FUNCIONES Y PROPIEDADES **VBA**

MANUALES PARA EL USUARIO

VisualChart
Real Time Financial Information & Trading Software

Funciones y Propiedades VBA

Alert
Angle
AvgBarsInTrade
AvgLosingTrade
AvgTrade
AvgWinningTrade
BestSeries
Buy
CalcDate
CalcTime
Close
ConfigStk
CurrentBar
CurrentContract
CurrentEntries
Date
DateSubstract
ExitLong
ExitShort
FeedFields
FilledOrders
GetBackgroundColor
GetBarColor
GetBarsSinceEntry
GetBarsSinceExit
GetBarStyle
GetBarWidth
GetConfigStk
GetDailyLosers
GetDailyWinners
GetEntryDate
GetEntryPrice
GetEntryTime
GetExitDate
GetExitOrder
GetExitPrice
GetExitTime
GetHighest
GetHighestBar
GetHistogramBand
GetIndicatorIdentifier - GII
GetIndicatorPos
GetIndicatorValue - GIV
GetLineName
GetLowest
GetLowestBar
GetMarketPosition
GetMaxContracts
GetMaxEntries
GetNthHighest
GetNthLowest
GetOrderCount
GetOrderDate
GetOrderLabel
GetOrderPrice
GetOrderSide
GetOrderSymbolCode
GetOrderType
GetOrderVolume
GetPivoDown
GetPivotUp
GetPositionProfit
GetPrice
GetStkLength
GetStkValue
GetStkValues
GetSymbolIdentifier - GSI
GetSymbolInfo
GetSymbolInfoEx
GetSystemIdentifier - GSYSI
GetSwingHigh
GetSwingHighBar
GetSwingLow
GetSwingLowBar
GetTrueHigh
GetTrueLow
GetTrueRange
GetTrueRangeCustom
GetVolatility
GetWndBackGroundColor
GrossLoss
GrossProfit
High
LargestLosingTrade
LargestWinningTrade
LC_Index
LimitOrder
LimitPrice
LimitVol
Low
MinutesToTime
NetProfit
NumberOfLines
NumberOfLosingTrades
NumberOfTrades
NumberOfWinningTrades
Open
OpenDay
OpenInt
PaintBar
PaintCandlestick
PaintMaxMin
PaintSeries
PercentProfitable
ProfitFactor
PRR
RegressionAngle
RegressionSlope
ReleaseDataIdentifier-RDI
Sell
SetBackgroundColor
SetBarColor
SetBarProperties
SetBarRepresentation
SetBarStyle
SetBarWidth
SetHistogramBand
SetIndicatorPos
SetIndicatorValue
SetLineName
SetWndBackgroundColor
ShouldTerminate
Slope
StandardDeviation
StarBar
Time
TimeEx
TimeToMinutes
This
Volume
WorstSeries

Alert

Descripción:
Esta función se utiliza en la programación de indicadores para emitir alertas. Al cumplirse unas determinadas condiciones, se mostrará un aviso en pantalla con el mensaje deseado.

Sintaxis:
.Alert(Description)

Parámetros:

Nombre	Defecto	Descripción
Description	"Indicator Alert"	Texto que se mostrará cuando se dispare la alerta.

Para que se visualice el mensaje de alerta en pantalla, es preciso que una vez que se haya insertado el indicador, se active la propiedad **Alertas Indicador** en el editor de propiedades de este.

Ejemplo:

.Alert("Atención, Cruce de Medias")

Mostrará en pantalla el mensaje "Atención, Cruce de Medias"

Angle

Descripción:
Devuelve el valor del ángulo que forma con la horizontal, la recta de regresión formada por los precios StartPrice y EndPrice, que relaciona las cotizaciones con la variable de tiempo.

Sintaxis:
.Angle(StartBar, EndBar, StartPrice, EndPrice, Identifier)

Parámetros:

Nombre	Defecto	Descripción
StarBar	-	Número de barra de inicio de la recta.
EndBar	-	Número de barra de fin de la recta.
StarPrice	-	Precio de inicio de la recta.
EndPrice	-	Precio de fin de la recta.
Identifier	-	Fuente de datos primaria. El sistema actúa sobre Data que es el nombre que toma la serie de datos del gráfico sobre el que se aplica la estrategia. Si en la ventana hay mas gráficos insertados se codificarán como Data2, Data3, Data4, etc.

Ejemplo:

Supongamos que deseamos conocer, para cada momento, el valor en radianes del ángulo entre el precio de cierre actual y el del cierre de hace 30 barras. En primer lugar tendríamos que definir la variable de salida.

Dim AnguloEnRadianes As Double

A esta variable se le asignará el valor que devuelva la llamada a esta propiedad:

AnguloEnRadianes = .Angle(Bar-30,Bar,.Close(30),.Close(0))

AvgBarsInTrade

Descripción:
Devuelve el promedio de número de barras existentes durante el periodo de vida de los negocios. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.AvgBarsInTrade

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

Supongamos que deseamos saber la media de barras por negocio. En primer lugar, se define una variable de salida:

Dim MediaBarsNeg as Double

Se calcula la media de barras por negocio cada vez que se genere uno nuevo, haciendo la siguiente pregunta:

If .NumberOfTrades > 1 then

Con esto nos aseguramos de que al menos se han generado dos negocios

MediaBarsNeg = .AvgBarsInTrade

Se asignará a la variable de salida el valor que devuelve la propiedad

Luego se puede almacenar en otra variable previamente definida, el número de negocios que llevamos:

NumNegActuales = .NumberOfTrades

Volveríamos a calcular la media de barras por negocio solo cuando .NumberOfTrades > NumNegActuales

AvgLosingTrade

Descripción:
Devuelve el promedio de resultados de los negocios perdedores. Este valor irá cambiando a medida que se generen nuevas barras y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.AvgLosingTrade(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje)

Ejemplo:

GananciaMediaPerdedores = .AvgLosingTrade(ByPoints)
Asigna a la variable GananciaMediaPerdedores (previamente definida) la ganancia media de todos los negocios perdedores (en puntos).

AvgTrade

Descripción:
Devuelve el promedio de resultados de los negocios. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.AvgTrade(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje)

Ejemplo:

GananciaMedia = .AvgTrade(Porcentual)

Asigna a la variable GananciaMedia (previamente definida) la ganancia media de todos los negocios (en %).

AvgWinningTrade

Descripción:
Devuelve el promedio de resultados de los negocios ganadores. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a esta propiedad.

Sintaxis:
.AvgWinningTrade(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje)

Ejemplo:

GananciaMediaGanadores = .AvgWinnigTrade(ByPoints)

Asigna a la variable GananciaMediaGanadores (previamente definida) la ganancia media de todos los negocios positivos (en puntos).

BestSeries

Descripción:
Devuelve el mejor valor de ganancia total que se ha llegado a alcanzar. Este irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a esta propiedad.

Sintaxis:
.BestSeries(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje)

Ejemplo:

.BestSeries(Porcentual)

Retornará en %, el mejor valor de ganancia total que se ha alcanzado hasta el momento de aplicar esta propiedad.

Buy

Descripción:
Esta función es utilizada para dar órdenes de compra. Es usa indistintamente para comprar acciones, futuros, etc.
Sintaxis:
.Buy Type, Contracts, Price, Label

Parámetros:

Nombre	Defecto	Descripción
Type	AtClose	Tipo de orden que se desea lanzar (AtClose, AtMarket, AtLimit, AtStop).
Contracts	1	Número de contratos/acciones. Las especificaciones numéricas sobre contratos pueden ser sustituidas por variables o por cualquier función definida previamente.
Price	-	Precio de compra. Sólo se indica este parámetro en las órdenes del tipo AtStop y AtLimit. El valor puede expresarse mediante un número, una variable, una función, o bien una mezcla de valor y función.
Label	-	Etiqueta de la orden en formato texto.

Ejemplo:

.Buy AtStop, 1, .High +10, "C1"

Lanza una orden de compra en stop de un contrato, siendo el precio del stop el máximo de la barra más 10 puntos, y la etiqueta identificativa de la orden "C1".

CalcDate

Descripción:
Suma una cantidad de días a un día en cuestión y devuelve la fecha resultante en **formato militar (AAAAMMDD).**

Sintaxis:
.CalcDate(Date, Days)

Parámetros:

Nombre	Defecto	Descripción
Date	-	Fecha (AAAAMMDD) a la cual se le sumará una cantidad de días.
Days	-	Cantidad de días a sumar.

Ejemplo:

.CalcDate(2010110,5)

Suma 5 días a la fecha 10/01/2010, que en formato militar es 20100110, por lo tanto retorna 20100115, que en formato de fecha es 15/01/2010.

CalcTime

Descripción:
Suma una cantidad de minutos a una hora en cuestión, y devuelve la hora resultante en formato militar (HHMM).

Sintaxis:
.CalcTime(Time, Minutes)

Parámetros:

Nombre	Defecto	Descripción
Time	-	Hora (HHMM) a la cual se le sumará una cantidad de minutos.
Minutes	-	Cantidad de minutos a sumar.

Ejemplo:

.CalcTime(1000,30)

Suma 30 minutos a las 10:00 am (formato militar 1000), por lo tanto retorna 1030, que en formato de hora es 10:30 am.

Close

Descripción:
Esta función devuelve el valor de cierre de una barra.

Sintaxis:
.Close(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. El valor por defecto hace referencia a la barra actual. En este parámetro es posible indicar un valor numérico contenido en una variable o teclear directamente el valor. Puede especificarse como función sustituyendo el valor numérico. Este parámetro sólo permite valores positivos.
Identifier	Data	Fuente de datos sobre la que se aplica la función. El sistema actúa sobre Data, que es el nombre que toma la serie de datos del gráfico sobre el que se aplica la estrategia. Si en la ventana hay mas gráficos insertados se codificarán como Data1, Data2, Data3, etc.

Ejemplo:

.Close(3,Data1)

Devuelve el cierre de hace 3 barras de la fuente de datos codificada como Data1.

ConfigStk

Descripción:
Permite establecer las propiedades iniciales de las estadísticas que se quieran extraer.

Sintaxis:
.ConfigStk(Sing, Unit, Filt, CompType, Compression, BeginDate, EndDate)

Parámetros:

Nombre	Defecto	Descripción
Sing	ssNets	Filtro de resultados según sean los negocios ganadores (ssWins) o perdedores (ssLoss).
Unit	suMoney	Representación de los datos en moneda (suMoney), en puntos (suPnts) o en porcentaje (ssPorc).
Filt	sfAll	Filtro de resultados por posición, es decir, según sean largos (sfLong) o cortos (sfShort).
CompType	sctTrades	Representación de los datos agrupados por negocios (sctTraders), días (sctDays), semanas (sctWeeks), meses (sctMoths) o años(sctYears).
Compressión	1	Unidad de compresión.

BeginDate	1	Fecha de inicio del intervalo de tiempo durante el cual se quieren extraer datos.
EndDate	-	Fecha fin del intervalo de tiempo durante el cual se quieren extraer datos.

Ejemplo:

.ConfigStk ssLoss, suMoney, sfShort, sctDays,1, cDate("09/08/2009"), cDate("09/08/2010")

Establece las propiedades de la estadística de un sistema, para obtener la información en moneda, de la pérdida por día (sólo aquellos en los que se ha entrado a corto), desde el 9 de agosto del 2009 al 9 de septiembre del 2010.

CurrentBar

Descripción:
Esta función retorna el número de orden de la barra en la que se están realizando los cálculos (barra actual). Considerando la primera barra de la serie de datos como la barra número 0, se irá incrementando en una unidad cada barra evaluada.

Cuando se trabaja con dos series de datos y una tiene más histórico que otra, los cálculos comenzarán cuando una barra de la primera serie coincida en el tiempo con una barra de la segunda serie. Esta barra será considerada como primera barra (CurrentBar=1).

Sintaxis:
.CurrentBar

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

.CurrentBar

Suponiendo que los cálculos se están efectuando en la séptima barra de la serie de datos, esta función retornaría el valor 6.

CurrentContract

Descripción:
Esta función se utiliza para conocer el número de contratos o de acciones que hay comprados o vendidos en la posición que está abierta.

Sintaxis:
.CurrentContract

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

.CurrentContract En el caso de que no haya, la función devuelve el valor 0.

CurrentEntries

Descripción:
Esta función se utiliza para conocer el número de entradas diferentes que hay abiertas en una posición. Una posición puede tener varias entradas diferentes en función de la etiqueta establecida en la orden y de la modalidad de casar operaciones elegida.

Sintaxis:
.CurrentEntries

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

.CurrentEntries

La función devuelve el número de entradas que haya en vigor en el momento de aplicar la función. Si no hay ninguna, retorna el valor 0.

Date

Descripción:
Todas las barras de un gráfico tienen una fecha, tanto si se trata de un gráfico de fin de día como si se trata de un gráfico intradiario. Permite obtener de cada barra la fecha en que se produjo. El formato en el que se retorna la fecha es militar (**AAAAMMDD)**. Por tanto, si la barra se produjo el día 10 de agosto de 2000, esta función retornará el valor numérico de 20000810.

Sintaxis:
.Date(BarsAgo,Data)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. El valor por defecto hace referencia a la barra actual. En este parámetro es posible indicar un valor numérico contenido en una variable o teclear directamente el valor. Puede especificarse como función sustituyendo el valor numérico. Este parámetro sólo permite valores positivos.
Data	Data	Fuente de datos sobre la que se aplica la función. El sistema actúa sobre Data, que es el nombre que toma la serie de datos del gráfico sobre el que se aplica la estrategia. Si en la ventana hay mas gráficos insertados se codificarán como Data1, Data2, Data3, etc.

Ejemplo:

.Date(3,Data)

La función devuelve la fecha correspondiente a hace 3 barras de la serie de datos codificada como Data.

