|  |  |
| --- | --- |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [ISeries<T>](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) >  **MaximumBarsLookBack** | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/isvaliddatapointat.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/onbarupdate.htm) |

**Definition**

Determines memory performance of custom [Series<T>](https://ninjatrader.com/es/support/helpGuides/nt8/seriest.htm) objects (such as Series<double>, Series<long>, etc.).  When using **MaximumBarsLookBack.TwoHundredFiftySix**, only the last 256 values of the series object will be stored in memory and be accessible for reference. This results in significant memory savings when using multiple series objects. In the rare case should you need older values you can use **MaximumBarsLookBack.Infinite** to allow full access of the series.

|  |
| --- |
| **Notes**:  •ISeries<T> objects that hold bar data (such as Close, High, Volume, Time, etc) always use **MaximumBarsLookBack.Infinite** which ensures all data points are always accessible during the lifetime of your NinjaScript indicator or strategy.  •Series<double> objects that hold indicator [plot values](https://ninjatrader.com/es/support/helpGuides/nt8/values.htm) always use **MaximumBarsLookBack.Infinite** which ensures that charts always display the entire indicator's calculated values. |

**Property Value**

A **MaximumBarsLookBack** enum value. Default value is **MaximumBarsLookBack.TwoHundredFiftySix**

Possible values are:

|  |  |
| --- | --- |
| MaximumBarsLookBack.TwoHundredFiftySix | Only the last 256 values of the series object will be stored in memory and accessible for reference (improves memory performance) |
| MaximumBarsLookBack.Infinite | Allow full access of the series, but you will then not be able to utilize the benefits of memory optimization |

|  |
| --- |
| **Tip**:  A **MaximumBarsLookBack.TwoHundredFiftySix** series works as a circular ring buffer, which will "loop" when the series reaches full capacity.  Specifically, once there are 256 entries in the series, new data added to the series overwrite the oldest data. |

**Syntax**

MaximumBarsLookBack

**Examples**

| ns **Setting all custom series to use the default MaximumBarsLookBack** |
| --- |
| Series<double> myDoubleSeries = null; Series<string> myStringSeries = null;   protected override void OnStateChange() {   if (State == State.SetDefaults)   {     Name = "Example Indicator";     // Store all series values instead of only the last 256 values     MaximumBarsLookBack = MaximumBarsLookBack.Infinite;   }   else if (State == State.DataLoaded)   {     // The custom Series<t> below are all constructed using only the NinjaScriptBase object (i.e., "this")     // therefore, the Series<T> MaximumBarsLookBack is taken from the NinjaScript's configured MaximumBarsLookBack property     myDoubleSeries = new Series<double>(this);     myStringSeries = new Series<string>(this);   } } |

| ns **Optimizing custom series to use unique MaximumBarsLookBack behavior** | |
| --- | --- |
| Series<double> myDoubleSeries = null; Series<string> myStringSeries = null;   protected override void OnStateChange() {   if (State == State.SetDefaults)   {     Name = "Example Indicator";   }   else if (State == State.DataLoaded)   {     // The custom Series<t> below are constructed using MaximumBarsLookBack parameter     // therefore, each Series<t> will use their uniquely specified MaximumBarsLookBack properites     myDoubleSeries = new Series<double>(this, MaximumBarsLookBack.Infinite); // stores all values     myStringSeries = new Series<string>(this, MaximumBarsLookBack.TwoHundredFiftySix); // only the last 256 values (better performance)   } } | |
| **Navigation:**  »No topics above this level«  **ForcePlotsMaximumBarsLookBackInfinite** | | [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/sharpdx_directwrite_textlayout.htm) |

**Definition**

Forces all [Series<T>](https://ninjatrader.com/es/support/helpGuides/nt8/seriest.htm) objects including plot series objects to hold the maximum amount of bars possible. This property supercedes the UI property of "Maximum bars look back".

|  |
| --- |
| **Warning**:  Setting this option to **true** uses more memory resources.  Please see [MaximumBarsLookBack](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm) for more information. |

**Method Return Value**

A bool value when **true**, any ISeries<T> interfaces contained in the scripts MaximumBarsLookBack property will be set to MaximumBarsLookBack.Infinite; otherwise will use the default properties configured.  Default is set to **false**.

**Syntax**

ForcePlotsMaximumBarsLookBackInfinite

**Examples**

| ns | |
| --- | --- |
| protected override void OnStateChange() {         if (State == State.SetDefaults)   {     Name = "Examples Indicator";       Description = @"An indicator used to demonstrate various NinjaScript methods and properties";     ForcePlotsMaximumBarsLookBackInfinite = true;   } } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [ISeries<T>](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) >  **Series<T>** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/reset.htm) |

**Definition**

A Series<T> is a special generic type of data structure that can be constructed with any chosen data type and holds a series of values equal to the same number of elements as bars in a chart. If you have 200 bars loaded in your chart with a moving average plotted, the moving average itself holds a Series<double> object with 200 historical values of data, one for each bar. Series<double> objects can be used as input data for all [indicator methods](https://ninjatrader.com/es/support/helpGuides/nt8/indicators.htm). The Series<T> class implements the ISeries<T> interface.

|  |
| --- |
| **Note**:  By default NinjaTrader limits the number of values stored for Series<T> objects to 256 from the current bar being processed. This drastically improves memory performance by not holding onto old values that are generally not needed. Should you need more values than the last 256 please be sure to create the Series<T> object so that it stores all values instead through the use of the [MaximumBarsLookBack](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm) property. |

**Parameters**

|  |  |
| --- | --- |
| ninjaScriptBase | The NinjaScript object used to create the Series |
| bars | The [Bars](https://ninjatrader.com/es/support/helpGuides/nt8/bars.htm) object used to create the Series |
| maximumBarsLookBack | A [MaximumBarsLookBack](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm) value used for memory performance |

**Methods and Properties**

|  |  |
| --- | --- |
| [GetValueAt()](https://ninjatrader.com/es/support/helpGuides/nt8/getvalueat.htm) | Returns the underlying input value at a specified bar index value. |
| [IsValidDataPoint()](https://ninjatrader.com/es/support/helpGuides/nt8/isvaliddatapoint.htm) | Determines if the specified input is set at a barsAgo value relative to the current bar. |
| [Reset()](https://ninjatrader.com/es/support/helpGuides/nt8/reset.htm) | Resets the internal marker which is used for [IsValidDataPoint()](https://ninjatrader.com/es/support/helpGuides/nt8/isvaliddatapoint.htm) back to false. |
| [Count](https://ninjatrader.com/es/support/helpGuides/nt8/count.htm) | The total number of bars or data points. |

**Creating Series<T> Objects**

When creating custom indicators, Series<double> objects are automatically created for you by calling the [AddPlot()](https://ninjatrader.com/es/support/helpGuides/nt8/addplot.htm) method and can be subsequently referenced by the [Value](https://ninjatrader.com/es/support/helpGuides/nt8/value.htm) and/or [Values](https://ninjatrader.com/es/support/helpGuides/nt8/values.htm) property. However, you may have a requirement to create a Series<T> object to store values that are part of an overall indicator value calculation. This can be done within a custom indicator or strategy.

|  |
| --- |
| **Note**:  Custom Series<T> objects will hold the number of values specified by the [MaximumBarsLookBack](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm) property when the custom series object is instantiated. |

To create a Series<T> object:

1.Determine the data type of the Series<T> object you wish to create. This could be double, bool, int, string or any other object type you want.

