD**) Evening Star, The Evening Star pattern is a top reversal signal.
CdlEveningstar (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**, penetration **VectorD**) Evening Star, The Evening Star pattern is a top reversal signal.

CdlGapsidesidewhite (returns **VectorD**)	(parameterless) Up/Down-gap side-by-side white lines, The Up Side By Side White Lines Pattern appears in a bullish market.
CdlGapsidesidewhite (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Up/Down-gap side-by-side white lines, The Up Side By Side White Lines Pattern appears in a bullish market.
CdlGravestonedoji (returns **VectorD**)	(parameterless) Gravestone Doji, The Gravestone Doji is formed when the open and the close occur at the low end of the trading range.
CdlGravestonedoji (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Gravestone Doji, The Gravestone Doji is formed when the open and the close occur at the low end of the trading range.
CdlHammer (returns **VectorD**)	(parameterless) Hammer, The Hammer is comprised of one candle.
CdlHammer (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Hammer, The Hammer is comprised of one candle.
CdlHangingman (returns **VectorD**)	(parameterless) Hanging Man, The Hanging Man is also comprised of one candle.
CdlHangingman (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Hanging Man, The Hanging Man is also comprised of one candle.
CdlHarami (returns **VectorD**)	(parameterless) Harami Pattern, The Harami is an often seen formation.
CdlHarami (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Harami Pattern, The Harami is an often seen formation.
CdlHaramicross (returns **VectorD**)	(parameterless) Harami Cross Pattern, A two day pattern similar to the Harami (see CDLHARAMI()), the difference is that the last day is a Doji.

CdlHaramicross (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Harami Cross Pattern, A two day pattern similar to the Harami (see CDLHARAMI()), the difference is that the last day is a Doji.
CdlHighwave (returns **VectorD**)	(parameterless) High-Wave Candle : This function is contained within the Pattern Recognition set of indicators.
CdlHighwave (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) High-Wave Candle : This function is contained within the Pattern Recognition set of indicators.
CdlHikkake (returns **VectorD**)	(parameterless) Hikkake Pattern
CdlHikkake (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Hikkake Pattern
CdlHikkakemod (returns **VectorD**)	(parameterless) Modified Hikkake Pattern
CdlHikkakemod (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Modified Hikkake Pattern
CdlHomingpigeon (returns **VectorD**)	(parameterless) Homing Pigeon, The Homing Pigeon is the same as the Harami, except for the colour of the second day's body.
CdlHomingpigeon (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Homing Pigeon, The Homing Pigeon is the same as the Harami, except for the colour of the second day's body.
CdlIdentical3crows (returns **VectorD**)	(parameterless) Identical Three Crows, The Three Identical Crows have the same criteria as the Three Black Crows.
CdlIdentical3crows (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Identical Three Crows, The Three Identical Crows have the same criteria as the Three Black Crows.

CdlInneck (returns **VectorD**)	(parameterless) In-Neck Pattern, The In Neck pattern is almost a Meeting Line pattern.
CdlInneck (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) In-Neck Pattern, The In Neck pattern is almost a Meeting Line pattern.
CdlInvertedhammer (returns **VectorD**)	(parameterless) Inverted Hammer, The Inverted Hammer is comprised of one candle.
CdlInvertedhammer (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Inverted Hammer, The Inverted Hammer is comprised of one candle.
CdlKicking (returns **VectorD**)	(parameterless) Kicking, The Kicker signal is the most powerful signal of all.
CdlKicking (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Kicking, The Kicker signal is the most powerful signal of all.
CdlKickingbylength (returns **VectorD**)	(parameterless) Kicking - bull/bear determined by the longer marubozu, The Kicker signal is the most powerful signal of all.
CdlKickingbylength (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Kicking - bull/bear determined by the longer marubozu, The Kicker signal is the most powerful signal of all.
CdlLadderbottom (returns **VectorD**)	(parameterless) Ladder Bottom, The downtrend is finishing with four consecutive black candles, each closing lower than the previous day.
MidPoint (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) MidPoint over period
MidPrice (returns **VectorD**)	(timePeriod **VectorD**) MidPoint Price over period
MidPrice (returns **VectorD**)	(high **VectorD**, low **VectorD**, timePeriod **VectorD**)

	MidPoint Price over period
Llv (returns **VectorD**)	(timePeriod **VectorD**) Lowest value over a specified period
Llv (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Lowest value over a specified period
Minus_Di (returns **VectorD**)	(timePeriod **VectorD**) Minus Directional Indicator
Minus_Di (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Minus Directional Indicator
Minus_Dm (returns **VectorD**)	(timePeriod **VectorD**) Minus Directional Movement
Minus_Dm (returns **VectorD**)	(high **VectorD**, low **VectorD**, timePeriod **VectorD**) Minus Directional Movement
Mom (returns **VectorD**)	(timePeriod **VectorD**) Momentum
Mom (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Momentum
Natr (returns **VectorD**)	(timePeriod **VectorD**) Normalized Average True Range
Natr (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Normalized Average True Range
Obv (returns **VectorD**)	(parameterless) On Balance Volume
Obv (returns **VectorD**)	(close **VectorD**, volume **VectorD**) On Balance Volume
Plus_Di (returns **VectorD**)	(timePeriod **VectorD**) Plus Directional Indicator