DateSubstract

Descripción:
Devuelve la diferencia (en días) entre dos fechas.

Sintaxis:
.DateSubstract(Left, Right)

Parámetros:

Nombre	Defecto	Descripción
Left	-	Minuendo (fecha, en formato militar, a la que se resta)
Right	-	Sustraendo (fecha, en formato militar, que resta)

Ejemplo:

.DateSubstract(20100110, 20100120) La función retorna el valor 10.

ExitLong

Descripción:
Esta función se utiliza cuando deseamos cerrar una posición de compra y no posicionarnos a corto (o hacer una venta a crédito). Por ejemplo, si se da una orden de compra de 500 acciones, y se cumplen las condiciones que se han establecido para salir de ese negocio, se utilizará esta función.

Sintaxis:
.ExitLong Type, Contracts, Price, Label

Parámetros:

Nombre	Defecto	Descripción
Type	AtClose	Tipo de orden que se desea lanzar (AtClose, AtMarket, AtLimit, AtStop).
Contracts	1	Número de contratos/acciones. Las especificaciones numéricas sobre contratos pueden ser sustituidas por variables o por cualquier función definida previamente.
Price	---	Precio de compra. Sólo se indica este parámetro en las órdenes del tipo AtStop y AtLimit. El valor puede expresarse mediante un número, una variable, una función, o bien una mezcla de valor y función.
Label	---	Etiqueta de la orden en formato texto.

- Si no se especifica el número de acciones/contratos (Contracts) ni la etiqueta (Label, suponiendo que se están usando varias órdenes de compra), se cerrará la posición entera y saldremos del mercado.
- Si no se especifica el número de acciones pero sí la etiqueta, en el supuesto de que estemos usando varias órdenes de compra, se cerrarán todos los contratos/acciones correspondientes a la orden que tenía esa etiqueta.
- Si se especifica el número de contratos y la etiqueta, se cerrará ese número de contratos/acciones de la orden cuya etiqueta se ha indicado.
- Si se especifica el número de contratos/acciones pero no se especifica la etiqueta, suponiendo que hay varias posiciones abiertas, se cerrarán los contratos/acciones especificados en la última orden.

Ejemplos:

.ExitLong AtStop, 1, .GetEntryPrice -10, "C1"

Se cerrará la posición (1 contrato), de la orden etiquetada como "C1", con una venta en stop al precio de entrada menos 10 puntos. En este caso se trata de limitar las pérdidas con un stop de protección.

.ExitLong AtLimit, 1, .GetEntryPrice+30

Se cerrará la posición (1 contrato) con una venta limitada al precio de entrada más 30 puntos. En este caso el objetivo es cerrar por beneficios.

ExitShort

Descripción:
Se utiliza cuando se quiere cerrar una posición de venta a crédito y no posicionarse a largo (o hacer una compra). Por ejemplo, si se da una orden de venta de 5 contratos de un futuro, y se cumplen las condiciones para liquidar la posición, será necesario usar esta función para cerrar dicha posición.

Sintaxis:
.ExitShort Type, Contracts, Price, Label

Parámetros:

Nombre	Defecto	Descripción
Type	AtClose	Tipo de orden que se desea lanzar (AtClose, AtMarket, AtLimit, AtStop).
Contracts	1	Número de contratos/acciones. Las especificaciones numéricas sobre contratos pueden ser sustituidas por variables o por cualquier función definida previamente.
Price	---	Precio de compra. Sólo se indica este parámetro en las órdenes del tipo AtStop y AtLimit. El valor puede expresarse mediante un número, una variable, una función, o bien una mezcla de valor y función.
Label	---	Etiqueta de la orden en formato texto.

- Si no se especifica el número de acciones/contratos (Contracts) ni la etiqueta (Label, en el supuesto de que se estén usando varias órdenes de venta), se cerrará la posición entera y saldremos del mercado.
- Si no se especifica el número de acciones pero sí la etiqueta (supuesto de que se estén usando varias órdenes de venta), se cerrarán todos los contratos/acciones correspondientes a la orden que tiene esa etiqueta.
- Si se especifica el número de contratos y la etiqueta, entonces se cerrará ese número de contratos/acciones de la orden cuya etiqueta se ha especificado.
- Si se especifica el número de contratos/acciones pero no se especifica la etiqueta, si hay varias posiciones abiertas, se cerrarán los contratos especificados de la última orden.

Ejemplos:

.ExitShort AtStop, 1, .High +10, "ES1"

Se cerrará la posición (1 contrato), de la orden etiquetada como "ES1", con una compra en stop al precio máximo de la barra más 10 puntos. En este caso se trata de limitar las pérdidas con un stop de protección.

FeedFields

Descripción:
Determina los campos de información que podemos obtener en tiempo real usando el método del mismo nombre.

Sintaxis:
.FeedFields(Field)

Parámetros:

Nombre	Defecto	Descripción
Field	FFLast	**FFAskSize** Cantidad de títulos ofertados **FFBidSize** Cantidad de títulos demandados **FFBuy1** Precio ofertado en la primera posición de compra **FFBuyAgency** Agencia compradora **FFBuyOrders** Órdenes de compra **FFDate** Fecha **FFDecimals** Decimales **FFDescription** Descripción del símbolo **FFDiff** Diferencia

		FFDiff_P Diferencia % **FFExpiry_Date** Fecha de vencimiento **FFHigh** Máximo **FFISIN** Código ISIN **FFLast** Precio último **FFLastVol** Volumen último **FFLow** Mínimo **FFMinimumMov** Mínimo movimiento **FFNumTrades** Número de negocios **FFOpen** Apertura **FFOpenInterest** Open Interest **FFPrevious** Anterior **FFSell1** Precio ofertado en la primera posición de venta **FFSellAgency** Agencia vendedora **FFSellOrders** Órdenes de venta **FFSubMarket** Submercado al que pertenece el símbolo **FFTime** Hora **FFVolume** Volumen

Ejemplo:

Si queremos obtener el último negocio cerrado, previamente definimos una variable de salida:

Dim UltNegocio as Double

A esta variable se le asignará el valor que retorne ésta propiedad:

UltNegocio = .FeedFields(FFLast)

FilledOrders

Descripción:
La propiedad FilledOrders permite conocer si en una barra se han ejecutado órdenes activas de compra o venta asociadas a una etiqueta en concreto. Si se han ejecutado, devolverá un 1, en otro caso devolverá un 0.

Sintaxis:
.FilledOrders(Label, Side, BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
Label	-	Etiqueta asociada a la orden que se desea consultar.
Side	-	Posición de la orden a consultar. (0 posición de compra y 1 posición de venta)
BarsAgo	0	Barra de la que se desea extraer la fecha. El valor por defecto es la barra actual.

Ejemplo 1:

.FilledOrders("C1", 0, 10)

En el caso de que hace 10 barras se haya ejecutado una orden de compra, etiquetada como "C1", devolverá el valor 1.

Ejemplo 2:

En un sistema, lanzamos tres órdenes en stop de compra con las etiquetas A, B y C.

En la barra siguiente, queremos saber cuántas de las tres órdenes se han ejecutado.

Para ello hacemos lo siguiente:

Definimos un contador de órdenes

Dim ContadorOrdenes As Integer

```
If .FilledOrders("A",0 , 0) =1 then
      ContadorOdenes= 1
End If

If .FilledOrders("B",0 , 0) = 1 then
      ContadorOrdenes=ContadorOrdenes+1
End If

If .FilledOrders("C",0 , 0) =1 then
      ContadorOrdenes=ContadorOrdenes+1
End If
```

Supongamos que las tres órdenes lanzadas son al cierre.

En la barra siguiente, si queremos saber cuántas se han ejecutado, haremos la pregunta pero respecto a la barra anterior, pues las órdenes fueron ATCLOSE, y de haberse ejecutado, lo habrían hecho en la barra de anterior:

```
If .FilledOrders("A",0 , 1) =1 then
      ContadorOdenes= 1
End If

If .FilledOrders("B",0 , 1) = 1 then
      ContadorOrdenes=ContadorOrdenes+1
End If

If .FilledOrders("C",0 , 1) =1 then
      ContadorOrdenes=ContadorOrdenes+1
End If
```

GetBackGroundColor

Descripción:
Retorna el color de fondo de la ventana de una barra determinada.

Sintaxis:
.GetBackGrounColor (BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.

Ejemplo:

.GetBackGroundColor(0) Devuelve el color de fondo de la ventana para la barra actual.

GetBarColor

Descripción:
Retorna el color de la línea de datos indicada en una barra determinada.

Sintaxis:
.GetBarColor (BarsAgo, Line)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line	-	Línea, cuyo color en la barra indicada, se desea obtener.

Ejemplo:

.GetBarColor(1,3) Devuelve el color de la barra anterior de la línea 3.

GetBarsSinceEntry

Descripción:
Esta función se utiliza para conocer el número de barras que se han completado desde que se abrió una posición determinada (de compra o de venta). Si no se ha abierto ninguna posición retorna el valor 0.

Sintaxis:
.GetBarsSinceEntry(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto hace referencia a la posición actual.

Ejemplos:

.GetBarsSinceEntry (0) Retorna el número de barras formadas desde que se abrió la posición actual.

GetBarsSinceExit

Descripción:
Esta función se utiliza para conocer el número de barras que se han completado desde que se cerró una posición determinada. Si no se ha salido en ninguna posición retorna el valor 0.

Sintaxis:
.GetBarsSinceExit(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto hace referencia a la posición actual.

Ejemplos:

.GetBarsSinceExit (1) Retorna el número de barras formadas desde cierre del negocio anterior.

GetBarStyle

Descripción:
Esta función devuelve el estilo de la línea que se utiliza en una barra determinada.

Sintaxis:
.GetBarStyle(BarsAgo, Line)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line	-	Identifica la línea a la que pertenece la barra de la que se quiere obtener el estilo de línea.

Ejemplo:

.GetBarStyle (0,1)

Retorna el estilo de línea en la barra actual de la línea 1. Los valores que puede devolver la función son:

- **lsSolid:** Línea contínua.
- **lsDash:** Línea discontinua.
- **lsDot:** Línea punteada.
- **lsDashDot:** Línea discontinua con punto.
- **lsDashDotDot:** Línea discontinua con dos puntos.

GetBarWidth

Descripción:
Retorna el grosor de una línea concreta para la barra indicada.

Sintaxis:
.GetBarWidth(BarsAgo, Line)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line		Identifica la línea de datos a la que pertenece la barra de la que se desea conocer el grosor.

Ejemplo:

.GetBarWidth(0,1) La función devuelve el grosor(1,2,…) de la línea 1 en la barra actual.

GetConfigStk

Descripción:
Este procedimiento retorna las propiedades iniciales de las estadísticas que en ese momento se estén utilizando.

Sintaxis:
.GetConfigStk(Sing, Unit, Filt, CompType,Compressión, BeginDate, EndDate)

Parámetros:

Nombre	Defecto	Descripción
Sing	-	Filtro de resultados según sean los negocios ganadores (ssWins) o perdedores (ssLoss).
Unit	-	Representación de los datos en moneda (suMoney), en puntos (suPnts) o en porcentaje (suPortc).
Filt	-	Filtro de resultados por posición, largo (sfLong) o corto(sfShort).
CompType	-	Representación de los datos agrupados por negocios (sctTrades), días (sctDays), semanas (sctWeeks), meses (scMoths) o años (sctYears).
Compressión	-	Unidad de compresión.
BeginDate	-	Fecha de inicio del intervalo de tiempo de las estadísticas.
EndDate	-	Fecha fin del intervalo de tiempo de las estadísticas.

Ejemplo:
Supongamos que hemos definido las siguientes variables de salida:

```
Dim Signo As StatisticSign
Dim Unidad As StatisticUnit
Dim Filtro As StatisticFilt
Dim TComp As StatisticCompType
Dim Compresion As Long
Dim FechaIni As Date
Dim FechaFin As Date
```

Al hacer la llamada a la propiedad GetConfigStk:

```
.GetConfigStk(Signo, Unidad, Filtro, TComp, Compresion, FechaIni, FechaFin)
```

Cada una de las variables definidas previamente, se rellenaran con los datos retornados por GetConfigStk.

Siguiendo con el ejemplo que se hacía para la propiedad ConfigStk, al utilizar GetConfigStk, la información que se retornaría sería:

Signo = ssLoss = 2	Filtro = sfShort = 2	Compresion = 1	FechaFin = 09/08/2010
Unidad = suMoney = 0	Tcomp = sctDays = 3	FechaIni = 09/08/2009	

GetDailyLosers

Descripción:
Devuelve número de negocios perdedores entre dos fechas dadas. Se compara siempre con la fecha de entrada del negocio.

Sintaxis:
.GetDailyLosers(FromDate, ToDate)

Parámetros:

Nombre	Defecto	Descripción
FromDate	-	Fecha de inicio del intervalo a analizar.
ToDate	-1	Fecha de fin del intervalo a analizar. Si no se especifica ningún valor o se asigna -1, significa que la fecha final del intervalo será la fecha actual.

Ejemplo:

Si queremos saber el número de negocios perdedores entre 2 fechas, p.e entre 10/06/2010 y 10/09/2010.

Definimos previamente la variable a la que asignaremos la cantidad de negocios perdedores:

```
Dim NegPerdedores As Long
```

Asignamos el valor que retorna la función haciendo lo siguiente:

```
NegPerdedores = .GetDailyLosers(20100610,20100910)
```

GetDailyWinners

Descripción:
Devuelve el número de negocios ganadores entre dos fechas dadas. Se compara siempre con la fecha de entrada del negocio.

Sintaxis:
.GetDailyWinners(FromDate, ToDate)

Parámetros:

Nombre	Defecto	Descripción
FromDate	-	Fecha de inicio del intervalo a analizar.
ToDate	-1	Fecha de fin del intervalo a analizar. Si no se especifica ningún valor o se asigna -1, significa que la fecha final del intervalo será la fecha actual.

