|  |  |
| --- | --- |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) >  **Code Breaking Changes** | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript_best_practices.htm) |

The following document is intended as a high level overview of the NinjaScript changes you can expect between NinjaTrader 7 and NinjaTrader 8.  For specific information on a particular method or property, you can refer to the dynamically formatted **Code Breaking table** at the bottom of this page.  We recommend using the **Filter** and **Sorting** features built into the table, as well checking the **Summary** column and expanding the **Details** section of each entry for general information.  Referring to the conveniently linked NinjaTrader 8 and NinjaTrader 7 documentation will provide specific information on syntax, usage, and examples of any new implementation or element names.

|  |
| --- |
| **Note**:  Information on this page focuses on **supported** **(documented)** NinjaTrader methods and properties shared between versions.  NinjaTrader 8 has seen a significant increase in supported NinjaTrader code, however if you were using previously **undocumented** NinjaTrader 7 methods or properties, they will **NOT** be covered in this topic.  You may be able to find more information on previously **undocumented** methods and properties in the NinjaTrader 8 Help Guide, or our support staff will also be happy to personally point you in the right direction. |

|  |
| --- |
| **Critical**:   If your product uses **unsupported (undocumented)** elements we strongly urge you to put your scripts through thorough testing to ensure they still behave as expected.  There is **NO** guarantee that previously **undocumented** method or property behavior has not changed in the new version of NinjaTrader 8. |

For questions or comments, please contact us at platformsupport@ninjatrader.com

tog_minus        [Implementation Changes Overview](javascript:HMToggle('toggle','ImplementationChangesOverview','ImplementationChangesOverview_ICON'))