Plus_Di (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Plus Directional Indicator
Plus_Dm (returns **VectorD**)	(timePeriod **VectorD**) Plus Directional Movement
Plus_Dm (returns **VectorD**)	(high **VectorD**, low **VectorD**, timePeriod **VectorD**) Plus Directional Movement
Ppo (returns **VectorD**)	(fastPeriod **VectorD**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Percentage Price Oscillator
Ppo (returns **VectorD**)	(close **VectorD**, fastPeriod **VectorD**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Percentage Price Oscillator
Roc (returns **VectorD**)	(timePeriod **VectorD**) Rate of change
Roc (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Rate of change
Rocp (returns **VectorD**)	(timePeriod **VectorD**) Rate of change Percentage
Rocp (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Rate of change Percentage
Rocr (returns **VectorD**)	(timePeriod **VectorD**) Rate of change ratio
Rocr (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Rate of change ratio
Rocr100 (returns **VectorD**)	(timePeriod **VectorD**) Rate of change ratio (scale 100)
Rocr100 (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Rate of change ratio (scale 100)
Rsi (returns **VectorD**)	(timePeriod **VectorD**)

	Relative Strength Index
Rsi (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Relative Strength Index
Sar (returns **VectorD**)	(acceleration **VectorD**, maximum **VectorD**) Parabolic SAR
Sar (returns **VectorD**)	(high **VectorD**, low **VectorD**, acceleration **VectorD**, maximum **VectorD**) Parabolic SAR
SarExt (returns **VectorD**)	(startValue **VectorD**, offsetOnReverse **VectorD**, accelerationInitLong **VectorD**, accelerationLong **VectorD**, accelerationMaxLong **VectorD**, accelerationInitShort **VectorD**, accelerationShort **VectorD**, accelerationMaxShort **VectorD**) Parabolic SAR - Extended
SarExt (returns **VectorD**)	(high **VectorD**, low **VectorD**, startValue **VectorD**, offsetOnReverse **VectorD**, accelerationInitLong **VectorD**, accelerationLong **VectorD**, accelerationMaxLong **VectorD**, accelerationInitShort **VectorD**, accelerationShort **VectorD**, accelerationMaxShort **VectorD**) Parabolic SAR - Extended
Sma (returns **VectorD**)	(timePeriod **VectorD**) Simple Moving Average
Sma (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Simple Moving Average
Stddev (returns **VectorD**)	(timePeriod **VectorD**) Standard Deviation
Stddev (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Standard Deviation
StochSlowd (returns **VectorD**)	(fastK_Period **VectorD**, slowK_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowD_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Slow D
StochSlowd (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, volume **VectorD**, fastK_Period **VectorD**, slowK_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowD_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**)

	Stochastic Slow D
StochSlowk (returns **VectorD**)	(fastK_Period **VectorD**, slowK_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Slow K
StochSlowk (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, volume **VectorD**, fastK_Period **VectorD**, slowK_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Slow K
StochRsiFastd (returns **VectorD**)	(timePeriod **VectorD**, fastK_Period **VectorD**, fastD_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Relative Strength Index Fast D
StochRsiFastd (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, fastK_Period **VectorD**, fastD_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Relative Strength Index Fast D
StochRsiFastk (returns **VectorD**)	(timePeriod **VectorD**, fastK_Period **VectorD**) Stochastic Relative Strength Index Fast K
StochRsiFastk (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, fastK_Period **VectorD**) Stochastic Relative Strength Index Fast K
StochFastd (returns **VectorD**)	(fastK_Period **VectorD**, fastD_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Fast D
StochFastd (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, fastK_Period **VectorD**, fastD_Period **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Stochastic Fast D
StochFastk (returns **VectorD**)	(fastK_Period **VectorD**) Stochastic Fast K
StochFastk (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, fastK_Period **VectorD**) Stochastic Fast K
T3 (returns **VectorD**)	(timePeriod **VectorD**, vFactor **VectorD**) Triple Exponential Moving Average (T3)

T3 (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, vFactor **VectorD**) Triple Exponential Moving Average (T3)
Tema (returns **VectorD**)	(timePeriod **VectorD**) Triple Exponential Moving Average
Tema (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Triple Exponential Moving Average
Trange (returns **VectorD**)	(parameterless) True Range
Trange (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**) True Range
Trima (returns **VectorD**)	(timePeriod **VectorD**) Triangular Moving Average
Trima (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Triangular Moving Average
Trix (returns **VectorD**)	(timePeriod **VectorD**) 1-day Rate-Of-Change (ROC) of a Triple Smooth EMA
Trix (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) 1-day Rate-Of-Change (ROC) of a Triple Smooth EMA
Tsf (returns **VectorD**)	(timePeriod **VectorD**) Time Series Forecast
Tsf (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Time Series Forecast
TypPrice (returns **VectorD**)	(timePeriod **VectorD**) Typical Price
TypPrice (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**) Typical Price
UltOsc (returns **VectorD**)	(timePeriod1 **VectorD**, timePeriod2 **VectorD**, timePeriod3 **VectorD**) Ultimate Oscillator

UltOsc (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod1 **VectorD**, timePeriod2 **VectorD**, timePeriod3 **VectorD**) Ultimate Oscillator
Var (returns **VectorD**)	(timePeriod **VectorD**) Variance
Var (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Variance
WclPrice (returns **VectorD**)	(parameterless) Weighted Close Price
WclPrice (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**) Weighted Close Price
Willr (returns **VectorD**)	(timePeriod **VectorD**) Williams' %R
Willr (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Williams' %R
Wma (returns **VectorD**)	(timePeriod **VectorD**) Weighted Moving Average
Wma (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Weighted Moving Average
Count (returns **VectorD**)	(parameterless) Returns the number of bars
DateTicks (returns **VectorD**)	(parameterless) Returns the number of ticks that represent the current date and time
TimeTicks (returns **VectorD**)	(parameterless) Returns the number of ticks that represent the current time
DayOfWeek (returns **VectorD**)	(parameterless) Returns the current bar's day of week (1 for Monday, 2 for Tuesday, 3 for Wednesday...)