Ejemplo:

Si queremos saber el número de negocios ganadores entre 2 fechas, p.e entre 10/06/2010 y 10/09/2010.

Definimos previamente la variable a la que asignaremos la cantidad de negocios ganadores:
Dim NegGanadores As Long

Asignamos el valor que retorna la función haciendo lo siguiente:
NegGanadores = .GetDailyWinners(20100610,20100910)

GetEntryDate

Descripción:
Esta función se utiliza para conocer la fecha en la que se ha abierto una posición. El formato en el que se retorna la fecha es militar (**AAAAMMDD).** Por lo tanto, si la posición se abrió el día 25 de Junio de 2010, la función indicada retornará el valor numérico 20100625. Si no se ha entrado en ninguna posición, retornará el valor 0.

Sintaxis:
.GetEntryDate(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto hace referencia a la posición actual.

Ejemplos:

.GetEntryDate(0) La función devuelve la fecha en la que se abrió la posición actual.

.GetEntryDate(5) La función devuelve la fecha de entrada de hace 5 negocios.

GetEntryPrice

Descripción:
Esta función se utiliza para conocer el precio al que ha realizado la compra/venta al abrir una posición. Retorna el precio de entrada de la posición indicada en el parámetro EntryAgo. Si no se ha entrado en ninguna posición, retorna 0.

Sintaxis:
.GetEntryPrice(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás que deseamos consultar. El valor por defecto hace referencia a la posición actual.

Ejemplos:

.GetEntryPrice(0) La función devuelve el precio de entrada de la posición actual.

.GetEntryPrice(5) La función devuelve el precio de entrada de hace 5 negocios.

GetEntryTime

Descripción:
Esta función se utiliza para conocer la hora a la que se ha entrado en una posición. Esta información se retornará en formato militar de 24 horas **(HHMM),** de forma que si la hora es 5:35 pm se considerará como el valor numérico número 1735. Si no se ha entrado en ninguna posición, la función devuelve el valor 0.

Sintaxis:
.GetEntryTime(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás que deseamos consultar. El valor por defecto hace referencia a la posición actual.

Ejemplos:

.GetEntryTime(0) La función devuelve la hora de entrada de la posición actual.

.GetEntryTime(4) La función devuelve la hora de entrada de hace 4 negocios.

GetExitDate

Descripción:
Esta función se utiliza para conocer la fecha en la que se ha cerrado una posición. El formato en el que se retorna la fecha es militar (**AAAAMMDD).** Si la posición se cerró el día 25 de Junio de 2010, la función indicada retornará el valor numérico 20100625. Si no se ha cerrado la posición, devolverá el valor 0.

Sintaxis:
.GetExitDate(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto hace referencia a la posición actual.

Observaciones:
Hay que tener en cuenta que a todos los efectos, el último negocio abierto se considera cerrado en la barra actual (barra en la que se están realizando los cálculos). Por tanto, si se indica en el parámetro **EntryAgo** el valor de 0, esta función retornará la fecha de la barra actual.

De este modo, si en la estrategia tan sólo se utilizan las funciones Buy y Sell, el valor 1 al parámetro **EntryAgo** devolverá la fecha de la última operación cerrada antes de la actual. Mientras que el valor 0 siempre devolverá la fecha de la barra actual, indicando que actualmente existe una operación activa.

Pero si se utilizan también las funciones ExitLong y ExitShort, puede darse el caso de que en una barra no haya ningún negocio abierto. En estos casos, el valor 0 al parámetro **EntryAgo** no devolverá nada, ya que no hay ninguna operación abierta. Mientras que el valor 1 devolverá la fecha de la barra donde se liquidó la última operación.

Ejemplo:

.GetExitDate(3) La función devuelve la fecha de salida de hace 3 negocios.

GetExitOrder

Descripción:
Especifica si la n-ésima orden dada durante una barra concreta para un Data (de tipo sistema) es de salida o no. En el caso de haber sido una orden de salida, la función retorna **True**. Mientras que si la orden ha sido de entrada en mercado devuelve **False**.

Sintaxis:
.GetExitOrder(Identifier, BarsAgo, NumOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos de la que se obtendrá la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la cual se desea obtener el dato. El valor por defecto hace referencia a la barra actual.
NumOrder	-	Posición de la orden de la que se desea obtener la información.

Ejemplo:

Supongamos que en el sistema "MiSistema" se realiza lo siguiente en una barra concreta:

1) Hemos entrado con una orden .Buy con etiqueta "Compra_1"

2) El sistema prepara tres órdenes nuevas:

   a. .ExitLong atlimit 1, .GetEntryPrice + 50, "Compra_1"
   b. .ExitLong atstop, 1, .GetEntryPrice - 20, "Compra_1"
   c. .Sell atstop, 1, .Low – 100, "Venta_1"

Partiendo de esto, suponemos también que desde otro desarrollo, hemos creado una variable de tipo DataIdentifier que hace referencia a MiSistema y a la que hemos llamado MisSistemaData.

Desde éste segundo desarrollo, definimos una variable booleana que nos permita saber si una orden del sistema MiSistema es de salida o no.

Definimos la variable:

Dim Salida As Boolean

Y le asignamos el resultado de la función GetExitOrder para el caso de la barra actual respecto a la segunda posición:

Salida = .GetExitOrder(MiSistemaData, 0,1)

Para la barra concreta a la que hacíamos referencia inicialmente, la variable Salida devolverá True, pues en dicha barra la segunda orden lanzada desde MiSistema fue de salida.

GetExitPrice

Descripción:
Esta función se utiliza para conocer el precio al que se ha cerrado una posición. Si no se ha cerrado ninguna posición, retornará 0.

Sintaxis:
.GetExitPrice(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás que deseamos consultar. El valor por defecto hace referencia a la posición actual.

Es importante tener en cuenta las observaciones de la función GetExitDate.

Ejemplo:

.GetExitPrice(5) La función devuelve el precio de salida de hace 5 negocios.

GetExitTime

Descripción:

Esta función se utiliza para conocer la hora a la que se ha cerrado una posición. La hora la retornará en formato militar de 24 horas **(HHMM),** de forma que si la hora es 5:35 pm se considerará como el valor numérico número 1735. Si no se ha cerrado ninguna posición, la función devuelve el valor 0.

Sintaxis:
.GetExitTime(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás que deseamos consultar. El valor por defecto hace referencia a la posición actual.

Ejemplo:

.GetExitTime(3) La función devuelve la hora de salida en hace 3 negocios.

Es importante tener en cuenta las observaciones de la función GetExitDate.

GetHighest

Descripción:
Esta función se utiliza para obtener el valor mayor de las últimas **n** barras.

Sintaxis:
.GetHighest(Identifier, TPrice, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Identificador de la fuente. Cualquier serie de datos (máximos, mínimos, cierres, indicadores…). Si en la ventana hay mas fuentes de datos , éstas se codificarán como Data1, Data2, Data3, etc.
TPrice	PriceClose	Campo de la barra al que se desea hacer referencia. Para ello se debe indicar en este campo cualquiera de los siguientes valores: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Length	1	Número de barras hacia atrás a considerar. Cualquier variable de tipo numérico puede ser puesta en lugar de un número.

Ejemplos:

.GetHighest(Data, PriceHigh, 10)

La función devuelve el valor del mayor máximo en las últimas 10 barras de la serie de datos indicada (Data).

.GetHighest(Data1, PriceVolume, 100)

La función devuelve el valor del mayor volumen en las últimas 100 barras de la serie de datos indicada (Data1).

GetHighestBar

Descripción:
Retorna el número de barra en el que se produce el valor más alto de una serie.

Sintaxis:
.GetHighestBar (Data,TPrice, Leght)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Identificador de la fuente. Cualquier serie de datos (máximos, mínimos, cierres, indicadores…). Si en la ventana hay mas fuentes de datos insertadas, se codificarán como Data1, Data2, Data3, etc.
Tprice	PriceClose	Campo de la barra al que se desea hacer referencia. Para ello se debe indicar en este campo cualquiera de los siguientes valores: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Leght	2	Número de barras hacia atrás a considerar. Cualquier variable de tipo numérico puede ser puesta en lugar de un número.

Ejemplo:

.GetHighestBar(Data1, PriceLow, 20)

La función devuelve el número de barra (hacia atrás) donde se obtiene el mayor mínimo de las 20 últimas barras de la serie de datos indicada (Data1).

GetHistogramBand

Descripción:
Devuelve el número de línea de banda asignado a la línea que se especifique en el parámetro Line.

Sintaxis:
.GetHistogramBand(Line)

Parámetros:

Nombre	Defecto	Descripción
Line	-	Identifica la línea de datos de la cual se desea conocer su línea de banda.

Ejemplo:

Supongamos que deseamos crear un indicador y a la hora de representarlo indicamos lo siguiente:

```
.SetIndicatorValue x, 1
.SetIndicatorValue 50, 2
.SetBarRepresentation 0,1,irHistogram
.SetHistogramBand 1, 2
```

Si definimos una variable de salida:

```
Dim MiLineaBanda as Integer
```

A esta podemos asignar el valor que devuelve la función:

```
MiLineaBanda = .GetHistogramBand(1)    La función retornará el valor 2 (la línea de banda).
```

GetIndicatorIdentifier - GII

Descripción:
Esta función se utiliza para crear la serie de datos correspondiente a un indicador cualquiera y obtener un identificador de esa serie. Para ello es necesario declarar previamente una variable de tipo DataIdentifier.

Una vez definida la variable, le asignaremos el valor de la función **GetIndicatorIdentifier** para crear la serie del indicador y obtener un identificador del mismo.

El identificador del indicador debe obtenerse en el procedimiento OnInitCalculate.

Para obtener posteriormente un valor del indicador, es preciso usar la función GetIndicatorValue e indicar en el parámetro **Data** la variable en la que hemos guardado el identificador del indicador en cuestión.

El identificador obtenido por esta función puede ser utilizado además en cualquier función de VBA en la que se solicite un Data (serie de datos sobre la que se calculan las distintas funciones).

Sintaxis:
.GetIndicatorIdentifier(Name, ParentDataIdentifier, ParamArray())

También se puede utilizar el modo abreviado **GII**:

.GII(Name, ParentDataIdentifier, ParamArray())

Parámetros:

Nombre	Defecto	Descripción
Name	-	Código del indicador.
ParentDataIdentifier	-	Identificador de la serie sobre la que se calculará el indicador. Si indicamos en este parámetro Data, estaremos calculando el indicador sobre el data o serie sobre la que se inserte la estrategia. Si deseamos obtener el identificador de un indicador que se calcula sobre otro, debemos indicar en este parámetro el identificador del indicador sobre el que deseamos que se calcule.

ParamArray	-	Primer parámetro específico del indicador. Representa un grupo de parámetros cuyo número no está especificado ya que cada indicador tiene un número variable. (una media móvil tiene 2 y sin embargo el RSI tiene 3). El orden en el que deben indicarse los parámetros es el mismo en que figuran en el cuadro de diálogo que se muestra cuando vamos a insertar el indicador sobre el gráfico. Si el parámetro es de tipo **Price**, hace referencia a uno de los campos de la barra (Cierre, Apertura, etc.), como sería el caso de una media móvil cuyo segundo parámetro es origen de los datos. En estos casos, debemos especificar el campo de la barra sobre la que se calculará el indicador. Tendremos que introducir cualquiera de las siguientes constantes que equivalen a esos campos: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen.
ParamArray	-	Segundo parámetro específico del indicador.
...	-	...
n-ésimo ParamArray	-	n-ésimo parámetro específico del indicador.

Ejemplo:

.GetIndicatorIdentifier(RSI, Data, 14, 70, 30)

La fuente devuelve el identificador del indicador RSI (14,70 y 30 son los parámetros del indicador).

GetIndicatorPos

Descripción:
Esta función obtiene la tendencia que tiene un indicador en una barra determinada.

Sintaxis:
.GetIndicatorPos(BarsAgo, Line)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás.
Line	-	Identifica la línea a la que pertenece la barra de la que se quiere obtener la tendencia.

Ejemplo:

.GetIndicatorPos(0,1)

Retornará la tendencia, en la barra actual, de la línea 1 del indicador, pudiendo ser :

- **ipBull**(alcista)
- **ipBear**(bajista)
- **ipNeutral**(neutrual)

GetIndicatorValue - GIV

Esta función se utiliza para obtener el valor de un indicador. Para ello es preciso previamente haber determinado el identificador del indicador en el procedimiento InitCalculate mediante la función GetIndicatorIdentifier.

Sintaxis:
.GetIndicatorValue(Identifier, BarsAgo, LineNumber)

También se puede utilizar el modo abreviado **GIV**

.GIV(Identifier, BarsAgo,LineNumber)

Parámetros:

Nombre	Defecto	Descripción
Identifier	-	Identificador del indicador.
BarsAgo	0	Representa el número de barras hacia atrás a la que deseamos hacer referencia para obtener el valor del indicador. Si estamos usando una media móvil, un valor igual a cero para este parámetro haría que esta función retornara el valor de la media en la barra actual. Si pasamos valor 1, retornaría el valor de la media de hace una barra y así sucesivamente.
LineNumber	1	Línea del Indicador a obtener. Hay algunos indicadores que tienen más de una línea de datos. En estos casos si pasamos a este parámetro 1, devolverá un valor referente a la primera línea de datos y si pasamos 2 devolverá un valor de la segunda línea y así sucesivamente.

Ejemplo:

.GetIndicatorValue(RSI, 0, 1) Devuelve el valor de la primera línea del indicador en la barra actual.

GetLineName

Descripción:

Esta función devuelve el nombre que tiene la línea de datos indicada.