2.Define a variable of type Series<T> that will hold a Series<T> object. This example will create "myDoubleSeries" as a Series<double>.

3.In the [OnStateChange()](https://ninjatrader.com/es/support/helpGuides/nt8/onstatechange.htm) method, in the State.DataLoaded create a new Series<T> object and assign it to the "myDoubleSeries" variable

| ns |
| --- |
| private Series<double> myDoubleSeries; // Define a Series<T> variable. In this instance we want it                                       // as a double so we created a Series<double> variable.   // Create a Series object and assign it to the variable protected override void OnStateChange() {     if (State == State.DataLoaded)     {         // "this" refers to the NinjaScript object itself. This syncs the Series object to historical data bars         // MaximumBarsLookBack determines how many values the Series<double> will have access to         myDoubleSeries = new Series<double>(this, MaximumBarsLookBack.Infinite);     } } |

|  |
| --- |
| **Tip***:*Series<T> objects can be used on supplementary series in a multi-time frame and instrument strategy. Please see our [support forum](http://www.ninjatrader.com/support/forum/showthread.php?t=3572" \t "_blank) NinjaScript reference samples section for further information. |

**Setting Values**

You can set the value for the current bar being evaluated by choosing a "barsAgo" value of "0" or, for historical bars, by choosing a "barsAgo" value that represents the number of bars ago that you want the value to be stored at.

| ns **Setting Series<T> values** |
| --- |
| protected override void OnBarUpdate() {     myDoubleSeries[0] = Close[0]; } |

|  |
| --- |
| **Note**:  The "barsAgo" value is only guaranteed to be in sync with the recent current bar during core data event methods, such as OnBarUpdate(), OnMarketUpdate(), and during strategy related order events such as OnOrderUpdate(), OnExecutionUpdate(), OnPositionUpdate().  For scenarios where you may need to set a value outside of a core data/order event, such as OnRender() or a custom event, you must first synchronize the "barsAgo" pointer via the [TriggerCustomEvent()](https://ninjatrader.com/es/support/helpGuides/nt8/triggercustomevent.htm) method. |

**Checking for Valid Values**  
It is possible that you may use a Series<T> object but decide not to set a value for a specific bar. However, you should *not* try to access a Series<T>value that has not been set. Internally, a dummy value does exists, but you want to check to see if it was a valid value that you set before trying to access it for use in your calculations.  Please see [IsValidDataPoint()](https://ninjatrader.com/es/support/helpGuides/nt8/isvaliddatapoint.htm) more information.

|  |
| --- |
| **Warning**:  Calling IsValidDataPoint() will only work a [MaximumBarsLookBackInfinite](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm) series.  Attempting to check IsValidDataPoint() MaximumBarsLookBack256 series throw an error.  Please check the Log tab of the Control Center |

**Getting Values**  
You can access Series<T> object values using the syntax Series<T>[int *barsAgo*] where barsAgo represents the data value *n* (number of bars ago).

| ns **Accessing Series object values** |
| --- |
| protected override void OnBarUpdate() {   // Prints the current and last bar value   Print("The values are " + myDoubleSeries[0] + " " + myDoubleSeries[1]); } |

Alternatively, you can access a value at an absolute bar index using the [GetValueAt()](https://ninjatrader.com/es/support/helpGuides/nt8/getvalueat.htm) method.

|  |
| --- |
| **Note**:  In most cases, you will access the historical price series using a core data event handler such as OnBarUpdate().  For more advance developers, you may find situations where you wish to access historical price series outside of the core data event methods, such as OnRender(), or your own custom event.  In these advanced scenarios, you may run into situations where the "barsAgo" pointer is not in sync with the current bar, and may result in errors when trying to obtain this information.  In those cases, please use the Bars.Get...() methods with the absolute bar index, e.g., [GetValueAt()](https://ninjatrader.com/es/support/helpGuides/nt8/getvalueat.htm). |

**Methods that Accept ISeries<T> as Arguments**  
All [indicator methods](https://ninjatrader.com/es/support/helpGuides/nt8/indicators.htm) accept ISeries<double> objects as arguments. Carrying from the prior examples, let's print out the 10 period simple moving average of range.

| ns **Using a custom Series object as indicator input** | |
| --- | --- |
| protected override void OnBarUpdate() {   // Calculate the range of the current bar and set the value     myDoubleSeries[0] = (High[0] - Low[0]);       // Print the current 10 period SMA of range     Print("Value is " + SMA(myDoubleSeries, 10)[0]);         } | |
| **Navigation:**  »No topics above this level«  **8.0.0.12 (Beta) Release Notes** | | [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/sharpdx_directwrite_textlayout.htm) |

**Release Date**

July 11, 2016

**Code Breaking Changes**

**Compile Errors**

•The NinjaScript Strategy "**AccountSize**" concept was removed due to limited functionality.  This change would also impact strategies which were coded to use [SetOrderQuantity.ByAccountSize](https://ninjatrader.com/es/support/helpGuides/nt8/setorderquantity.htm).  Please use your own variables to set quantities by account size. **Tip**:  NinjaTrader 8 can read real-world account values for live trading purposes through the Account class, e.g., Account.Get(AccountItem.CashValue, Currency.UsDollar) provides a method for returning a double value representing the current cash value of the account.

**Implementation changes**

•The concept used to force plot series objects through **ForcePlotsMaximumBarsLookBackInfinite** was disabled and tagged as obsolete.  You should set any indicator/strategy plots to [MaximumBarsLookBack.Infinite](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm) during its construction.