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Initialize(), OnStartUp(), OnTermination()**  NinjaTrader 8 has simplified the methods used to set or release various resources during the lifetime of a NinjaTrader object to a single **[OnStateChange()](https://ninjatrader.com/es/support/helpGuides/nt8/onstatechange.htm)** method. This single method is guaranteed to be called for every change in **State** of the object.  It is from this method you can monitor the progression of the object throughout its lifetime in order to setup various resources, set properties, or take action the moment **State** has changed.  This method also exposes a **[State](https://ninjatrader.com/es/support/helpGuides/nt8/state.htm)** variable which can be used in various other methods, such as**OnBarUpdate(),** in order to tell your indicator or strategy to process data depending on the actual **State** of the object.    For example, pushing settings to the UI, or setting initial values for public properties can now be done use **OnStateChange()** when the state has reached**State.SetDefaults**:     | ns | | --- | | protected override void OnStateChange() {   if (State == State.SetDefaults)   {     // set the default properties     Name = "My Indicator";     Fast = 10;     Slow = 25;     IsOverlay = true;     IsAutoScale = true;   } } |       If you have custom resources that need to be setup before the NinjaTrader object is active and processing data, instead of using the**Initialize()** method, you can now set this up once the **OnStateChange()** method has reached **State.Configure** state:     | ns | | --- | | protected override void OnStateChange() {   if (State == State.Configure)   {     // Add a 5 minute Bars object to the strategy     AddDataSeries(Data.BarsPeriodType.Minute, 5);     // setup a custom data series     spread = new Series<double>(this);     // setup a 20-period EMA indicator     ema = EMA(20);     // add indicator to strategy for visual purposes     AddChartIndicator(ema);     } } |       NinjaTrader 7 had no concept to detect when your NinjaTrader object was transitioning from processing Historical data to processing Real-time data.  Now with NinjaTrader 8, the **OnStateChange()** method provides a **State.Transition** state which will notify you when this change is about to occur.  If your NinjaTrader 7 indicators or strategies were using custom methods to try to detect this transition, your custom methods may be refactored under this new state:     | ns | | --- | | protected override void OnStateChange() {   if (State == State.Transition)   {     Print("We're going to real-time data...");     // setup your real-time data resources here   } } |       When your NinjaTrader object is shutting down, and you need clean up any custom device resources, instead of using **OnTermination()**, you should now clean up these resources once the **OnStateChange()** method has reached the **State.Terminated** state:     | ns | | --- | | protected override void OnStateChange() {   if (State == State.Terminated)   {     // release any device resources     if(myTimer != null)         myTimer = null;   } } |     NinjaTrader previously used a **Historical** bool property to notify when an indicator or strategy bar was being processed historically or real-time.  The NinjaTrader 8 **OnStateChange()** approach has now introduced a class level variable **State** where you can check for **State.Historical** or**State.Realtime**in any of the other event methods which will allow you to take action depending on the desired state:     | ns | | --- | | protected override void OnBarUpdate() {   // only process on real-time data   if (State == State.Historical)     return;     else if (State >= State.Realtime)       // rest of logic } |     **Strategies, Orders, and Accounts**  Low level access has been provided to allow more flexibility with the information pertaining to trade data.    •IOrders, IExecution, and IPosition interfaces have all been replaced directly with the corresponding object  •The signatures of the related NinjaScript events have changed to match the NinjaTrader internal Update events  •Methods now return and update with the object instance generated, instead of the previously used interface     |  | | --- | | **Tip**:  Since NinjaTrader 8 now exposes the direct **Order** object, rather than an **IOrder** interface, it is possible to receive **null object reference errors** if you attempt to access an order object before the entry or exit order method has returned.  To prevent these situations, it is recommended to assign your strategies **Order** variables in the **OnOrderUpdate()** method and match them by their **signal name** (order.Name).  Please see the example beginning on line #22 below for demonstration of assigning order objects to private variables. |      | ns | | --- | | Order myOrder = null;   protected override void OnBarUpdate() {           if (Position.MarketPosition == MarketPosition.Flat && myOrder == null)     EnterLongLimit(Low[0], "Entry");     if (myOrder != null)   {     Print(myOrder.OrderState);           if (myOrder.OrderState == OrderState.Cancelled || myOrder.OrderState == OrderState.Filled)         myOrder = null;               } }         protected override void OnOrderUpdate(Cbi.Order order, double limitPrice, double stopPrice,   int quantity, int filled, double averageFillPrice,   Cbi.OrderState orderState, DateTime time, Cbi.ErrorCode error, string comment) {         // compare the order object created via EnterLongLimit by the signal name   if (myOrder == null && order.Name == "Entry")   {     // assign myOrder to matching order update     myOrder = order;           } } |     **Data Series**  Previously there had been type specific Data Series implementations (e.g., IntSeries, TimeSeries, BoolSeries, etc).  Now there just is a template [Series<T>](https://ninjatrader.com/es/support/helpGuides/nt8/seriest.htm) class which could be used generically and even allows support for additional types:     | ns | | --- | | Series<double> mySeries = new Series<double>(this); Series<DateTime> myTimeSeries = new Series<DateTime>(this); |     The **DataSeries.Set()** method used to assign Data Series or Plot values has been removed and values can now be stored using a single assignment operator:     | ns | | --- | | protected override void OnBarUpdate() {   // set public plotting data series close value of current bar   MyPlot[0] = Close[0];   // set custom Series<DateTime> to time value of current bar   myTimeSeries[0] = Time[0];         } |     **Drawing**  The DrawObjects used in NinjaTrader have received a number of changes:    •All DrawObjects have been moved to a separate **NinjaScript.DrawingTools** namespace and are properly known as **DrawingTools**  •Drawing Methods called from indicators or strategies have been moved to a new static partial **Draw** class  •Drawing Methods have all received a signature change which requires you specify the owner (object) which drew the **DrawingTool** object  •Drawing Methods no longer returns an interface but rather an instance of the **DrawingTool** object itself  •Drawing Methods now use the [System.Windows.Media.Brushes](https://msdn.microsoft.com/en-us/library/system.windows.media.brushes%28v=vs.110%29.aspx" \t "_blank) class instead of the [System.Drawing.Color](https://msdn.microsoft.com/en-us/library/system.drawing.color(v=vs.110).aspx" \t "_blank) structure     |  | | --- | | **Tip**:  DrawingTools are now completely unprotected and you can review their source code from the DrawingTools folder of the NinjaScript Editor's explorer menu |      | ns | | --- | | // example syntax Draw.Line(NinjaScriptBase owner, string tag, int startBarsAgo, double startY, int endBarsAgo, double endY, Brush brush)   // example usage Draw.Line(this, "tag1", true, 10, Low[0], 0, Brushes.Red); |     Casting a member of the **DrawObjects[]**collection must be done safely using the "as" keyword, otherwise you may receive exceptions at run time should another instance of the object (e.g., matching tag) exist from another owner:     | ns | | --- | | NinjaScript.DrawingTools.Line myLine = DrawObjects["tag1"] as DrawingTools.Line; |     **DrawingTools** anchor fields such as "Time" or "Price", etc have been moved to a **ChartAnchor** object owned by the drawing tool, rather than a direct field on the drawing object interface.  Please refer to the NinjaTrader 8 documentation for specific changes for each drawing tool:     | ns | | --- | | double linePrice = myLine.StartAnchor.Price; |     Objects which previously used**System.Drawing.Font** now uses new **NinjaTrader.Gui.Tools.SimpleFont** class:     | ns | | --- | | Gui.Tools.SimpleFont myFont = new Gui.Tools.SimpleFont("Arial", 12); |     Properties and other methods/objects which previously [System.Drawing.Color](https://msdn.microsoft.com/en-us/library/system.drawing.color(v=vs.110).aspx" \t "_blank) structure now use the [System.Windows.Media.Brushes](https://msdn.microsoft.com/en-us/library/system.windows.media.brushes%28v=vs.110%29.aspx" \t "_blank) class:     | ns | | --- | | BackBrush = Brushes.Blue; |      |  | | --- | | **Note**:  For custom **Brush** objects, it is important to .**Freeze()** the **Brush** due to the multi-threaded architecture of NinjaTrader 8.  Please be sure to review the new information on using [Brushes](https://ninjatrader.com/es/support/helpGuides/nt8/brushes.htm) |     **Namespaces**  The NinjaTrader 7 namespaces**NinjaTrader.Indicator** and **NinjaTrader.Strategy**have been renamed and moved to single **NinjaTrader.NinjaScript** namespace     | ns | | --- | | //This namespace holds indicators in this folder and is required. Do not change it. namespace NinjaTrader.NinjaScript.Indicators {   public class MyCustomIndicator : Indicator   {   } }   //This namespace holds Strategies in this folder and is required. Do not change it. namespace NinjaTrader.NinjaScript.Strategies {   public class MyCustomStrategy : Strategy   {   } } |     **Partial Classes (Porting methods and properties from UserDefinedMethods.cs)**  NinjaTrader 7 used a "UserDefinedMethods" class to define methods to be used across multiple NinjaScript indicators or strategies. In NinjaTrader 8, these pre-built partial classes have been removed to reduce a number of issues which could result from users sharing their UserDefinedMethods.cs files, or overwriting their existing files with copies from a new vendor. Partial classes are now best built manually and saved in the C:\Users\<user>\Documents\NinjaTrader 8\bin\Custom\AddOns folder.     |  | | --- | | **Warning**: If a partial class is saved in one of the folders used for specific NinjaScript objects other than AddOns (e.g., Indicators folder), auto-generated NinjaScript code may be appended to the end of the class by the NinjaScript Editor when compiled, which will cause a compilation error.  Saving these files in the AddOns folder will ensure they are still accessible and will not generate code which may be cause conflicts. |     You can use the template below as a starting point to create your partial class. If your partial class needs to inherit from a parent class, you can append the name of your desired parent class after the " : " to change the inheritance.     |  | | --- | | **Note**: Methods within your partial classes should be using the "public" modifier. |      | ns**Partial Class Example Template** | | --- | | namespace NinjaTrader.NinjaScript.Indicators {   public partial class MyMethods *// : parent class to inherit from*   {       //Sample method which calculates the delta of two prices       public double calculateDelta(double firstPrice, double secondPrice)       {           return Math.Abs(firstPrice - secondPrice);       }         //Sample method which prints Position information       public void printPositionInfo(Position position)       {           Print(String.Format("{0}: {1} {2} at {3}", position.Instrument, position.Quantity, position.MarketPosition, position.AveragePrice));       }           } } |     Below is an example of using one of the methods in this partial class from within an Indicator:     | ns**Partial Class Usage** | | --- | | protected override void OnBarUpdate() {   if (CurrentBar < 1) return;     // Use the static calculateDelta method to calculate the difference between the close of each bar   double delta = MyMethods.calculateDelta(Close[0], Close[1]);     Print(delta); } |      |  | | --- | | **Tip**:  At the time of the Beta implementation, the NinjaScript Editor does **NOT** include a partial class generator wizard, as it does for core NinjaScript Types such as Drawing Tools, Market Analyzer Columns, or Strategies. However, we are currently tracking a suggestion to implement a wizard for partial classes, under ID # **SFT-341**.   Please feel free to contact platformsupport@ninjatrader.com if you would like to add your vote for this enhancement. |     **Prevention of Redundant Data Loading**  In NinjaTrader 7, multiple Data Series could be added within a script, such as an indicator, and that script could then be hosted by another script, such as a strategy. While this is still possible in NinjaTrader 8, there is a new safeguard in place to prevent redundant data loading in both the hosting script and the hosted indicator.    When hosting an indicator which adds Data Series programmatically, the hosting script must include the same calls to the AddDataSeries() method as the hosted script. Without this, an error will result, which reads *"A hosted indicator tried to load additional data. All data must first be loaded by the hosting NinjaScript in its Configure state."* Without this safegaurd in place, it would be possible for unnecessarily large amounts of data to be loaded concurrently, as would be the case in a direct call to an indicator method on each OnBarUpdate(). By adding the calls to AddDataSeries() to the hosting script, you can ensure that the data is loaded when needed. Also, when this is done in the hosting script, all identical calls to AddDataSeries() in the hosted script will be ignored, as the data is already available.    The examples below show this in action:     | ns**Hosted Indicator Loads Additional Data** | | --- | | public class MyCustomIndicator : Indicator {   protected override void OnStateChange()   {     if (State == State.Configure)     {           AddDataSeries("AAPL", BarsPeriodType.Day, 1);           AddDataSeries("EURUSD", BarsPeriodType.Minute, 15);       }   } } |      | ns**Hosting Strategy Mirrors AddDataSeries() calls** | | --- | | public class MyCustomStrategy : Strategy {   // Define a MyCustomIndicator   MyCustomIndicator myIndicator;     protected override void OnStateChange()   {     if (State == State.Configure)     {         // Instantiate the MyCustomIndicator and add it to the chart         myIndicator = MyCustomIndicator();         AddChartIndicator(myIndicator);           // These calls to AddDataSeries() mirror the calls in the hosted indicator         AddDataSeries("AAPL", BarsPeriodType.Day, 1);         AddDataSeries("EURUSD", BarsPeriodType.Minute, 15);     }   } } |     **Bars with 0 Volume**  In previous versions, the NinjaTrader core was designed to replace a tick with a volume of 0 with a volume of 1.  This resulted in all ticks having a volume value of at least 1.  NinjaTrader 8 has removed that design policy and will now allow ticks with a volume of 0 to be processed.  This policy change may require logic changes to any custom bar types, indicators, or strategies which may have previously assumed volume would always be greater than 0.    **Multi-Series default "Trading Hours" templates**  The default behavior in NinjaTrader 8 will ensure that a bars series added to a script using [AddDataSeries()](https://ninjatrader.com/es/support/helpGuides/nt8/adddataseries.htm) will use the same "[TradingHours](https://ninjatrader.com/es/support/helpGuides/nt8/tradinghours.htm)" template as the primary series configured by the user. In contrast, the NinjaTrader 7 behavior was highly dependent on a number of variables.  We have updated this behavior to help with consistences and synchronization issues between multiple series; however if you your script relies on two times frames using different trading hours templates, you may consider using one of the new **tradingHours**string overloaded used in [AddDataSeries()](https://ninjatrader.com/es/support/helpGuides/nt8/adddataseries.htm):     | ns | | --- | | protected override void OnStateChange() {   if (State == State.Configure)   {     // adds a 1 minute AAPL bars with a default 24/7 session tempalte.     AddDataSeries("AAPL", new BarsPeriod { BarsPeriodType = BarsPeriodType.Minute, Value = 1 }, "Default 24 x 7");   } } |     **Miscellaneous**  All of the NinjaTrader 7 reference samples posted in our support forum have been updated to demonstrate NinjaTrader 8 functionality.  Please be sure to check the reference sample section to see other undocumented features and concepts which may not have been covered in the help guide:    [Official NinjaScript reference code samples](http://www.ninjatrader.com/support/forum/forumdisplay.php?f=30)    There are several other changes to implementation which are not covered in detail on this overview, please see the code breaking changes table at the bottom of this page which will compare the implementation changes between both versions. |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?code_breaking_changes.htm#ImplementationChangesOverview)