Sintaxis:

.GetLineName(Line)

Parámetros:

Nombre	Defecto	Descripción
Line	0	Identifica la línea de la que se desea obtener el nombre.

Ejemplo:

.GetLineName(1)

Suponiendo que estuviéramos programando un indicador y le hubiéramos asignado a la línea 2 el nombre "UpperBand", la función .GetLineName(1) devolverá "UpperBand".

GetLowest

Descripción:
Esta función se utiliza para obtener el valor menor de las últimas **n** barras de una serie de datos.

Sintaxis:
.GetLowest (Identifier, TPrice, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Identificador de la fuente. Cualquier serie de datos (máximos, mínimos, cierres, indicadores…). Si en la ventana hay mas fuentes de datos insertadas, se codificarán como Data1, Data2, Data3, etc.
TPrice	PriceClose	Campo de la barra al que se desea hacer referencia. Para ello se debe indicar en este campo cualquiera de los siguientes valores: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Length	1	Número de barras hacia atrás a considerar. Cualquier variable de tipo numérico puede ser puesta en lugar de un número.

Ejemplo:

.GetLowest (Data, PriceLow, 10) Devuelve el precio mínimo de las últimas 10 barras de la serie (Data).

GetLowestBar

Descripción:
Retorna el número de barra en el que se produce el valor más bajo de una serie.

Sintaxis:
.GetLowestBar (Data,TPrice, Leght)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Identificador de la fuente. Cualquier serie de datos (máximos, mínimos, cierres, indicadores…). Si en la ventana hay mas fuentes de datos insertadas, se codificarán como Data1, Data2, Data3, etc.
Tprice	PriceClose	Campo de la barra al que se desea hacer referencia. Para ello se debe indicar en este campo cualquiera de los siguientes valores: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Leght	2	Número de barras hacia atrás a considerar.

Ejemplo:

.GetLowestBar(Data1, PriceVolume, 20)

La función devuelve el número de barra (hacia atrás) donde se obtiene el volumen más bajo de las 20 últimas barras de la serie de datos indicada (Data1).

GetMarketPosition

Descripción:
Se utiliza para conocer, mientras se está calculando el sistema, en qué posición nos encontramos, es decir, si hay abierta una posición de compra o venta, o bien estamos fuera de mercado. Esta función es muy útil si estamos trabajando con órdenes en stop o limitadas, ya que no sabemos cuándo se han ejecutando.

Sintaxis:
.GetMarketPosition (EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de negocios hacia atrás. Por defecto es la posición actual.

Ejemplo:

.GetMarketPosition(0)

La función devuelve el tipo de posición actual. Si es de compra retornará 1, si es de venta retornará -1 y en caso de estar fuera de mercado retornará 0.

GetMaxContracts

Descripción:
Esta función se utiliza para conocer el mayor número de contratos que ha habido comprados o vendidos en una posición.

Sintaxis:
.GetMaxContracts (EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto hace referencia a la posición actual.

Ejemplo:
.GetMaxContracts(3) La función devuelve el número de contratos comprados/vendidos hace 3 negocios.

GetMaxEntries

Descripción:
Esta función se utiliza para conocer el mayor número de entradas diferentes que ha habido en una posición. Una posición puede tener varias entradas diferentes en función de la etiqueta establecida en la orden y de la modalidad de casar operaciones elegida.

Sintaxis:
.GetMaxEntries (EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto hace referencia a la posición actual.

Ejemplo:

.GetMaxEntries(5) La función devuelve el mayor número de entradas de hace 5 negocios.

GetNthHighest

Descripción:
Esta función se utiliza para obtener el valor mayor de las últimas **n** barras con un orden concreto.

Sintaxis:
. GetNthHighest (Identifier, Nth, TPrice, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Identificador de la fuente. Cualquier serie de datos (máximos, mínimos, cierres, indicadores…). Si en la ventana hay mas fuentes de datos insertadas, se codificarán como Data1, Data2, Data3, etc.
Nth	1	Es el ordinal que representa qué valor deseamos obtener. Si Nth vale 1, obtendremos el primer valor mayor de las **n** ultimas barras de la serie, si Nth vale 2 obtendremos el segundo mayor valor de la serie y así sucesivamente.
TPrice	PriceClose	Campo de la barra al que se desea hacer referencia. Para ello se debe indicar en este campo cualquiera de los siguientes valores: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Length	50	Número de barras hacia atrás a considerar.

Ejemplo:

.GetNthHighest(Data, 3, PriceHigh, 10) Devuelve el tercer mayor máximo de las últimas 10 barras.

GetNthLowest

Descripción:
Esta función se utiliza para obtener el valor menor de las últimas **n** barras con un orden concreto.

Sintaxis:
.GetNthLowest(Identifier, Nth, TPrice, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Identificador de la fuente. Cualquier serie de datos (máximos, mínimos, cierres, indicadores…). Si en la ventana hay mas fuentes de datos insertadas, se codificarán como Data1, Data2, Data3, etc.
Nth	1	Es el ordinal que representa qué valor deseamos obtener. Si Nth vale 1, obtendremos el primer valor menor de las **n** ultimas barras de la serie, si Nth vale 2 obtendremos el segundo menor valor de la serie y así sucesivamente.
TPrice	PriceClose	Campo de la barra al que se desea hacer referencia. Para ello se debe indicar en este campo cualquiera de los siguientes valores: **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Length	50	Número de barras hacia atrás a considerar.

Ejemplo:

.GetNthLowest (Data, 2, PriceLow, 10) Devuelve el 2 mínimo más bajo de las últimas 10 barras.

GetOrderCount

Descripción:
Devuelve la cantidad de órdenes activas de un sistema dadas en una barra concreta.

Sintaxis:
.GetOrderCount(Identifier, BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
Identifier	-	Serie de datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea extraer la cantidad de órdenes. El valor por defecto hace referencia a la barra actual.

Ejemplo:

Imaginemos que el sistema al que hacemos referencia desde Identifier en la barra actual, envía a mercado las siguientes órdenes:

- Una orden para girarse a corto porque el sistema está comprado
- Una orden para salirse por ganancias
- Una orden para salirse por pérdidas

Podemos asignar a una variable, previamente definida, el valor que retorna esta función:

```
Dim NumOrdenes as Integer

NumOrdenes = .GetOrderCount(MiSistemaData, 0)
```

La función retornará el valor 3 en la barra actual, ya que son las órdenes ACTIVAS en la misma.

GetOrderDate

Descripción:
Devuelve la fecha asignada a la n-ésima orden activa de un sistema, dada en una barra concreta.

Sintaxis:
.GetOrderDate(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea extraer la fecha. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Siguiendo con el ejemplo empleado para la función .GetOrderCount, si de las tres órdenes sólo nos interesa saber la fecha de la última, entonces haríamos lo siguiente:

Dim FechaUlOrden as Date

FechaUlOrden = .GetOrderDate(MiSistemaData,0, 2)

La función retorna una fecha del tipo DD/MM/YYYY. Normalmente la fecha que devolverá será la correspondiente a la barra actual (en caso de BarsAgo=0), o a la fecha de la barra a la que hagamos referencia (en caso de BarsAgo >0)

La posición a la que hacemos referencia en este caso es 2, ya que como se indica en el apartado Parámetros de esta función, el valor 0 se asigna a la primera orden, el valor 1 a la segunda y así sucesivamente.

GetOrderLabel

Descripción:
Devuelve la etiqueta asignada a la n-ésima orden activa de un sistema dada en una barra.

Sintaxis:
.GetOrderLabel(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea extraer la etiqueta. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Supongamos que en el sistema "MiSistema" se realiza lo siguiente en una barra concreta:

1) Hemos entrado con una orden .Buy con etiqueta "Compra_1"
2) El sistema prepara tres órdenes nuevas:

    a. .ExitLong atlimit 1, .GetEntryPrice + 50, "Compra_1"
    b. .ExitLong atstop, 1, .GetEntryPrice - 20, "Compra_1"
    c. .Sell atstop, 1, .Low – 100, "Venta_1"

Partiendo de esto, si desde un segundo sistema (suponiendo que hemos creado un DataIdentifier que use a "MiSistema"), indicamos lo siguiente:

Dim Etiqueta as String

Etiqueta = .GetOrderLabel(MiSistemaData, 0, 1)

En Etiqueta se obtendrá "Compra_1", que es la etiqueta asignada a la segunda orden de "MiSistemaData".

GetOrderPrice

Descripción:
Devuelve el precio de la n-ésima orden activa de un sistema dada en una barra concreta.

Sintaxis:
.GetOrderPrice(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de Datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea extraer la posición. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Siguiendo con el ejemplo empleado para la función .GetOrderCount, si de las tres órdenes, para la barra actual nos interesa saber el precio de la primera de ellas, podemos definir una variable de tipo Double, y a esta le asignaremos el valor que retorna la función .GetOrderPrice:

```
Dim Precio as Double

Precio = .GetOrderPrice(MiSistemaData, 0, 0)
```

GetOrderSide

Descripción:
Devuelve la posición (compra o venta) de la n-ésima orden activa de un sistema, dada en una barra concreta.

Sintaxis:
.GetOrderside(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea obtener el símbolo del valor asociado. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Siguiendo con el ejemplo empleado para la función .GetOrderLabel, si de las tres órdenes, para la barra actual nos interesa saber la posición de la segunda de ellas, podemos definir una variable de tipo OrderSide(osBuy, osSell), y a esta le asignaremos el valor que retorna la función .GetOrderside:

```
Dim Pos as OrderSide

Pos = .GetOrderSide(MiSistemaData, 0, 1)
```

La función retornará OsSell, ya que la segunda orden es una venta en stop.

GetOrderSymbolCode

Descripción:
Devuelve el código (en formato de Visual Chart, p.e 010072MFXI) del valor asociado de la n-ésima orden activa de un sistema, dada en una barra concreta.

Sintaxis:
.GetOrderSymbolCode(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea obtener el símbolo del valor asociado. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Supongamos que estamos operando sobre el Futuro Dax con el sistema utilizado para la función .GetOrderLabel. Si deseamos obtener el símbolo asociado a la tercera orden en la barra actual, podríamos definir una variable y asignar a esta el código del valor. Se haría de la siguiente forma:

Dim Simbolo as String

Simbolo=.GetOrderSymbolCode(MiSistemaData,0,2)

La función retornará "010015DX" que es el código del Futuro Dax en formato de Visual Chart.

GetOrderType

Descripción:
Devuelve el tipo de orden (stop, limitada, al cierre) de la n-ésima orden activa de un sistema, dada en una barra concreta.

Sintaxis:
.GetOrderType(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea extraer el tipo de orden. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Siguiendo con el ejemplo utilizado para la función .GetOrderLabel, si deseamos saber el tipo de orden asociado a la primera de ellas en la barra anterior, podríamos definir una variable de tipo TradeType (AtClose, AtLimit, AtStop, AtMarket) y asignar a ésta ese valor. Se haría de la siguiente forma:

Dim TipoOrden as TradeType

TipoOrden =.GetOrderType(MiSistemaData,1,0) La función retornará AtLimit.

GetOrderVolume

Descripción:
Devuelve la cantidad de contratos/acciones de la n-ésima orden activa de un sistema, dada en una barra concreta.

Sintaxis:
.GetOrderVolume(Identifier, BarsAgo, NumberOrder)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de Datos del que se obtiene la información. Debe tratarse de un Data asociado a un sistema que haya sido llamado a través del método GetSystemIdentifier.
BarsAgo	0	Barra de la que se desea extraer el volumen. El valor por defecto hace referencia a la barra actual.
NumberOrder	0	Número de orden de referencia asignado a la orden de la que se quiere extraer la información. El valor por defecto hace referencia a la primera orden activa que se ha realizado una barra concreta.

Ejemplo:

Siguiendo con el ejemplo utilizado para la función .GetOrderLabel, si deseamos saber el volumen de contratos/acciones de la segunda orden en la barra actual, podríamos definir una variable y asignar a esta el valor que retorne la función.

```
Dim VolumenOrden as Long

VolumenOrden =.GetOrderVolume(MiSistemaData,0,1)        La función retornará el valor 1.
```

GetPivotDow

Descripción:
Esta función se utiliza para obtener el valor de un pivot. Un pivot es un pico que se da en la cotización, en este caso podríamos considerarlo un soporte

Sintaxis:
.GetPivotDown(Identifier, Occur, TPrice, LeftCount, RightCount, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de la que obtendremos el pivot (cotizaciones, indicadores...).
Occur	1	Valor numérico que representa el número de pivot hacia atrás que deseamos obtener. Si Occur vale 1, estaremos obteniendo el primer pivot que haya desde la barra actual, si vale 2 el segundo pivot y así sucesivamente.
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
LeftCount	-	Es el número de barras a la izquierda del pivot.
RightCount	-	Es el número de barras a la derecha del pivot.
Length	50	Número de barras hacia atrás a considerar en la búsqueda del pivot.

Ejemplo:

.GetPivotDown(Data, 1, PriceLow, 2,4, 50)

Buscará, en las 50 barras anteriores a la actual, el valor del pivot más cercano (calculado sobre los mínimos), siendo en las 2 barras a la izquierda del pivot, y en las 4 barras a la derecha de éste, el valor del mínimo superior al de éste.

GetPivotUp

Descripción:
Esta función se utiliza para obtener el valor de un pivot. Un pivot es un pico que se da en la cotización, en este caso podríamos considerarlo una resistencia.

Sintaxis:
.GetPivotUp(Identifier, Occur, TPrice, LeftCount, RightCount, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de la que obtendremos el pivot (cotizaciones, indicadores...).
Occur	1	Valor numérico que representa el número de pivot hacia atrás que deseamos obtener. Si Occur vale 1, estaremos obteniendo el primer pivot que haya desde la barra actual, si vale 2 el segundo pivot y así sucesivamente.
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
LeftCount	-	Es el número de barras a la izquierda del pivot.
RightCount	-	Es el número de barras a la derecha del pivot.
Length	50	Número de barras hacia atrás a considerar en la búsqueda del pivot.

