Charts with Rest API

Last updated: May 30, 2014

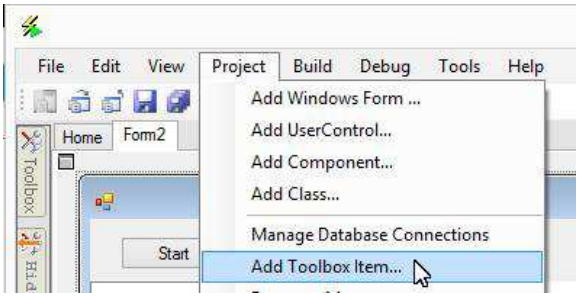
Contents

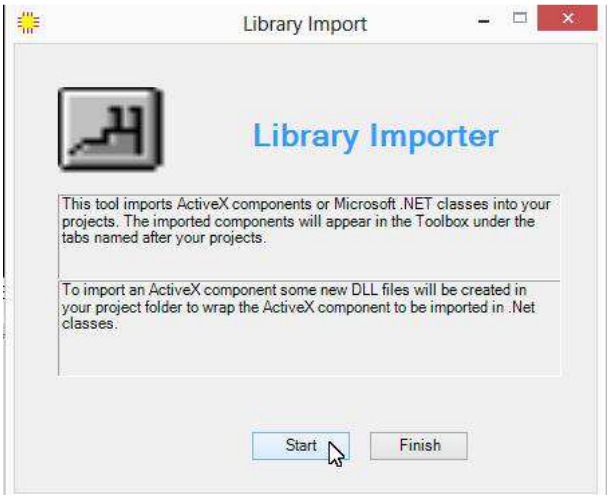
Introduction

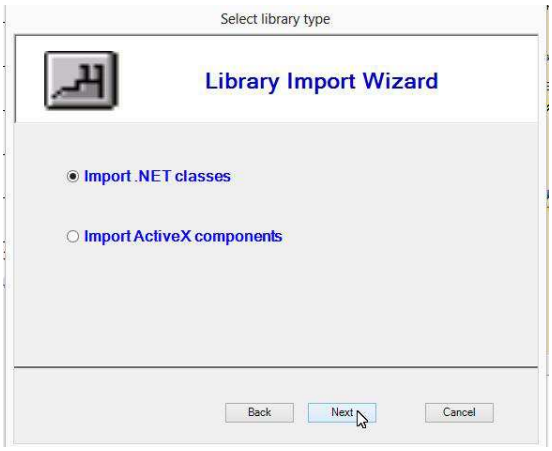
See http://www.limnor.com/support/RestAPIsamples.pdf for how to load Rest API DLL into your project, and samples of using the Rest API. Below are some chart samples using data from Rest API. Please read entire chapter “Price Streaming” in the above document to get important information on how to using price streaming before trying the samples in this document.

Add Chart Control

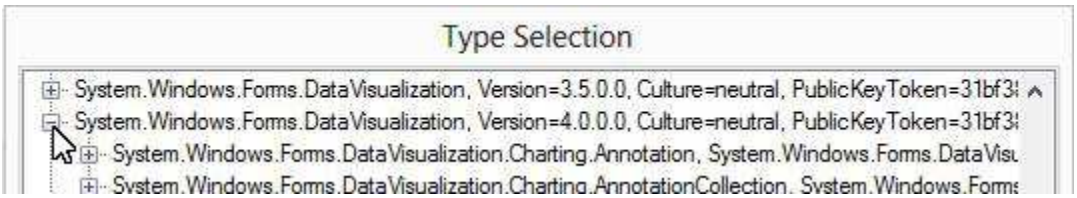
While a form is opened in the designer, select menu “Add Toolbox Item …”



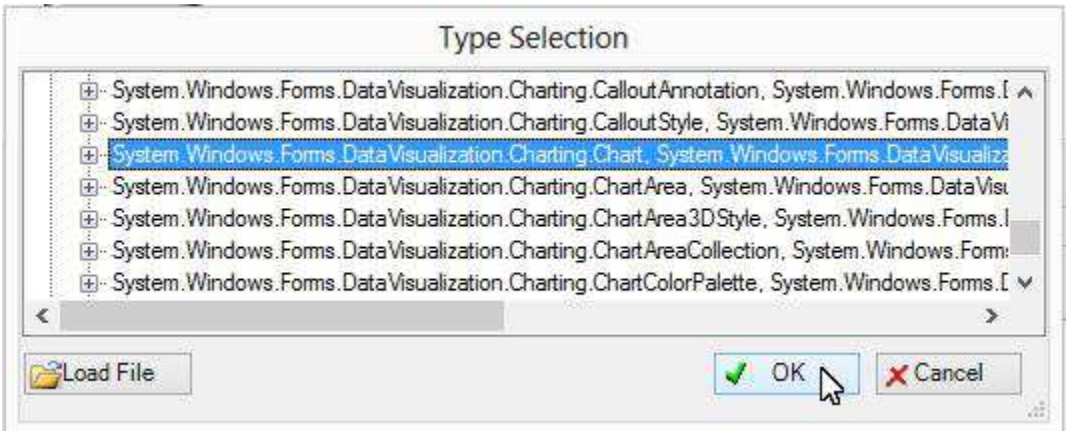


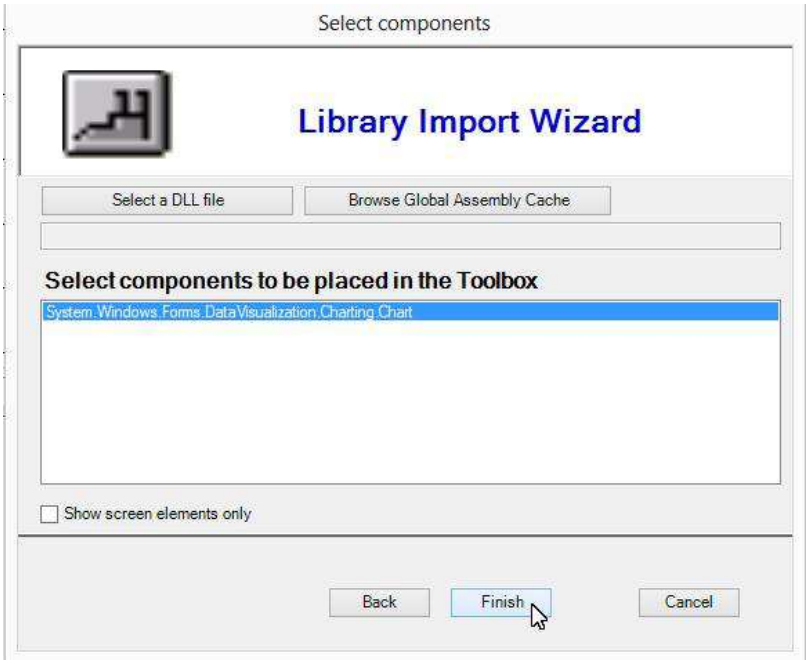


Suppose we want to use the Chart control for .Net 4:

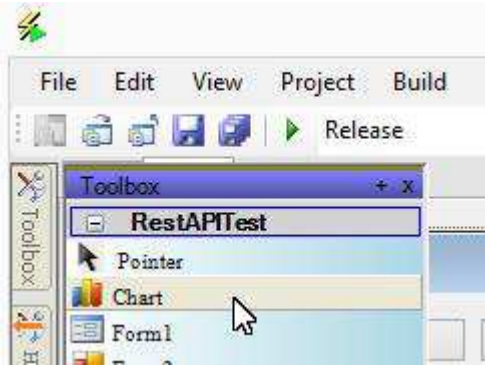


Select Chart:





The Chart control appears in the Toolbox:

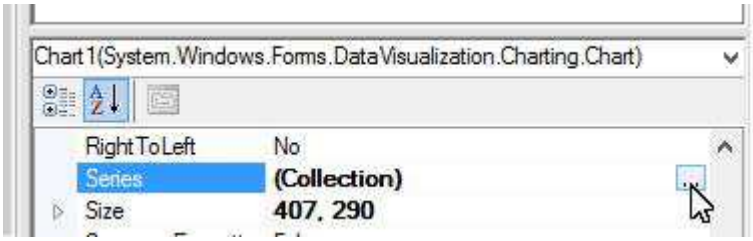


We may drop it to forms to use it.

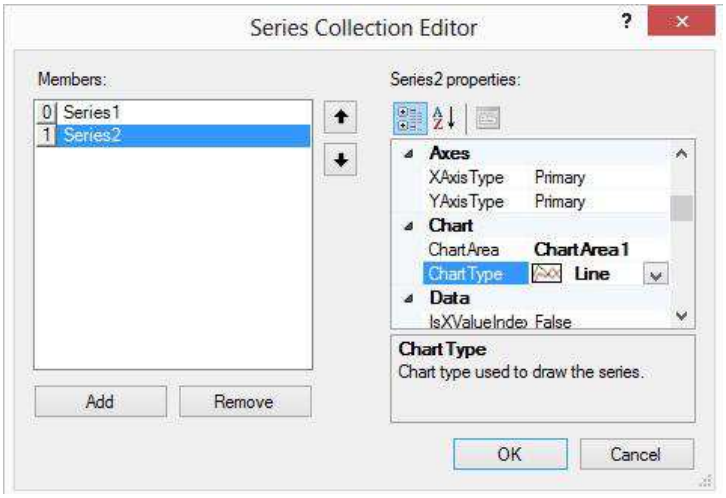
Live Prices

Suppose we want to use a Chart to show live prices. We may create two data series, one for “Ask prices” and the other for “Bid prices”.

Create Series

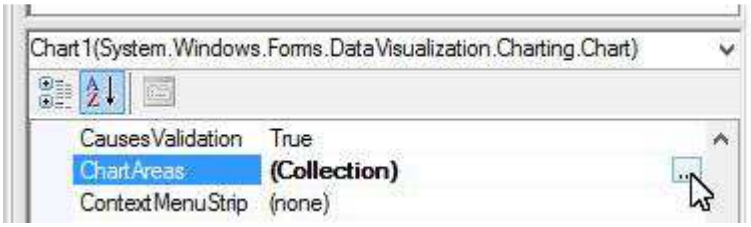


Set chart type to Line:

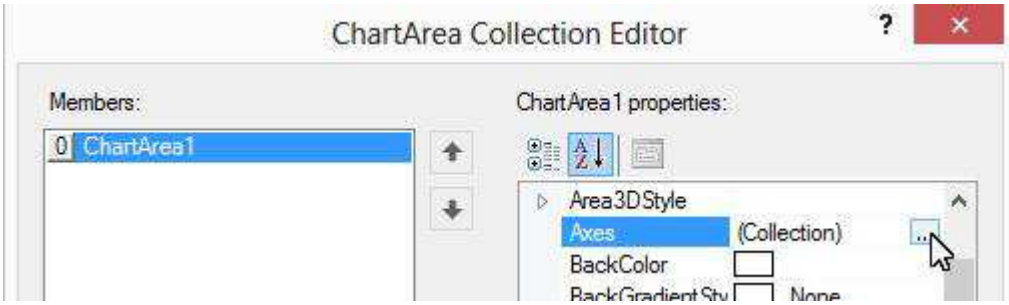


Set Value Scope

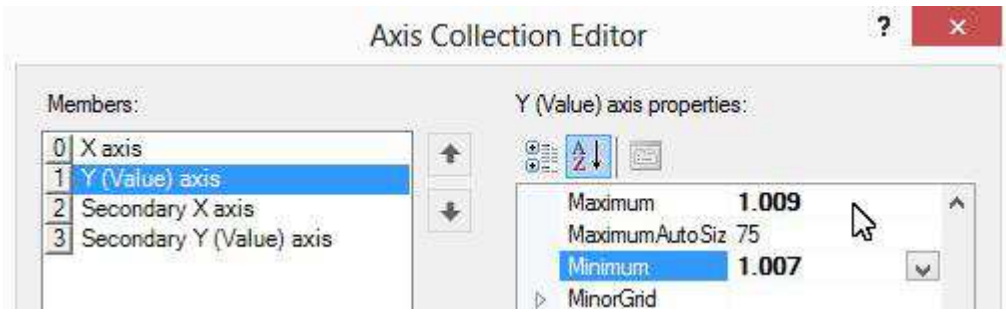
Suppose we want to show prices in a range of 1.007 and 1.009.



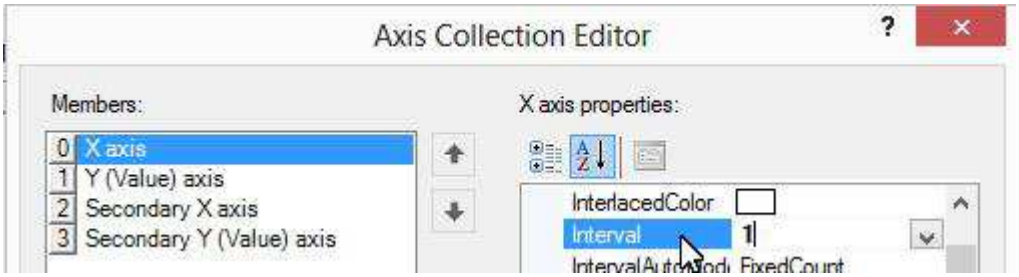
Adjust Axes:



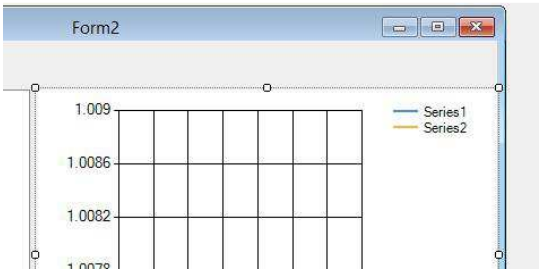
Set the Maximum and the Minimum for the Y-axes:



Set the Interval of X-axes to 1:

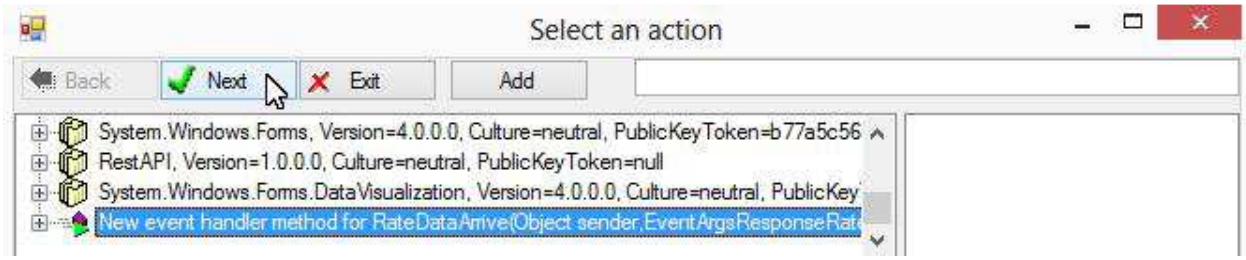


This is our chart control on a form:



Show Live Data in Chart

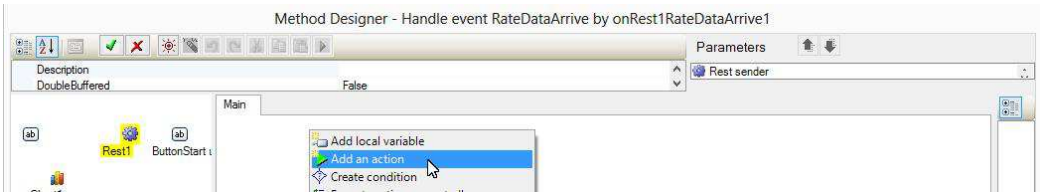
We'll handle event RateDataArrive to pass live data to the chart.