tog_minus        [Signature Changes Overview](javascript:HMToggle('toggle','SignatureChangesOverview','SignatureChangesOverview_ICON'))

|  |  |
| --- | --- |
| **Signature**  A large number of the NinjaTrader methods which were available in NinjaTrader 7 have remained largely the same and should not generate any errors on compilation.  However there are a handful of existing methods signatures which have been updated in NinjaTrader 8 in order to fit within new framework which you would need to be aware of in order to transfer these functions from NinjaTrader 7 to NinjaTrader 8.  In most cases, the fundamental argument type has been restructured, which may result in compile errors depending on the type of object that is being used within the methods signature.     |  | | --- | | **Tip**:  Methods may now have additional signatures which add functionality which was not previously available.  Be sure to check the NinjaTrader 8 documentation which will cover all the available signatures available. | |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?code_breaking_changes.htm#SignatureChangesOverview)

tog_minus        [Name Changes Overview](javascript:HMToggle('toggle','NameChangesOverview','NameChangesOverview_ICON'))

|  |
| --- |
| **Renamed**  During the NinjaTrader 8 development process, one of our goals to make sure that our core framework matched various coding standards which have been set out in the industry.  As a result of meeting these coding standards, many NinjaTrader methods and properties needed to been renamed.    While the functionality of these methods and properties remains the same, we chose to rename these variables to follow a semantically context specific naming convention which is generally agreed upon to favor readability.  We feel that the renaming of these properties and methods more explicitly describes the intended function to the developer who may be reviewing code.  The largest number of changes is in response to the name convention of bools, where they now follow a more strict verb-adjective or verb-noun structure.    For an example:    •The property **FirstTickOfBar** may have been hard to distinguish precisely what it represented without having to look up documentation.  In NinjaTrader 8, this property has been renamed to **IsFirstTickOfBar**, which now gives this property a more readable identifier name when you read this line of code as "*is the first tick of bar true?*"  •Another example is the case of **BarsSinceEntry()** which was renamed to**BarsSinceEntryExecution()**, which now specifies that this method is looking for an entry *execution*.  •NinjaTrader 7 sometimes had methods or properties which shared names, but references different data or actions.  For example **Add()** could have been used in reference to adding **DataSeries** to a script, adding a **Plot**, or adding a **Line**.  To be more specific, NinjaTrader 8 has renamed these to **AddDataSeries()**,**AddPlot()**, and **AddLine()** respectively.  •There may be cases where the property or method name has changed simply because the type of data it interacted with has changed.  (e.g., **BarColor** vs. **BarBrush**)  •There are other cases where properties may have used unnecessary brevity and was renamed to favor readability (e.g., **AvgPrice** vs **AveragePrice**)    These are just a few examples of the many name changes found in NinjaTrader 8 and some of the rational behind the number of these changes.  For simplicity, you will find a list of all the renamed properties in the table at the bottom of this document by filtering by the "Renamed" keyword. |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?code_breaking_changes.htm#NameChangesOverview)

**Code Breaking Table**

Below you will find a reference table which lists all of the supported NinjaScript changes between NinjaTrader 7 and NinjaTrader 8.