Ejemplo:

.GetPivotUp(Data, 1, PriceHigh, 2,4, 50)

Buscará, en las 50 barras anteriores a la actual, el valor del pivot más cercano (calculado sobre los máximos) siendo en las 2 barras a la izquierda del pivot, y en las 4 barras a la derecha de éste, el valor del máximo inferior al de éste.

GetPositionProfit

Descripción
Se utiliza para conocer el valor (en puntos) de la ganancia que se ha obtenido en una posición. Esta función considera el número de contratos/acciones que hayamos comprado/ vendido.

El valor que retorne será la diferencia entre el cierre de la última barra y el precio de entrada, multiplicado por el número de contratos/acciones. Si éste valor es positivo, la función indicará beneficio, y en caso de ser negativo, pérdida.

Por otra parte, si no se ha abierto ninguna posición devolverá el valor 0.

Sintaxis:
.GetPositionProfit(EntryAgo)

Parámetros:

Nombre	Defecto	Descripción
EntryAgo	0	Número de posiciones hacia atrás. El valor por defecto indica la posición actual.

Ejemplo:

.GetPositionProfit(1) Retorna la ganancia obtenida en el negocio anterior.

GetPrice

Descripción:
Esta función se utiliza para conocer el precio de un campo de una barra determinada perteneciente a una serie.

Sintaxis:
.GetPrice(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
TPrice	PriceClose	El campo de la serie del que deseamos obtener el valor. **PriceHigh:** Máxima. **PriceLow:** Mínima. **PriceOpen:** Apertura. **PriceClose:** Cierre. **PriceVolume:** Volumen. **PriceOpInt:** OpenInterest
BarsAgo	0	Número de barras hacia atrás.
Identifier	Data	Serie de datos sobre la que se aplica la función. Si en la ventana hay varios gráficos insertados (cotizaciones, indicadores etc..), éstos se codificarán como Data1, Data2, Data3, etc.

Ejemplo:

.GetPrice(PriceHigh, 22, Data1) Devuelve el precio máximo de 22 barras atrás, de la serie Data1.

GetStkLength

Descripción:
Devuelve la cantidad total de valores dados para un tipo de dato estadístico concreto.

Sintaxis:
.GetStkLength (Statistic)

Parámetros:

Nombre	Defecto	Descripción
Statistic	-	Permite seleccionar el tipo de dato estadístico del que se quiere extraer el valor.

Ejemplo:

Dentro de un sistema, queremos saber en un momento determinado, la longitud del Drawdown, es decir, cuantos datos (negocios) lleva calculados. Haríamos lo siguiente:

Longitud_DD = .GetStkLength(svAvg_Drawdown)

Longitud_DD devolverá la cantidad de negocios que lleva el sistema en el momento de llamar a la función, y que se han usado para calcular el drawdown.

GetStkValue

Descripción:
Devuelve el valor n-ésimo de un tipo de dato estadístico concreto.

Sintaxis:
.GetStkValue(Statistic, Index)

Parámetros:

Nombre	Defecto	Descripción
Statistic	-	Permite seleccionar el tipo de dato estadístico del que se quiere extraer el valor.
Index	-	Permite establecer una posición concreta en la lista de datos, de modo que se seleccionará el valor dado en una posición determinada.

Ejemplo:

Trabajando con un sistema, si queremos saber en un momento determinado el Drawdown que lleva realizado el indicador, podríamos asignar este valor a una variable previamente definida de la siguiente forma:

Actual_DD = .GetStkValue(svAvg_Drawdown)

Devolverá el valor del Drawdown de dicha barra en ese momento.

GetStkValues

Descripción:
Devuelve el conjunto de valores totales para un tipo de dato estadístico concreto.

Sintaxis:
.GetStkValues(Statistic, aValues)

Parámetros:

Nombre	Defecto	Descripción
Statistic	-	Permite seleccionar el tipo de dato estadístico del que se quiere extraer el valor.
aValues	-	Búffer donde se almacenan los valores retornados para el tipo de dato estadístico solicitado.

Ejemplo:

```
Dim Cantidad as long
Dim Values() as Double

Cantidad = .GetStkValues(svDrawdown,Values)
```

En este caso, devuelve en el array **Values** todos los datos. Y en **Cantidad** la cantidad de valores escritos en el array.

GetSymbolIdentifier-GSI

Descripción:
Esta función es necesaria sólo en los casos en los que deseemos utilizar los datos de un símbolo que no esté en pantalla a la hora de insertar la estrategia de que se trate, pues si así fuera, sería más cómodo utilizar los **Datas** correspondientes.

Esta función también es útil para hacer referencia a valores de un símbolo determinado en una Macro sobre Tabla, pues en este caso no tenemos disponibles los datos históricos en pantalla.

Para crear una fuente de datos y obtener el identificador de la misma es preciso declarar previamente una variable de tipo DataIdentifier.

Una vez definida la variable, le asignaremos el valor de la función **GetSymbolIdentifier** para obtener el identificador del símbolo. El identificador del símbolo debe obtenerse en el procedimiento OnInitCalculate.

El identificador obtenido por esta función puede ser utilizado luego en cualquier función de VBA en la que se nos solicite un Data (serie de datos sobre la que se calculan las distintas funciones).

Sintaxis:

.GetSymbolIdentifier(Symbol, Compresion, Cr, FromDate, ToFinalDate)

También se puede utilizar su modo abreviado **GSI:**

.GSI(Symbol, Compresion, Cr, FromDate, ToFinalDate)

Parámetros:

Nombre	Defecto	Descripción
Symbol	-	Código del símbolo deseado.
Compresion	-	Unidad de compresión (2, 5 ,10...).
Cr	-	Tipo de compresión. Hay cuatro tipos posibles de compresión: **CrMinutos:** Para obtener un gráfico de minutos. **CrDias:** Para obtener un gráfico diario. **CrSemanas:** Para obtener un gráfico de barra semanal. **CrMeses:** Para obtener un gráfico de barra mensual.
FromDate	-	Fecha inicial de la fuente de datos cuyo identificador vamos solicitamos.
ToFinalDate	-	Es la fecha final de los datos históricos que vamos a cargar. Normalmente esta fecha deseamos que sea siempre superior a la fecha actual por lo que aconsejamos que se indique en este parámetro la siguiente fecha 01/01/2037 para asegurarnos de que siempre se actualizan los datos de la fuente cuyo identificador solicitamos.

Ejemplo:

.GetSymbolIdentifier(010072MFXI, 5, CrMinutos, Date-30, Date)

La función utiliza los últimos 30 días del histórico del futuro del IBEX(MFXI) en 5 minutos.

GetSymbolInfo

Descripción:

Esta función devuelve una serie de características propias del producto en cuestión, y no sólo de una barra en concreto. Estos valores permanecen constantes durante todo el gráfico. La información que retorna la función viene determinada por el tipo de valor **SymbolInfo**, el cual puede tomar los siguientes valores:

SbiBarCompresion	Compresión utilizada para las barras.
SbiCode	Código del activo.
SbiCompresion	Tipo de compresión que se está usando (ticks, minutos, días,…)
SbiFirstSessionEnd	Hora de cierre de sesión del activo (formato HHMM).
SbiFirstSession Start	Hora de inicio de sesión del activo (formato HHMM).
SbiMarket	Mercado al que pertenece el producto.
SbiMinMov	Mínimo movimiento (pipos) que puede llegar a dar el producto.
SbiName	Nombre del producto.
SbiNumDec	Número de decimales considerados en la escala que usa el producto.
SbiPath	Todos los símbolos registrados en nuestro ordenador se encuentran en la carpeta RealServer\Data de VisualChart. Con este dato se puede extraer

	la ruta dentro de la carpeta Data de un activo en cuestión.
SbiPointValue	Valor por punto del producto.
SbiTimeD if	Diferencia horaria del producto sobre el que se aplica (en segundos)
SbiVendor	Especifica el Vendor del activo.

Sintaxis:
.GetSymbolInfo(Info, Data, Day)

Parámetros:

Nombre	Defecto	Descripción
Info	SbiName	Tipo de SymbolInfo que deseamos consultar.
Symbol	Data	Fuente de datos de la que se quiere extraer la información. Si no se especifica, se considera el gráfico sobre el que se inserta el sistema.
Day	-	Día de la semana del que se quiere extraer la información. Algunos valores pueden variar dependiendo del día de la semana.(no obligatorio)

Ejemplos:

.GetSymbolInfo(SbiTimeDif)

Si se está aplicando al Futuro e-mini S&P (diferencia horaria 7 horas), ésta función retornará 25200(segundos).
.GetSymbolInfo(SbiMinMov)

Si se está aplicando la función al Futuro DX(010015DX), ésta retornará 0.5 que es su mínimo movimiento.

GetSymbolInfoEx

Descripción:
Esta propiedad devuelve una estructura con todos los datos de tipo **SymbolInfo** relativos a la serie de datos asignado. El uso de esta propiedad es el de evitar tener que hacer más de una llamada al recurso para extraer todos los datos **SymbolInfo**.

Sintaxis:
.GetSymbolInfoEx(Symbol, Day)

Parámetros:

Nombre	Defecto	Descripción
Symbol	Data	Fuente de datos de la que se quiere extraer la información. Si no se especifica, se considera el gráfico sobre el que se inserta el sistema.
Day	-	Parámetro opcional. Día de la semana del que se quiere extraer la información. Algunos valores pueden variar dependiendo del día de la semana: **vbFriday:** Viernes **vbMonday:** Lunes **vbSaturday:** Sábado **vbSunday:** Domingo **vbThursday:** Martes **vbTuestday:** Jueves **vbWednesday:** Miercoles

GetSystemIdentifier-GSYSI

Descripción:
Es una función que permite obtener internamente la información de un sistema determinado. De este modo, se puede extraer información de dicho sistema sin necesidad de calcularlo nuevamente.

Sintaxis:
.GetSystemIdentifier(Name, ParentDataIdentifier, ParamArray)

También se puede utilizar el método abreviado **GSYSI.**

.GSYSI(Name, ParentDataIdentifier, ParamArray)

Parámetros:

Nombre	Defecto	Descripción
Name	-	Código del sistema del cual se desea obtener la información.
ParentDataIdentifier	Data	Fuente de datos sobre la que se estaría cargando el sistema.
ParamArray	-	Colección de parámetros de entrada que exige el sistema (opcional).

Ejemplo:

Podemos asignar a una variable de tipo DataIdentifier la información de un sistema determinado, y utilizarla para el uso de otras funciones, por ejemplo, .GetOrderCount, .GetOrderLabel etc.

```
Dim MiSistemaData as DataIdentifier

MiSistemaData =.GetSystemIdentifier(MiSistema, Data, NumeroContratos, Objetivo, Perdidas, HoraInicio, HoraFin)

Dim GOC As Long

GOC = .GetOrderCount(MisistemaData)
```

GetSwingHigh

Descripción:
Esta función es similar a GetPivotUp. Se utiliza para obtener el valor de un pivot. Un pivot es un pico que se da en la cotización. Los pivots pueden calcularse sobre cualquier serie de datos, por tanto pueden ser calculados sobre símbolos e indicadores. En el caso de no existir un pivot, la función devolverá el valor 0.

Sintaxis:
.GetSwinHigh(Identifier, Occur, Tprice, Strength, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de la que obtendremos el pivot (cotizaciones, indicadores...).
Occur	1	Valor numérico que representa el número de pivot hacia atrás que deseamos obtener. Si Occur vale 1, estaremos obteniendo el primer pivot que haya desde la barra actual, si vale 2 el segundo pivot y así sucesivamente.
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Strength	2	Número de barras a considerar ambos lados del pivot.
Length	50	Número de barras hacia atrás a considerar en la búsqueda del pivot.

Esta función es de gran utilidad pues nos ayuda a buscar puntos de resistencia o figuras de vuelta. Decimos que existe un pívot Superior cuando un valor de una serie de datos es mayor que un número de valores anteriores y posteriores especificados en el parámetro Strength.

Ejemplo:

.GetSwingHigh(Data, 1, PriceHigh, 2, 50)

Buscará, en las 50 barras anteriores a la actual, el valor del pivot más cercano (calculado sobre los máximos), siendo en las 2 barras a cada lado del pivot, el valor del máximo inferior al de éste.

GetSwingHighBar

Descripción:
Esta función devuelve la longitud, en número de barras, desde el último pívot alcista aparecido. Entendiendo como pívot alcista a un máximo que es mayor tanto a su izquierda como a su derecha.

Sintaxis:
.GetSwinHighBar(Identifier, Occur, Tprice, Strength, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de la que obtendremos el pivot (cotizaciones, indicadores...).
Occur	1	Valor numérico que representa el número de pivot hacia atrás que deseamos obtener. Si Occur vale 1, estaremos obteniendo el primer pivot que haya desde la barra actual, si vale 2 el segundo pivot y así sucesivamente.
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Strength	5	Número de barras a considerar ambos lados del pívot para que se considere como tal.
Length	50	Número de barras hacia atrás a considerar en la búsqueda del pívot. Si se supera el número de barras la función devuelve el valor 0.

Ejemplo:

.GetSwingHighBar(Data, 1, PriceHigh, 2, 50)

Buscará, en las 50 barras anteriores a la actual, el valor del pivot más cercano (calculado sobre los máximos), siendo en las 2 barras a cada lado del pivot, el valor del máximo inferior al de éste. Si dicho pivot se encuentra a 5 barras desde la barra actual en la que se usa la función, devolverá un 5. En la barra siguiente, devolverá un 6 y así sucesivamente hasta que aparezca un nuevo pivot.