•It is recommended that any strategies generated with the **'Strategy Builder'** tool via Control Center > New > Strategy Builder be removed and recreated after installing B12. If you had invested significant effort in a **Strategy Builder** generated strategy in B11 which you would like to continue to use, please contact us [platformsupport@ninjatrader.com](mailto:platformsupport@ninjatrader.com)

**Notes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Status** | **Issue #** | **Category** | **Comments** |
| Added | 9127 | NinjaScript | Added XML Comments to reflect Supported NinjaScript and NinjaTrader Core Methods (ongoing) |
| Added | 9897 | NinjaScript Editor | Added Multi-Select to the NS Editor References Win File Dialog |
| Added | 9908 | Rithmic | Added additional Rithmic Adapter Account Items |
| Added | 9933 | Window Linking | SFT-162 - Added more link button colors (up to 11) |
| Changed | 10066 | Account Data | Change Position Display option to "Show gross realized PnL when flat." |
| Changed | 10076 | FXCM | Reverted NT8 FXCM Forex volume to NT7 implementation |
| Added | 9839 | Interactive Brokers | Added TWS File picker to allow users to the specific location of TWS/Gateway executable which can vary in some scenarios. [See the NinjaTrader 8 Interactive Brokers connection guide](http://ninjatrader.com/ConnectionGuides/Interactive-Brokers-Connection-Guide" \t "_blank). |
| Changed | 9976 | NinjaScript | Disabled the .ForcePlotsMaximumBarsLookBackInfinite concept |
| Fixed | 10022 | NinjaScript Editor | Delimiter matching was hard to see |
| Fixed | 9946 | Orders | Trace Order reported old price after order change |
| Changed | 9875 | Strategy | Removed of "ByAccountSize" Strategy Order Option |
| Fixed | 9887 | Strategy Builder | Strategy Builder additional data window did not show instrument |
| Fixed | 9996 | UI | Removed UI Formatting of Market Data Volume (real-time and historical bar data) |
| Fixed | 9879 | Account Data | Account Data Window default tab order did not match order of Add Tab Menu Sub Items |
| Fixed | 9992 | Account Data | Excess Position Margin Column width was not saved/restored correctly |
| Fixed | 9441 | Adapter | Kinetick adapter was not properly resolving market data settlement time updates |
| Fixed | 9977 | Alerts | Alerts Log Window "priority" was not working as expected with some localization |
| Fixed | 9876 | Attach Order To Indicator, Chart | IsSuspendedWhileInactive did not always re-suspend after manually attached to orders |
| Fixed | 9906 | Bars | Range Bars were built differently using custom data range |
| Fixed | 9889 | Bars | Renko Bars produced undesired ticks bars at EOD |
| Fixed | 9854 | Bars | Break EOD could cause issues with bar pool series when requested after session EOD |
| Fixed | 9803 | Bars, Chart | Bar spacing issues identified when scrolled into future which resulted in the "Automatically choose time-based series for x-axis (non-equidistant bar spacing)" concept to be removed.  We will revisit this feature in the future. |
| Fixed | 9859 | Bars, NinjaScript | BarsSinceNewTradingDay was not operating as expected |
| Fixed | 9985 | Bars, Playback | Exception in multi-series Playback |
| Fixed | 9856 | Chart | Data Box did not update values as you scroll |
| Fixed | 9843 | Chart | Drag and drop series to tab was not creating new series as expected |
| Fixed | 9934 | Chart | Unable to use Instrument Search from new Data Series window |
| Fixed | 9868 | Chart | Fixed Data Box volume formatting issues for FX |
| Fixed | 9858 | Chart | Gap in Chart and Chart Window at 150% DPI |
| Fixed | 10025 | Chart | Multi-Series chart bar width was incorrect |
| Fixed | 10078 | Chart | Crosshair time label rendering incorrect on left side |
| Fixed | 10058 | Chart | Price market incorrect in condition with days to load and session template |
| Fixed | 9988 | Chart Trader | Changing colors of the action, buy, and sell buttons did not save |
| Fixed | 9884 | Chart, Drawing | Draw objects in future would move when chart timeframe changed |
| Fixed | 9943 | Chart, DrawingTool | Mouse cursor was not reverting to normal when canceling the creation of a draw object by right-clicking |
| Fixed | 9964 | Chart, Indicator | Out of range exception was thrown using Horizontal  Line plot on Multi-Series chart |
| Fixed | 9837 | Chart, NinjaScript, Tick Replay | Tick Replay sometimes caused SynchronizationLockException when loading indicator |
| Fixed | 9831 | Chart, Sessions | Rollover date was missing on Default 24/7 with Break at EOD unchecked |
| Fixed | 9931 | Chart, Strategy Analyzer | Strategy Analyzer Data Box opened from trade review chart doesn't show trade execution |
| Fixed | 9929 | Chart, Strategy Analyzer | Aborting in Strategy Analyzer leaves disabled script on chart |
| Fixed | 10048 | Chart, Strategy Analyzer | Unhandled exception when switching to chart in Strategy Analyzer |
| Fixed | 9864 | Chart, Templates | Drawing tool option selection "Stay in Draw Mode" was not included when saving chart template |
| Fixed | 10069 | Code Wizard, Strategy Builder | Additional data option missed week and year type as choice |
| Fixed | 10012 | Commissions | Some commission values displayed with trailing 0's |
| Fixed | 9890 | Control Center | Restoring live executions were sorted reverse |
| Fixed | 9882 | Control Center, Database | Restoring executions were working differently from NT7 |
| Fixed | 9695 | Core | FXCM Adapter reported duplicate trades in a race condition |
| Fixed | 9894 | CQG | Invalid error occurred when trying to connect to CQG after failed CQG logon |
| Fixed | 9910 | Data Grids | Tri-State Sorting was not working in all grids |
| Fixed | 10075 | Data Grids | Account Tab Filter With Positions Was Not Updating as Account position changes |
| Fixed | 9936 | Database | Performance improvements during first ever application startup using default workspaces |
| Fixed | 10042 | Database, Instruments | Removed instrument in instrument manager would sometimes reappear |
| Fixed | 9932 | Database, Trade Performance | TradesPerformance showing incorrect Bars value for some trades |
| Fixed | 9939 | DrawingTool | Ruler was incorrectly rounding pip sizes |
| Fixed | 9945 | DrawingTool | Stay In Draw mode did not apply to Text tool |
| Fixed | 9970 | DrawingTool | Additional settings displayed when selecting drawing tools in the Configured list. |
| Fixed | 9963 | DrawingTool | Changing drawing tool EndTime into future leaves drawing tool drawn as it was |
| Fixed | 9895 | DrawingTool | Drawing tools when switching contract months do not reappear on the chart. |
| Fixed | 9983 | DrawingTool | Performance improvements on default text drawing tool |
| Fixed | 10035 | DrawingTool | Ruler exhibited localization issue |
| Fixed | 9738 | DrawingTool | Setting DrawObject ChartAnchor.BarsAgo did not change rendered object |
| Fixed | 10009 | DrawingTool | Drawing Tools lost auto scale as chart scrolled into future |
| Fixed | 10052 | DrawingTool | TrendChannel Parallel Line Start Anchor did not snap when using bar and price mode |
| Fixed | 9804 | DrawingTool | Trend Lines second line did not respect Snap Mode. |
| Fixed | 9925 | DrawingTool, NinjaScript | DrawObjects collection was not updating with global draw object until NinjaScript reloaded |
| Fixed | 10038 | DrawingTool, Strategy Analyzer | AddChartIndicator indicator was always rendering draw objects on primary panel in Strategy Analyzer |
| Fixed | 10049 | DrawingTool, Templates | PriceLevels reverted after re-compile in Fibonacci Drawing Tools |
| Fixed | 9962 | DrawingTool, Workspaces | Global lines inconsistently saved across workspaces using" workspace save as" |
| Fixed | 10006 | eSignal | Esignal was not processing real-time index data |
| Fixed | 9935 | FX Board | Unable to load ATM Strategy Template Error from FXBoard |
| Fixed | 9950 | FX Board | FXBoard Unrealized PnL field did not update until unrealized PnL changes |
| Fixed | 10067 | FXCM | FXCM order rejection scenario |
| Fixed | 10008 | Hot Key | Removing custom drawing tool code did not remove assigned hot key |
| Fixed | 9912 | Indicator | Bar timer did not function on Playback connection |
| Fixed | 9953 | Indicator | Indicator was plotting on the second bar on chart despite BarsRequiredToPlot = 0 |