Show 255075100All entries

Filter results:

| **Category** | **Base** | **NT7 Method/Property** | **NT8 Method/Property** | **Summary** |  |
| --- | --- | --- | --- | --- | --- |
| Implementation | Strategy | [GetAccountValue()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?getaccountvalue.htm) | [Account.Get()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?get.htm) | Access to Account values have been directly exposed | Details |
| Renamed | Strategy | [Add() - Strategy](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?add2.htm) | [AddChartIndicator()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addchartindicator.htm) | Method renamed to be more specific |  |
| Implementation | General | [Add() - Data](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?add3.htm) | [AddDataSeries()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?adddataseries.htm) | Method was renamed to be more specific, received a number of enhancements. | Details |
| Signature | General | [AddKagi()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?addkagi.htm) | [AddKagi()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addkagi.htm) | Received a number of signature changes | Details |
| Renamed | Indicator | [Add() - Line](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?add.htm) | [AddLine()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addline.htm) | Method renamed to be more specific |  |
| Signature | General | [AddLineBreak()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?addlinebreak.htm) | [AddLineBreak()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addlinebreak.htm) | Received a number of signature changes | Details |
| Renamed | Indicator | [Add() - Plot](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?add.htm) | [AddPlot()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addplot.htm) | Method renamed to be more specific |  |
| Signature | General | [AddPointAndFigure()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?addpointandfigure.htm) | [AddPointAndFigure()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addpointandfigure.htm) | Received a number of signature changes | Details |
| Signature | General | [AddRenko()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?addrenko.htm) | [AddRenko()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?addrenko.htm) | Received a number of signature changes | Details |
| Signature | General | [Alert()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?alert.htm) | [Alert()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?alert.htm) | Color no longer used, use Brushes instead; soundLocation now requires absolute file path | Details |
| Implementation | Drawing | [IAndrewsPitchfork](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?iandrewspitchfork.htm) | [AndrewsPitchfork](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?andrewspitchfork.htm) | IDrawingObjects have been replaced | Details |
| Implementation | Drawing | [IArc](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?iarc.htm) | [Arc](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?arc.htm) | IDrawingObjects have been replaced | Details |
| Renamed | Indicator | [LinesConfigurable](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?linesconfigurable.htm) | [AreLinesConfigurable](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?arelinesconfigurable.htm) | Property renamed to meet naming conventions |  |
| Renamed | Indicator | [PlotsConfigurable](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?plotsconfigurable.htm) | [ArePlotsConfigurable](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?areplotsconfigurable.htm) | Property renamed to meet naming conventions |  |
| Implementation | Drawing | [IArrowDown](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?iarrowdown.htm) | [ArrowDown](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?arrowdown.htm) | IDrawingObjects have been replaced | Details |
| Implementation | Drawing | [IArrowLine](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?iarrowline.htm) | [ArrowLine](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?arrowline.htm) | IDrawingObjects have been replaced | Details |
| Implementation | Drawing | [IArrowUp](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?iarrowup.htm) | [ArrowUp](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?arrowup.htm) | IDrawingObjects have been replaced | Details |
| Implementation | General | [DataSeries.Set()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?dataseries_class.htm) | [Assignment Operator (=)](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?seriest.htm) | The .Set() method has been replaced | Details |
| Implementation | Strategy | [AtmStrategyCreate()](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?atmstrategycreate.htm) | [AtmStrategyCreate()](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?atmstrategycreate.htm) | Added a callback signature parameter | Details |
| Renamed | Strategy | [AvgBarsInTrade](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?avgbarsintrade.htm) | [AverageBarsInTrade](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?averagebarsintrade.htm) | Property renamed to favor readability |  |
| Renamed | Strategy | [AvgEtd](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?avgetd.htm) | [AverageEtd](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?averageetd.htm) | Property renamed to favor readability |  |
| Renamed | Strategy | [AvgMae](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?avgmae.htm) | [AverageMae](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?averagemae.htm) | Property renamed to favor readability |  |
| Renamed | Strategy | [AvgMfe](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?avgmfe.htm) | [AverageMfe](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?averagemfe.htm) | Property renamed to favor readability |  |
| Renamed | Strategy | [AvgPrice](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?avgprice.htm) | [AveragePrice](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?position_averageprice.htm) | Property renamed to favor readability |  |
| Renamed | Strategy | [AvgProfit](http://www.ninjatrader.com/support/helpGuides/nt7/index.html?avgprofit.htm) | [AverageProfit](http://www.ninjatrader.com/support/helpGuides/nt8/en-us/index.html?averageprofit.htm) | Property renamed to favor readability |  |

Showing 1 to 25 of 197 entries

Previous12345…8Next

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?code_breaking_changes.htm#codebreakingtable)

|  |  |
| --- | --- |
| **Navigation:**  [NinjaScript](https://ninjatrader.com/es/support/helpGuides/nt8/ninjascript.htm) > [Language Reference](https://ninjatrader.com/es/support/helpGuides/nt8/language_reference_wip.htm) > [Indicator](https://ninjatrader.com/es/support/helpGuides/nt8/indicator.htm) >  **IndicatorBaseConverter Class** | [Previous page](https://ninjatrader.com/es/support/helpGuides/nt8/drawverticalgridlines.htm) [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/indicator.htm) [Next page](https://ninjatrader.com/es/support/helpGuides/nt8/ischartonly.htm) |

**Definition**

A custom [TypeConverter](https://msdn.microsoft.com/en-us/library/system.componentmodel.typeconverter%28v=vs.110%29.aspx) class handling the designed behavior of an indicator's property descriptor collection.  Use this as a base class for any custom **TypeConverter** you are applying to an indicator class.

|  |
| --- |
| **Notes:**  •A working NinjaScript demo can be found through the reference sample on "[Using a TypeConverter to Customize Property Grid Behavior](http://ninjatrader.com/support/forum/showthread.php?t=97919" \t "_blank)"  •When applying the custom converter, you must fully qualify the name (e.g., "NinjaTrader.NinjaScript.Indicators.MyCustomConveter")  •Additional **TypeConverter** information can be found from the [MSDN documentation](https://msdn.microsoft.com/en-us/library/system.componentmodel.typeconverter%28v=vs.110%29.aspx)  •See also [TypeConverterAttribute](https://ninjatrader.com/es/support/helpGuides/nt8/typeconverterattribute.htm)  •For Strategies, see the [StrategyBaseConverter](https://ninjatrader.com/es/support/helpGuides/nt8/strategybaseconverter.htm) class |

**Relevant base methods**

|  |  |
| --- | --- |
| [TypeConverter.GetProperties()](https://msdn.microsoft.com/en-us/library/system.componentmodel.typeconverter.getproperties(v=vs.110).aspx) | When overriding **GetProperties()**, calling base.GetProperties() ensures that all default property grid behavior works as designed |
| [TypeConverter.GetPropertiesSupported()](https://msdn.microsoft.com/en-us/library/system.componentmodel.typeconverter.getpropertiessupported(v=vs.110).aspx) | In your custom converter class, you must override **GetPropertiesSupported()**and return a value of **true** in order for your custom type converter to work |