DayOfYear (returns **VectorD**)	(parameterless) Returns the current bar's day of the year
Week (returns **VectorD**)	(parameterless) Returns the current bar's week
Day (returns **VectorD**)	(parameterless) Returns the current bar's day of the month
Month (returns **VectorD**)	(parameterless) Returns the current bar's month
Year (returns **VectorD**)	(parameterless) Returns the current bar's year
Hour (returns **VectorD**)	(parameterless) Returns the current bar's hour
Adxr (returns **VectorD**)	(timePeriod **VectorD**) Average Directional Movement Index Rating
Adxr (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Average Directional Movement Index Rating
Apo (returns **VectorD**)	(fastPeriod **VectorD**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Absolute Price Oscillator
Apo (returns **VectorD**)	(high **VectorD**, fastPeriod **VectorD**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Absolute Price Oscillator
AroonDown (returns **VectorD**)	(timePeriod **VectorD**) Aroon Down
AroonDown (returns **VectorD**)	(high **VectorD**, low **VectorD**, timePeriod **VectorD**) Aroon Down
AroonOsc (returns **VectorD**)	(timePeriod **VectorD**) Aroon Oscillator
AroonOsc (returns **VectorD**)	(high **VectorD**, low **VectorD**, timePeriod **VectorD**)

	Aroon Oscillator
AroonUp (returns **VectorD**)	(timePeriod **VectorD**) Aroon Up
AroonUp (returns **VectorD**)	(high **VectorD**, low **VectorD**, timePeriod **VectorD**) Aroon Up
Atr (returns **VectorD**)	(timePeriod **VectorD**) Average True Range
Atr (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Average True Range
AvgPrice (returns **VectorD**)	(parameterless) Average Price
AvgPrice (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Average Price
BbandsLower (returns **VectorD**)	(timePeriod **VectorD**, nbDevDown **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Bollinger Bands (Lower band)
BbandsLower (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, nbDevDown **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Bollinger Bands (Lower band)
BbandsMiddle (returns **VectorD**)	(timePeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Bollinger Bands (Middle band)
BbandsMiddle (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Bollinger Bands (Middle band)
BbandsUpper (returns **VectorD**)	(timePeriod **VectorD**, nbDevUp **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Bollinger Bands (Upper band)
BbandsUpper (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, nbDevUp **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Bollinger Bands (Upper band)

Bop (returns **VectorD**)	(parameterless) Balance Of Power
Bop (returns **VectorD**)	(open **VectorD**, high **VectorD**, low **VectorD**, close **VectorD**) Balance Of Power
Cci (returns **VectorD**)	(timePeriod **VectorD**) Commodity Channel Index
Cci (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Commodity Channel Index
Cmo (returns **VectorD**)	(timePeriod **VectorD**) Chande Momentum Oscillator
Cmo (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Chande Momentum Oscillator
Correl (returns **VectorD**)	(array **VectorD**, timePeriod **VectorD**) Correlation Analysis
Correl (returns **VectorD**)	(array **VectorD**, array **VectorD**, timePeriod **VectorD**) Correlation Analysis
Dema (returns **VectorD**)	(timePeriod **VectorD**) Double Exponential Moving Average
Dema (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Double Exponential Moving Average
Dx (returns **VectorD**)	(timePeriod **VectorD**) Directional Movement Index
Dx (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Directional Movement Index
Ema (returns **VectorD**)	(timePeriod **VectorD**) Exponential Moving Average
Ema (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**)

	Exponential Moving Average
Fama (returns **VectorD**)	(fastLimit **VectorD**, slowLimit **VectorD**) Following Adaptive Moving Average
Fama (returns **VectorD**)	(array **VectorD**, fastLimit **VectorD**, slowLimit **VectorD**) Following Adaptive Moving Average
Ht_Dcperiod (returns **VectorD**)	(parameterless) Hilbert Transform - Dominant Cycle Period
Ht_Dcperiod (returns **VectorD**)	(close **VectorD**) Hilbert Transform - Dominant Cycle Period
Ht_Dcphase (returns **VectorD**)	(parameterless) Hilbert Transform - Dominant Cycle Phase
Ht_Dcphase (returns **VectorD**)	(close **VectorD**) Hilbert Transform - Dominant Cycle Phase
Ht_Leadsine (returns **VectorD**)	(parameterless) Hilbert Transform - SineWave
Ht_Leadsine (returns **VectorD**)	(close **VectorD**) Hilbert Transform - SineWave
Ht_Phasorphase (returns **VectorD**)	(parameterless) Hilbert Transform - Instantaneous Trendline
Ht_Phasorphase (returns **VectorD**)	(close **VectorD**) Hilbert Transform - Instantaneous Trendline
Ht_Phasorquadrature (returns **VectorD**)	(parameterless) Hilbert Transform - Trend vs Cycle Mode
Ht_Phasorquadrature (returns **VectorD**)	(close **VectorD**) Hilbert Transform - Trend vs Cycle Mode
Ht_Sine (returns **VectorD**)	(parameterless) Hilbert Transform - SineWave
Ht_Sine (returns **VectorD**)	(close **VectorD**)

	Hilbert Transform - SineWave
Ht_Trendline (returns **VectorD**)	(parameterless) Hilbert Transform - Phasor Components
Ht_Trendline (returns **VectorD**)	(close **VectorD**) Hilbert Transform - Phasor Components
Ht_Trendmode (returns **VectorD**)	(parameterless) Hilbert Transform - Phasor Components
Ht_Trendmode (returns **VectorD**)	(close **VectorD**) Hilbert Transform - Phasor Components
Kama (returns **VectorD**)	(timePeriod **VectorD**) Kaufman Adaptive Moving Average
Kama (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Kaufman Adaptive Moving Average
LinearReg (returns **VectorD**)	(timePeriod **VectorD**) Linear Regression
LinearReg (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Linear Regression
LinearReg_Angle (returns **VectorD**)	(timePeriod **VectorD**) Linear Regression Angle
LinearReg_Angle (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Linear Regression Angle
LinearReg_Intercept (returns **VectorD**)	(timePeriod **VectorD**) Linear Regression Intercept
LinearReg_Intercept (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Linear Regression Intercept
LinearReg_Slope (returns **VectorD**)	(timePeriod **VectorD**) Linear Regression Slope
LinearReg_Slope (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**)