| Fixed | 9965 | Indicator | @VolumeProfile.cs results were skewed towards buy volume using Tick Replay |
| Fixed | 10021 | Indicator, SuperDOM | SuperDOM Indicator Data series input changes using multi-time frame indicators |
| Fixed | 9869 | Interactive Brokers | Unable to receive data from the instrument on the SEHK exchange. |
| Fixed | 9961 | Interactive Brokers | Unexpected behavior with chart "Price Based On" was set to Bid/Ask |
| Fixed | 9893 | Interactive Brokers | Was not processing Index data due to no volume reported |
| Fixed | 9995 | Interactive Brokers | Interactive Brokers MarketDataType.DailyHigh/Low was not coming in |
| Fixed | 9994 | Kinetick | Adapter was processing volume of "1" for indexes instead of "0" |
| Fixed | 10034 | Licensing | VendorLicense did not always show all configured free trials |
| Fixed | 9820 | Licensing, NinjaScript | VendorLicense in AddOn prevented OnWindowCreated from working as expected |
| Fixed | 10040 | Market Replay, Time and Sales | Playback connection T&S Window did not scroll precisely to user actions while using Scroll Wheel |
| Fixed | 9898 | NinjaScript | Multi-series indicator was running into deadlocks |
| Fixed | 9857 | NinjaScript | Drawing objects were still visible to NinjaScript after changing instrument on the chart |
| Fixed | 9795 | NinjaScript | Errors on calling 'EventHandlerBarsUpdate' method from Strategy |
| Fixed | 9937 | NinjaScript | Remove NinjaScript assembly window needed vertical scrollbar |
| Fixed | 9955 | NinjaScript | User configured indicator input series changed after exception |
| Fixed | 9978 | NinjaScript | PlotBrushes were not working as expected with AddChartIndicator() |
| Fixed | 9823 | NinjaScript | Indicator plot did not match when using a custom bar type |
| Fixed | 10041 | NinjaScript | Exception calling Draw.Triangle() from AddChartIndicator() |
| Fixed | 10070 | NinjaScript | Blank string used for Name property resulted in crash |
| Fixed | 9928 | NinjaScript Editor | NinjaScript Editor F3 did not work if Find window was closed |
| Fixed | 9971 | NinjaScript Editor | NinjaScript Editor invalid char allowed as name |
| Fixed | 9865 | NinjaScript, Orders | Stop price checks for strategies with OrderFillResolution High had been executed in the wrong bars series |
| Fixed | 9956 | NinjaScript, Strategy | Strategy disabled by SetState was considered as enabled in Strategies tab when attempting to edit |
| Fixed | 10037 | NinjaScript, Strategy Analyzer | Adding indicator to backtest chart with AddChartIndicators caused indicator plots to disappear |
| Fixed | 9874 | NinjaScript, Strategy Builder | Strategy Wizard did not utilize secondary series for input correctly. **Note**:  This fixed resulted in change which preventing "Plot on chart" from working correctly, which is schedule to be fixed in the next beta release (# 10090) |
| Fixed | 9942 | Options | Changing Auto Close Position settings did not create warning if already connected |
| Fixed | 10051 | Orders | Exception on placing order to non-USD sim account created after connecting to provider |
| Fixed | 9915 | Playback | Unhanded exception on Playback disconnect |
| Fixed | 9907 | Playback | Tick Replay on Playback was frozen when using Go To |
| Fixed | 9947 | Playback | Empty warning pop-up when changing Playback start date |
| Fixed | 9944 | Playback | Playback did not "Go To" available date before current start  date. |
| Fixed | 9968 | Playback | Execution markers were not plotting correctly in Playback Connection |
| Fixed | 9975 | Playback | Minor issues with Playback behavior if no data was available |
| Fixed | 9899 | Playback | Playback data auto-replaying on chart when created after Playback connected |
| Fixed | 9989 | Playback | Strategy exception with multiple range series during Playback |
| Fixed | 9990 | Playback | Did not disconnect all the way when disconnected before loaded |
| Fixed | 9999 | Playback | "Error in real time market data handling" when running Playback |
| Fixed | 9986 | Playback | Gaps on dragging slider on Playback data |
| Fixed | 9979 | Strategy | Strategies grid changes to respect region denomination |
| Fixed | 9987 | Strategy | Unexpected strategy accounts issue on Playback data |
| Fixed | 9870 | Strategy Analyzer | Data Box exception using chart display |
| Fixed | 9855 | Strategy Analyzer | Settings display was not updating include commissions param on subsequent backtests |
| Fixed | 9836 | Strategy Analyzer | Parameter tooltip displayed incorrectly |
| Fixed | 9828 | Strategy Analyzer | Control Center logo was sometimes being displayed in chart |
| Fixed | 9980 | Strategy Analyzer | Taskbar previews were not working |
| Fixed | 9922 | Strategy Analyzer | Analysis displayed incorrectly after running separate backtest on optimization result |
| Fixed | 9973 | Strategy Analyzer | Chart Exception: Cannot call DragMove or Activate before a Window is shown |
| Fixed | 9842 | Strategy Analyzer, Templates | Strategy template saved on chart did not restore correctly in Strategy Analyzer |
| Fixed | 9917 | Strategy Builder | Lookback period defaulted to 0 |
| Fixed | 9916 | Strategy Builder | TimeSpan did not use user-applied settings |
| Fixed | 9891 | Strategy Builder | Various Indicator Syntax changes |
| Fixed | 9888 | Strategy Builder | Did not retain added data series in conditions |
| Fixed | 9885 | Strategy Builder | Unlocking code did not open NinjaScript Editor |
| Fixed | 9883 | Strategy Builder | Strategy opened in NinjaScript Editor sometimes produced unhandled exception |
| Fixed | 9867 | Strategy Builder | Did not prevent input of invalid Int value |
| Fixed | 9861 | Strategy Builder | Duplicate input series displayed in selector |
| Fixed | 9938 | Strategy Builder | Conditions window incorrectly scrolled to the top of the list of options when opening configured condition |
| Fixed | 9984 | Strategy Builder | Used incorrect Set methods OnStateChange() |
| Fixed | 9991 | Strategy Builder | Used incorrect value of 0 for selected order quantity |
| Fixed | 9981 | Strategy Builder | Consistency improvements regarding Condition/action copy/save |
| Fixed | 10011 | Strategy Builder | Did not confirm on removing strategy |
| Fixed | 10015 | Strategy Builder | Could not remove Strategy Builder Strategy when opened from NinjaScript Editor |
| Fixed | 10043 | Strategy Builder | Did not automatically compile strategy on clicking Finish |
| Fixed | 10017 | Strategy Builder | Save As' rejected a new name |
| Fixed | 10050 | Strategy Builder | Print() action was not accepting any arguments |
| Fixed | 10055 | Strategy Builder | File names were cut off in the strategy builder load/save strategy popup |
| Fixed | 10039 | Strategy Builder | Did not perform bars check on barsAgo and multi-series scripts |
| Fixed | 10073 | Strategy Builder | Sometimes lost active button highlight |
| Fixed | 10045 | Strategy Builder | String variable did not escape characters |
| Fixed | 9951 | Strategy Builder | Opening screen referred to "wizard" which caused confusion |
| Fixed | 10077 | Strategy Builder | German region decimal format caused compile errors |
| Fixed | 9871 | Strategy, Workspaces | Switching workspaces do not maintain realized PnL and randomly sorts rows in Strategies tab |
| Fixed | 9924 | TD AMERITRADE | Partial Position  was not displaying while connected to TDA |
| Fixed | 10056 | Trade Performance | Changing start date/end date on trade performance prevented generating report for only current day |
| Fixed | 10057 | Trade Performance | Trade Performance column sorting reset when regenerated |
| Fixed | 10027 | Data Grids | Performance Improvement on data grids (Changes require users to restore custom sorting/filtering manually) |
| Changed | 10063 | Indicator | Removed redundant logic of BuySellVolume and BuySellPressure indicators |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [ISeries<T>](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) >  **IsValidDataPoint()** | | | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/getvalueat.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/iseriest.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/isvaliddatapointat.htm) |