**Syntax**

public class IndicatorBaseConverter : TypeConverter

|  |
| --- |
| **Warning**:  Failure to apply a type of **IndicatorBaseConverter** on an indicator class can result in unpredictable behavior of the standard NinjaTrader WPF property grid. |

|  |
| --- |
| **Tip**: Common indicator functions like Print() are not available to a type converter instance.  To debug a type converter class, you can use the AddOn [Debug Concepts](https://ninjatrader.com/es/support/helpGuides/nt8/alert_and_debug_concepts.htm) or [attach to a debugger](https://ninjatrader.com/es/support/helpGuides/nt8/visual_studio_debugging.htm) (recommended) |

**Examples**

| ns | |
| --- | --- |
| //This namespace holds Indicators in this folder and is required. Do not change it. namespace NinjaTrader.NinjaScript.Indicators {   // When applying the type converter, you must fully qualify the name   [TypeConverter("NinjaTrader.NinjaScript.Indicators.MyCustomConveter")]   public class MyCustomIndicator : Indicator   {     protected override void OnStateChange()     {         if (State == State.SetDefaults)         {           Name   = "MyCustomIndicator";         }     }       protected override void OnBarUpdate()     {         //Add your custom indicator logic here.     }   }     public class MyCustomConveter : IndicatorBaseConverter   {     // A general TypeConveter method used for converting types     public override PropertyDescriptorCollection GetProperties(ITypeDescriptorContext context, object component, Attribute[] attrs)     {         // sometimes you may need the indicator instance which actually exists on the grid         MyCustomIndicator indicator = component as MyCustomIndicator;           // base.GetProperties ensures we have all the properties (and associated property grid editors)         // NinjaTrader internal logic handles for a given indicator         PropertyDescriptorCollection propertyDescriptorCollection = base.GetPropertiesSupported(context)                 ? base.GetProperties(context, component, attrs) : TypeDescriptor.GetProperties(component, attrs);           if (indicator == null || propertyDescriptorCollection == null)           return propertyDescriptorCollection;           // example of why you may need the instance that exists on the grid....         if (indicator.EntryHandling == EntryHandling.UniqueEntries)         {           // do something in the event a property contains some value...         }           // Loop all of the properties of the indicator         foreach (PropertyDescriptor property in propertyDescriptorCollection)         {           // do something with a specific property             // cannot call Print() here           // but you can call the static Output window "Process()"           NinjaTrader.Code.Output.Process(property.Name, PrintTo.OutputTab1);         }           // must return the collection after making changes         return propertyDescriptorCollection;     }       // Important:  This must return true otherwise the type converter will not be called     public override bool GetPropertiesSupported(ITypeDescriptorContext context)     { return true; }     }   } } | |
| **Navigation:**  »No topics above this level«  **Welcome to NinjaTrader 8 Beta** | | [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/sharpdx_directwrite_textlayout.htm) |

**Welcome to the NinjaTrader 8 Beta Release!**

We are very excited to openly release NinjaTrader 8 Beta to our general user base. The purpose of this program is to have you start testing the platform to in order to solicit feedback in the form of bug reports and constructive criticism. The beta process is a critical step in the software development cycle during which time we jointly work towards improving quality and performance with a goal of reaching a stable product ready for general release.

This release marks the introduction of our next generation trading platform. NinjaTrader 8 adds over 500+ new features and functionality on top of a re-designed architecture built to take advantage of new development practices and modern technologies. This is our most ambitious project to date and we are excited to get it in the hands of our customers.

|  |  |
| --- | --- |
| **NinjaTrader 8 Download** |  |
| [8.0.0.14](http://www.ninjatrader.com/PlatformDirect) | [Release Notes](https://ninjatrader.com/es/support/helpGuides/nt8/8_0_0_14.htm) |
| [8.0.0.13](http://www.ninjatrader.com/ninjatrader/nt8/NinjaTrader.Install.B13.msi) | [Release Notes](https://ninjatrader.com/es/support/helpGuides/nt8/8_0_0_13.htm) |
| [8.0.0.12](http://www.ninjatrader.com/ninjatrader/nt8/NinjaTrader.Install.B12.msi) | [Release Notes](https://ninjatrader.com/es/support/helpGuides/nt8/8_0_0_12.htm) |

**Bug reports and providing feedback**

Support will be driven via our[NinjaTrader Support Forum](http://www.ninjatrader.com/support/forum/forumdisplay.php?f=57" \t "_blank) NinjaTrader 8 section. When running into unexpected behavior please create a bug report in the NinjaTrader support forum's bug report section. Bug reports will be reviewed by our support and development staff inside of a 48 hour period. If you are unable to post to the bug section of the forum please contact [platformsupport@ninjatrader.com](mailto:platformsupport@ninjatrader.com).

tog_minus        [Installation Guide](javascript:HMToggle('toggle','Beta','Beta_ICON'))

|  |  |
| --- | --- |
| **Minimum PC Requirements**  You must meet the below minimum PC requirements to run NinjaTrader 8 Beta.    •Windows Vista (SP2) w/ Platform Update, Windows 7, Windows 8, Windows 10, Windows Server 2008 w/ Platform Update, Windows Server 2008 R2 or later.  •1 (GHz) or faster 32-bit or 64-bit processor  •2 GB RAM  •Microsoft .NET Framework 4.5  •Screen resolution of 1024 x 768 (16-bit color depth)    **Recommended PC Requirements**  NinjaTrader 8 was designed to take full advantage of modern PC architecture. To achieve the highest possible level of performance, NinjaTrader 8 will utilize all available CPU cores and additional memory resources. Depending on your actual usage with NinjaTrader you may need more or less resources as there is no one size fits all recommendation. Note: If you do plan on running strategy optimizations then having additional memory will be of direct benefit, the amount of additional memory needed is proportional to the number of CPU cores available.    •2 (GHz) or faster quad core 64-bit processor  •8 GB RAM  •DirectX10 compatible graphics card  •SSD Hard Drive    **Installation**  Please follow the below steps to get up and running on the beta.    1.If you do not have the Microsoft .NET Framework 4.5 installed you must download and install (Pre-installed on most PC’s and can be downloaded here: [Microsoft .NET Framework](http://www.microsoft.com/en-us/download/details.aspx?id=30653" \t "_blank)).  2.[Download](http://ninjatrader.com/PlatformDirect) and install the NinjaTrader 8 Beta.  3.If you are running a 64-bit PC you will want to start the 'NinjaTrader 8 (64-bit)' version, on a 32-bit PC start the 'NinjaTrader 8 (32-bit)' version.  4.Firewall Software – NinjaTrader contacts our license server on application start up for license key validation. If you have a firewall, spyware or other such software running on your PC, please ensure that you grant NinjaTrader permission to access the internet or you may receive an invalid or license expired message.  5.Platform Activation for Live Trading – If you registered to use the free simulation version, or do not plan on trading live with NinjaTrader 8 Beta, you can skip this step.  oA license key to activate live trading is sent via email within minutes from the time you complete your purchase (check your junk/spam folder if you have not received it)  oStart NinjaTrader, select the menu Help > License Key within the Control Center and enter your license key  6.If you are a user of NijaTrader 7 you have the option to migrate NinjaTrader 7 data. The migration process will import historical data, replay data, instruments, trade history, account connections, market analyzer templates and strategy wizard templates. Any workspaces, chart templates, NinjaScript will unfortunately not be migrated over.  7.If you choose to migrate your connection should be immediately available in the Control Center -> Connections menu. If you did not migrate and need to setup your first connection please review the [connect to your account](https://ninjatrader.com/es/support/helpGuides/nt8/connecting_to_youraccount.htm) page to get started.     |  | | --- | | **Note:**Simulation color by default is transparent in NinjaTrader 8. If you migrate your settings from NinjaTrader 7 then simulation color will be maintained as is from NinjaTrader 7. You can change simulation color via tools > Option > Trading Category > Simulation Color. | |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?beta.htm#Beta)