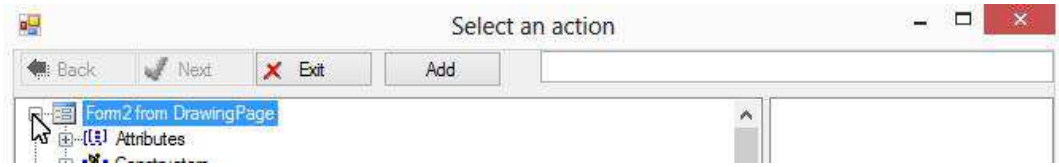
A Method Editor appears for adding actions to be executed at the event.

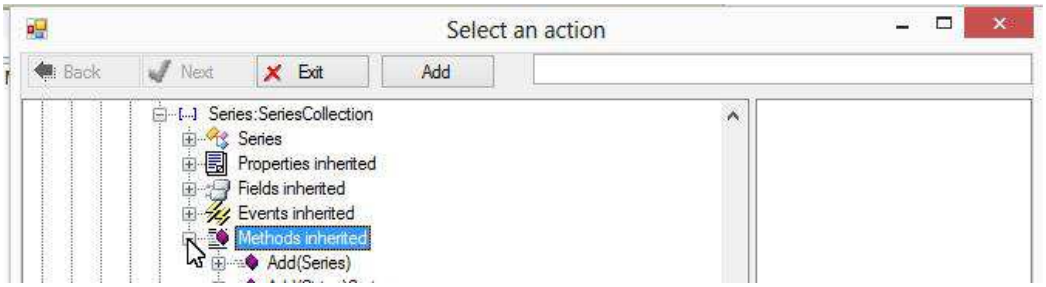
Pass "Ask Price"

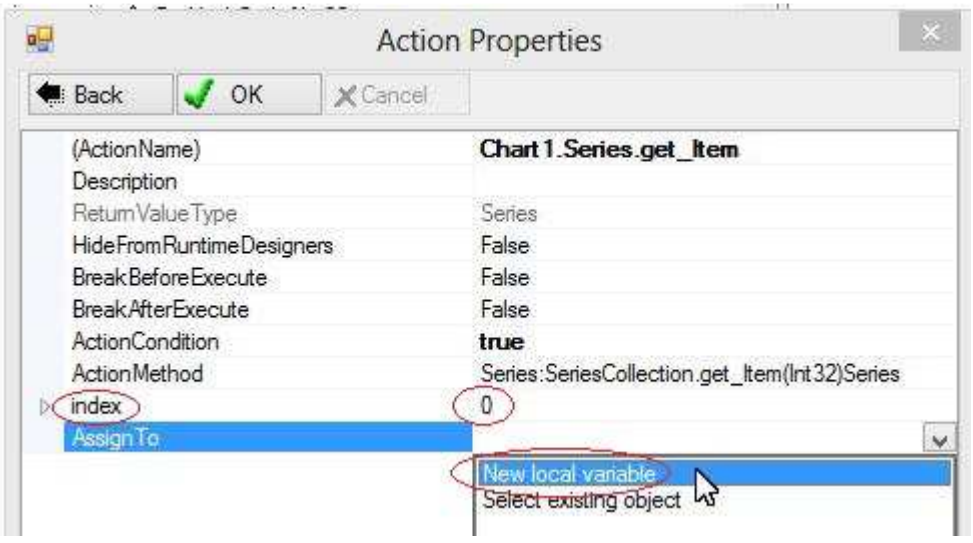
Add an action to get Series1 which is for showing "Ask" prices:



Select get_Item method of the Series property of the chart control:

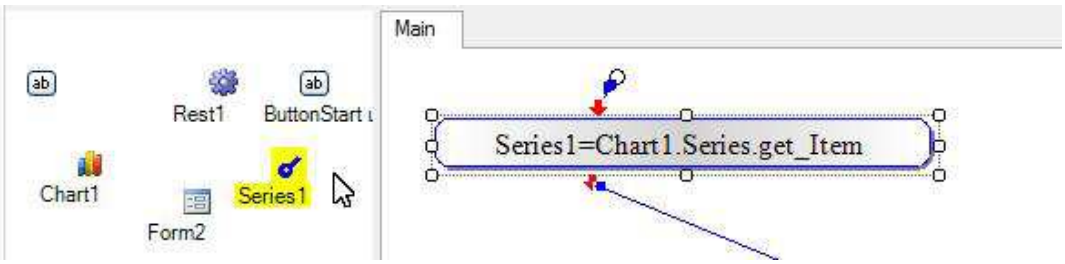






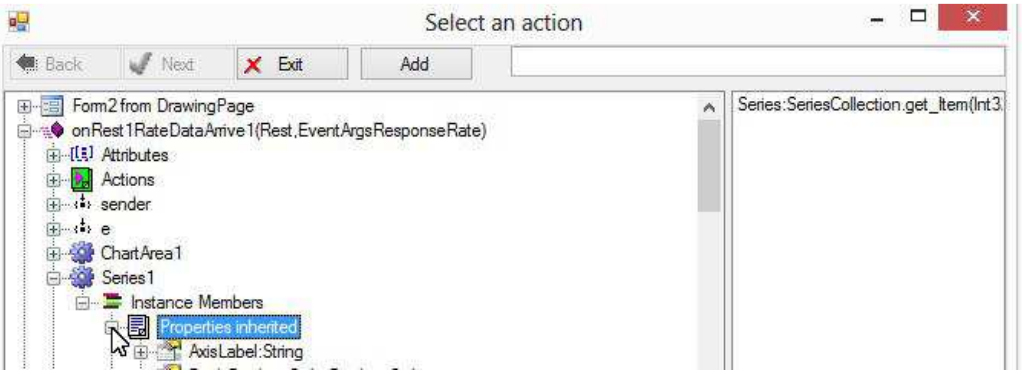
Note that 0 for “index” represents the first data series, Series1. The result of the action will be saved in a variable.

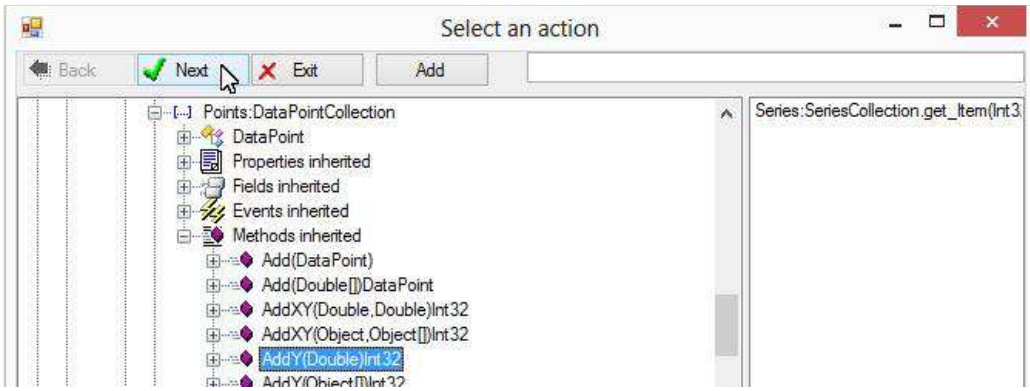
The Action and the variable appear.



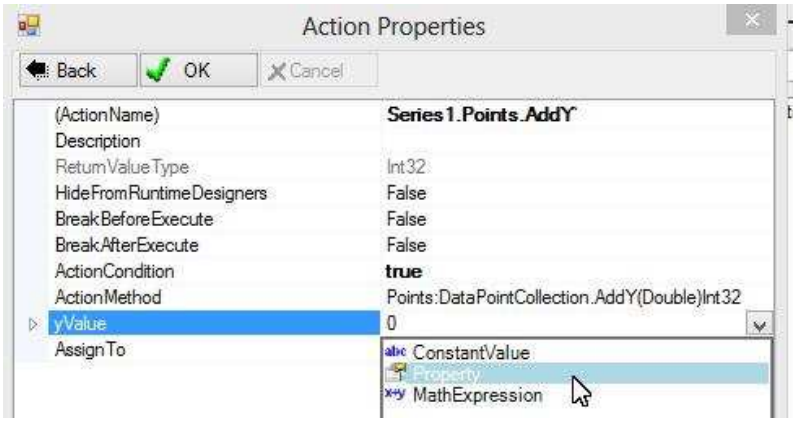
Each data series has a property Points, which has an AddY method for adding value to the series:

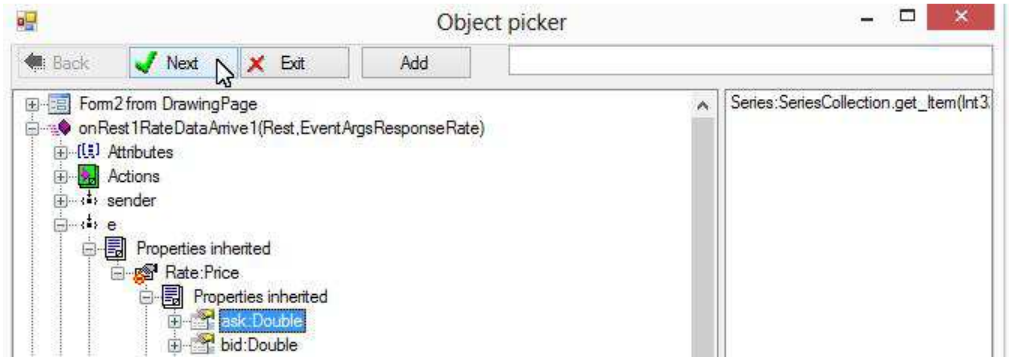
Select AddY of the Points of Series1:





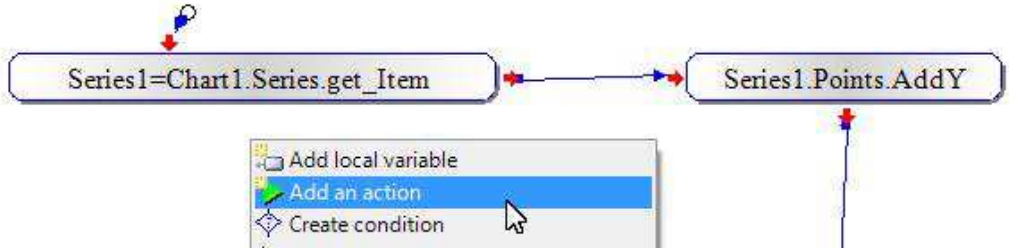
Use the “ask” price for the “yValue”:

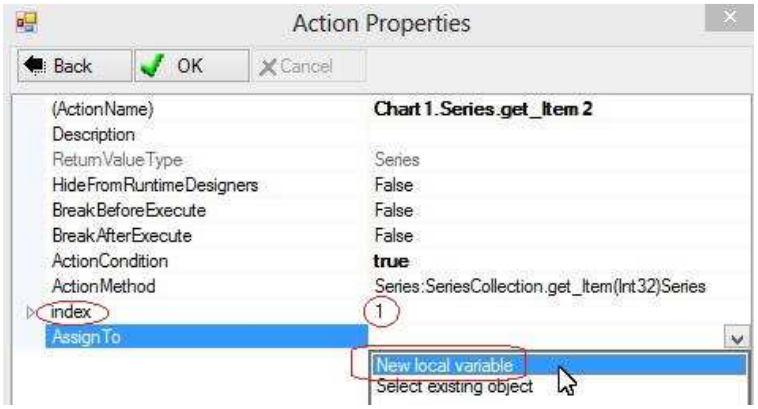




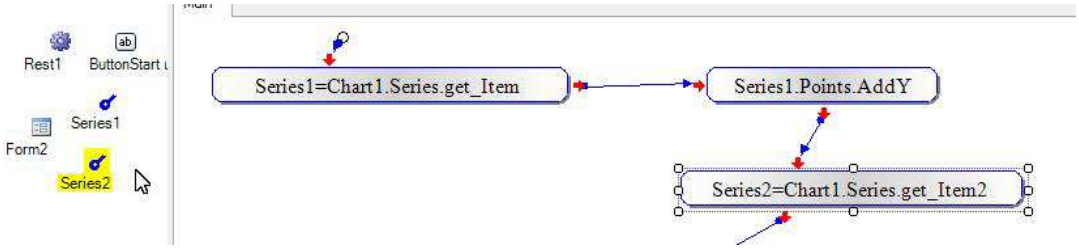
Pass "Bid Price"

Use "get_Item" to get the data series for "Bid" prices:

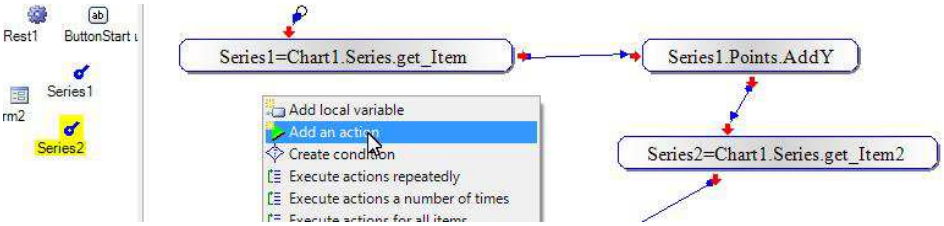


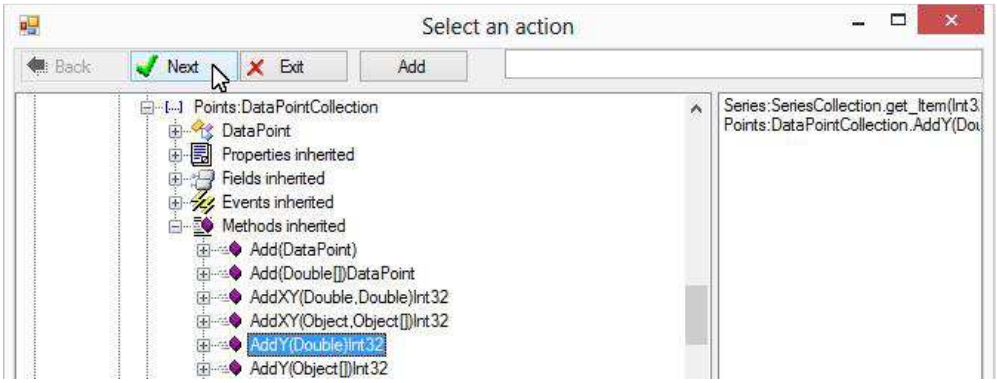


Note that this time we use 1 for “index” to indicate that we want to get the second data series. The action and the new variable appear.