	Linear Regression Slope
Ma (returns **VectorD**)	(timePeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) Moving Average
Ma (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) All Moving Average
Macd (returns **VectorD**)	(parameterless) Moving Average Convergence/Divergence
Macd (returns **VectorD**)	(close **VectorD**) Moving Average Convergence/Divergence
MacdExt (returns **VectorD**)	(fastPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) MACD with controllable MA type
MacdExt (returns **VectorD**)	(close **VectorD**, fastPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) MACD with controllable MA type
MacdExtHist (returns **VectorD**)	(fastPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, signalPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) MACD Historigramme with controllable MA type
MacdExtHist (returns **VectorD**)	(close **VectorD**, fastPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, signalPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) MACD Historigramme with controllable MA type
MacdExtSignal (returns **VectorD**)	(fastPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, signalPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) MACD Signal with controllable MA type

MacdExtSignal (returns **VectorD**)	(close **VectorD**, fastPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, slowPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**, signalPeriod **VectorD**, System.Collections.Generic.List`1[System.String] **String**) MACD Signal with controllable MA type
MacdHist (returns **VectorD**)	(signalPeriod **VectorD**) MACD Historigramme
MacdHist (returns **VectorD**)	(close **VectorD**, signalPeriod **VectorD**) MACD Historigramme
MacdSignal (returns **VectorD**)	(signalPeriod **VectorD**) MACD Signal
MacdSignal (returns **VectorD**)	(close **VectorD**, signalPeriod **VectorD**) MACD Signal
Mama (returns **VectorD**)	(fastLimit **VectorD**, slowLimit **VectorD**) MESA Adaptive Moving Average
Mama (returns **VectorD**)	(array **VectorD**, fastLimit **VectorD**, slowLimit **VectorD**) MESA Adaptive Moving Average
Hhv (returns **VectorD**)	(timePeriod **VectorD**) Highest value over a specified period
Hhv (returns **VectorD**)	(close **VectorD**, timePeriod **VectorD**) Highest value over a specified period
MedPrice (returns **VectorD**)	(parameterless) Median Price
MedPrice (returns **VectorD**)	(high **VectorD**, low **VectorD**) Median Price
Mfi (returns **VectorD**)	(timePeriod **VectorD**) Money Flow Index
Mfi (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, volume **VectorD**, timePeriod **VectorD**) Money Flow Index

MidPoint (returns **VectorD**)	(timePeriod **VectorD**) MidPoint over period
FFT (returns **VectorCustom**)	(parameterless) Performs Fast Fourier Transform
Date (returns **VectorS**)	(parameterless) Returns the bar's Date
Now (returns **VectorS**)	(parameterless) Returns the current Date
StringExtract (returns **VectorS**)	(string **VectorS**, value **VectorD**, value1 **VectorD**) Extract a string from the string in the first parameter, starting at the specified number in the second parameter and ending at the specified number in the third parameter
StringExtractStart (returns **VectorS**)	(string **VectorS**, value **VectorD**) Extract a string from the string in the first parameter, starting at 0 and ending at the specified number in the second parameter
StringExtractEnds (returns **VectorS**)	(string **VectorS**, value **VectorD**) Extract a string from the string in the first parameter, starting from (Length of the first parameter string - number in the second parameter) and ending at the length of the first parameter string
StringReplace (returns **VectorS**)	(string **VectorS**, oldValue **VectorS**, newValue **VectorS**) Replace in the first parameter string, oldValue with newValue
StringInsert (returns **VectorS**)	(string **VectorS**, ToInsert **VectorS**, index **VectorD**) Insert in the first parameter the second parameter string at the specified index number
IffStr (returns **VectorS**)	(**VectorD**, **VectorS**, **VectorS**) A conditional function that returns the value of the first string parameter if condition is true (different from 0), if condition is false then it returns the value of the second string parameter.
Market (returns **VectorS**)	(parameterless) Returns the symbol's Market
Industry (returns **VectorS**)	(parameterless)

	Returns the symbol's Industry
Sector (returns **VectorS**)	(parameterless) Returns the symbol's Sector
Group (returns **VectorS**)	(parameterless) Returns the symbol's Group
Index (returns **VectorS**)	(parameterless) Returns the symbol's Index
Name (returns **VectorS**)	(parameterless) Returns the symbol's Name
FullName (returns **VectorS**)	(parameterless) Returns the symbol's FullName
Website (returns **VectorS**)	(parameterless) Returns the symbol's Website
Address (returns **VectorS**)	(parameterless) Returns the symbol's Address
Country (returns **VectorS**)	(parameterless) Returns the symbol's Country
Currency (returns **VectorS**)	(parameterless) Returns the symbol's Currency
Name1 (returns **VectorS**)	(parameterless) Returns the symbol's Name1
Name2 (returns **VectorS**)	(parameterless) Returns the symbol's Name2
Name3 (returns **VectorS**)	(parameterless) Returns the symbol's Name3
States (returns **VectorD**)	(array **VectorD**, states (ex : 70|50|30) **String**) Returns an array containing different states.|Example: States(perf(close, 20), 10|0|-10) | Returns 0 if the array element value is lower then -10, returns 1 if the array element value is between -10 and 0 and returns 2 if the array element value is higher than 10