**Definition**

Indicates if the specified input is set at a barsAgo value relative to the current bar.  Please also see the [Reset()](https://ninjatrader.com/es/support/helpGuides/nt8/reset.htm) method for more information.

|  |
| --- |
| **Notes**:  •If called directly from the instance of the NinjaScript object, the value returned corresponds to the Input Series (e.g., Close, High, Low, SMA, etc.)  •When checking a [Bar](https://ninjatrader.com/es/support/helpGuides/nt8/bars.htm) or [PriceSeries](https://ninjatrader.com/es/support/helpGuides/nt8/priceseries.htm), IsValidDataPoint() returns **true** as long as the barAgo value falls between 0 and the total count for that series.  These are special series which always contain a value set at every slot index for multi-series scripting purposes (e.g., comparing two price series with various session templates, or one series has more ticks than the other)  •For a [Value](https://ninjatrader.com/es/support/helpGuides/nt8/value.htm) series or custom [Series<T>](https://ninjatrader.com/es/support/helpGuides/nt8/seriest.htm), IsValidPlot() returns **true** or **false** depending on if you have set a value at that index location |

**Method Return Value**

A bool value, when **true** indicates that specified data point is set; otherwise **false**.

**Syntax**

IsValidDataPoint(int barsAgo)

ISeries<T>.IsValidDataPoint(int barsAgo)

|  |
| --- |
| **Warning**:  Calling IsValidDataPoint() will only work a MaximumBarsLookBackInfinite series.  Attempting to check IsValidDataPoint() MaximumBarsLookBack256 series throw an error.  Please check the Log tab of the Control Center. In addition since this method references BarsAgo data, and therefore cannot be used during [OnRender (see note 5)](https://ninjatrader.com/es/support/helpGuides/nt8/onrender.htm).- instead please use the [IsValidDataPointAt](https://ninjatrader.com/es/support/helpGuides/nt8/isvaliddatapointat.htm) during OnRender. |

**Parameters**

|  |  |
| --- | --- |
| barsAgo | An int representing from the current bar the number of historical bars the method will check. |

**Examples**

| ns | |
| --- | --- |
| protected override void OnBarUpdate() {   // only set plot value if hosted indicator is not reset   if(SMA(20).IsValidDataPoint(0))     MyPlot[0] = SMA(20)[0];     } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Drawing](https://ninjatrader.com/es/support/helpGuides/nt8/drawing.htm) >  **Draw.Region()** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/rectangle.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/drawing.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/region.htm) |

**Definition**

Draws a region on a chart.

**Method Return Value**

A [Region](https://ninjatrader.com/es/support/helpGuides/nt8/region.htm) object that represents the draw object.