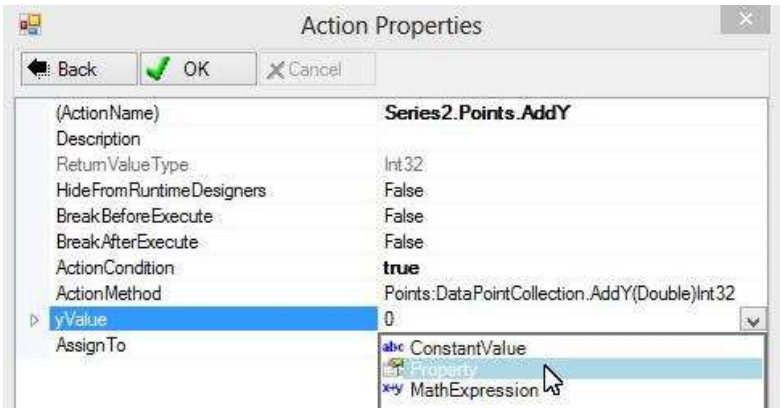


Use AddY to pass “bid” price:

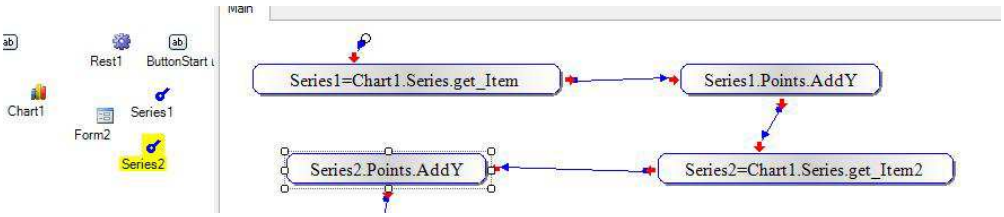




Use “bid” for “yValue”:



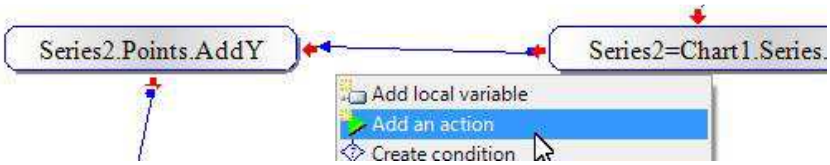
We thus show “ask” and “bid” in the chart:

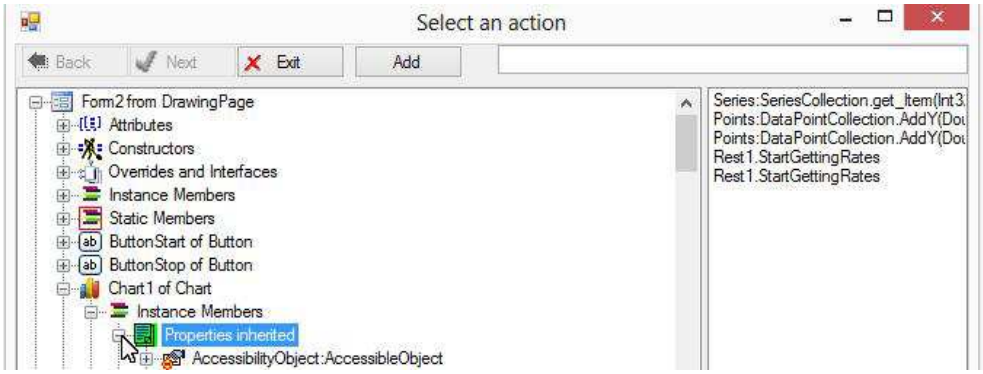


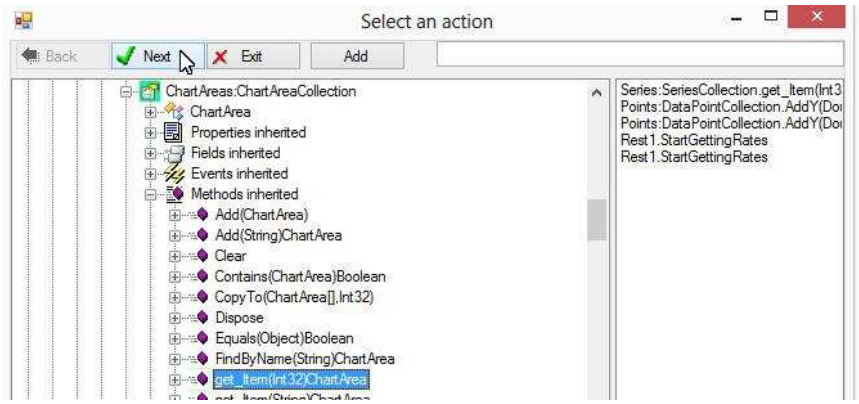
Remove old data

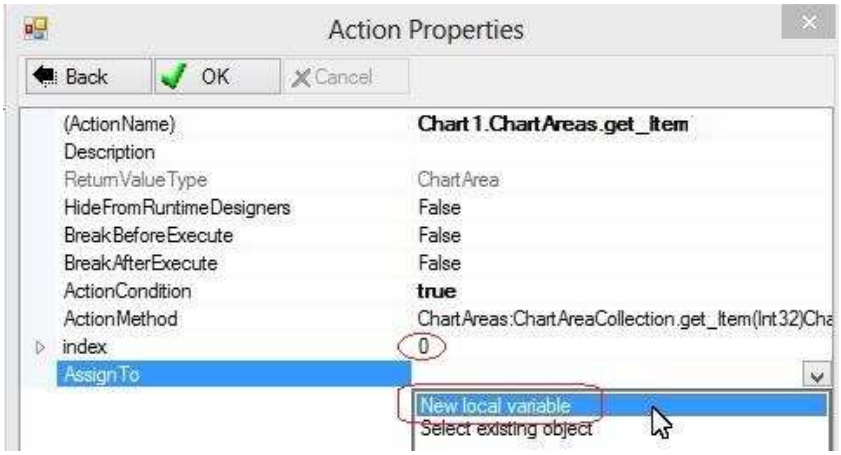
While the new data keep coming into the chart, we may want to remove older data from the chart. We can do it by setting the minimum value of the X-axes.

First we need to get the chart area so that we may access the x-axes.

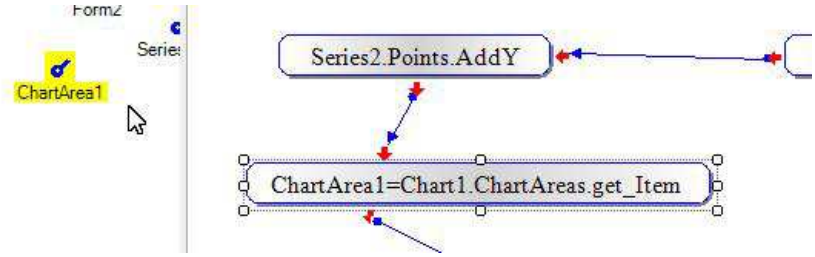




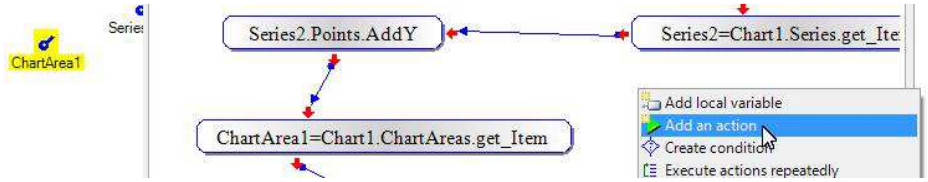


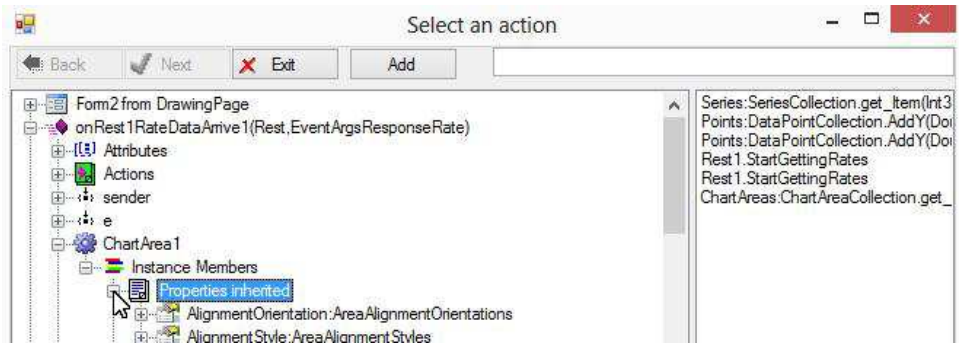


The action and the variable appear.

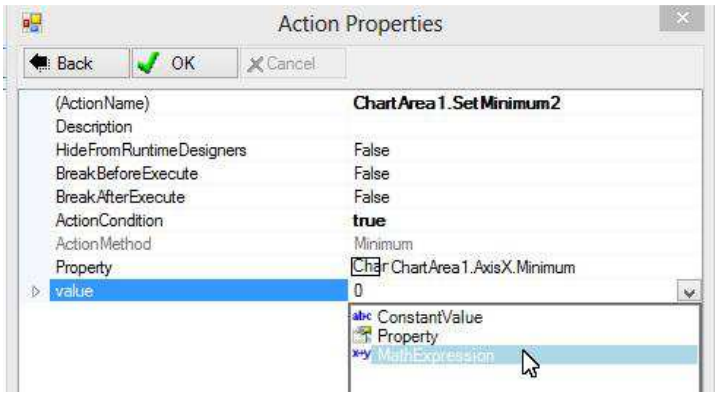


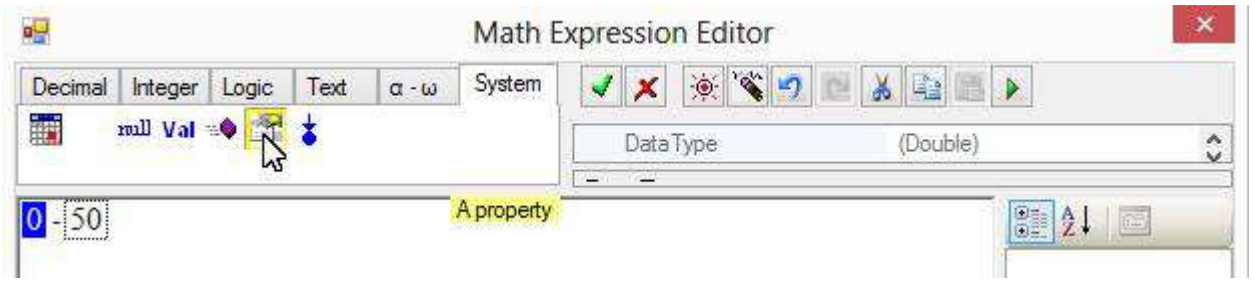
Add another action to set the minimum value of the x-axes:

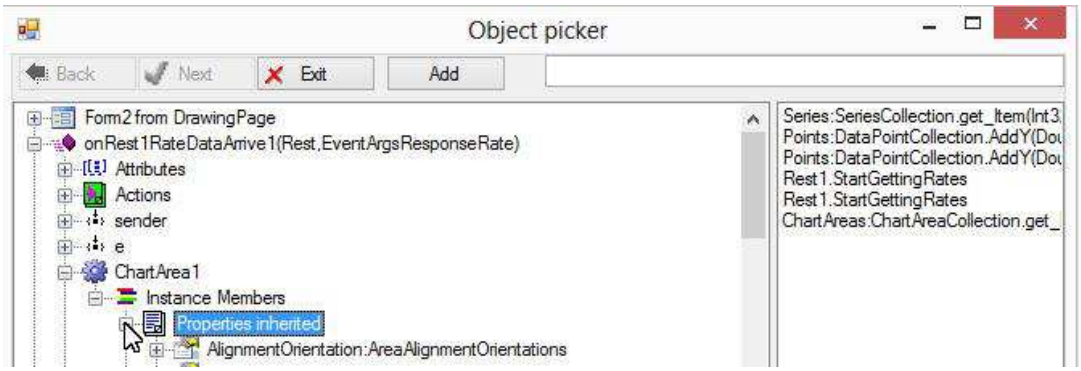


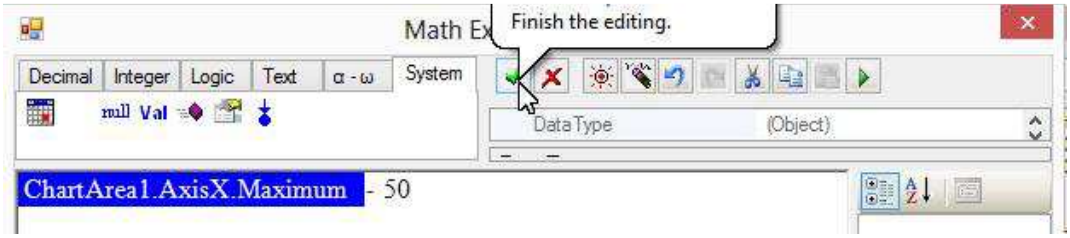


Suppose we want to show 50 values on the chart. We calculate the minimum value by the maximum value minus 50. Use an expression to set the minimum value:

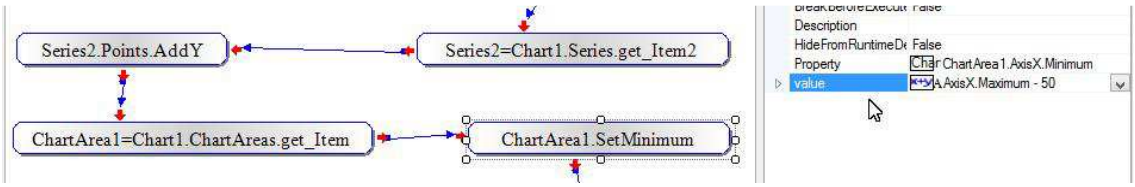






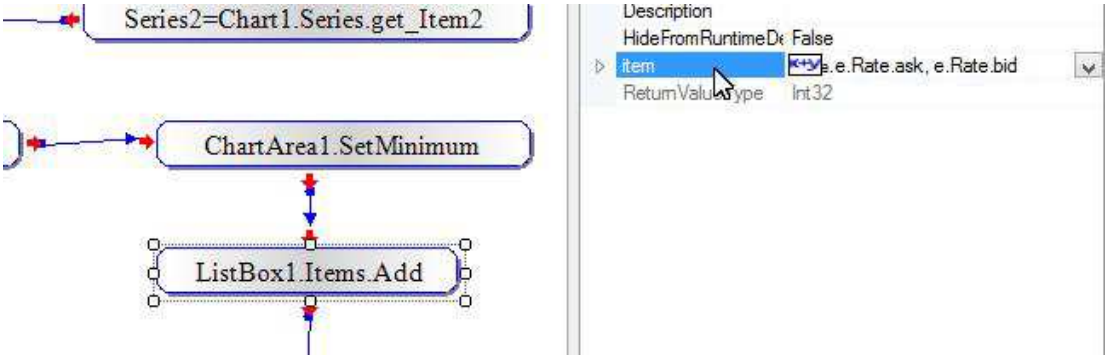


Link the action to the last action:

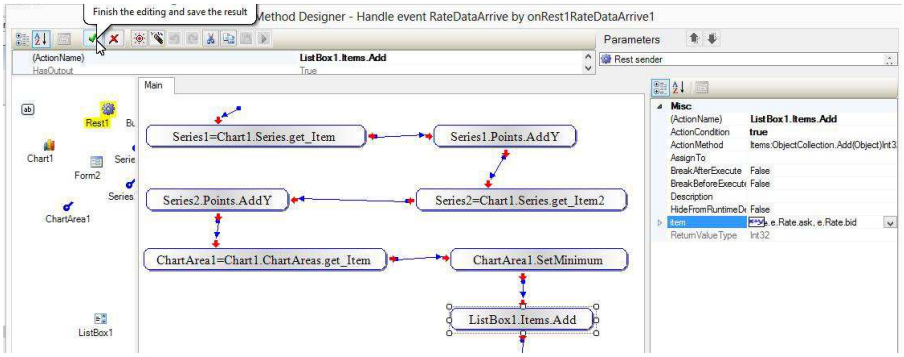


Show values in a list box

We may also show the live data in a list box:

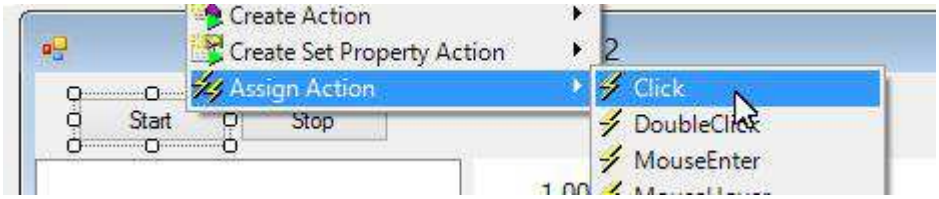


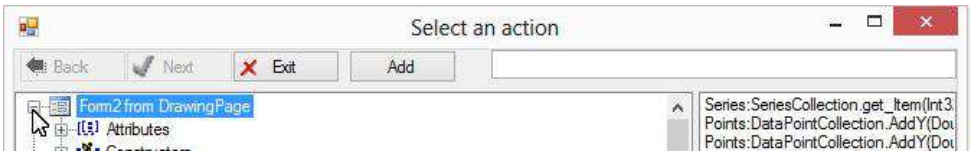
That is all for handling RateDataArrive event:

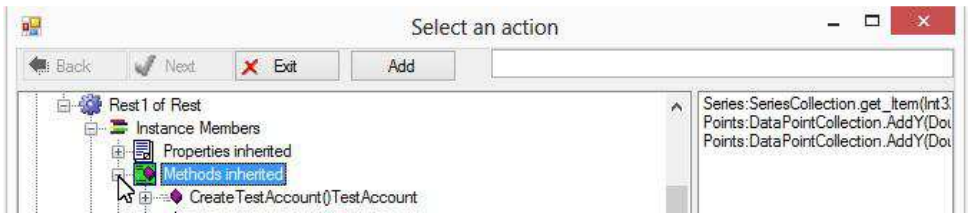


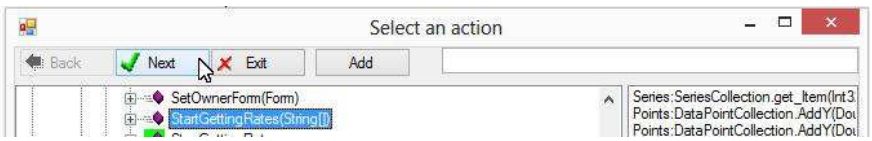
Start/Stop Price Streaming

We use a button to do that:

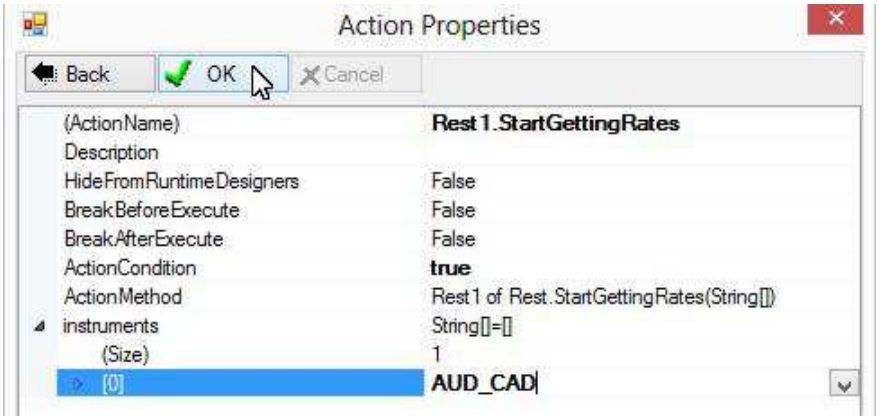






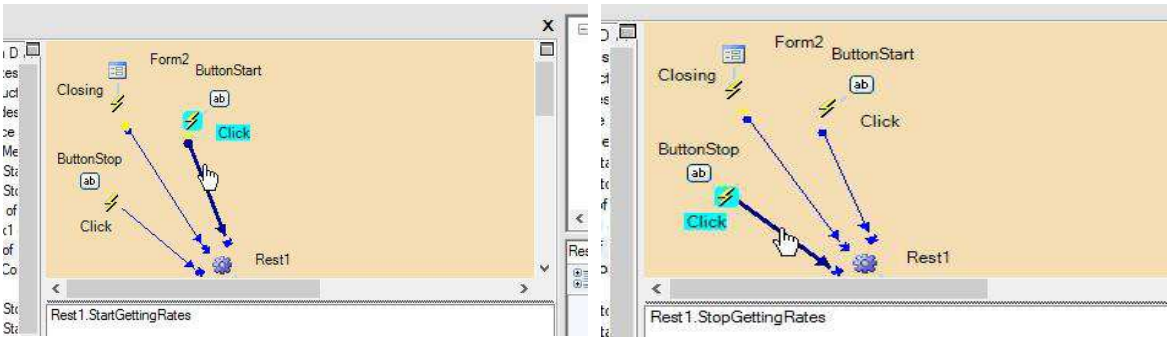


In this sample we only get “ask” and “bid” for “AUD_CAD”:



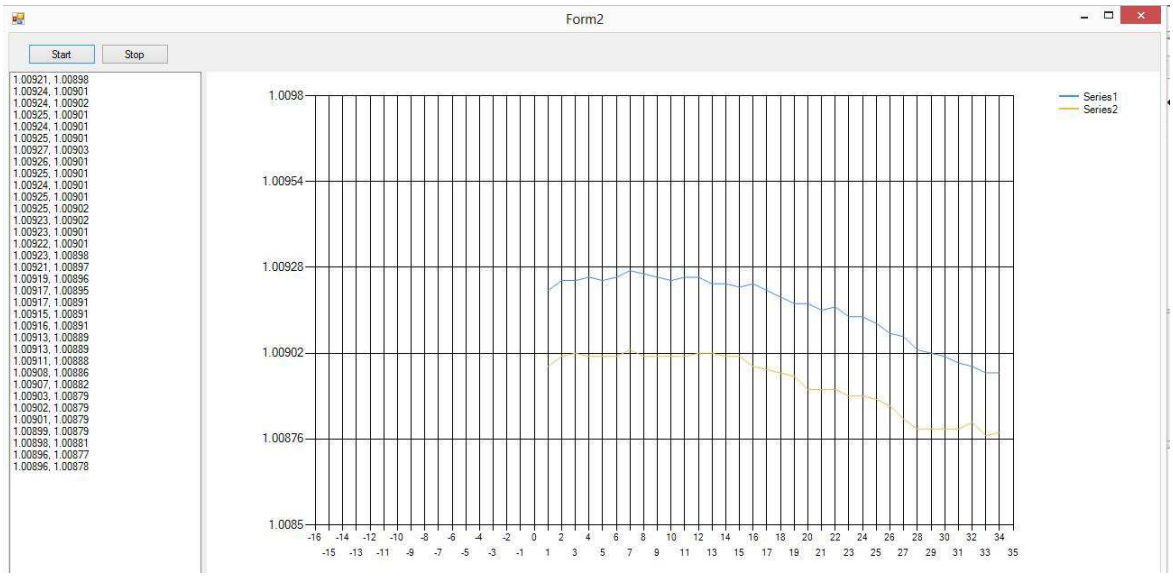
The action is created and assigned to the button.

To stop price streaming, we create a StopGettingRates action and assign it to a button and also the Closing event of the form.



Test

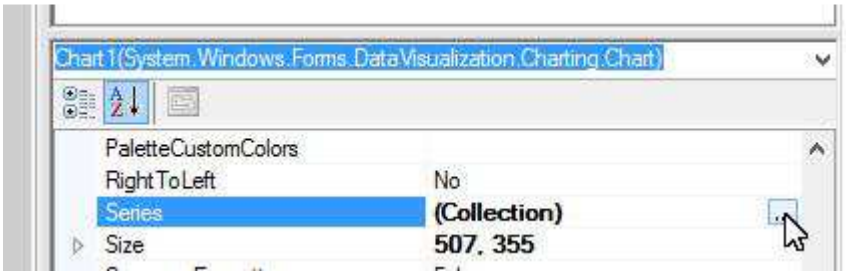
Compile and run the application. The form appears. Click the Start button. Live data are getting into the chart.

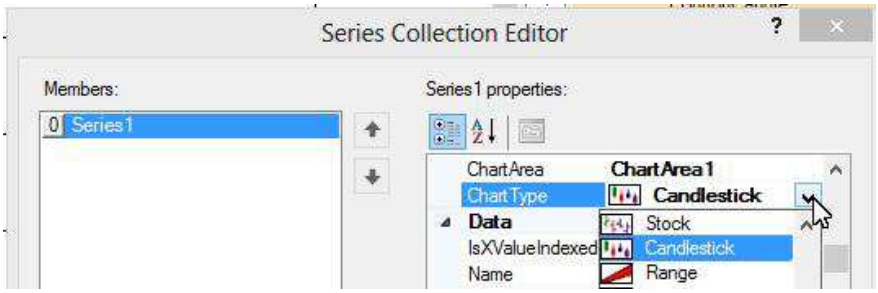


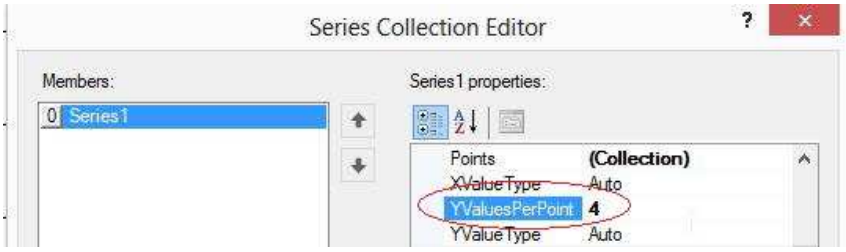
Candle Stick Chart

The Rest API has a GetCandles2 method to get (high, low, open close) prices. Chart control has a Candlestick chart type. We may use a Chart control to display candle sticks from the Rest API.

Use Candlestick Chart Type

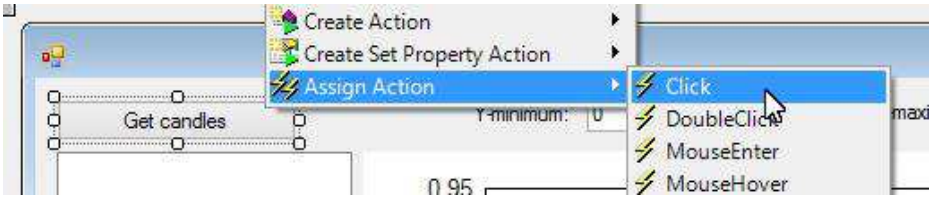




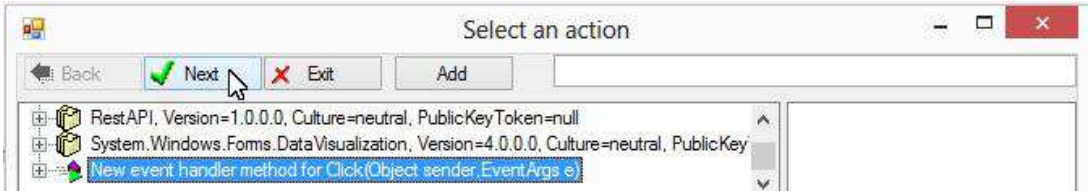


Show Candlestick

Use a button to do it:

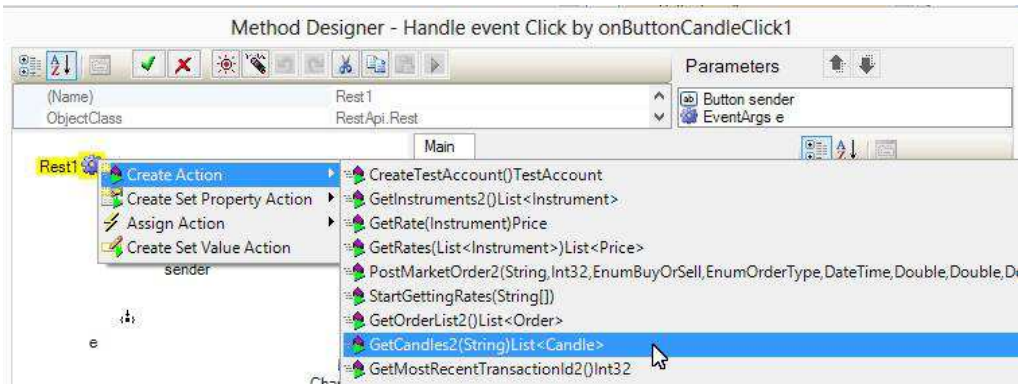


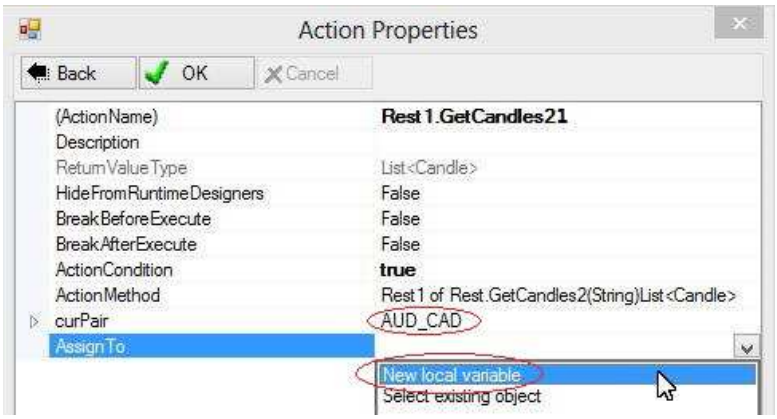
Create an event handler:



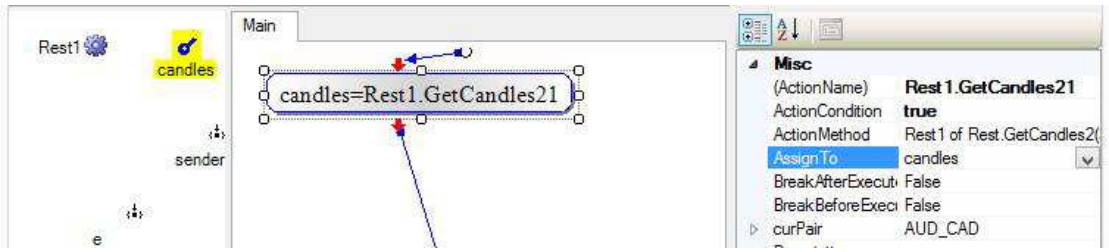
Get Candle data

Use GetCandles2 to get candlestick data:



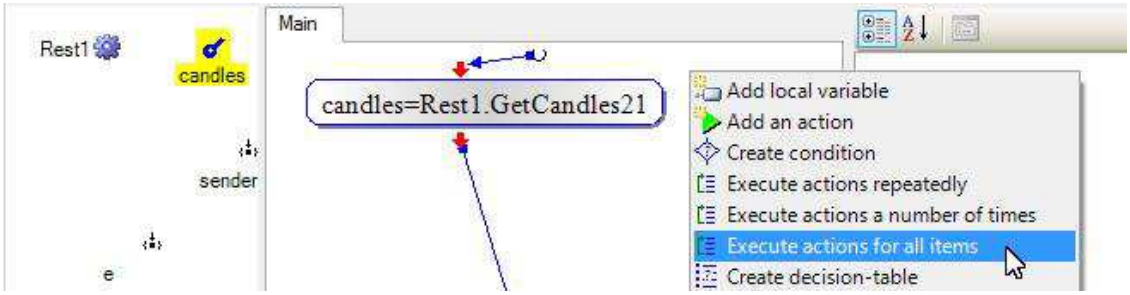


The action and a variable appear.