GetSwingLow

Descripción:
Esta función es similar a GetPivotDown. Se utiliza para obtener el valor de un pivot. Un pivot es un pico que se da en la cotización, en este caso podríamos considerarlo un soporte ya que sería un pico invertido. Los pivots pueden calcularse sobre cualquier serie de datos, por tanto pueden ser calculados sobre símbolos e indicadores. Si no se encuentra ningún pivot, la función devolverá el valor 0.

Sintaxis:
.GetSwingLow(Identifier, Occur, Tprice, Strength, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de la que obtendremos el pivot (cotizaciones, indicadores...).
Occur	1	Valor numérico que representa el número de pivot hacia atrás que deseamos obtener. Si Occur vale 1, estaremos obteniendo el primer pivot que haya desde la barra actual, si vale 2 el segundo pivot y así sucesivamente.
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Strength	2	Número de barras a considerar ambos lados del pivot.
Length	50	Número de barras hacia atrás a considerar en la búsqueda del pivot.

Esta función es de gran utilidad pues nos ayuda a buscar soportes o figuras de vuelta. Decimos que existe un pivot inferior cuando un valor de una serie de datos es menor o igual a un número de valores anteriores y posteriores especificados en el parámetro Strength.

Ejemplo:

.GetSwingLow(Data, 1, PriceLow, 2, 50)

Buscará, en las 50 barras anteriores a la actual, el valor del pivot más cercano (calculado sobre los mínimos), siendo en las 2 barras a cada lado del pivot, el valor del mínimo superior al de éste.

GetSwingLowBar

Descripción:
Esta función devuelve la longitud, en número de barras, desde el último pivot bajista aparecido. Entendiendo como pívot bajista a un mínimo que es menor tanto a su izquierda como a su derecha.

Sintaxis:
.GetSwingLowBar(Identifier, Occur, Tprice, Strength, Length)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de la que obtendremos el pivot (cotizaciones, indicadores...).
Occur	1	Valor numérico que representa el número de pivot hacia atrás que deseamos obtener. Si Occur vale 1, estaremos obteniendo el primer pivot que haya desde la barra actual, si vale 2 el segundo pivot y así sucesivamente.
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Equivale a la Máxima. **PriceLow:** Equivale a la Mínima. **PriceOpen:** Equivale a la Apertura. **PriceClose:** Equivale al Cierre. **PriceVolume:** Equivale al Volumen. Si se calcula esta función sobre un indicador, se pasará como parámetro en **TPrice** el valor **PriceClose**, que hace referencia al cierre de la serie de datos del identificador. Si se indicara **PriceHigh** o cualquier otro sería indiferente, pues retornaría siempre el mismo valor.
Strength	2	Número de barras a considerar ambos lados del pivot para que se considere como tal.
Length	50	Número de barras hacia atrás a considerar en la búsqueda del pivot.

Ejemplo:

.GetSwingLow(Data, 1, PriceLow, 2, 50)

Buscará, en las 50 barras anteriores a la actual, el valor del pivot más cercano (calculado sobre los mínimos), siendo en las 2 barras a cada lado del pivot, el valor del mínimo superior al de éste. Si el pivot se encuentra a 5 barras desde la barra actual en la que se usa la función, devolverá un 5. En la barra siguiente, devolverá un 6 y así sucesivamente hasta que aparezca un nuevo pívot.

GetTrueHigh

Descripción:
Devuelve el precio más alto entre la máxima de una barra y el cierre de la barra anterior.

Sintaxis:
.GetTrueHigh (Identifier, BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos sobre la que se aplica el cálculo.
BarsAgo	10	Número de barra hacia atrás que servirá de referencia para el cálculo.

Ejemplo:

.GetTrueHigh (Data, 1)

Compara en el histórico de la serie Data, el máximo de una barra con el cierre de la anterior, indicándose el mayor de los 2 (tomando como referencia para comenzar el cálculo la barra anterior a la actual (1)).

GetTrueLow

Descripción:
Devuelve el precio más bajo entre la mínima de una barra y el cierre de la barra anterior.

Sintaxis:
.GetTrueLow (Identifier, BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos sobre la que se aplica el cálculo.
BarsAgo	10	Número de barras hacia atrás que servirá como referencia para el cálculo.

Ejemplo:

.GetTrueLow (Data1, 5)

Compara en el histórico de la serie Data1, el mínimo de una barra con el cierre de la anterior, indicándose el mínimo de los 2 (tomando como referencia para comenzar el cálculo 5 barras hacia atrás).

GetTrueRange

Descripción:
Retorna la diferencia entre .GetTrueHigh y .GetTrueLow en una serie de datos.

Sintaxis:
.GetTrueRange (Identifier, BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos sobre la que se aplica el cálculo.
BarsAgo	10	Número de barra sobre la que se realiza el cálculo.

Ejemplo:

Supongamos que creamos un indicador que extrae el siguiente dato:

Dim Diferencia As Double

Diferencia = .GetTrueRangeCustom(Data, 0, 10700, 10400)

Si insertamos el indicador sobre el gráfico del IBEX FUT. CONTINUO, por lo general Diferencia valdrá 300, que es la diferencia entre 10700 y 10400. Sin embargo, en aquellas barras que superen por encima o por debajo este margen, la diferencia cambiará. Por ejemplo, si el cierre vale 10809, el máximo pasa a ser 10809, que es el verdadero máximo, y en este caso la Diferencia será 409

GetTrueRangeCustom

Descripción:
Devuelve la diferencia entre un valor máximo y otro valor mínimo establecidos. Si el valor máximo es menor que el valor mínimo, el valor retornado será 0.
Sintaxis:
.GetTrueRangeCustom (Identifier, BarsAgo, High, Low)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos de la que se extrae la información.
BarsAgo	10	Número de barra de referencia.
High	-	Valor máximo.
Low	-	Valor mínimo.

Ejemplo:

.GetTrueRange (Data, 2)

Retorna la diferencia entre el valor que devuelve la función GetTrueHigh y GetTrueLow en la segunda barra hacia atrás, de la serie Data.

GetVolatility

Descripción:
Devuelve la volatilidad entre la barra actual y la n-ésima barra hacia atrás, en términos de diferencia de puntos.

Sintaxis:
.GetVolatility (Identifier, Tprice, Lenght)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Serie de datos sobre la que se aplica el cálculo (cotización, indicador..).
TPrice	PriceClose	El campo de la serie del que deseamos obtener el pivot. **PriceHigh:** Máxima. **PriceLow:** Mínima. **PriceOpen:** Apertura.

		PriceClose: Cierre. **PriceVolume:** Volumen.
Lenght	1	Distancia con respecto a la barra actual para calcular la volatilidad.

Ejemplo:

.GetVolatility(Data1, PriceLow, 5)

Devuelve la diferencia entre mínimo de la barra actual y el de hace 5 barras (de la serie de datos Data1).

GetWndBackGroundColor

Descripción:
Retorna el color de fondo de la ventana de un indicador.

Sintaxis:
.GetWndBackGrounColor()

Parámetros:

Nombre	Defecto	Descripción
-	-	-

GrossLoss

Descripción:
Obtiene el valor de las pérdidas brutas de los negocios negativos que nuestro sistema lleva acumulado hasta la barra actual (en la que se están realizando los cálculos). Para obtenerla consideramos que el último negocio abierto concluye en el cierre de la barra actual.

Sintaxis:
.GrossLoss(Show As SttRepresentation)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

.GrossLoss(Porcentual) Retornará el porcentaje de pérdida (bruta) obtenida por el sistema.

GrossProfit

Descripción:
Obtiene el valor de la ganancia brutas de los negocios positivos que nuestro sistema lleva acumulado hasta la barra actual (barra en la que se están realizando los cálculos). Para obtenerla consideramos que el último negocio abierto concluye en el cierre de la barra actual.

Sintaxis:
.GrossProfit(Show As SttRepresentation)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

.GrossProfit(Puntos) Retornará en puntos la ganancia bruta obtenida por el sistema.

High

Descripción:
Esta función devuelve el valor de la máxima de una barra.

Sintaxis:
.High(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. El valor por defecto hacer referencia a la barra actual. En este parámetro se puede utilizar cualquier valor numérico contenido en una variable o introducir directamente el valor. Puede especificarse como función sustituyendo el valor numérico.
Identifier	Data	Fuente de datos sobre la que se aplica la función. Si en la ventana hay más gráficos insertados, se codificarán como Data1, Data2, Data3 etc.

Ejemplo:

.High(4, Data1) Retornará el máximo de 4 barras atrás (de la fuente de datos Data1).

LargestLosingTrade

Descripción:
Devuelve el resultado de la peor operación realizada. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.LargestLosingTrade(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

Deseamos saber en un momento determinado, el resultado de la peor operación (en puntos) realizada por nuestro sistema. Podríamos definir una variable:

Dim PeorOperacion

Y a esta asignarle el valor que devuelve la propiedad:

PeorOperacion=.LargestLosingTrade(ByPoints)

LargestWinningTrade

Descripción:
Devuelve el resultado de la mejor operación realizada. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.LargestWinningTrade(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

Deseamos saber en un momento determinado, el resultado de la mejor operación (en porcentaje) realizada por nuestro sistema. Podríamos definir una variable:

Dim MejorOperacion

Y a esta asignarle el valor que devuelve la propiedad:

MejorOperacion=.LargestWinningsTrade(Porcentual)

Lc_Index

Descripción:
Devuelve el ratio **Ganancias a Largo/Ganancias a Corto**. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.Lc_Index(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

Deseamos saber en un momento determinado, el ratio ganancias a largo/ganancias a corto(porcentual) de nuestro sistema. Podríamos definir una variable:

Dim ratio_ganancia

Y a esta asignarle el valor que devuelve la propiedad:

ratio_ganancia=.Lc_Index(Porcentual)

LimitOrder

Descripción:
Esta propiedad permite obtener la cantidad existente de órdenes activas de oferta y demanda para una posición concreta, en una barra determinada. Sólo devuelve resultados durante el tiempo real, pues no existe histórico de las posiciones de oferta y demanda.

Sintaxis:
.LimitOrder(Level, Side, BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
Level	1	Indica la n-ésima posición de oferta/demanda de la que deseamos saber la cantidad.
Side	osBuy	Posición de la orden , osBuy (compra), osSell (venta).
BarsAgo	0	Solo devuelve resultados en tiempo real.
Identifier	Data	Serie de datos de la que se extrae la información.

Ejemplo:

.LimitOrder(4, osSell, 0, Data1)

Se obtiene es la cantidad de títulos/contratos ofertados en la cuarta posición de venta de la fuente Data1.

LimitPrice

Descripción:
La propiedad permite obtener el precio de las órdenes activas de oferta y demanda para una posición concreta en una barra determinada. Sólo devuelve resultados durante el tiempo real, pues no existe histórico de las posiciones de oferta y demanda.

Sintaxis:
.LimitePrice(Level, Side, BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
Level	1	Indica la n-ésima posición de oferta o demanda de la que deseamos saber el precio.
Side	osBuy	Posición de la orden , osBuy (compra), osSell (venta).
BarsAgo	0	Solo devuelve resultados en tiempo real.
Identifier	Data	Dato del que se extrae la información.

Ejemplo:

.LimitOrder(4, osSell, 0, Data1) Devuelve el precio ofertado en la cuarta posición de venta del Data1.

LimitVol

Descripción:
La propiedad LimitVol permite obtener el volumen de contratos existente de órdenes activas de oferta y demanda para una posición concreta en una barra determinada. Esta propiedad sólo devuelve resultados durante el tiempo real, pues no existe histórico de las posiciones de oferta y demanda.

Sintaxis:
.LimiteVol(Level, Side, BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
Level	1	Indica la n-ésima posición de oferta o demanda de la que deseamos saber la cantidad ofertada.
Side	osBuy	Posición de la orden , osBuy (compra), osSell (venta).
BarsAgo	0	Solo devuelve resultados en tiempo real.
Identifier	Data	Dato del que se extrae la información.

Ejemplo:

.LimitVol(2, osBuy, 0, Data)

Se obtiene el volumen ofertado de contratos/títulos en la segunda posición de compra de la fuente Data.

Low

Descripción:
Esta función devuelve el valor del mínimo de una barra.

Sintaxis:
.Low(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. El valor por defecto hacer referencia a la barra actual. En este parámetro se puede utilizar cualquier valor numérico contenido en una variable o introducir directamente el valor. Puede especificarse como función sustituyendo el valor numérico.
Identifier	Data	Fuente de datos sobre la que se aplica la función. Si en la ventana hay más gráficos insertados, se codificarán como Data1, Data2, Data3 etc.

Ejemplo:

.Low(0, Data2) Retornará el mínimo de la barra actual (de la fuente de datos Data2).

MinutesToTime

Descripción:
Es utilizada para transformar minutos en tiempo standard.

Sintaxis:
.MinutesToTime(Minutes)

Parámetros:

Nombre	Defecto	Descripción
Minutes	-	Tiempo en formato numérico

Ejemplo:

.MinutesToTime(300) Retornará el valor 5 (300 minutos equivalen a 5 horas).

NetProfit

Descripción:
Esta función se utiliza para obtener el valor de la ganancia neta que nuestro sistema lleva acumulada hasta la barra actual (barra en la que se están realizando los cálculos). Para obtener la ganancia total consideramos que el último negocio abierto concluye en el cierre de la barra actual.

Sintaxis:
.NetProfit(Show As SttRepresentation)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

.NetProfit(punto) Retornará en puntos la ganancia neta obtenida por el sistema.

NumberOfLines

Descripción:
Esta función devuelve el número de líneas que tiene el indicador sobre el que se aplica. El indicador debe haber asignado valores a las líneas para que el valor devuelto las incluya (solo devuelve el número de líneas que actualmente existen).