GapDown (returns **VectorD**)	(parameterless) Returns 1, when a down gap happen
GapUp (returns **VectorD**)	(parameterless) Returns 1, when an up gap happen
Inside (returns **VectorD**)	(parameterless) Returns 1, when an inside day occurs
Outside (returns **VectorD**)	(parameterless) Returns 1, when an outside day occurs
HhvLb (returns **VectorD**)	(array **VectorD**) Returns the number of bars since the array reached its highest value
HhvLb (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the number of bars since the array reached its highest value over the specified period
LlvLb (returns **VectorD**)	(array **VectorD**) Returns the number of bars since the array reached its lowest value
LlvLb (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the number of bars since the array reached its lowest value over the specified period
ZigZag (returns **VectorD**)	(array **VectorD**, change **VectorD**) Calculates the Zig Zag indicator given the percent change
Peak (returns **VectorD**)	(array **VectorD**, change **VectorD**) Returns the value of the array in the last peak
Peak (returns **VectorD**)	(array **VectorD**, change **VectorD**, N **VectorD**) Returns the value of the array N peak(s) ago
Trough (returns **VectorD**)	(array **VectorD**, change **VectorD**) Returns the value of the array in the last trough
Trough (returns **VectorD**)	(array **VectorD**, change **VectorD**, N **VectorD**) Returns the value of the array N trough(s) ago

PeakBars (returns **VectorD**)	(array **VectorD**, change **VectorD**) Returns the number of bars since the last peak
PeakBars (returns **VectorD**)	(array **VectorD**, change **VectorD**, N **VectorD**) Returns the number of bars since N peak(s) ago
TroughBars (returns **VectorD**)	(array **VectorD**, change **VectorD**) Returns the number of bars since the last trough
TroughBars (returns **VectorD**)	(array **VectorD**, change **VectorD**, N **VectorD**) Returns the number of bars since N trough(s) ago
BarsSince (returns **VectorD**)	(array **VectorD**) Calculates the number of bars that have passed since the array was different from 0
Cross (returns **VectorD**)	(**VectorD**, **VectorD**) Returns '1', when the first array crosses above the second array, otherwise the result is '0'
RemoveSameSignals (returns **VectorD**)	(array **VectorD**, array **VectorD**) Removes excessive signals.|Returns 1 on the first occurence of 'true' signal in the first array.|then Returns 0 until the second array is 'true' even if there are 'true' signals in the first array.
RemoveSameSignals (returns **VectorD**)	(array **VectorD**) Removes excessive signals.|Returns 1 on the first occurence of 'true' signal in the first array.|then Returns 0 until the array become 'false' then 'true'.
ValueWhen (returns **VectorD**)	(expression **VectorD**, array **VectorD**, period **VectorD**) Returns the value of the array, when the expression is TRUE, of the n -th most recent occurrence
ValueWhen (returns **VectorD**)	(expression **VectorD**, array **VectorD**) Returns the value of the array, when the expression is TRUE, of the last occurrence
Perf (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the performance of the array for the specified period (k-step Return)
PerfD (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the average daily performance of the array for the specified period (k-step Trend)

GFun (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the gaussian for the specified array
RFun (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the relative volatility of the specified array
Random (returns **VectorD**)	(min **VectorD**, max **VectorD**) Returns a random number between min and max
Sharpe (returns **VectorD**)	(array **VectorD**, period **VectorD**) Returns the sharpe ratio for the specified array and period
Iff (returns **VectorD**)	(**VectorD**, **VectorD**, **VectorD**) A conditional function that returns the value of the first parameter if condition is true (different from 0), if condition is false then it returns the value of the second parameter.
SumIf (returns **VectorD**)	(condition **VectorD**, array **VectorD**) Calculates, when condition is TRUE, a cumulative sum of the ARRAY for all the lookback periods (current bar is included)
SumIf (returns **VectorD**)	(condition **VectorD**, array **VectorD**, period **VectorD**) Calculates, when condition is TRUE, a cumulative sum of the ARRAY for the specified number of lookback periods (current bar is included)
Avg (returns **VectorD**)	(array **VectorD**) Calculates the average of the ARRAY for all the lookback periods (current bar is included)
Avg (returns **VectorD**)	(array **VectorD**, period **VectorD**) Calculates the average of the ARRAY for the specified number of lookback periods (current bar is included)
AvgIf (returns **VectorD**)	(condition **VectorD**, array **VectorD**) Calculates, when condition is TRUE, the average of the ARRAY for all the lookback periods (current bar is included)
AvgIf (returns **VectorD**)	(condition **VectorD**, array **VectorD**, period **VectorD**) Calculates, when condition is TRUE, the average of the ARRAY for the specified number of lookback periods (current bar is included) (When the condition is true)
Sum (returns **VectorD**)	(array **VectorD**) Calculates a cumulative sum of the ARRAY for all the lookback periods (current bar is included)

Sum (returns **VectorD**)	(array **VectorD**, period **VectorD**) Calculates a cumulative sum of the ARRAY for the specified number of lookback periods (current bar is included)
Min (returns **VectorD**)	(array **VectorD**, array **VectorD**) For each bar of the ARRAY, returns the smallest one
Max (returns **VectorD**)	(array **VectorD**, array **VectorD**) For each bar of the ARRAY, returns the largest one
LastValue (returns **VectorD**)	(array **VectorD**) Returns the last value of the ARRAY
LastNotNullValue (returns **VectorD**)	(array **VectorD**) Returns the last (not equal to 0) value of the ARRAY
Ref (returns **VectorD**)	(array **VectorD**, period **VectorD**) References a previous or subsequent element in a ARRAY. A negative period references X periods in the future; a positive period references X periods ago (Replace empty values in the ARRAY with NA)
RefZero (returns **VectorD**)	(array **VectorD**, period **VectorD**) References a previous or subsequent element in a ARRAY. A negative period references X periods in the future; a positive period references X periods ago (Replace empty values in the ARRAY with 0)
Value (returns **VectorD**)	(array **VectorD**, bar number **VectorD**) Returns the value of the specified ARRAY for the bar number X
StringContains (returns **VectorD**)	(string **VectorS**, string **VectorS**) Returns 1 if the first string parameter contains the second string parameter, 0 otherwise
StringLength (returns **VectorD**)	(string **VectorS**) Returns the length of the string
StringEqual (returns **VectorD**)	(string **VectorS**, string **VectorS**) Returns 1 if the two strings are equal, 0 otherwise
Ad (returns **VectorD**)	(parameterless) Chaikin A/D Line