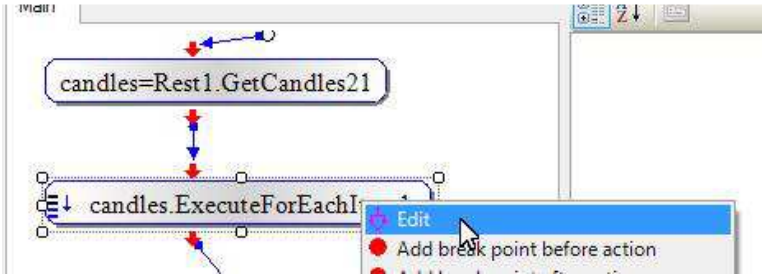
Go through candles data

Create an "Execute actions for all items" action to go through every candle data:



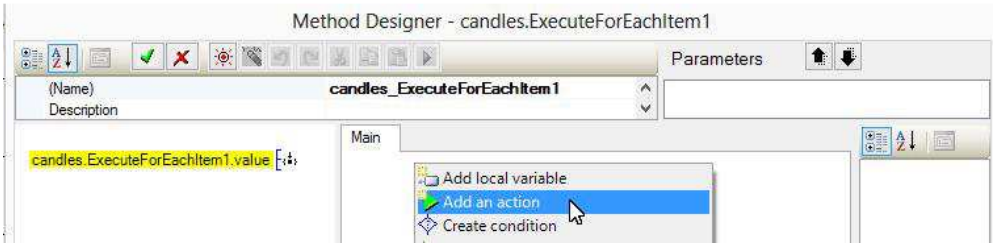
Select variable "candles":

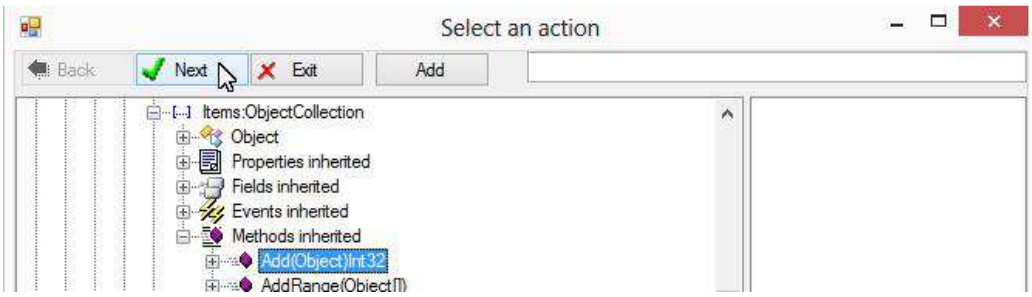
Edit the action to create actions to process each candle data:

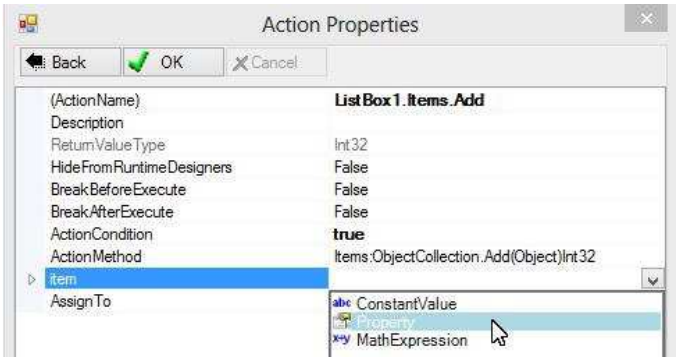


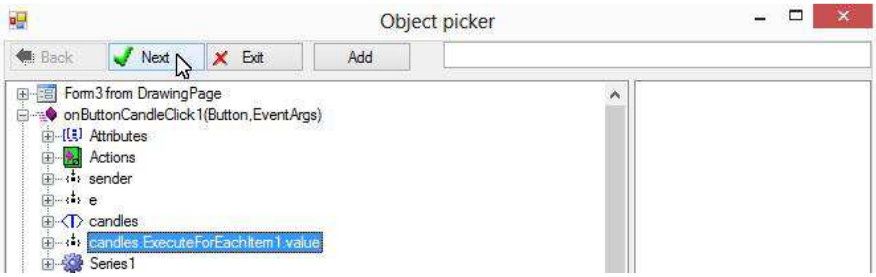
A Method Editor appears. Note that the "value" variable represents the candle data to be processed.

Show candle in list box



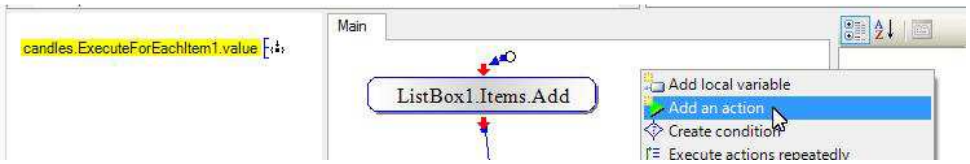


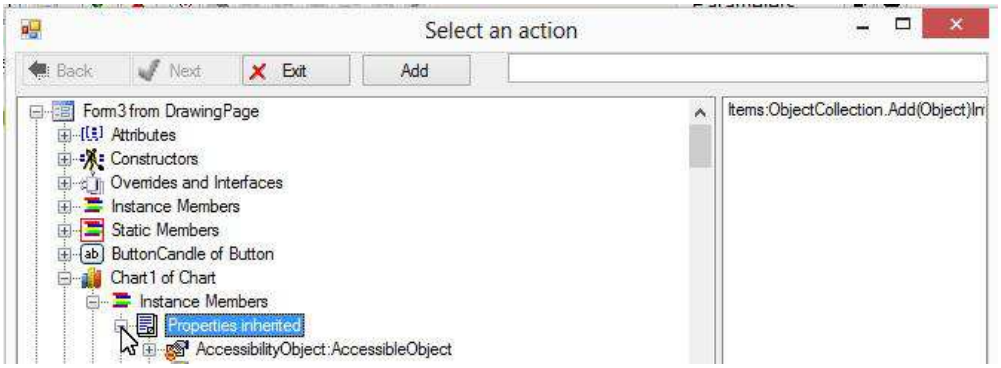


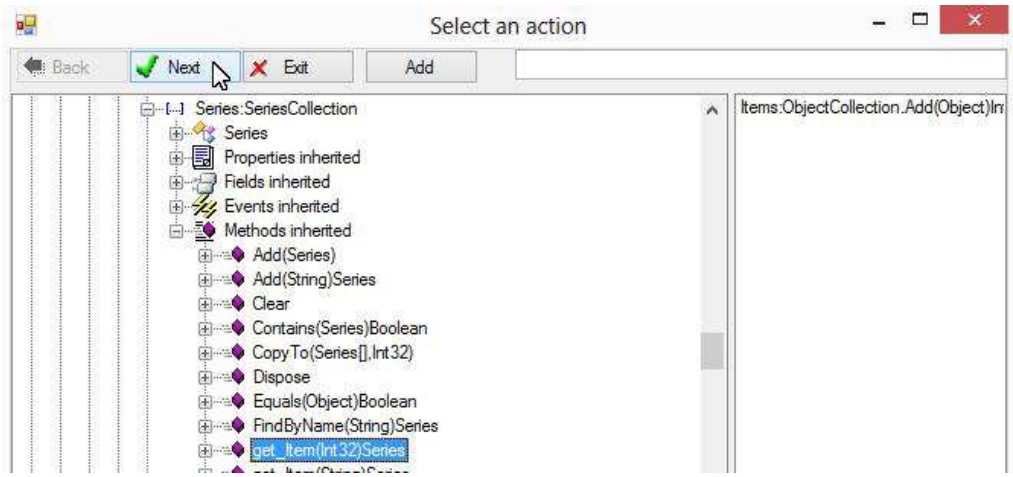


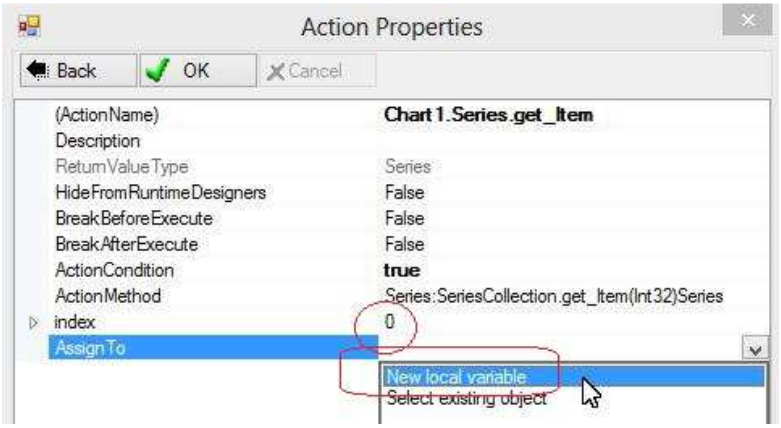
Get chart series

To add the candle data to the chart, get the data series:

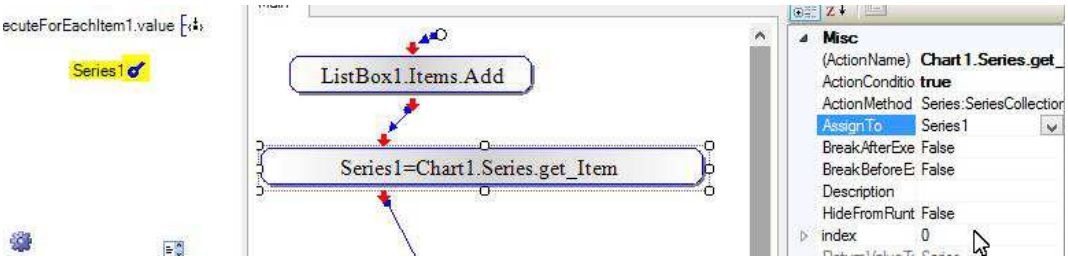




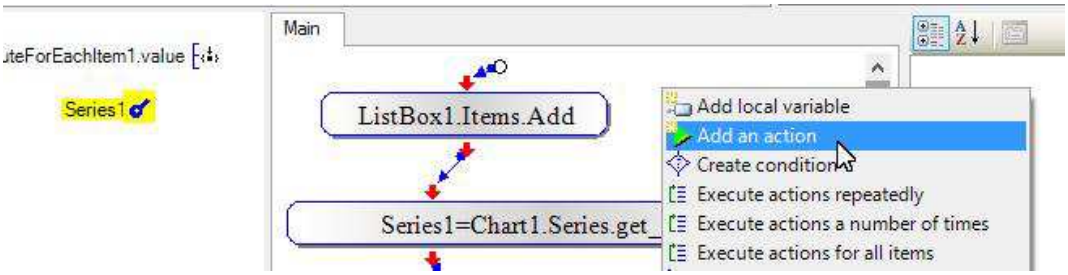


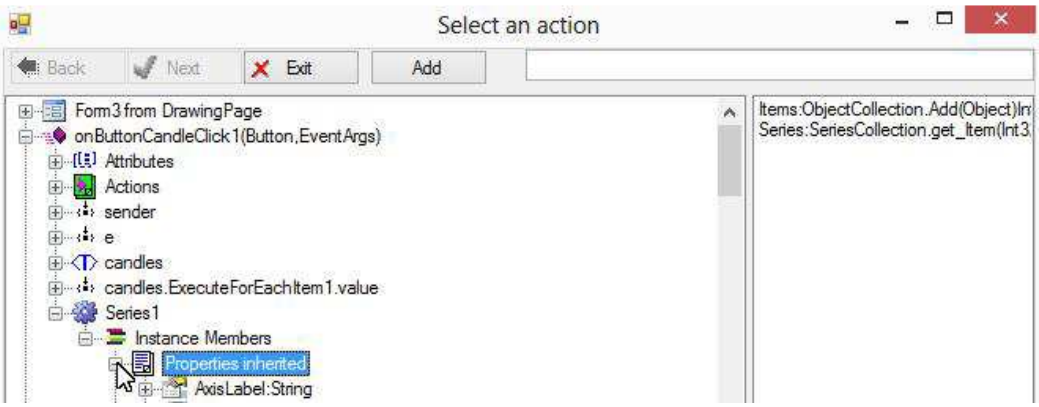


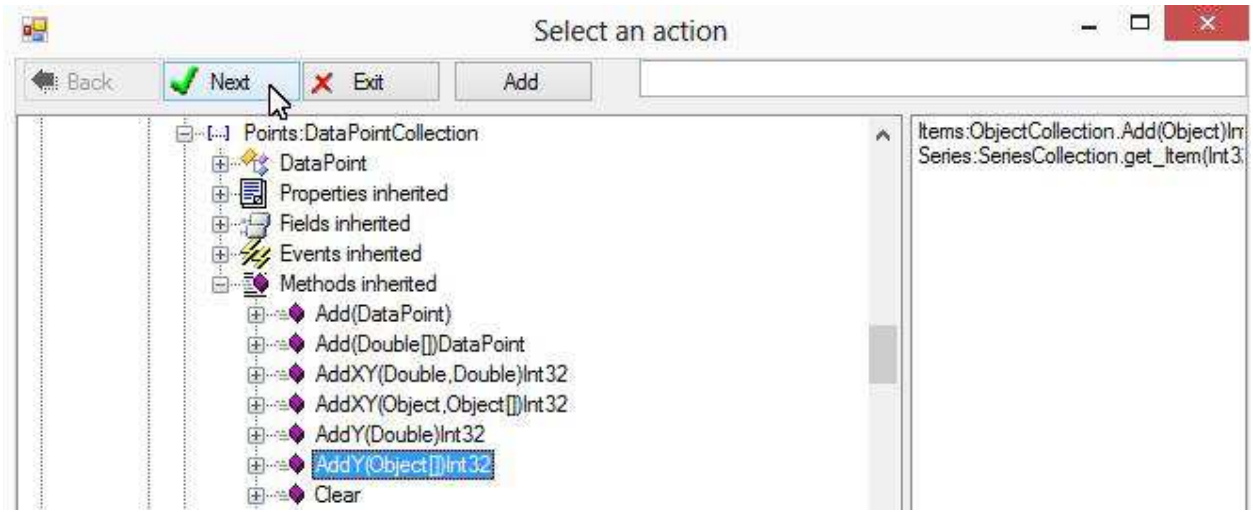
Note that 0 for “index” represents the first data series. Click OK. The action and a variable appear:



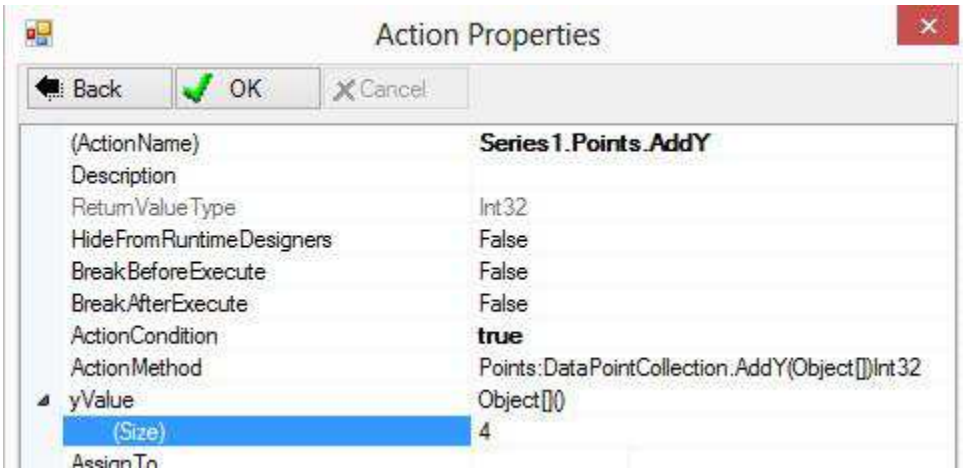
Add candle to series

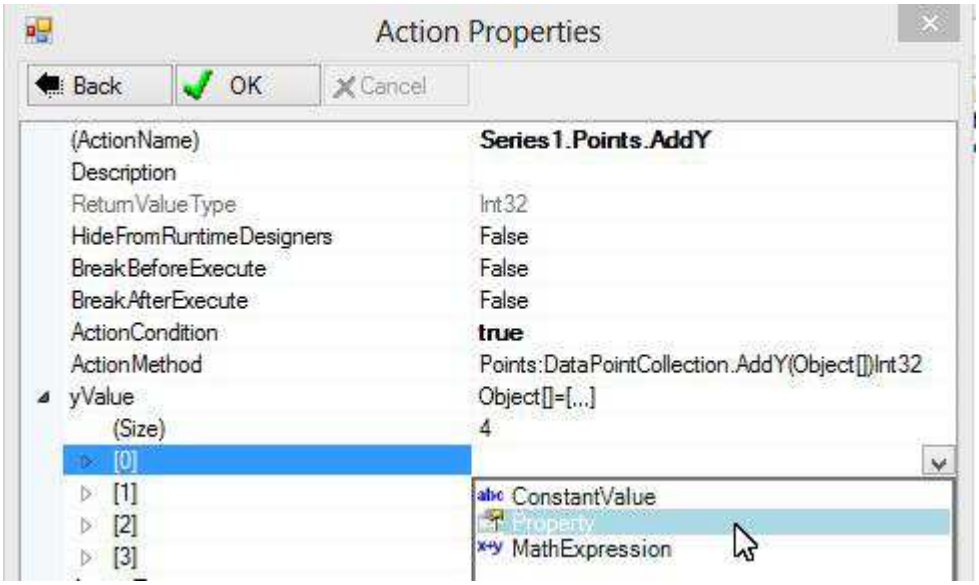


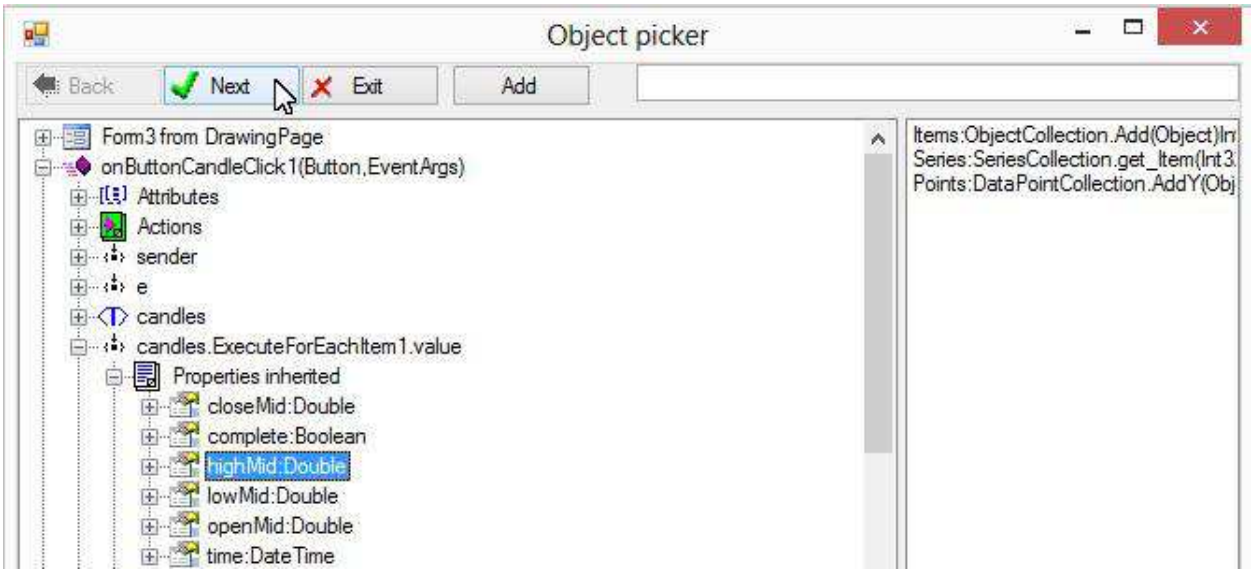


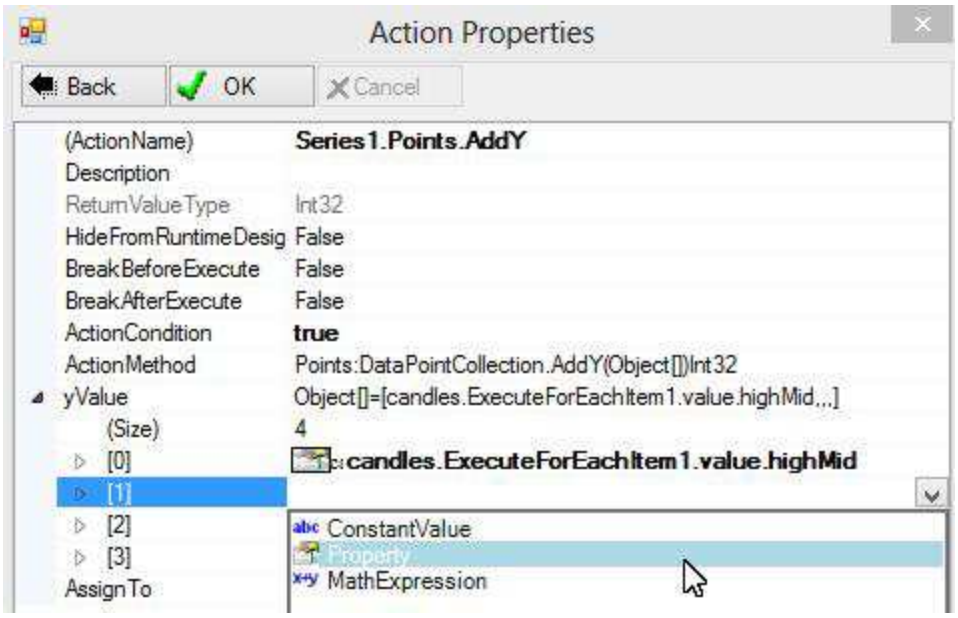


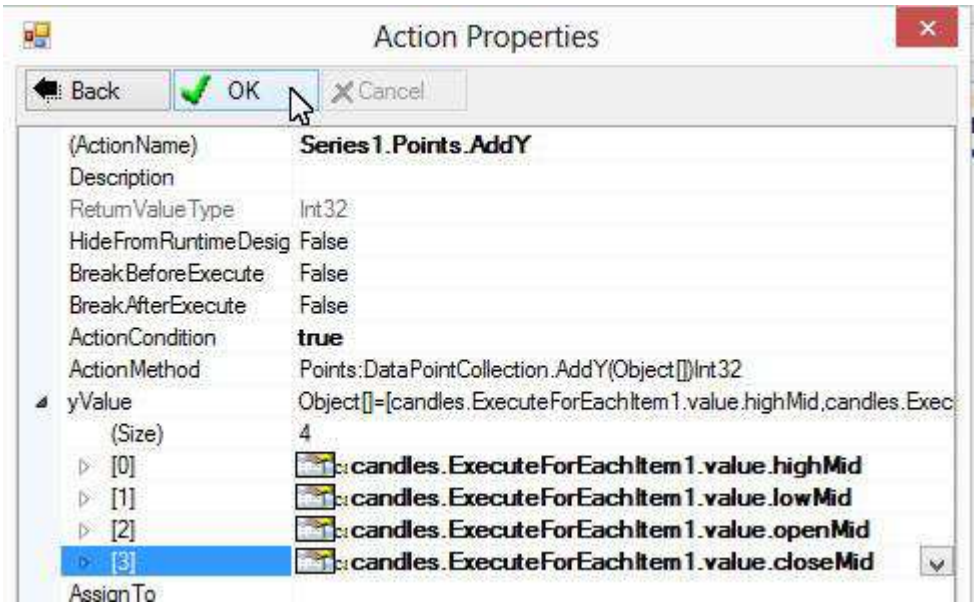
Add high, low, open and close to the data point:


Action Properties
Back
OK
Cancel
(ActionName)
Series1.Points.AddY
Description
ReturnValueType
Int32
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Points:DataPointCollection.AddY(Object[])Int32
yValue
Object[]()
(Size)
4


Action Properties
Back
OK
Cancel
(ActionName)
Series1.Points.AddY
Description
ReturnValueType
Int32
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Points:DataPointCollection.AddY(Object[])Int32
yValue
Object[]=[...]
(Size)
4
[0]
[1]
[2]
[3]
ConstantValue
Property
MathExpression

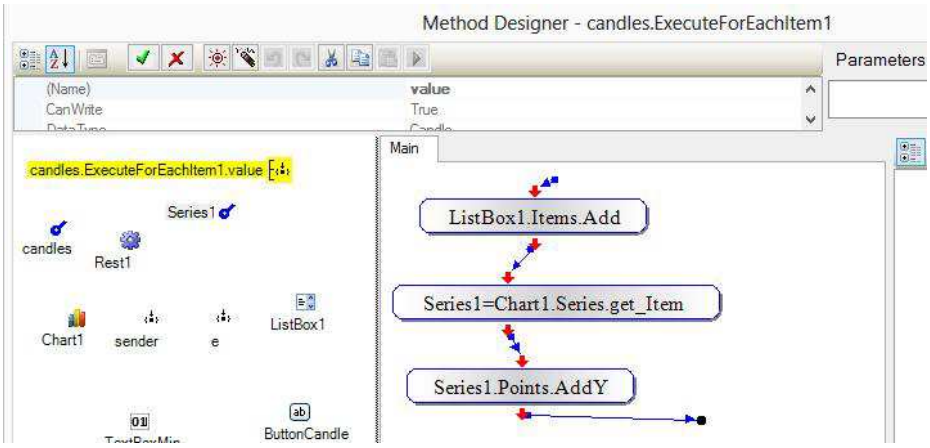

Object picker
Back
Next
Exit
Add
Form3 from DrawingPage
onButtonCandleClick1(Button,EventArgs)
Attributes
Actions
sender
e
candles
candles.ExecuteForEachItem1.value
Properties inherited
closeMid:Double
complete:Boolean
highMid:Double
lowMid:Double
openMid:Double
time:DateTime
Items:ObjectCollection.Add(Object)In
Series:SeriesCollection.get_Item(Int3
Points:DataPointCollection.AddY(Obj



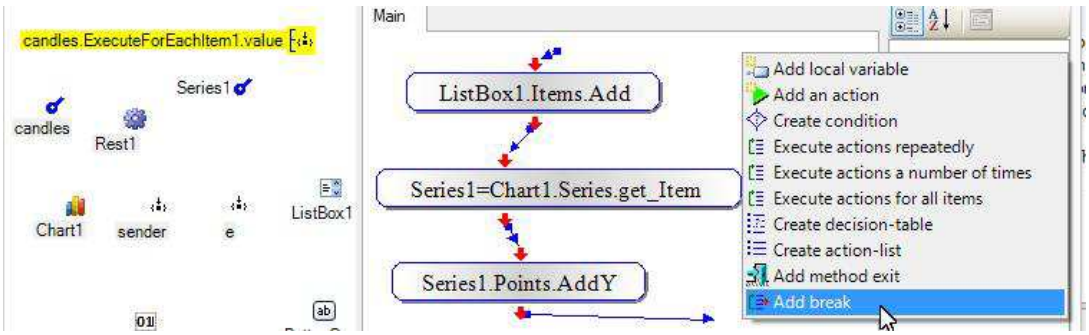


Break "go through"