Sintaxis:
.NumberOfLines

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

Supongamos que creamos un indicador llamado MiIndicador que realiza el siguiente cálculo:

```
Valor1 = .High + .Low + .Close / 3
Valor2 = .High + .Low / 2
Valor3 = .Open + .Close + .Close(1) / 3
```

Y luego pintamos:

```
.SetIndicatorValue Valor1, 1
.SetIndicatorValue Valor2, 2
.SetIndicatorValue Valor3, 3
```

Si dentro de nuestro código utilizamos ésta función:

```
NLineas=.NumberOfLines
```

Nlineas será igual a 3 que es el número de líneas que hemos pintado.

NumberOfLosingTrades

Descripción:
Devuelve el número de negocios perdedores realizados hasta ese momento. Este valor irá cambiando a medida que se generen nuevas barras y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.NumberOfLosingTrades

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

Supongamos que deseamos saber, en una barra determinada, el número de negocios perdedores acumulados. Podriamos definir una variable:

```
Dim perdedores as  long

Perdedores = .NumberOfLosingTrades
```

Devuelve el número de negocios perdedores acumulado hasta el momento en que se realiza la llamada a la propiedad.

NumberOfTrades

Descripción:
Devuelve el número de negocios realizados hasta ese momento. Este valor irá cambiando a medida que se generen nuevas barras y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.NumberOfTrades

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

Supongamos que deseamos saber, en una barra determinada, el número de negocios totales acumulados. Podriamos definir una variable:

```
Dim totalnegocios as  long

totalnegocios =.NumberOfTrades
```

Devuelve el número de negocios acumulado hasta el momento en que se realiza la llamada a la propiedad.

NumberOfWinningTrades

Descripción:
Devuelve el número de negocios ganadores realizados hasta ese momento. Este valor irá cambiando a medida que se generen nuevas barras y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.
Sintaxis:
.NumberOfWinningTrades

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

Supongamos que deseamos saber, en una barra determinada, el número de negocios ganadores acumulados. Podriamos definir una variable:

```
Dim ganadores as  long

ganadores =.NumberOfWinningTrades
```

Devuelve el número de negocios ganadores acumulado hasta el momento en que se realiza la llamada a la propiedad.

Open

Descripción:
Esta función devuelve el valor de la apertura de una barra.

Sintaxis:
.Open(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. El valor por defecto hace referencia a la barra actual. En este parámetro es posible indicar un valor numérico contenido en una variable o teclear directamente el valor. Puede especificarse como función sustituyendo el valor numérico. Este parámetro sólo permite valores positivos.
Identifier	Data	Fuente de datos primaria. El sistema actúa sobre Data que es el nombre que toma la serie de datos del gráfico sobre el que se aplica la estrategia. Si en la ventana hay mas gráficos insertados se codificarán como Data2, Data3, Data4, etc.

Ejemplo:

.Open(3, Data1) Devuelve la apertura de 3 barras atrás de la fuente de datos codificada como Data1.

OpenDay

Descripción:
Esta función devuelve el valor 1 en el caso de que la barra a la que se hace referencia sea la última de la sesión, o bien 0 en caso contrario.

Sintaxis:
.OpenDay(Identifier, BarsAgo)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Fuente de datos primaria. El sistema actúa sobre Data que es el nombre que toma la serie de datos del gráfico sobre el que se aplica la estrategia. Si en la ventana hay mas gráficos insertados se codificarán como Data2, Data3, Data4, etc. Puede tratarse también de un indicador.
BarsAgo	-	Específica la barra de referencia para substraer el dato. La barra actual es 0.

OpenInt

Descripción:
Esta función devuelve el valor del **OpenInterest** de una barra (en el caso de contratos de futuros, donde se facilita esta información).

Sintaxis:
.OpenInt(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. El valor por defecto hace referencia a la barra actual. En este parámetro es posible indicar un valor numérico contenido en una variable o teclear directamente el valor. Puede especificarse como función sustituyendo el valor numérico. Este parámetro sólo permite valores positivos.
Identifier	Data	Fuente de datos primaria. El sistema actúa sobre Data que es el nombre que toma la serie de datos del gráfico sobre el que se aplica la estrategia. Si en la ventana hay mas gráficos insertados se codificarán como Data2, Data3, Data4, etc.

Ejemplo:

.OpenInt(5, Data)

Devuelve el OpenInteres de hace 5 barras (de la serie de datos Data).

PaintBar

Descripción:

Mediante esta opción, es posible pintar barras en el gráfico con los valores deseados para apertura, máxima, mínima, cierre.

Sintaxis:
.PaintBar(Open, High, Low, Close, Color, LineNumber, Width, nBars)

Parámetros:

Nombre	Defecto	Descripción
Open	-	Apertura de la barra. Permite utilizar una función o variable.
High	-	Máxima de la barra. Permite utilizar una función o variable.
Low	-	Mínima de la barra. Permite utilizar una función o variable.
Close	-	Cierre de la barra. Permite utilizar una función o variable.
Color	-	Color que se utilizará para pintar las barras. Se utiliza la función RGB.
LineNumber	-	Especifica el número de orden de pintar estudio. Esto es para estudios que utilizan diferentes órdenes de pintar (p.e si queremos incluir en un estudio una orden de pintar tipo PintarBarra (.PaintBar) y otra del tipo Pintar Línea (.PaintSeries), debemos poner a la primera orden, número de línea 0 y a la segunda, número de línea 1.
Width	1	Grosor de la barra que se pintará.
nBarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.

Ejemplo:

.PaintBar .Open, .High, .Low, .Close, RGB(0,0,225), 0, 1, 0

Pinta la barra actual de color azul manteniendo el grosor original.

PaintCandlestick

Descripción:

Mediante esta opción, es posible pintar velas con los valores deseados para apertura, máxima, mínima, cierre..

Sintaxis:
.PaintCandlestick(Open, High, Low, Close, Color, LineNumber, Width, nBars)

Parámetros:

Nombre	Defecto	Descripción
Open	-	Apertura de la barra. Permite utilizar una función o variable.
High	-	Máxima de la barra. Permite utilizar una función o variable.
Low	-	Mínima de la barra. Permite utilizar una función o variable.
Close	-	Cierre de la barra. Permite utilizar una función o variable.
Color	-	Color que se utilizará para pintar las barras. Se utiliza la función RGB.

LineNumber	-	Especifica el número de orden de pintar estudio. Esto es para estudios que utilizan diferentes órdenes de pintar (p.e si queremos incluir en un estudio una orden de pintar tipo PintarBarra (.PaintBar) y otra del tipo Pintar Línea (.PaintSeries), debemos poner a la primera orden, número de línea 0 y a la segunda, número de línea 1.
Width	1	Grosor que se utilizará para pintar la vela.
nBarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.

Ejemplo:

.PaintCandlestick .Open, .High, .Low, .Close, RGB(32,151,25), 0, 1, 0

Pinta sobre la barra actual una vela en tono verde manteniendo el grosor original.

PaintMaxMin

Descripción:

Esta función es similar a PaintBar/PaintCandlestick. La diferencia es que en ésta solo podremos establecer valores para la máxima y para la mínima de la barra que deseamos pintar.

Sintaxis:
.PaintMaxMin(Top, Bottom, Color, LineNumber, Width, nBars)

Parámetros:

Nombre	Defecto	Descripción
Top	-	Máxima de la barra que vamos a pintar. Permite usar una función o variable.
Bottom	-	Mínima de la barra. Permite utilizar una función o variable.
Color	-	Color que se utilizará para pintar las barras. Se utiliza la función RGB.
LineNumber	-	Especifica el número de orden de pintar estudio. Esto es para estudios que utilizan diferentes órdenes de pintar (p.e si queremos incluir en un estudio una orden de pintar tipo PintarBarra (.PaintBar) y otra del tipo Pintar Línea (.PaintSeries), debemos poner a la primera orden, número de línea 0 y a la segunda, número de línea 1.
Width	1	Grosor que se utilizará para pintar.
nBarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.

Ejemplo:

.PaintMaxMin .Low, .Low, RGB(0, 0, 255), 0, 8, 0

Pinta en el minimo de la barra un círculo (grosor 8) de color azul.

PaintSeries

Descripción:

Esta función se utiliza para pintar líneas de datos en pantalla. Esta tarea también se puede realizar haciendo un indicador, pero en el supuesto de que queramos mezclar líneas con opciones de pintar barras, o pintar figuras, podemos optar por crear un estudio que incluya todos estos tipos. Hay que recordar aquí que los estudios no pueden ser utilizados en otro tipo de estrategia como sistemas, indicadores etc. Por tanto, si deseamos usar después la línea de datos debemos crear un indicador.

Sintaxis:
.PaintSeries(Price, Color, LineNumber, Width, nBars)

Parámetros:

Nombre	Defecto	Descripción
Price	-	En este parámetro se indica el valor que deseamos que tenga la línea en la barra actual. Cualquier valor numérico o una variable de tipo numérico son aceptados en este parámetro.
Color	-	Color que se utilizará para pintar. Se utiliza la función RGB.
LineNumber	-	Especifica el número de orden de pintar estudio. Esto es para estudios que utilizan diferentes órdenes de pintar (p.e si queremos incluir en un estudio una orden de pintar tipo PintarBarra (.PaintBar) y otra del tipo Pintar Línea (.PaintSeries), debemos poner a la primera orden, número de línea 0 y a la segunda, número de línea 1.
Width	1	Grosor que se utilizará para pintar la vela.
nBarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.

Ejemplo:

.PaintSeries (.High + .Low) / 2, RGB(255, 0, 0), 0, 1, 0

Dibuja una línea de color rojo que va representando el punto medio de la barra.

PercentProfitable

Descripción:
Devuelve el ratio de fiabilidad. Este valor irá cambiando a medida que se generen nuevas barras y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.PercentProfitable

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

Queremos saber en un momento determinado, el ratio de fiabilidad de nuestro sistema. Para esto, previamente definiaríamos una variable:

Dim Fiabilidad as double

A ésta, en un momento dado, se le asignará el valor de la propiedad .PercentProfitable:

Fiabilidad =.PercentProfitable

ProfitFactor

Descripción:
Devuelve el ratio **Factor de ganancia**. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.ProfitFactor(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

Queremos saber en un momento determinado, el factor de ganancia (porcentual) de nuestro sistema. Para esto, previamente definiaríamos una variable:

Dim FGanancia as double

A ésta, en un momento dado, se le asignará el valor de esta función:

FGanancia =.ProfitFactor(porcentual)

PRR

Descripción:
Devuelve el ratio **Factor de ganancia ajustada**. Este valor irá cambiando a medida que se generen nuevas barras, y su valor dependerá de la barra en la que se haga la llamada a dicha propiedad.

Sintaxis:
.PRR(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

Queremos saber en un momento determinado, el factor de ganancia ajustada(en puntos) de nuestro sistema. Para esto, previamente definiaríamos una variable:

Dim FGanAjus as double

A ésta, en un momento dado, se le asignará el valor de esta función:

FGanAjus =.PRR(ByPoints)

RegressionAngle

Descripción:
Devuelve el valor del ángulo que forma con la horizontal la recta de regresión formada por los precios de cierre situados entre las barras **StartBar** y **EndBar**.

Sintaxis:
.RegressionAngle(StarBar, EndBar, Data)

Parámetros:

Nombre	Defecto	Descripción
StarBar	-	Barra de inicio de la recta de regresión.
EndBar	-	Barra de inicio de la recta de regresión.
Data	-	Serie de datos sobre la que se extrae la información. Puede ser un activo o un indicador.

RegressionSlope

Descripción:
Devuelve el valor de la pendiente de la recta de regresión formada por los precios de cierre situados entre las barras **StartBar** y **EndBar**.

Sintaxis:
.RegressionSlope(StarBar, EndBar, Data)

Parámetros:

Nombre	Defecto	Descripción
StarBar	-	Barra de inicio de la recta de regresión.
EndBar	-	Barra de inicio de la recta de regresión.
Data	-	Serie de datos sobre la que se extrae la información. Puede ser un activo o un indicador.

ReleaseDataIdentifier - RDI

Descripción:
Permite liberar un **DataIndentifier** que haya sido llamado previamente mediante el método GetSymbolIndentifier.

Sintaxis:
.ReleaseDataIdentifier(Identifier)

También se puede utilizar el modo abreviado **RDI**.

.RDI(Identfier)

Parámetros:

Nombre	Defecto	Descripción
Identifier	Data	Data que deseamos liberar.

Ejemplo:

Primero, creamos un data identifier que vamos a usar para extraer el minimo movimiento del vencimiento FDAXZ0 (contrato de diciembre-2010 del Futuro Dax)

Dim NuevoData as DataIdentifer

NuevoData = .GetSymbolIdentifier("010015FDAXZ0", 1, crDias, CDate("01/01/2010"), CDate("01/01/2036"))

Luego extraemos el minimo movimiento que vamos a usar:

Dim pip as Double

pip = .GetSymbolInfo(SbiMinMov, NuevoData)

Por último, como el nuevodata está ocupando espacio en memoria, lo liberamos pues no vamos a usarlo más. Para ello usamos la función ReleaseDataIdentifier

.ReleaseDataIdentifier (NuevoData)

Sell

Descripción:

Es utilizada indistintamente para venta de futuros y venta a crédito de acciones. Es importante tener en cuenta que esta función se utiliza para abrir posiciones de venta, no para cerrar las de compra solamente. Por tanto, si deseamos cerrar una compra sin abrir una posición de venta a crédito, debemos usar la función ExitLong.