**Syntax**

Draw.Region(NinjaScriptBase owner, string tag, int startBarsAgo,  
        int endBarsAgo, ISeries<double> series, double price, Brush areaBrush, int areaOpacity, int displacement = 0)  
Draw.Region(NinjaScriptBase owner, string tag, int startBarsAgo,  
        int endBarsAgo, ISeries<double> series1, ISeries<double> series2, Brush outlineBrush,  
        Brush areaBrush, int areaOpacity, [int displacement])  
Draw.Region(NinjaScriptBase owner, string tag, DateTime startTime,  
        DateTime endTime, ISeries<double> series, double price, Brush areaBrush, int areaOpacity)  
Draw.Region(NinjaScriptBase owner, string tag, DateTime startTime,  
        DateTime endTime, ISeries<double> series1, ISeries<double> series2, Brush outlineBrush, Brush areaBrush, int areaOpacity)

**Parameters**

|  |  |
| --- | --- |
| owner | The hosting NinjaScript object which is calling the draw method    Typically will be the object which is calling the draw method (e.g., "this") |
| tag | A user defined unique id used to reference the draw object.    For example, if you pass in a value of "myTag", each time this tag is used, the same draw object is modified. If unique tags are used each time, a new draw object will be created each time. |
| startBarsAgo | The starting bar (x axis co-ordinate) where the draw object will be drawn. For example, a value of 10 would paint the draw object 10 bars back. |
| startTime | The starting time where the draw object will be drawn. |
| endBarsAgo | The end bar (x axis co-ordinate) where the draw object will terminate |
| endTime | The end time where the draw object will terminate |
| series, series1, series2 | Any Series<double> type object such as an indicator, Close, High, Low etc.. The value of the object will represent a y value. |
| price | Any double value |
| outlineBrush | The brush used to color the region outline of draw object ([reference](https://msdn.microsoft.com/en-us/library/system.windows.media.brushes%28v=vs.110%29.aspx" \t "_blank)) |
| areaBrush | The brush used to color the fill region area of the draw object ([reference](https://msdn.microsoft.com/en-us/library/system.windows.media.brushes%28v=vs.110%29.aspx" \t "_blank)) |
| areaOpacity | Sets the level of transparency for the fill color. Valid values between 0 - 100. (0 = completely transparent, 100 = no opacity) |
| displacement | An optional parameter which will offset the barsAgo value for the Series<double> value used to match the desired [Displacement](https://ninjatrader.com/es/support/helpGuides/nt8/displacement.htm).  Default value is 0. |

**Example**

| ns |
| --- |
| // Draw a region between upper and lower Bollinger bands  Draw.Region(this, "tag1", CurrentBar, 0, Bollinger(2, 14).Upper, Bollinger(2, 14).Lower, null, Brushes.Blue, 50); |

|  |  |
| --- | --- |
| **Tips**: 1. Pass in null to the "outlineColor" parameter if you do not want to have an outline color. 2. If you wanted to fill a region between a value (20 period simple moving average) and the upper edge of the chart, pass in an extreme value to the "y" parameter such as 1000000.  3. Should you be drawing regions based on Series<double> objects instead of indicator plots, be sure to create the Series<double> with the MaximumBarsLookBack.Infinite parameter if the region you are drawing would be maintained on the chart for more than 256 bars back. | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Educational Resources](https://ninjatrader.com/es/support/helpGuides/nt8/educational_resources.htm) > [Tips](https://ninjatrader.com/es/support/helpGuides/nt8/tips.htm) >  **Referencing the correct bar** | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/parameter_sequencing.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/tips.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/strategy_position_vs__account_.htm) |

When coding an indicator or strategy it is important to be able to access the intended bars for correct calculations. In NinjaScript we are able to access the bars we want through proper use of the bar's indexing.

The bar's indexing is setup in a reverse chronological order. This means "0" refers to the most recent bar, "1" refers to the previous bar, "2" refers to the bar before that one, etc.

For example, if we wanted to subtract the high and low of 10 bars ago from each other we would do this:

| ns |
| --- |
| double value = High[10] - Low[10]; |

Now that we know how the indexing works there are several properties and methods at our disposal that can help us access important keystone bars. The more important ones are [CurrentBar](https://ninjatrader.com/es/support/helpGuides/nt8/currentbar.htm) and [BarsSinceNewTradingDay](https://ninjatrader.com/es/support/helpGuides/nt8/barssincenewtradingday.htm).

**CurrentBar**

CurrentBar returns an int representing the number of bars existing on the chart. This property is most useful when you want to run calculations from the very beginning of the chart.

For example, if you wanted to find the average high value of the first 10 bars on the chart you could do this:

| ns |
| --- |
| double highValue = 0;  int x = CurrentBar;  while (x > CurrentBar - 10)  {     highValue += High[x];     x--;  }  Print("The average high value: " + highValue/10); |

|  |
| --- |
| **Note**: A common mistake in using CurrentBar is using it in the index to access the most recent bar. In this situation, instead of doing something like Close[CurrentBar] you will want to do Close[0]. |

**BarsSinceNewTradingDay**

BarsSinceNewTradingDay is another property that can help you find the first bar of the current trading day. The difference between BarsSinceNewTradingDay and CurrentBar is that BarsSinceNewTradingDay resets its count whenever a new session begins. This means if you use it in an index it will only get you to the beginning of the current session and not any previous sessions.

For example, if you wanted to find the open of the current session you could do this:

| ns |
| --- |
| double openValue = Open[Bars.BarsSinceNewTradingDay]; |

The example used in the discussion about CurrentBar can also be done with Bars.BarsSinceNewTradingDay if you wanted to calculate values based on the current session instead of the start of the chart too.

|  |
| --- |
| **Note**: If you wish to access values older than 256 bars ago you will need to ensure the [MaximumBarsLookBack](https://ninjatrader.com/es/support/helpGuides/nt8/maximumbarslookback.htm)is set to .Infinite. |

**Other Properties and Methods**

There are also a number of other properties and methods that can be useful in helping you locate the correct bars index to reference. Please take a look at these in the help guide:

[BarsSinceEntryExecution()](https://ninjatrader.com/es/support/helpGuides/nt8/barssinceentryexecution.htm)

[BarsSinceExitExecution()](https://ninjatrader.com/es/support/helpGuides/nt8/barssinceexitexecution.htm)

[GetBar()](https://ninjatrader.com/es/support/helpGuides/nt8/getbar.htm)

[GetDayBar()](https://ninjatrader.com/es/support/helpGuides/nt8/getdaybar.htm)

[HighestBar()](https://ninjatrader.com/es/support/helpGuides/nt8/highestbar.htm)

[LowestBar()](https://ninjatrader.com/es/support/helpGuides/nt8/lowestbar.htm)

[LRO()](https://ninjatrader.com/es/support/helpGuides/nt8/least_recent_occurence_lro.htm)

[MRO()](https://ninjatrader.com/es/support/helpGuides/nt8/most_recent_occurence_mro.htm)

|  |  |
| --- | --- |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Strategy](https://ninjatrader.com/es/support/helpGuides/nt8/strategy.htm) >  **BarsSinceEntryExecution()** | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/barsrequiredtotrade.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/strategy.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/barssinceexitexecution.htm) |

**Definition**

Returns the number of bars that have elapsed since the last specified entry.