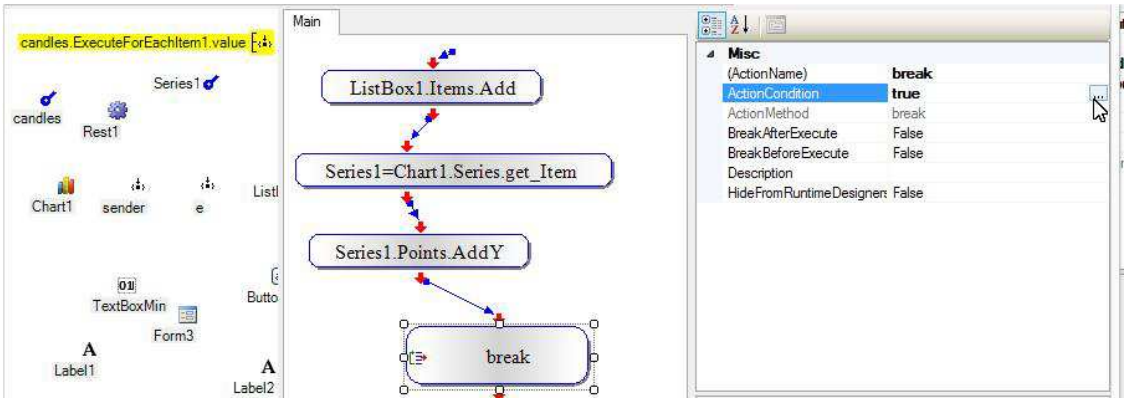
So far, we created 3 actions for processing every candle represented by variable "value":



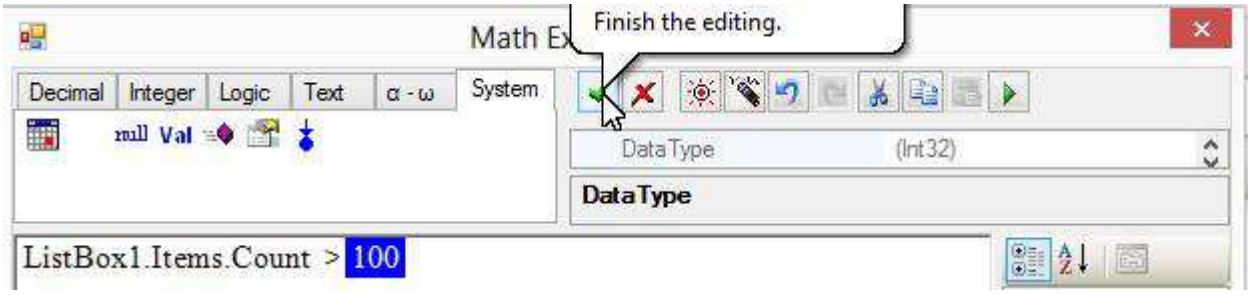
If “candles” has 500 items then these 3 actions will be executed 500 times. Thus, 500 items will be added to the list box; 500 candles will be added to the data series of the chart. A “break” action can be used to cancel going through the rest of the items:



We need to set the ActionCondition of the “break” action:

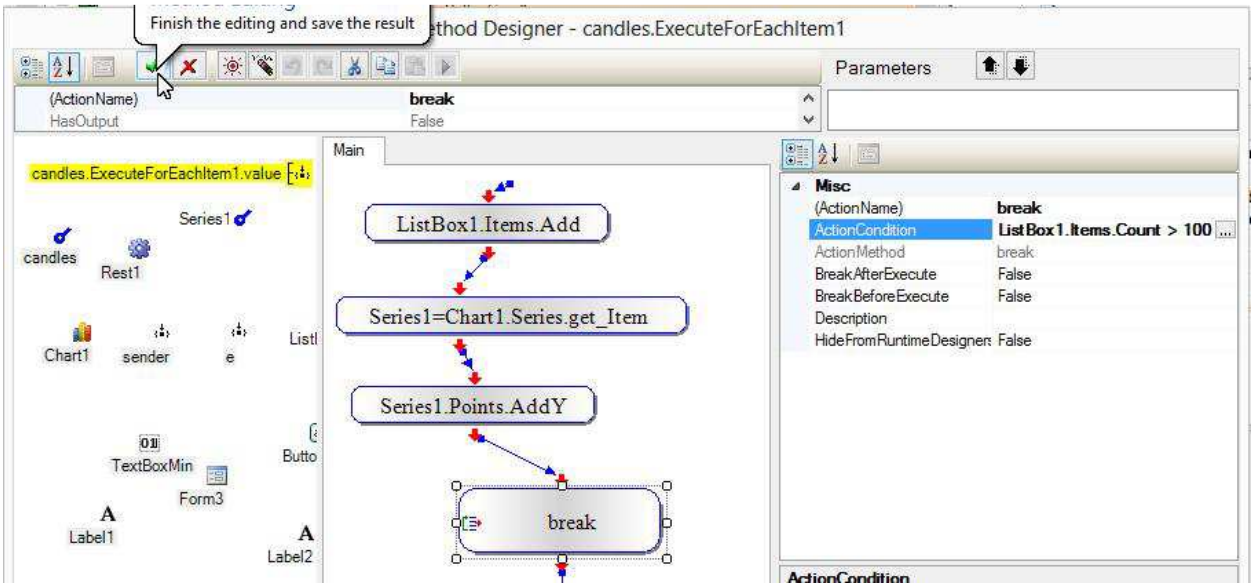


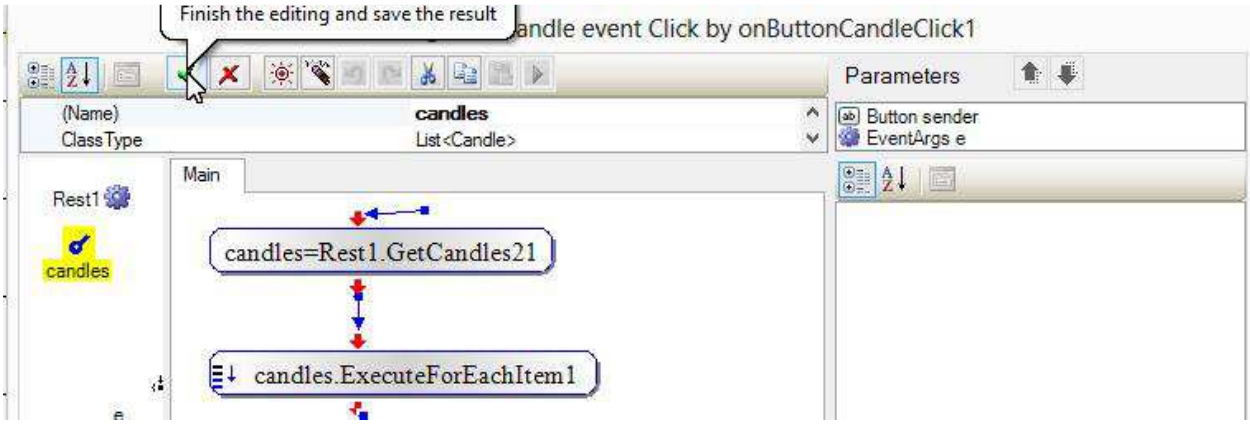
For this sample, we simply set the condition to be that the number of items added to the list box is greater than 100:



Thus only the first candle data are used.

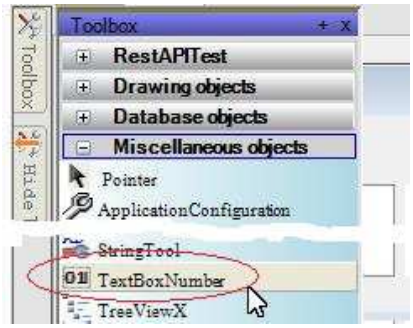
That is all we need for this sample.





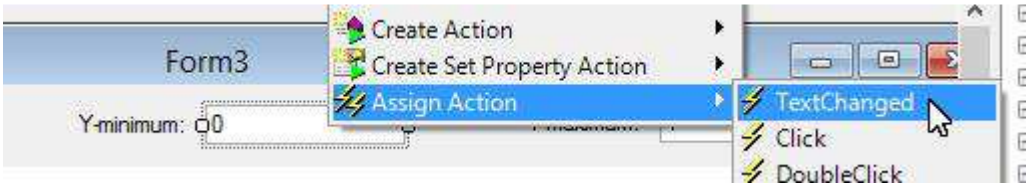
Adjust Y-axes scope

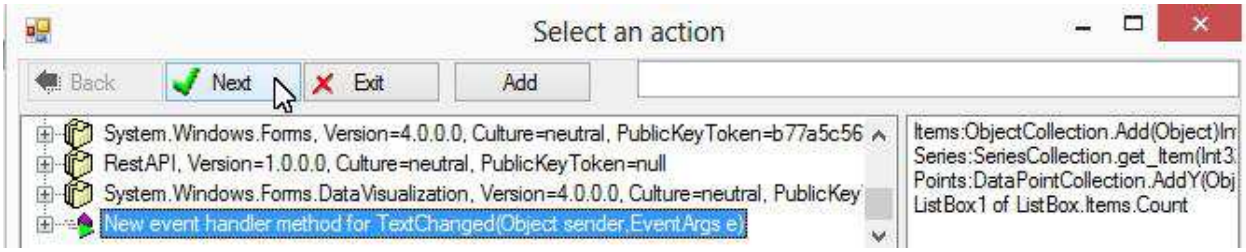
We use two text boxes to specify the minimum value and the maximum value of the Y-axes:



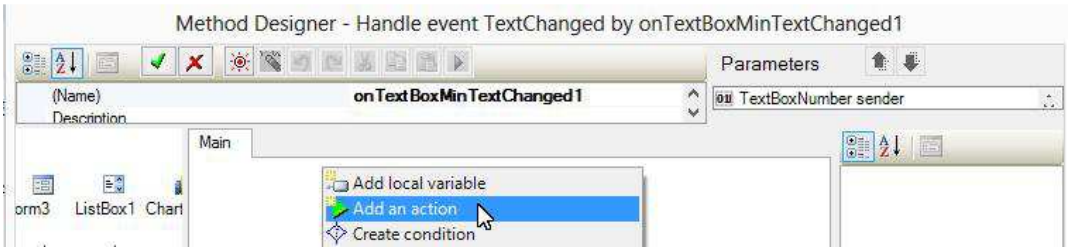
Set minimum/maximum values

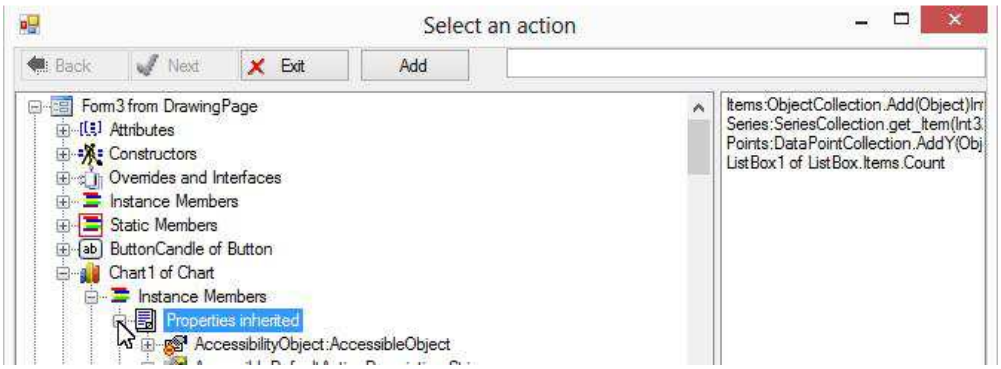
We do it at the event of TextChanged:

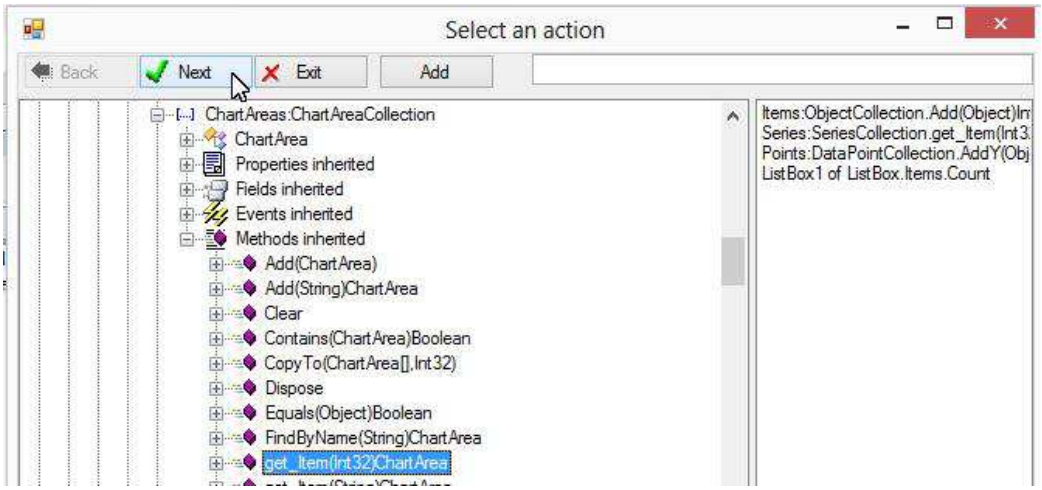


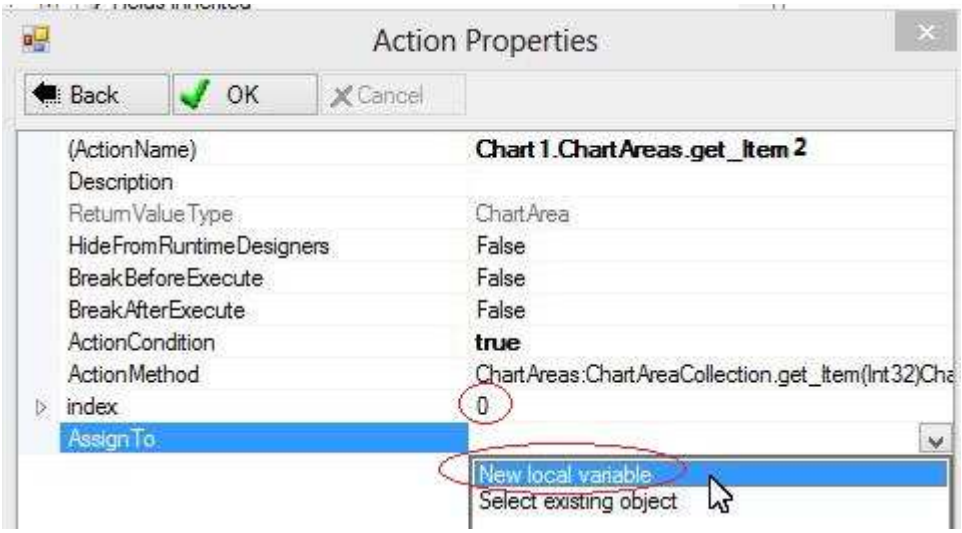


To access the y-axes, we need to get the chart area:

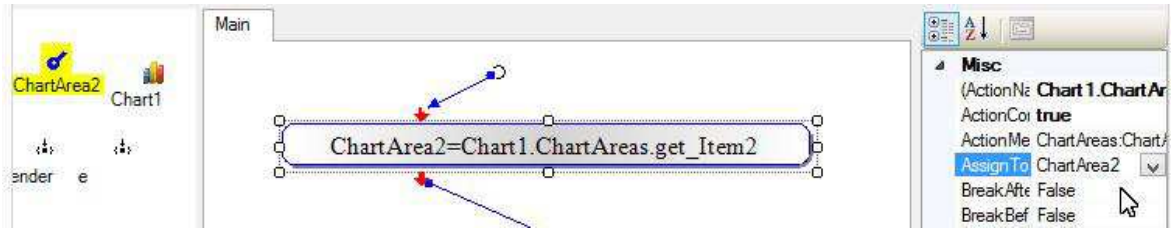




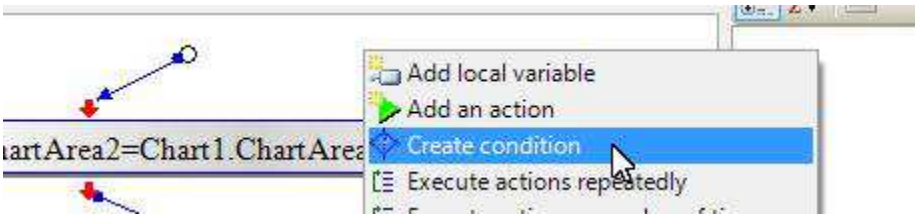


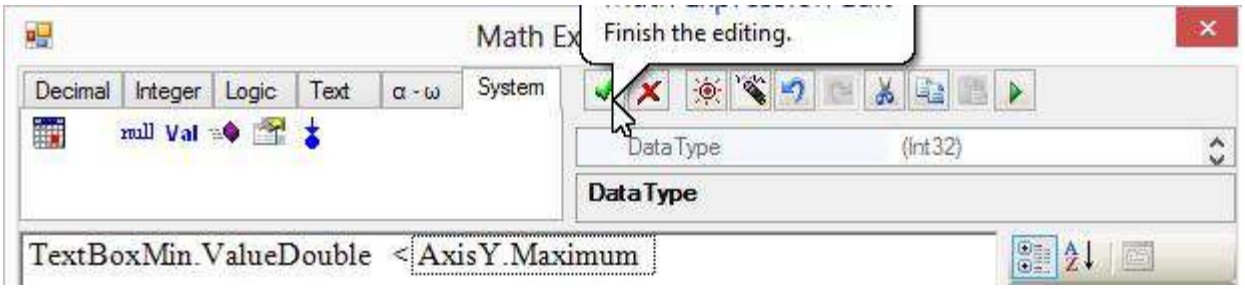


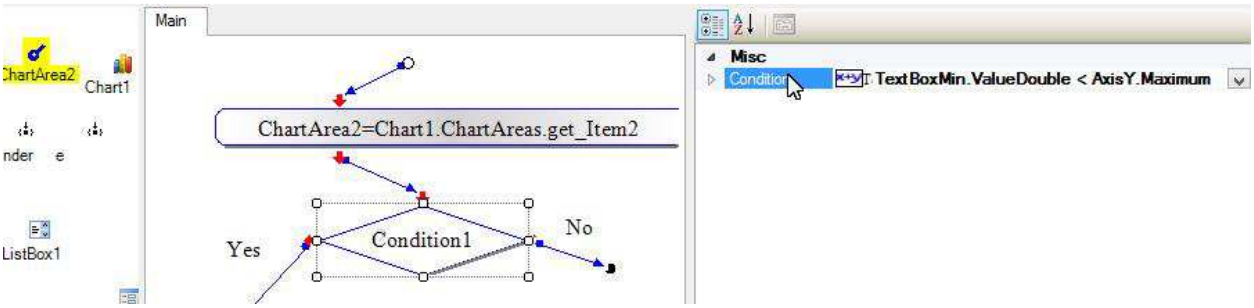
0 indicate the first chart area. Click OK. The action and a chart area variable appear.



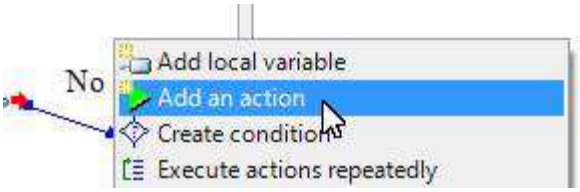
Add a condition to check whether the value of the text box is less than the maximum value of the y-axes:

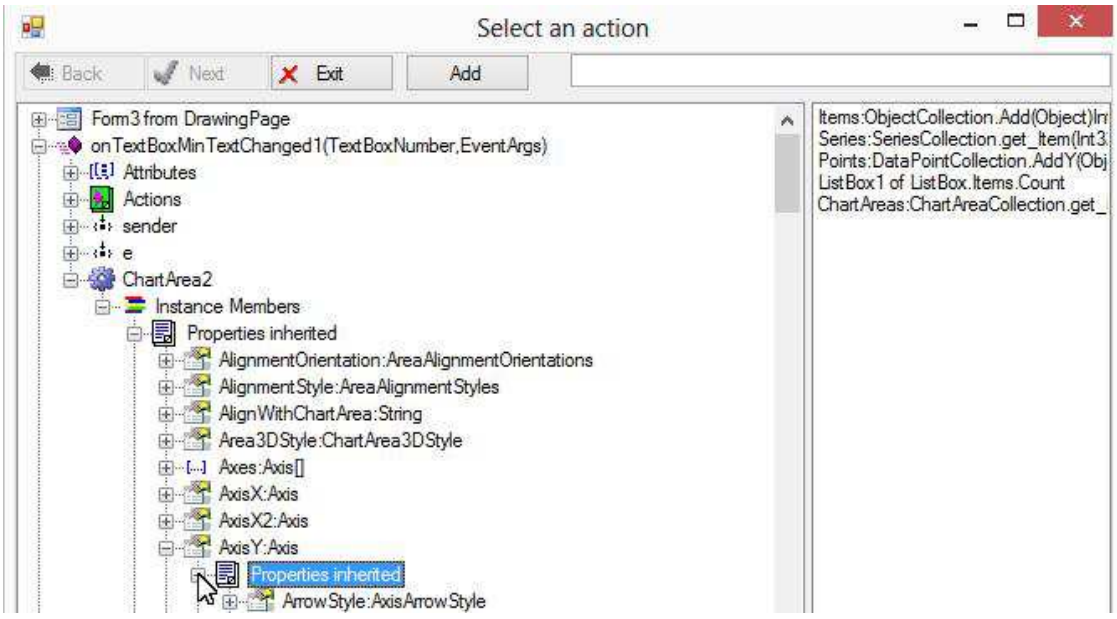




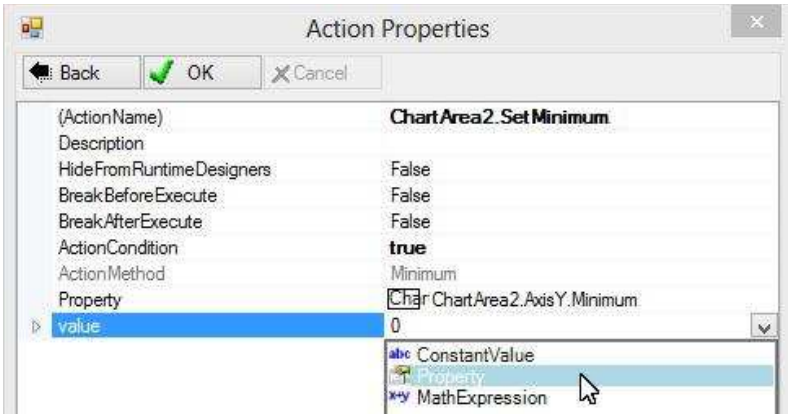


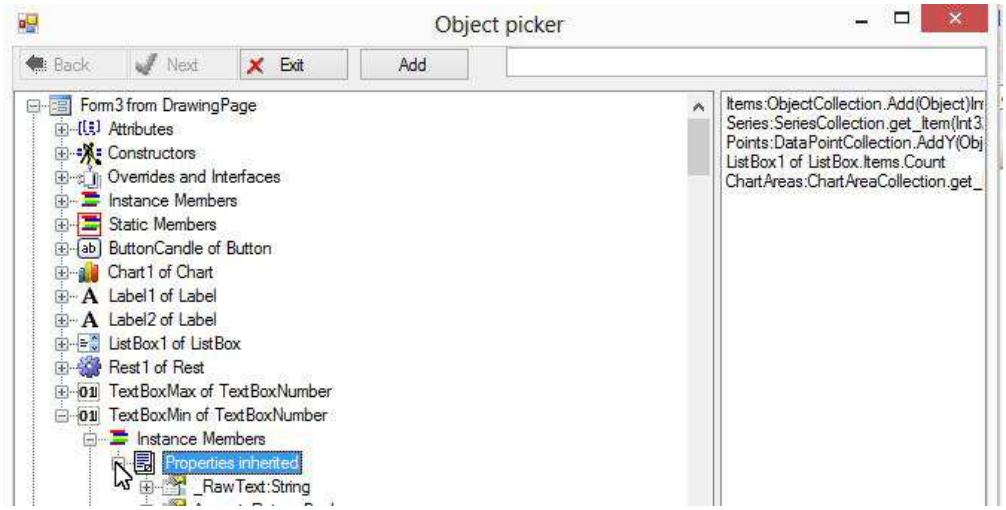
Add an action to set the Minimum value of the y-axes:



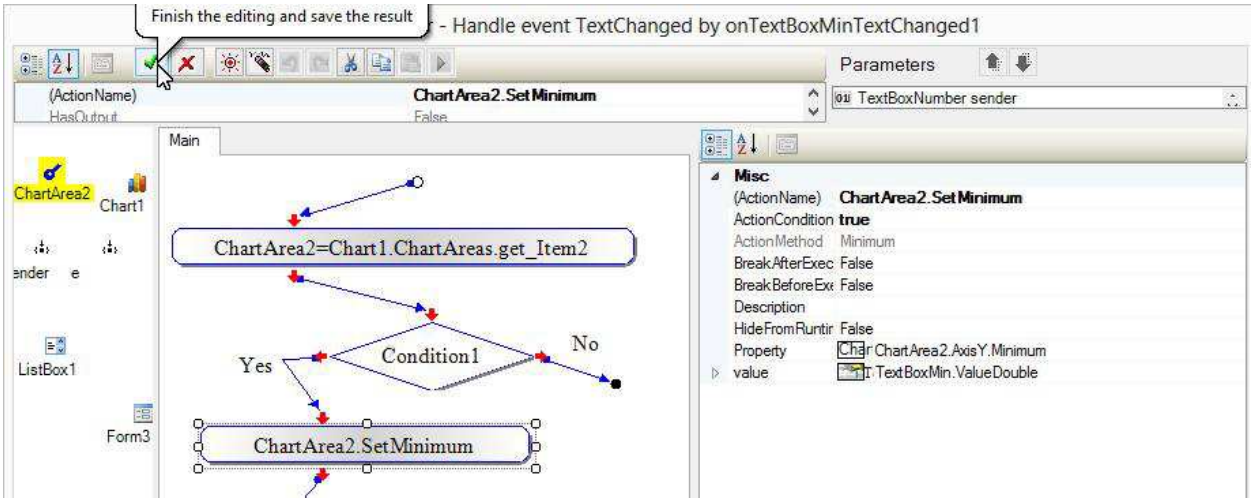


Use the value from the text box:

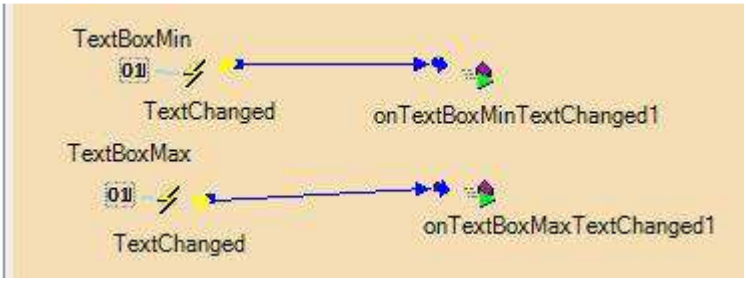




That is all we need to handle the TextChanged event of the text box:

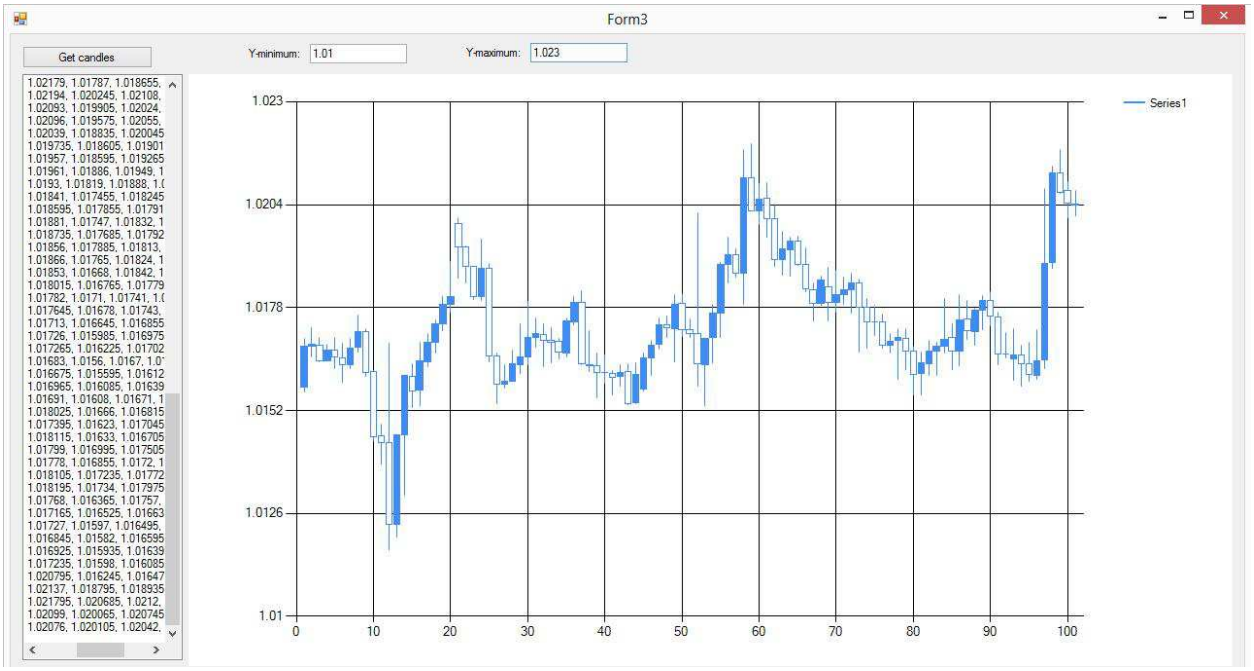


We can do the similar programming for the text box for setting the maximum value of the y-axes.



Test

Compile and run the application. Click “Candles” button. We can see that 100 candles are added to the list box and the chart control. We may adjust the minimum and maximum values of the y-axes to make a better view of the candlestick chart.



Feedback

Please send your feedback to support@limnor.com