Ad (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, volume **VectorD**) Chaikin A/D Line
AdOsc (returns **VectorD**)	(fastPeriod **VectorD**, slowPeriod **VectorD**) Chaikin A/D Oscillator
AdOsc (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, volume **VectorD**, fastPeriod **VectorD**, slowPeriod **VectorD**) Chaikin A/D Oscillator
Adx (returns **VectorD**)	(timePeriod **VectorD**) Average Directional Movement Index
Adx (returns **VectorD**)	(high **VectorD**, low **VectorD**, close **VectorD**, timePeriod **VectorD**) Average Directional Movement Index

[TOP]

1.5.2.7 Symbol

QuantShare API Documentation

Symbol Members

Parent

Properties

Name (of type **String**)	Security name
Name1 (of type **String**)	Security name1
Name2 (of type **String**)	Security name2
Name3 (of type **String**)	Security name3
FullName (of type **String**)	Security full name
Group (of type **String**)	Security group
Market (of type **String**)	Security market
Industry (of type **String**)	Security industry
Sector (of type **String**)	Security sector
Description (of type **String**)	Security description
Country (of type **String**)	Security country
Address (of type **String**)	Company address
Website (of type **String**)	Company website
IsActive (of type **Boolean**)	Specify whether the current security is active or not
IsTradable (of type **Boolean**)	Specify whether the current security is tradable or not

Currency (of type **String**)	Security currency
TickSize (of type **Double**)	Security Tick Size
RoundLotSize (of type **Double**)	Security Round Lot Size
PointValue (of type **Double**)	Security Point Value
MarginDeposit (of type **Double**)	Security Margin Deposit
Multiplier (of type **Int32**)	Security Multiplier
Strike (of type **Double**)	Security strike price
ExpirationDate (of type **String**)	Security expiration date

Methods

GetIndexesFromList (returns **String[]**)	(parameterless) Get a list of all the indexes where the current security belongs to
AddIndex (returns **Void**)	(Index **String**) Add the current symbol to an index
RemoveIndex (returns **Void**)	(Index **String**) Remove the current symbol from an index

[TOP]

1.5.2.8 VectorCustom

QuantShare API Documentation

VectorCustom Members

Parent

Properties

[Int32] (of type **CustomDatabaseValues**)	Get the elements at the specified bar index
Length (of type **Int32**)	Get the vector length

Methods

GetValues (returns **CustomDatabaseValues**)	(index **Int32**) Get the elements at the specified bar index

[TOP]

1.5.2.9 VectorCustomDouble

QuantShare API Documentation

VectorCustomDouble Members

Parent

Properties

[Int32] (of type **CustomDatabaseValuesDouble**)	Get the elements at the specified bar index
Length (of type **Int32**)	Get the vector length

Methods

GetValues (returns **CustomDatabaseValuesDouble**)	(index **Int32**) Get the elements at the specified bar index
ToVectorD (returns **VectorD**)	(parameterless) Convert this object to a VectorD object

[TOP]

1.5.2.10 VectorD

QuantShare API Documentation

VectorD Members

Parent

Properties

Length (of type **Int32**)	Get the vector length
BackwardBars (of type **Int32**)	Get the number of previous bars required to calculate the value of a particular bar
ForwardBars (of type **Int32**)	Get the number of future bars required to calculate the value of a particular bar

Methods

GetValue (returns **Double**)	(index **Int32**) Get an element value from the vector
SetValue (returns **Void**)	(index **Int32**, value **Double**) Assign a value to an element of the vector
Assign (returns **Void**)	(value **Double**) Assign a value to all the elements of the vector
Assign (returns **Void**)	(value **VectorD**) Assign a vector to the current vector
StoreData (returns **Void**)	(name **String**, value **Object**) Store a value
RetrieveData (returns **Object**)	(name **String**) Retrieve a value stored using the 'StoreData' function

[TOP]

1.5.2.11 VectorDate

QuantShare API Documentation

VectorDate Members

Parent

Properties

Length (of type **Int32**)	Get the vector length

Methods

SetValue (returns **Void**)	(index **Int32**, value **DateTime**) Assign a value to an element of the vector
GetValue (returns **DateTime**)	(index **Int32**) Get an element value from the vector

[TOP]

1.5.2.12 VectorR

QuantShare API Documentation

VectorR Members

Parent

Properties

[Int32] (of type **Object**)	Get an element value from the vector
Length (of type **Int32**)	Get the vector length

Methods

GetValue (returns **Object**)	(index **Int32**) Get an element value from the vector
GetDate (returns **DateTime**)	(index **Int32**) Get a date from the vector

[TOP]

1.5.2.13 VectorS

QuantShare API Documentation

VectorS Members

Parent

Properties

Length (of type **Int32**)	Get the vector length

Methods

SetValue (returns **Void**)	(index **Int32**, value **String**) Assign a value to an element of the vector
GetValue (returns **String**)	(index **Int32**) Get an element value from the vector
Assign (returns **Void**)	(value **String**) Assign a value to all the elements of the vector
Assign (returns **Void**)	(value **VectorS**) Assign a vector to the current vector

[TOP]

1.6 Parser Post-Script

1.6.1 Downloader Post-Script

QuantShare API Documentation

Parser Post-Script Members

Script that is executed after parsing the downloaded data.