tog_minus        [Expectations during beta](javascript:HMToggle('toggle','Betaexpectation','Betaexpectation_ICON'))

|  |  |
| --- | --- |
| **Help guide status**  We have made significant progress on the Help Guide and will continue to work on it throughout the beta process but it is still “Work In Progress”.  Any section marked at the top of the page with "Under Construction" is still being worked on and my contain errors; sections marked as "Needs Review" should be considered complete, but may change in the future.  Please feel free to report feedback to [platformsupport@ninjatrader.com](mailto:platformsupport@ninjatrader.com)    **Historical data status**  If you are connecting to CQG, Continuum or the FXCM then you are using historical data provided by NinjaTrader. Our Historical Data Servers for NinjaTrader 8 have been redesigned and are also in a beta testing period. As such, it is possible to experience outages and gaps in data.    **Ongoing Performance Optimization**  We are not 100% complete in our User Interface performance optimizations and as such, you may notice on large workspaces and/or slower hardware areas that will be improved.    **Live Trading with NinjaTrader 8 Beta**  There are risks associated with live trading on any [electronic trading platform](https://ninjatrader.com/es/support/helpGuides/nt8/risks_of_electronic_trading_wi.htm). While the NinjaTrader 8 live trading components are feature complete, there could be some undiscovered bugs which could adversely impact your ability to trade live. As such, the risks associated with live trading should be heavily considered.  We do have users that are reporting positive experiences with live trading and we have been using NinjaTrader 8 in a production capacity with CQG/Continuum on our own order desk for the past eight months. We continue to work through reported issues in anticipation of giving our users the best possible experience when NinjaTrader 8 is released into production.    **Reporting Bugs**  When running into unexpected behavior please note the following if possible and include it in your bug report as it will increase our ability to diagnose and resolve the issue.    •The time and date the bug occurred.  •Step by step actions (if any) that you took prior to the bug.  •Screen shot or video of the bug.     |  | | --- | | **Tip**:  Internally we use a free service called ['Jing'](https://www.techsmith.com/jing.html) which is very helpful to allow you to take a screen shot or record a video and share it. |     **How long will the beta last?**  Beta is a fluid process where we will continue to stay in beta for as long as we are receiving critical bug reports. We will be sure to stay in contact with you to provide you with new builds and keep you updated as to the progress of the beta. |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?beta.htm#Betaexpectation)

tog_minus        [NinjaScript](javascript:HMToggle('toggle','BetaHistoricalData','BetaHistoricalData_ICON'))

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **What's New with NinjaScript in NinjaTrader 8**  One of the primary goals of NinjaTrader 8 is to expand the number of NinjaScript objects, methods, and properties that are documented and officially supported, in order to provide our customers more control and flexibility than ever before. To that end, we have increased support across the board and unlocked additional NinjaScript types.     |  |  | | --- | --- | | **NinjaScript Type** | **Description** | | [Add Ons](https://ninjatrader.com/es/support/helpGuides/nt8/add_on.htm) | Add Ons run as a window is created. Add Ons can be used to extend the functionality of an existing NinjaTrader window, or to create your own window that enjoys all the features of others, such as workspace persistence and styling. | | [Bars Type](https://ninjatrader.com/es/support/helpGuides/nt8/bars_type.htm) | Bar Types are now fully supported in NinjaTrader 8. Although we allowed them to be customized in NinjaTrader 7, we offered no official support and provided only a small number of available slots. We have addressed both of these items with NinjaTrader 8. | | [Chart Style](https://ninjatrader.com/es/support/helpGuides/nt8/chart_style.htm) | Chart Styles are now fully supported and customizable. | | [Drawing Tools](https://ninjatrader.com/es/support/helpGuides/nt8/drawing_tools.htm) | You can now create your own custom drawing tools for use on NinjaTrader charts. | | [Import Type](https://ninjatrader.com/es/support/helpGuides/nt8/import_type.htm) | Allows you to customize the Historical Data Importer to allow different data formats to be imported in NinjaTrader 8. | | [Indicator](https://ninjatrader.com/es/support/helpGuides/nt8/indicator.htm) | Indicators were fully supported in NinjaTrader 7, and continue to be fully supported with increased functionality in NinjaTrader 8. | | [Market Analyzer Column](https://ninjatrader.com/es/support/helpGuides/nt8/market_analyzer_column.htm) | Market Analyzer Columns were allowed to be customized in NinjaTrader 7, but there was no official support or documentation. We have added these as fully supported in NinjaTrader 8. | | [Optimization Fitness](https://ninjatrader.com/es/support/helpGuides/nt8/optimization_fitness.htm) | An Optimization Fitness allows you to expand the optimizer by defining new ways to find the best strategy performance. | | [Optimizer](https://ninjatrader.com/es/support/helpGuides/nt8/optimizer.htm) | We support the standard (Brute Force) optimizer and a Genetic Optimizer, and we've opened up full support for writing your own optimizer. | | [Performance Metrics](https://ninjatrader.com/es/support/helpGuides/nt8/performance_metrics.htm) | A new NinjaScript component which allows you to expand performance metrics and define your own custom metrics. These will be viewable in the Trade Performance window and the Strategy Analyzer. | | [Share Service](https://ninjatrader.com/es/support/helpGuides/nt8/share_service.htm) | A new NinjaScript component to allow the creation of additional sharing service adapters. By default we support email, Facebook, Twitter, and Stocktwits. | | [Strategy](https://ninjatrader.com/es/support/helpGuides/nt8/strategy.htm) | Strategies were fully supported in NinjaTrader 7, and continue to be fully supported with increased functionality in NinjaTrader 8. | | [SuperDOM Column](https://ninjatrader.com/es/support/helpGuides/nt8/superdom_column.htm) | New NinjaScript component that allows you to create your own custom column to be added to a SuperDOM window. |       **Converting NinjaScript From NinjaTrader 7 to NinjaTrader 8**  In line with the goals of providing expanded support, consistency, performance, and improved functionality, we found it necessary to make code breaking changes. This means that existing scripts for NinjaTrader 7 will have to go through a porting process to get the code compiled and working in NinjaTrader 8. The changes required are typically mild and primarily constitute renaming methods and moving code to the new "State" system. We have compiled an overview guide of the new concepts in NinjaTrader and a list of the [code breaking changes](https://ninjatrader.com/es/support/helpGuides/nt8/code_breaking_changes.htm) to help you get started. We are very welcome to feedback to help improve these documents. If you run into issues porting a script, please post in our beta forums so that we can assist and improve documentation where needed.    **DLL Protection with CLI Secure**  For those who distribute NinjaScript assemblies, we upgraded our protection to use the latest Agile.net 6.3 protection. As the protection is stronger, the service is no longer free, and we have negotiated a discounted price of $100 for protecting NinjaScript assemblies. Please visit the [Protection / DLL Security](https://ninjatrader.com/es/support/helpGuides/nt8/protection_dll_security.htm) section of the help guide for more information. |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?beta.htm#BetaHistoricalData)