Sintaxis:

.Sell(Type, Contracts, Price, Label)

Parámetros:

Nombre	Defecto	Descripción
Type	AtClose	Tipo de orden que se desea lanzar (AtClose, AtMarket, AtLimit, AtStop).
Contracts	1	Número de contratos/acciones. Las especificaciones numéricas sobre contratos pueden ser sustituidas por variables o por cualquier función definida previamente.
Price	-	Precio de venta. Sólo se indica este parámetro en las órdenes del tipo AtStop y AtLimit. El valor puede expresarse mediante un número, una variable, una función, o bien una mezcla de valor y función.
Label	-	Etiqueta de la orden en formato texto.

Ejemplo:

.Sell(AtStop, 1, .Close-10, "V1")

Lanza una orden de venta en stop (un contrato), siendo el precio del stop el cierre de la barra menos 10 puntos, y la etiqueta identificativa de la orden "V1".

SetBackGroundColor

Descripción:
Pinta el fondo de la ventana, de una barra determinada, en el color indicado.

Sintaxis:
.SetBackGrounColor (BarsAgo, Color)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Color	-	Color en el que se debe pintar el fondo de la ventana en la barra indicada (BarsAgo). Utilización de la función **RGB.**

Ejemplo:

.SetBackGroundColor(1, (RGB 255,0,0)) Pintará el fondo de la ventana (en la barra anterior) en color rojo.

SetBarColor

Descripción:
Asigna a la barra indicada, de una línea concreta del indicador, un color determinado.

Sintaxis:
.SetBarColor (BarsAgo, Line, Color)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line		Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
Color	-	Color en el que se debe pintar la barra indicada. Utilización de la función **RGB**.

Ejemplo:

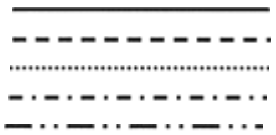
.SetBarColor(0,1, RGB(255,0,0)) Pintará de color rojo la barra actual de la línea 1.

SetBarProperties

Descripción:
Asigna a la barra indicada, de una línea concreta del indicador, el color, grosor y tipo de de línea y representación indicados en el resto de parámetros.

Sintaxis:
.SetBarProperties (BarsAgo, Line, Color, Width, Style, Representation)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line	-	Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
Color	-	Color en el que se debe pintar la barra indicada. Función **RGB**.
Width	-	Indica el grosor que se aplica a la barra (1,2,..).
Style	-	Estilo que se utilizará para la representación: **lsSolid** línea continua **lsDash** línea discontinua **lsDot** línea punteada **lsDashdot** línea discontinua con punto **lsDashdotdot** línea discontinua con 2 puntos 
Representation	-	Tipo de representación que se utilizará: **irBars** barras **irCandlestic** velas **irDottedLine** línea punteada **irFilledHistogram** histograma relleno **irHistogram** histograma **irLineal** lineal **irParabolic** parabólico **irVolume** volumen

Ejemplo:

.SetBarProperties(0,1, RGB(255,0,0),2,lsSolid,irLineal)

Pintará en color rojo la línea 1 de la barra actual (en formato de línea continua y grosor 2).

SetBarRepresentation

Descripción:
Esta función se encarga de establecer el tipo de representación a una barra especificada.

Sintaxis:
.SetBarRepresentation(BarsAgo, Line, Representation)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line		Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
Representation	-	Tipo de representación que se utilizará (pueden consultarse los diferentes tipos en la función SetBarProperties).

Ejemplo:

.SetBarRepresentation(0,1,irLineal)

La barra actual de la línea 1 del indicador, se representará en formato lineal.

SetBarStyle

Descripción:
Esta función se encarga de establecer el estilo a una barra especificada.

Sintaxis:
.SetBarStyle(BarsAgo, Line, Style)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line		Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
Representation	-	Estilo de línea que se utilizará (pueden consultarse los diferentes tipos en la función SetBarProperties).

Ejemplo:

.SetBarStyle(0,1,irSolid)

La barra actual de la línea 1 del indicador, se representará en formato de línea continua.

SetBarWidth

Descripción:
Asigna a la barra indicada, de una línea determinada, el grosor deseado.

Sintaxis:
.SetBarWidth(BarsAgo, Line, Width)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line		Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
Width	-	Grosor en el que se representará la barra.

Ejemplo:

.SetBarWidth(0,1, 2) La barra actual de la línea 1 del indicador, se representará con grosor 2.

SetHistogramBand

Descripción:
Asigna a la barra indicada, de una línea de datos concreta, el valor que tiene la línea del histograma, es decir, el valor hasta el cual se pinta el histograma (si la representación utilizada es histograma). Esta función está estrictamente asociada al uso de la propiedad SetBarRepresentation.

Sintaxis:
.SetHistogramBand(BarsAgo, Line, BandLine)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line		Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
BandLine	-	Línea de datos que servirá de referencia para dibujar el histograma.

Ejemplo:

Supongamos que utilizamos la función .SetBarRepresentation(0,1,irHistogram) . El indicador necesitará un valor de referencia sobre el que oscilar para generar el histograma.

En este caso:

.SetIndicatorValue (.Close – Close(1),1,0) Con la línea 1 se representará la diferencia entre cierres
.SetIndicatorValue(0,2,0) Con la línea 2 se representará el valor 0.

A continuación utilizando la función SetBarRepresentation, se indicará que la línea 1 se pintará por defecto con un histograma:

.SetBarRepresentation (0,1,irHistogram)

Y por último se indicará la línea de referencia sobre la que oscilar:

.SetHistogramBand (1,2)

De esta forma la línea 1 usará como línea de banda (línea de referencia) la línea 2.

SetIndicatorPos

Descripción:
Indica a la barra indicada, de una línea determinada, una posición concreta.

Sintaxis:
.SetIndicatorPos(BarsAgo, Line, IndicatorPosition)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
Line	-	Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
IndicatorPosition	-	La tendencia puede ser **ipBull** (alcista), **ipBear** (bajista) o **ipNeutral** (neutra)

Ejemplo:

.SetIndicatorPos(3,2,ipNeutral) Asigna a la barra 3 hacia atrás, de la línea 2, la posición neutra.

SetIndicatorValue

Descripción:
Lo que se hace es asignar un valor en una barra indicada. A estae valor se le asignará una determinada tendencia.

Sintaxis:
.SetIndicatorPos(Value, Line, BarsAgo, IndicatorPosition)

Parámetros:

Nombre	Defecto	Descripción
Value	-	Variable de tipo doublé.
Line	1	Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
BarsAgo	-	Número de barras hacia atrás. El valor 0 hace referencia a la barra actual.
IndicatorPosition	-	La tendencia puede ser ipBull (alcista), ipBear bajista o ipNeutral neutra.

Ejemplo:

Supongamos que definimos una variable de tipo Double:

```
Dim buffer3 as Double
```

A esta le asignamos el valor que retorna el siguiente cálculo:

```
buffer3 = .High(0) - .Low(0) / .High(0) * 100
```

A continuación, podemos utilizar la función .SetIndicatorValue para pintar en cada barra el valor calculado para dicha variable:

```
If buffer3 > 50 then
        .SetIndicatorValue buffer3, 1,0,ipBull
    Else
        .SetIndicatorValue buffer3, 1,0,ipBear
End If
```

La diferencia está en que dependiendo de cómo sea este valor, se le asignará al indicador tendencia alcista o bajista.

SetLineName

Descripción:
Asigna un nombre a la línea indicada.

Sintaxis:
.SetLineName(Line, LineName)

Parámetros:

Nombre	Defecto	Descripción
Line	-	Identifica la línea de datos a la que pertenece la barra sobre la que se establecerá la propiedad.
LineName	-	El nombre que se le asigna a la línea correspondiente.

Ejemplo:

.SetLineName 2, "DT" Asigna el nombre "DT" a la línea 2.

SetWndBackGroundColor

Descripción:
Asigna el color de fondo de la ventana de un indicador.

Sintaxis:
.SetWndBackGrounColor()

Parámetros:

Nombre	Defecto	Descripción
Color	-	Utilización de la función **RGB** para indicar el color de fondo de la ventana.

Ejemplo:

.SetWndBackgroundColor(RGB(255,0,0)) Asigna el color rojo a la ventana del indicador.

ShouldTerminated

Descripción:
Se utiliza para interrumpir los cálculos de una estrategia. **ShouldTerminated** es una variable booleana inicializada con el valor False, de forma que para la interrupción de los cálculos, se tiene que asignar el valor True a la misma.
Resulta útil cuando se trabaja con históricos largos y se desea detener el cálculo de un sistema cuando determinadas circunstancias se produzcan.

Sintaxis:
.ShouldTerminated = True

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

```
If .CurrentBar = 1000 And  .NetProfit < 100 Then
.ShouldTerminate = True
End If
```

En este caso, los cálculos se detendrán cuando hayan transcurrido 1000 barras de cálculo y la ganancia neta sea inferior a 100 euros.

Slope

Descripción:
Devuelve el valor de la pendiente de la recta de regresión, formada por los precios indicados, situados entre las barras **StartBar** y **EndBar**.

Sintaxis:
.Slope(StarBar. EndBar, StarPrice, EndPrice, Data)

Parámetros:

Nombre	Defecto	Descripción
StarBar	-	Barra de inicio de la recta de regresión.
EndBar	-	Barra de inicio de la recta de regresión.
StarPrice	-	Precio de inicio de la recta.
EndPrice	-	Precio de fin de la recta.
Data	-	Serie de datos sobre la que se extrae la información. Puede ser un activo o un indicador.

StandardDeviation

Descripción:
Retorna la desviación estándar.

Sintaxis:
.StandardDeviation(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual** (porcentaje).

Ejemplo:

Dim DE as double

DE =.StandardDeviation(ByPoints)

En el momento en que se hace uso de esta propiedad, se le asignará a la variable DE, la desviación standard en puntos.

StarBar

Descripción:
Permite especificar a partir de qué barra se va a empezar a calcular el sistema.

Sintaxis:
.StarBar = (Número de barras)

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Ejemplo:

.StarBar=25 El sistema comenzará a calcularse a partir de la barra 25 del histórico.

Time

Descripción
Devuelve el valor de campo Tiempo (hora) de una barra. La hora de una barra viene dada por la hora de finalización del periodo temporal que resume la barra. La hora de una barra es considerada en formato militar (HHMM), es decir, si la hora de una barra es 5:35pm, en visual chart se considera como el valor numérico 17:35h.

Sintaxis:
.Time(BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barras hacia atrás. Por defecto hace referencia a la barra actual.
Identifier	Data	Fuente de datos sobre la que se utiliza la función. Si existen más gráficos insertados, estarán codificados como Data1, Data2, etc.

Ejemplo:

.Time(3, Data1) Retorna la hora en formato militar (HHMM) de hace 3 barras en la fuente de datos Data1.

TimeEx

Descripción
Retorna la fecha de la barra de referencia en formato fecha (DD/MM/AAAA HH:MM:SS).

Sintaxis:
.TimeEx(BarsAgo, Identifier, TickIndex)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Barra de la que se desea extraer la fecha. El valor por defecto hace referencia a la barra actual.
Identifier	Data	Fuente de datos sobre la que se utiliza la función. Si existen más gráficos insertados, estarán codificados como Data1, Data2, etc.

TickIndex es un parámetro de Salida. Cuando se opera sobre un gráfico de ticks, puede ocurrir que algunos de éstos tengan la misma fecha. Este parámetro se rellena indicando la n-ésima posición de ticks referente a la misma fecha.

Ejemplo:

.TimeExe(1, Data2)

Retorna la fecha (DD/MM/AAAA HH:MM:SS) de la barra anterior de la serie Data2.

TimeToMinutes

Descripción:
Retorna el número de minutos transcurridos desde las 00:00 horas.

Sintaxis:
.TimeToMinutes(Time)

Parámetros:

Nombre	Defecto	Descripción
Time	-	Hora en formato militar (HHMM)

Ejemplo:

.TimeToMinutes(1735)

Retorna 1055 minutos.

This

Descripción:
Propiedad con la cual se hace referencia al propio dato sobre el que se está trabajando. Se utilizar para asignarlo a cualquier elemento que solicite como parámetro de entrada un **DataIdentifier** (por ejemplo, la función .GetIndicatorIdentifier(Code_Ind, .This, Period, PriceClose, etc.)

Sintaxis:
.This

Parámetros:

Nombre	Defecto	Descripción
-	-	-

Volume

Descripción:
Devuelve el valor del volumen (acciones/contratos negociados) de una barra.

Sintaxis:
.Volume (BarsAgo, Identifier)

Parámetros:

Nombre	Defecto	Descripción
BarsAgo	0	Número de barra. El valor por defecto hace referencia a la barra actual. En este parámetro podemos indicar cualquier valor numérico contenido en una variable o teclear directamente el valor. Puede especificarse como función sustituyendo el valor numérico.
Identifier	Data	Serie de datos de la que se obtiene el volumen de la barra especificada. Si en la ventana hay más gráficos insertados, se codificarán como Data1, Data2, etc.

Ejemplo:

.Volume(15, Data2)

Retorna el volumen negociado 15 barras atrás, en la serie de datos Data2.

WorstSeries

Descripción:
Devuelve la peor serie de negocios según sus resultados.

Sintaxis:
.Worstseries(Show)

Parámetros:

Nombre	Defecto	Descripción
Show	Bypoints	Permite indicar el formato en el que se mostrará la información, **ByPoints** (en puntos), o **Porcentual**(porcentaje).

VisualChart

Real Time Financial Information & Trading Software

España: 902 34 11 34 • USA: 866 497 5537 • UK: 020 7153 8936 - 020 7153 8937
020 7153 8938 - 020 7153 8939 • Deutschland: 01805 35 40 30 • Österreich: 0820
40 0038 • Schweiz: 0 800 56 10 15 France: 0 821 23 00 99 • Belgique: 078 79 01 79
Suisse: 0 800 56 10 15 • Canada: 800 210 7889 • Italy: 848 390 458

Visualchart Group
C/ Freniche 4, Cp: 04009
Almería • ESPAÑA
www.visualchart.com