Fields

Data (of type **PostScript**)	Contains the data that was parsed

[TOP]

1.6.2 Members

1.6.2.1 PostScript

QuantShare API Documentation

PostScript Members

Contains the data that was parsed

Parent

Properties

Rows (of type **PostScriptRow[]**)	Get the content rows
SymbolInfo (of type **Symbol**)	Symbol information

[TOP]

1.6.2.2 PostScriptRow

QuantShare API Documentation

PostScriptRow Members

Parent

Properties

IsIgnoreLine (of type **Boolean**)	Specify whether to ignore the current row or not
Data (of type **Object[]**)	Get the elements of the current row

[TOP]

1.6.2.3 Symbol

QuantShare API Documentation

Symbol Members

Parent

Properties

Name (of type **String**)	Security name
Name1 (of type **String**)	Security name1
Name2 (of type **String**)	Security name2
Name3 (of type **String**)	Security name3
FullName (of type **String**)	Security full name
Group (of type **String**)	Security group
Market (of type **String**)	Security market
Industry (of type **String**)	Security industry
Sector (of type **String**)	Security sector
Description (of type **String**)	Security description
Country (of type **String**)	Security country
Address (of type **String**)	Company address
Website (of type **String**)	Company website
IsActive (of type **Boolean**)	Specify whether the current security is active or not
IsTradable (of type **Boolean**)	Specify whether the current security is tradable or not

Currency (of type **String**)	Security currency
TickSize (of type **Double**)	Security Tick Size
RoundLotSize (of type **Double**)	Security Round Lot Size
PointValue (of type **Double**)	Security Point Value
MarginDeposit (of type **Double**)	Security Margin Deposit
Multiplier (of type **Int32**)	Security Multiplier
Strike (of type **Double**)	Security strike price
ExpirationDate (of type **String**)	Security expiration date

Methods

GetIndexesFromList (returns **String[]**)	(parameterless) Get a list of all the indexes where the current security belongs to
AddIndex (returns **Void**)	(Index **String**) Add the current symbol to an index
RemoveIndex (returns **Void**)	(Index **String**) Remove the current symbol from an index

[TOP]

1.7 Parser Pre-Script

1.7.1 Downloader Pre-Script

QuantShare API Documentation

Parser Pre-Script Members

Script that is executed before parsing the downloaded data.

Fields

Content (of type **PreScript**)	Contains the content to parse

[TOP]

1.7.2 Members

1.7.2.1 PreScript

QuantShare API Documentation

PreScript Members

Contains the content to parse

Parent

Properties

Rows (of type **PreScriptRow[]**)	Get the content rows

Methods

GetURLORFileName (returns **String**)	(parameterless) Get the URL or the file path that was used to get the current content
GetSymbolNameFromNameX (returns **String**)	(0 for name1... **Int32**, symbol nameX **String**) Get symbol name from name1, name2 or name3
GetDatabaseName (returns **String**)	(index **Int32**) Get the database name for the specified column index
GetFieldName (returns **String**)	(index **Int32**) Get the field name for the specified column index
SetDatabaseAndField (returns **Boolean**)	(index **Int32**, database name **String**, field name **String**, database descriptive name **String**, field descriptive name **String**, default value **String**, Is historical database **Boolean**, Is numeric field **Boolean**, Is get snapshot **Boolean**) Updates the Database and field names for the specified column index
GetNumericValue (returns **Double**)	(Database name **String**, Field name **String**, Symbol **String**, Date **DateTime**) Returns the last value from a database/field for the specified symbol and the specified date (Value should occur before the specified date)
AddRow (returns **Void**)	(Elements to add **String[]**)

	Add a new row
GetContent (returns **String**)	(parameterless) Get the downloaded content
GetContentLines (returns **String[]**)	(parameterless) Get the downloaded content

[TOP]

1.7.2.2 PreScriptRow

QuantShare API Documentation

PreScriptRow Members

Parent

Properties

IsIgnoreLine (of type **Boolean**)	Specify whether to ignore the current line or not
Data (of type **String[]**)	Get the elements of the current line

[TOP]

1.8 Prediction Model Metrics

1.8.1 Prediction Model Metrics

QuantShare API Documentation

Prediction Model Metrics Members

Fields

Cycle (of type **Int32**)	The current cycle number
NetworkError (of type **Double**)	The network error
StrategyMetric (of type **Double**)	The strategy metric (The default metric is the Output Per Bar)
LearningErrors (of type **PredictionErrors**)	Metrics for the learning samples
ValidationErrors (of type **PredictionErrors**)	Metrics for the validation samples
TestErrors (of type **PredictionErrors**)	Metrics for the test samples
OptimizerMetrics (of type **OptimizerMetricsClass**)	Optimizer Metrics - Should be used only when working with the AI Optimizer
Fitness (of type **Double**)	The fitness value

[TOP]

1.8.2 Members

1.8.2.1 OptimizerMetricsClass

QuantShare API Documentation

OptimizerMetricsClass Members

Optimizer Metrics - Should be used only when working with the AI Optimizer

Parent

Properties

Generation (of type **Double**)	Current generation number
AvgFitness (of type **Double**)	Average fitness for the current generation
HighestFitness (of type **Double**)	Highest fitness for the current generation
LowestFitness (of type **Double**)	Lowest fitness for the current generation

[TOP]

1.8.2.2 PredictionErrors

PredictionErrors Members

Metrics for the test samples

Parent

Properties

MSE (of type **Double**)	Mean squared error
POCD (of type **Double**)	Prediction of correct direction
POCID (of type **Double**)	Prediction of change in direction
UTHEIL (of type **Double**)	Theil's U statistic assesses predictive accuracy relative to a naive no-change model. It is unitary when the MSFE (mean absolute forecast error) equals the mean square error of naive no-change forecasts, and it is greater than 1.0 if predictions are less accurate than no-change forecasts.