**Method Return Value**

An int value that represents a number of bars. A value of -1 will be returned if a previous entry does not exist.

**Syntax**  
BarsSinceEntryExecution()  
BarsSinceEntryExecution(string signalName)

The following method signature should be used when working with [multi-time frame and instrument strategies](https://ninjatrader.com/es/support/helpGuides/nt8/multi-time_frame__instruments.htm):

BarsSinceEntryExecution(int barsInProgressIndex, string signalName, int entryExecutionsAgo)

|  |
| --- |
| **Note**: When working with a multi-series strategy the BarsSinceEntryExecution() will return you the elapsed bars as determined by the first Bars object for the instrument specified by the barsInProgressIndex. |

**Parameters**

|  |  |
| --- | --- |
| signalName | The signal name of an entry order specified in an order entry method. |
| barsInProgressIndex | The index of the Bars object the entry order was submitted against.    **Note**:  See the [BarsInProgress](https://ninjatrader.com/es/support/helpGuides/nt8/barsinprogress.htm) property. |
| entryExecutionsAgo | Number of entry executions ago. Pass in 0 for the number of bars since the last entry execution. |

**Examples**

| ns | |
| --- | --- |
| protected override void OnBarUpdate() {     if (CurrentBar < BarsRequiredToTrade)         return;       // Only enter if at least 10 bars has passed since our last entry     if ((BarsSinceEntryExecution() > 10 || BarsSinceEntryExecution() == -1) && CrossAbove(SMA(10), SMA(20), 1))         EnterLong();     } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Strategy](https://ninjatrader.com/es/support/helpGuides/nt8/strategy.htm) >  **BarsSinceExitExecution()** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/barssinceentryexecution.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/strategy.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/chartindicators.htm) |

**Definition**

Returns the number of bars that have elapsed since the last specified exit.

**Method Return Value**

An int value that represents a number of bars. A value of -1 will be returned if a previous exit does not exist.

**Syntax**  
BarsSinceExitExecution()  
BarsSinceExitExecution(string *signalName*)

The following method signature should be used when working with [multi-time frame and instrument strategies](https://ninjatrader.com/es/support/helpGuides/nt8/multi-time_frame__instruments.htm):

BarsSinceExitExecution(int *barsInProgressIndex*, string *signalName*, int *exitExecutionsAgo*)

|  |
| --- |
| **Note**: When working with a multi-series strategy the BarsSinceExitExecution() will return you the elapsed bars as determined by the first Bars object for the instrument specified in the barsInProgressIndex. |

**Parameters**

|  |  |
| --- | --- |
| signalName | The signal name of an exit order specified in an order exit method. |
| barsInProgressIndex | The index of the Bars object the entry order was submitted against.    **Note**:  See the [BarsInProgress](https://ninjatrader.com/es/support/helpGuides/nt8/barsinprogress.htm) property. |
| exitExecutionsAgo | Number of exit executions ago. Pass in 0 for the number of bars since the last exit execution. |

|  |
| --- |
| **Tip**:  Please see [SetStopLoss()](https://ninjatrader.com/es/support/helpGuides/nt8/setstoploss.htm), [SetProfitTarget()](https://ninjatrader.com/es/support/helpGuides/nt8/setprofittarget.htm) or [SetTrailStop()](https://ninjatrader.com/es/support/helpGuides/nt8/settrailstop.htm) for their corresponding signal name |

**Examples**

| ns | |
| --- | --- |
| protected override void OnBarUpdate() {   if (CurrentBar < BarsRequiredToTrade)       return;     // Only enter if at least 10 bars has passed since our last exit or if we have never traded yet   if ((BarsSinceExitExecution() > 10 || BarsSinceExitExecution() == -1) && CrossAbove(SMA(10), SMA(20), 1))       EnterLong(); } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Bars](https://ninjatrader.com/es/support/helpGuides/nt8/bars.htm) >  **GetBar()** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/getask.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/bars.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/getbid.htm) |

**Definition**

Returns the first bar that matches the time stamp of the "time" parameter provided.

|  |
| --- |
| **Note**:  If the time parameter provided is older than the first bar in the series, a bar index of 0 is returned. If the time stamp is newer than the last bar in the series, the last absolute bar index is returned. |

**Method Return Value**

An int value representing an absolute bar index value.

**Syntax**  
Bars.GetBar(DateTime time)

**Parameters**

|  |  |
| --- | --- |
| time | Time stamp to be converted to an absolute bar index |

**Examples**

| ns | |
| --- | --- |
| // Check that its past 9:45 AM if (ToTime(Time[0]) >= ToTime(9, 45, 00)) {   // Calculate the bars ago value for the 9 AM bar for the current day   int barsAgo = CurrentBar - Bars.GetBar(new DateTime(2006, 12, 18, 9, 0, 0));     // Print out the 9 AM bar closing price   Print("The close price on the 9 AM bar was: " + Close[barsAgo].ToString()); } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Bars](https://ninjatrader.com/es/support/helpGuides/nt8/bars.htm) >  **GetDayBar()** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/getclose.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/bars.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/gethigh.htm) |

**Definition**

Returns a virtual historical Bar object that represents a trading day whose properties for open, high, low, close, time and volume can be accessed.

|  |
| --- |
| **Notes:**  1.The bar object returned is a "virtual bar" built from the underlying bar series and its configured session.  Since the bar object is virtual, its property values are calculated based on session definitions contained in the trading day only. The returned bar object does **NOT** necessarily represent the actual day.  For accessing a true "Daily" bar, please see use [AddDataSeries()](https://ninjatrader.com/es/support/helpGuides/nt8/adddataseries.htm)and use the BarsPeriodType.Day as the bars period.  2.GetDayBar() should **ONLY** be used for accessing prior trading day data. To access current trading day data, use the [CurrentDayOHL()](https://ninjatrader.com/es/support/helpGuides/nt8/current_day_ohl.htm) method. |

**Method Return Value**

A virtual bar object representing the current configured session. Otherwise null if there is insufficient intraday data

**Syntax**  
The properties below return double values:

Bars.GetDayBar(int tradingDaysBack).Open  
Bars.GetDayBar(int tradingDaysBack).High  
Bars.GetDayBar(int tradingDaysBack).Low  
Bars.GetDayBar(int tradingDaysBack).Close

The property below returns a [DateTime](http://msdn.microsoft.com/en-us/library/system.datetime.aspx" \t "_blank) structure:

Bars.GetDayBar(int tradingDaysBack).Time

The property below returns an int value:

Bars.GetDayBar(int tradingDaysBack).Volume

|  |
| --- |
| **Warning**:  You must check for a null reference to ensure there is sufficient intraday data to build a trading day bar. |

**Parameters**

|  |  |
| --- | --- |
| tradingDaysBack | An int representing the number of the trading day to get OHLCV and time information from |

**Examples**

| ns | |
| --- | --- |
| protected override void OnBarUpdate() {   // Check to ensure that sufficient intraday data was supplied   if(Bars.GetDayBar(1) != null)     Print("The prior trading day's close is: " + Bars.GetDayBar(1).Close); } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Analytical](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) >  **HighestBar()** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/getmedian.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/falling.htm) |

**Definition**

Returns the number of bars ago the highest price value occurred within the specified look-back period.