|  |  |
| --- | --- |
| **Navigation:**  »No topics above this level«  **Welcome to NinjaTrader 8 Beta** | [Return to chapter overview](https://ninjatrader.com/es/support/helpGuides/nt8/sharpdx_directwrite_textlayout.htm) |

**Welcome to the NinjaTrader 8 Beta Release!**

We are very excited to openly release NinjaTrader 8 Beta to our general user base. The purpose of this program is to have you start testing the platform to in order to solicit feedback in the form of bug reports and constructive criticism. The beta process is a critical step in the software development cycle during which time we jointly work towards improving quality and performance with a goal of reaching a stable product ready for general release.

This release marks the introduction of our next generation trading platform. NinjaTrader 8 adds over 500+ new features and functionality on top of a re-designed architecture built to take advantage of new development practices and modern technologies. This is our most ambitious project to date and we are excited to get it in the hands of our customers.

|  |  |
| --- | --- |
| **NinjaTrader 8 Download** |  |
| [8.0.0.14](http://www.ninjatrader.com/PlatformDirect) | [Release Notes](https://ninjatrader.com/es/support/helpGuides/nt8/8_0_0_14.htm) |
| [8.0.0.13](http://www.ninjatrader.com/ninjatrader/nt8/NinjaTrader.Install.B13.msi) | [Release Notes](https://ninjatrader.com/es/support/helpGuides/nt8/8_0_0_13.htm) |
| [8.0.0.12](http://www.ninjatrader.com/ninjatrader/nt8/NinjaTrader.Install.B12.msi) | [Release Notes](https://ninjatrader.com/es/support/helpGuides/nt8/8_0_0_12.htm) |

**Bug reports and providing feedback**

Support will be driven via our[NinjaTrader Support Forum](http://www.ninjatrader.com/support/forum/forumdisplay.php?f=57" \t "_blank) NinjaTrader 8 section. When running into unexpected behavior please create a bug report in the NinjaTrader support forum's bug report section. Bug reports will be reviewed by our support and development staff inside of a 48 hour period. If you are unable to post to the bug section of the forum please contact [platformsupport@ninjatrader.com](mailto:platformsupport@ninjatrader.com).

tog_minus        [Installation Guide](javascript:HMToggle('toggle','Beta','Beta_ICON'))

|  |  |
| --- | --- |
| **Minimum PC Requirements**  You must meet the below minimum PC requirements to run NinjaTrader 8 Beta.    •Windows Vista (SP2) w/ Platform Update, Windows 7, Windows 8, Windows 10, Windows Server 2008 w/ Platform Update, Windows Server 2008 R2 or later.  •1 (GHz) or faster 32-bit or 64-bit processor  •2 GB RAM  •Microsoft .NET Framework 4.5  •Screen resolution of 1024 x 768 (16-bit color depth)    **Recommended PC Requirements**  NinjaTrader 8 was designed to take full advantage of modern PC architecture. To achieve the highest possible level of performance, NinjaTrader 8 will utilize all available CPU cores and additional memory resources. Depending on your actual usage with NinjaTrader you may need more or less resources as there is no one size fits all recommendation. Note: If you do plan on running strategy optimizations then having additional memory will be of direct benefit, the amount of additional memory needed is proportional to the number of CPU cores available.    •2 (GHz) or faster quad core 64-bit processor  •8 GB RAM  •DirectX10 compatible graphics card  •SSD Hard Drive    **Installation**  Please follow the below steps to get up and running on the beta.    1.If you do not have the Microsoft .NET Framework 4.5 installed you must download and install (Pre-installed on most PC’s and can be downloaded here: [Microsoft .](http://www.microsoft.com/en-us/download/details.aspx?id=30653" \t "_blank)[NET Framework](http://www.microsoft.com/en-us/download/details.aspx?id=30653)).  2.[Download](http://ninjatrader.com/PlatformDirect) and install the NinjaTrader 8 Beta.  3.If you are running a 64-bit PC you will want to start the 'NinjaTrader 8 (64-bit)' version, on a 32-bit PC start the 'NinjaTrader 8 (32-bit)' version.  4.Firewall Software – NinjaTrader contacts our license server on application start up for license key validation. If you have a firewall, spyware or other such software running on your PC, please ensure that you grant NinjaTrader permission to access the internet or you may receive an invalid or license expired message.  5.Platform Activation for Live Trading – If you registered to use the free simulation version, or do not plan on trading live with NinjaTrader 8 Beta, you can skip this step.  oA license key to activate live trading is sent via email within minutes from the time you complete your purchase (check your junk/spam folder if you have not received it)  oStart NinjaTrader, select the menu Help > License Key within the Control Center and enter your license key  6.If you are a user of NijaTrader 7 you have the option to migrate NinjaTrader 7 data. The migration process will import historical data, replay data, instruments, trade history, account connections, market analyzer templates and strategy wizard templates. Any workspaces, chart templates, NinjaScript will unfortunately not be migrated over.  7.If you choose to migrate your connection should be immediately available in the Control Center -> Connections menu. If you did not migrate and need to setup your first connection please review the [connect to your account](https://ninjatrader.com/es/support/helpGuides/nt8/connecting_to_youraccount.htm) page to get started.     |  | | --- | | **Note:**Simulation color by default is transparent in NinjaTrader 8. If you migrate your settings from NinjaTrader 7 then simulation color will be maintained as is from NinjaTrader 7. You can change simulation color via tools > Option > Trading Category > Simulation Color. | |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?beta.htm#Beta)

tog_minus        [Expectations during beta](javascript:HMToggle('toggle','Betaexpectation','Betaexpectation_ICON'))