[TOP]

1.9 Simulation Metrics

1.9.1 Simulation Metrics

QuantShare API Documentation

Simulation Metrics Members

Fields

InitialCapital (of type **Double**)	Initial Capital
EndCapital (of type **Double**)	End Capital
NetProfit (of type **Double**)	Net Profit
NetProfitInPercentage (of type **Double**)	Net Profit In Percentage
Exposure (of type **Double**)	Exposure
NetRiskAdjustedReturn (of type **Double**)	Net Risk Adjusted Return
AnnualReturn (of type **Double**)	Annual Return
RiskAdjustedReturn (of type **Double**)	Risk Adjusted Return
AverageProfitLoss (of type **Double**)	Average ProfitLoss
AverageProfitLossInPercentage (of type **Double**)	Average ProfitLoss In Percentage
AverageBarsHeld (of type **Double**)	Average Bars Held
TotalProfit (of type **Double**)	Total Profit
NumberOfWinners (of type **Double**)	Number Of Winners
AverageProfit (of type **Double**)	Average Profit
AverageProfitInPercentage (of type **Double**)	Average Profit In Percentage

AverageBarsHeldForWinners (of type **Double**)	Average Bars Held For Winners
MaxConsecutiveWinner (of type **Double**)	Max Consecutive Winner
LargestWinner (of type **Double**)	Largest Winner
NumberOfBarsInLargestWinner (of type **Double**)	Number Of Bars In Largest Winner
TotalLoss (of type **Double**)	Total Loss
NumberOfLosers (of type **Double**)	Number Of Losers
AverageLoss (of type **Double**)	Average Loss
AverageLossInPercentage (of type **Double**)	Average Loss In Percentage
AverageBarsHeldForLosers (of type **Double**)	Average Bars Held For Losers
MaxConsecutiveLoser (of type **Double**)	Max Consecutive Loser
LargestLoser (of type **Double**)	Largest Loser
NumberOfBarsInLargestLoser (of type **Double**)	Number Of Bars In Largest Loser
MaximumTradeDrawdown (of type **Double**)	Maximum Trade Drawdown
MaximumTradeDrawdownInpercentage (of type **Double**)	Maximum Trade Drawdown In percentage
MaximumSystemDrawdown (of type **Double**)	Maximum System Drawdown
MaximumSystemDrawdownInpercentage (of type **Double**)	Maximum System Drawdown In percentage
RecoveryFactor (of type **Double**)	Recovery Factor
CarMaxDD (of type **Double**)	Car MaxDD
RarMaxDD (of type **Double**)	Rar MaxDD
NumberOfTrades (of type **Double**)	Number Of Trades

PercentOfWinners (of type **Double**)	Percent Of Winners
AnnualTurnover (of type **Double**)	Annual Turnover
TotalTradingCosts (of type **Double**)	Total Trading Costs
VolumeActivity (of type **Double**)	Volume Activity
PercentTradingCostOfVolume (of type **Double**)	Percent Trading Cost Of Volume
PercentPositiveDays (of type **Double**)	Percent Positive Days
PercentPositiveWeeks (of type **Double**)	Percent Positive Weeks
PercentPositiveMonths (of type **Double**)	Percent Positive Months
PercentPositiveYears (of type **Double**)	Percent Positive Years
AverageDailyReturn (of type **Double**)	Average Daily Return
AverageWeeklyReturn (of type **Double**)	Average Weekly Return
AverageMonthlyReturn (of type **Double**)	Average Monthly Return
UlcerIndex (of type **Double**)	Ulcer Index
UlcerPerformanceIndex (of type **Double**)	Ulcer Performance Index
KRatio (of type **Double**)	K Ratio
StandardDeviation (of type **Double**)	Standard Deviation
DownsideStandardDeviation (of type **Double**)	Downside Standard Deviation
SharpeRatio (of type **Double**)	Sharpe Ratio
SortinoRatio (of type **Double**)	Sortino Ratio
ProfitFactor (of type **Double**)	Profit Factor

PayoffRatio (of type **Double**)	Payoff Ratio
Beta (of type **Double**)	Beta
Alpha (of type **Double**)	Alpha
RSquared (of type **Double**)	R Squared
Correlation (of type **Double**)	Correlation
Fitness (of type **Double**)	Fitness value
DailyReturn (of type **SeriesStats**)	Return per Day
WeeklyReturn (of type **SeriesStats**)	Return per Week
MonthlyReturn (of type **SeriesStats**)	Return per Month
YearlyReturn (of type **SeriesStats**)	Return per Year
OptimizerMetrics (of type **OptimizerMetricsClass**)	Optimizer Metrics - Should be used only when working with the AI Optimizer

[TOP]

1.9.2 Members

1.9.2.1 OptimizerMetricsClass

QuantShare API Documentation

OptimizerMetricsClass Members

Optimizer Metrics - Should be used only when working with the AI Optimizer

Parent

Properties

Generation (of type **Double**)	Current generation number
AvgFitness (of type **Double**)	Average fitness for the current generation
HighestFitness (of type **Double**)	Highest fitness for the current generation
LowestFitness (of type **Double**)	Lowest fitness for the current generation

[TOP]

1.9.2.2 SeriesStats

SeriesStats Members

Return per Year

Parent

Properties

Count (of type **Int32**)	Total number of values
[Int32] (of type **Double**)	Get a value at the specified index

Methods

GetDate (returns **DateTime**)	(index **Int32**) Get the date at the specified index

[TOP]