**Method Return Value**

An int value representing a value of bars ago.

**Syntax**  
HighestBar(ISeries<double> series, int period)

**Parameters**

|  |  |
| --- | --- |
| period | The number of bars to include in the calculation |
| series | Any Series<double> type object such as an indicator, Close, High, Low, etc... |

**Examples**

| ns | |
| --- | --- |
| protected override void OnBarUpdate() {     // store the highest bars ago value   int highestBarsAgo = HighestBar(High, Bars.BarsSinceNewTradingDay);     //evaluate high price from highest bars ago value   double highestPrice = High[highestBarsAgo];             //Printed result:  Highest price of the session: 2095.5 - occurred 24 bars ago   Print(string.Format("Highest price of the session: {0} - occurred {1} bars ago", highestPrice, highestBarsAgo));             } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Analytical](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) >  **LowestBar()** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/least_recent_occurence_lro.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/most_recent_occurence_mro.htm) |

**Definition**

Returns the number of bars ago the lowest price value occurred within the specified look-back period.

**Method Return Value**

An int value representing a value of bars ago.

**Syntax**  
LowestBar(ISeries<double> series, int period)

**Parameters**

|  |  |
| --- | --- |
| period | The number of bars to check for the test condition |
| series | Any Series<double> type object such as an indicator, Close, High, Low, etc... |

**Examples**

| ns | |
| --- | --- |
| protected override void OnBarUpdate() {     // store the lowest bar ago value   int lowestBar = LowestBar(Low, Bars.BarsSinceNewTradingDay);     //evaluate low price from lowest bar ago value   double lowestPrice = Low[lowestBar];             //Printed result:  Lowest price of the session: 2087.25 - occurred 362 bars ago   Print(string.Format("Lowest price of the session: {0} - occurred {1} bars ago", lowestPrice, lowestBar));             } | |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Analytical](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) >  **Least Recent Occurrence (LRO)** | | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/rising.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/lowestbar.htm) |

**Definition**

Returns the number of bars ago that the test condition evaluated to true within the specified look back period expressed in bars. The**LRO()** method start from the furthest bar away and works toward the current bar.

|  |
| --- |
| **Note**: This method does **NOT** work on [multi-series](https://ninjatrader.com/es/support/helpGuides/nt8/multi-time_frame__instruments.htm) strategies and indicators. |

**Method Return Value**

An int value representing the number of bars ago. Returns a value of -1 if the specified test condition did not evaluate to true within the look-back period.

**Syntax**

LRO(Func<bool> *condition*, int *instance*, int lookBackPeriod)

|  |
| --- |
| **Warnings**:  1.  The "instance" parameter **MUST** be greater than 1.  2.  The "lookBackPeriod" parameter **MUST** be greater than 0.  3.  Please check the Log tab for any other exceptions that may be thrown by the condition function parameter. |

**Parameters**

|  |  |
| --- | --- |
| condition | A true/false expression |
| instance | The occurrence to check for (1 is the least recent, 2 is the 2nd least recent, etc...) |
| lookBackPeriod | The number of bars to look back to check for the test condition. The test evaluates on the current bar and the bars within the look-back period. |

|  |
| --- |
| **Tip**:  The syntax for the "condition" parameter uses [lambda expression](http://msdn.microsoft.com/en-us/library/bb397687.aspx) syntax |

**Examples**

| ns |
| --- |
| protected override void OnBarUpdate() {   // Prints the high price of the least recent up bar over the last 10 bars (current bar + look back period's 9 bars before that)   int barsAgo = LRO(() => Close[0] > Open[0], 1, 9);   if (barsAgo > -1)       Print("The bar high was " + High[barsAgo]);         } |

**See Also**  
[Most Recent Occurrence(MRO)](https://ninjatrader.com/es/support/helpGuides/nt8/most_recent_occurence_mro.htm)

|  |  |
| --- | --- |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Common](https://ninjatrader.com/es/support/helpGuides/nt8/common.htm) > [Analytical](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) >  **Most Recent Occurrence (MRO)** | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/lowestbar.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/market_data.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/slope.htm) |

**Definition**

Returns the number of bars ago that the test condition evaluated to true within the specified look back period expressed in bars.  The **MRO()** method starts from the current bar works away (backward) from it.

|  |
| --- |
| **Note**: This method does **NOT** work on [multi-series](https://ninjatrader.com/es/support/helpGuides/nt8/multi-time_frame__instruments.htm) strategies and indicators. |

**Method Return Value**

An int value representing the number of bars ago. Returns a value of -1 if the specified test condition did not evaluate to true within the look-back period.

**Syntax**

MRO(Func<bool> *condition*, int instance, int lookBackPeriod)

|  |
| --- |
| **Warnings**:  1.  The "instance" parameter **MUST** be greater than 1.  2.  The "lookBackPeriod" parameter **MUST** be greater than 0.  3.  Please check the Log tab for any other exceptions that may be thrown by the condition function parameter. |

**Parameters**

|  |  |
| --- | --- |
| condition | A true/false expression |
| instance | The occurrence to check for (1 is the most recent, 2 is the 2nd most recent, etc...) |
| lookBackPeriod | The number of bars to look back to check for the test condition. The test evaluates on the current bar and the bars within the look-back period. |

|  |
| --- |
| **Tip**:  The syntax for the "condition" parameter uses [lambda expression](http://msdn.microsoft.com/en-us/library/bb397687.aspx) syntax |

**Examples**

| ns |
| --- |
| protected override void OnBarUpdate() {   // Prints the high price of the most recent up bar over the last 10 bars (current bar + look back period's 9 bars before that)   int barsAgo = MRO(() => Close[0] > Open[0], 1, 9);   if (barsAgo > -1)       Print("The bar high was " + High[barsAgo]);   } |

**See Also**  
[Least Recent Occurrence(LRO)](https://ninjatrader.com/es/support/helpGuides/nt8/least_recent_occurence_lro.htm)