|  |  |
| --- | --- |
| **Help guide status**  We have made significant progress on the Help Guide and will continue to work on it throughout the beta process but it is still “Work In Progress”.  Any section marked at the top of the page with "Under Construction" is still being worked on and my contain errors; sections marked as "Needs Review" should be considered complete, but may change in the future.  Please feel free to report feedback to [platformsupport@ninjatrader.com](mailto:platformsupport@ninjatrader.com)    **Historical data status**  If you are connecting to CQG, Continuum or the FXCM then you are using historical data provided by NinjaTrader. Our Historical Data Servers for NinjaTrader 8 have been redesigned and are also in a beta testing period. As such, it is possible to experience outages and gaps in data.    **Ongoing Performance Optimization**  We are not 100% complete in our User Interface performance optimizations and as such, you may notice on large workspaces and/or slower hardware areas that will be improved.    **Live Trading with NinjaTrader 8 Beta**  There are risks associated with live trading on any [electronic trading platform](https://ninjatrader.com/es/support/helpGuides/nt8/risks_of_electronic_trading_wi.htm). While the NinjaTrader 8 live trading components are feature complete, there could be some undiscovered bugs which could adversely impact your ability to trade live. As such, the risks associated with live trading should be heavily considered.  We do have users that are reporting positive experiences with live trading and we have been using NinjaTrader 8 in a production capacity with CQG/Continuum on our own order desk for the past eight months. We continue to work through reported issues in anticipation of giving our users the best possible experience when NinjaTrader 8 is released into production.    **Reporting Bugs**  When running into unexpected behavior please note the following if possible and include it in your bug report as it will increase our ability to diagnose and resolve the issue.    •The time and date the bug occurred.  •Step by step actions (if any) that you took prior to the bug.  •Screen shot or video of the bug.     |  | | --- | | **Tip**:  Internally we use a free service called ['Jing'](https://www.techsmith.com/jing.html) which is very helpful to allow you to take a screen shot or record a video and share it. |     **How long will the beta last?**  Beta is a fluid process where we will continue to stay in beta for as long as we are receiving critical bug reports. We will be sure to stay in contact with you to provide you with new builds and keep you updated as to the progress of the beta. |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?beta.htm#Betaexpectation)

tog_minus        [NinjaScript](javascript:HMToggle('toggle','BetaHistoricalData','BetaHistoricalData_ICON'))

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **What's New with NinjaScript in NinjaTrader 8**  One of the primary goals of NinjaTrader 8 is to expand the number of NinjaScript objects, methods, and properties that are documented and officially supported, in order to provide our customers more control and flexibility than ever before. To that end, we have increased support across the board and unlocked additional NinjaScript types.     |  |  | | --- | --- | | **NinjaScript Type** | **Description** | | [Add Ons](https://ninjatrader.com/es/support/helpGuides/nt8/add_on.htm) | Add Ons run as a window is created. Add Ons can be used to extend the functionality of an existing NinjaTrader window, or to create your own window that enjoys all the features of others, such as workspace persistence and styling. | | [Bars Type](https://ninjatrader.com/es/support/helpGuides/nt8/bars_type.htm) | Bar Types are now fully supported in NinjaTrader 8. Although we allowed them to be customized in NinjaTrader 7, we offered no official support and provided only a small number of available slots. We have addressed both of these items with NinjaTrader 8. | | [Chart Style](https://ninjatrader.com/es/support/helpGuides/nt8/chart_style.htm) | Chart Styles are now fully supported and customizable. | | [Drawing Tools](https://ninjatrader.com/es/support/helpGuides/nt8/drawing_tools.htm) | You can now create your own custom drawing tools for use on NinjaTrader charts. | | [Import Type](https://ninjatrader.com/es/support/helpGuides/nt8/import_type.htm) | Allows you to customize the Historical Data Importer to allow different data formats to be imported in NinjaTrader 8. | | [Indicator](https://ninjatrader.com/es/support/helpGuides/nt8/indicator.htm) | Indicators were fully supported in NinjaTrader 7, and continue to be fully supported with increased functionality in NinjaTrader 8. | | [Market Analyzer Column](https://ninjatrader.com/es/support/helpGuides/nt8/market_analyzer_column.htm) | Market Analyzer Columns were allowed to be customized in NinjaTrader 7, but there was no official support or documentation. We have added these as fully supported in NinjaTrader 8. | | [Optimization Fitness](https://ninjatrader.com/es/support/helpGuides/nt8/optimization_fitness.htm) | An Optimization Fitness allows you to expand the optimizer by defining new ways to find the best strategy performance. | | [Optimizer](https://ninjatrader.com/es/support/helpGuides/nt8/optimizer.htm) | We support the standard (Brute Force) optimizer and a Genetic Optimizer, and we've opened up full support for writing your own optimizer. | | [Performance Metrics](https://ninjatrader.com/es/support/helpGuides/nt8/performance_metrics.htm) | A new NinjaScript component which allows you to expand performance metrics and define your own custom metrics. These will be viewable in the Trade Performance window and the Strategy Analyzer. | | [Share Service](https://ninjatrader.com/es/support/helpGuides/nt8/share_service.htm) | A new NinjaScript component to allow the creation of additional sharing service adapters. By default we support email, Facebook, Twitter, and Stocktwits. | | [Strategy](https://ninjatrader.com/es/support/helpGuides/nt8/strategy.htm) | Strategies were fully supported in NinjaTrader 7, and continue to be fully supported with increased functionality in NinjaTrader 8. | | [SuperDOM Column](https://ninjatrader.com/es/support/helpGuides/nt8/superdom_column.htm) | New NinjaScript component that allows you to create your own custom column to be added to a SuperDOM window. |       **Converting NinjaScript From NinjaTrader 7 to NinjaTrader 8**  In line with the goals of providing expanded support, consistency, performance, and improved functionality, we found it necessary to make code breaking changes. This means that existing scripts for NinjaTrader 7 will have to go through a porting process to get the code compiled and working in NinjaTrader 8. The changes required are typically mild and primarily constitute renaming methods and moving code to the new "State" system. We have compiled an overview guide of the new concepts in NinjaTrader and a list of the [code breaking changes](https://ninjatrader.com/es/support/helpGuides/nt8/code_breaking_changes.htm) to help you get started. We are very welcome to feedback to help improve these documents. If you run into issues porting a script, please post in our beta forums so that we can assist and improve documentation where needed.    **DLL Protection with CLI Secure**  For those who distribute NinjaScript assemblies, we upgraded our protection to use the latest Agile.net 6.3 protection. As the protection is stronger, the service is no longer free, and we have negotiated a discounted price of $100 for protecting NinjaScript assemblies. Please visit the [Protection / DLL Security](https://ninjatrader.com/es/support/helpGuides/nt8/protection_dll_security.htm) section of the help guide for more information. |

[permalink](https://ninjatrader.com/es/support/helpGuides/nt8/index.html?beta.htm#BetaHistoricalData)