How to Owner-draw List Items

Contents

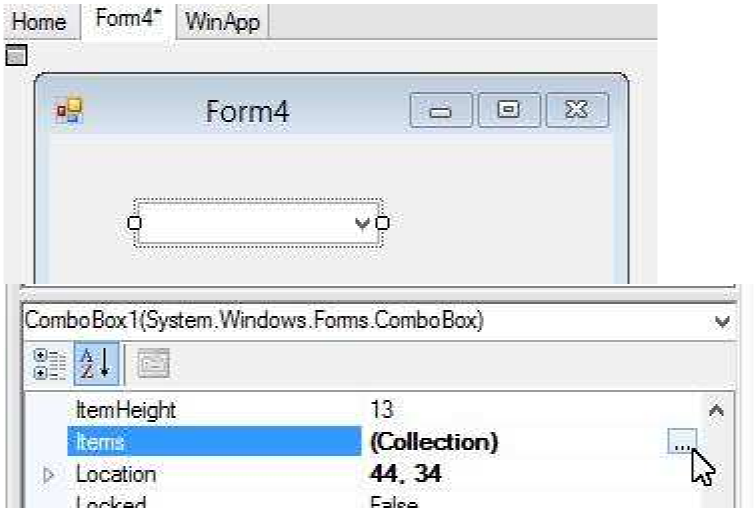
Introduction

Usually items in list boxes and combo boxes are displayed on the screen automatically. You may take over the drawing of each item if the automatic display is not good enough for your needs. This article demonstrates the process.

Take over the drawing task

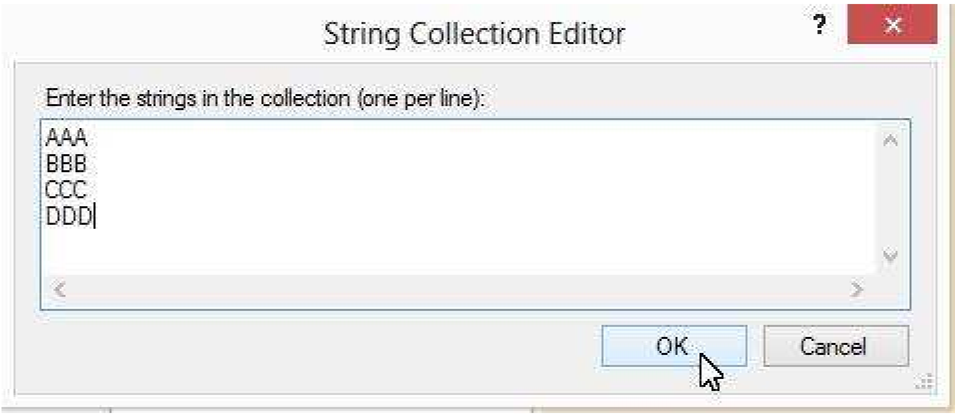
Setting DrawMode property of a list other than Normal will take over the item drawing task. Let’s use a combo box to show its effect.

Suppose we have a combo box and populate it with some items for testing purpose:



ComboBox1(System.Windows.Forms.ComboBox)

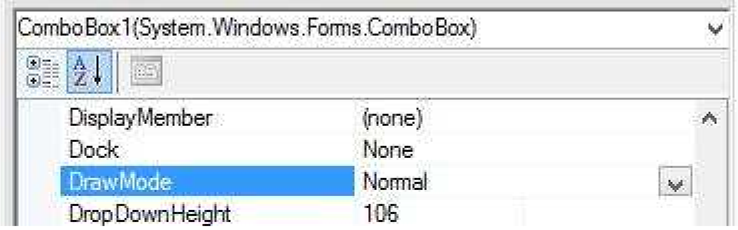
ItemHeight 13
Items (Collection)
Location 44, 34
Locked False



Enter the strings in the collection (one per line):

AAA
BBB
CCC
DDD

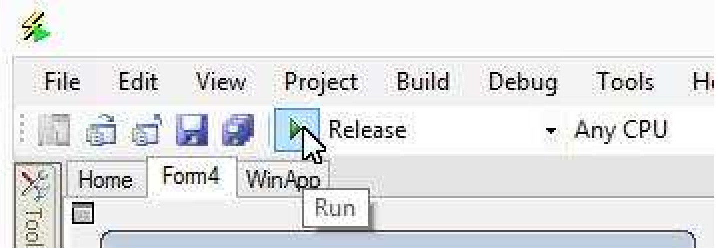
By default, DrawMode is Normal:

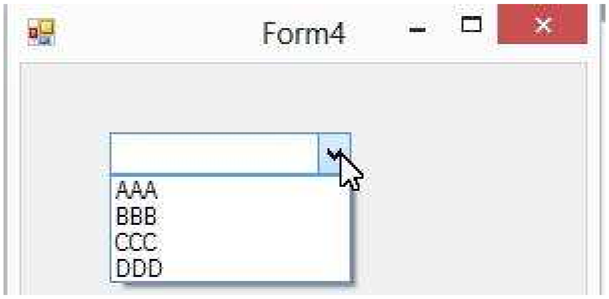


ComboBox1(System.Windows.Forms.ComboBox)

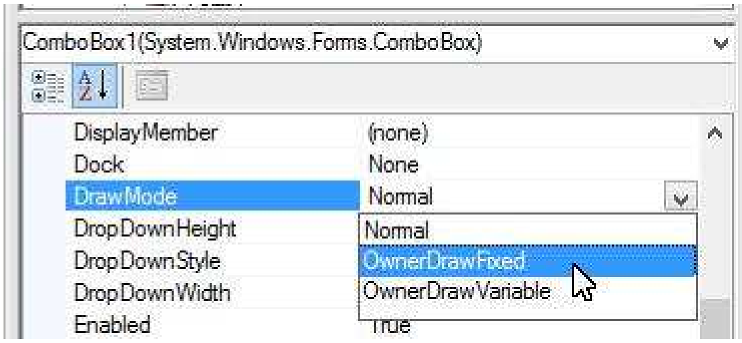
DisplayMember (none)
Dock None
DrawMode Normal
DropDownHeight 106

Run the project; we see the normal drawing of the items:



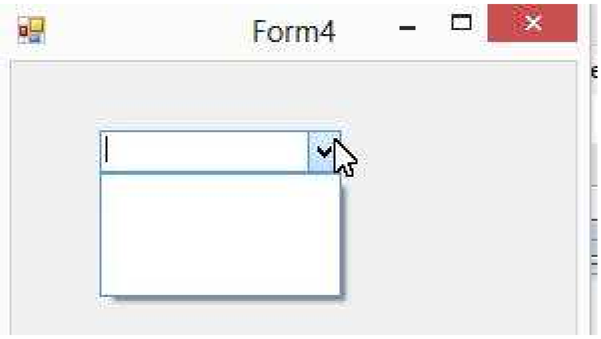


Now set DrawMode to OwnerDrawFixed


ComboBox1 (System.Windows.Forms.ComboBox)

DisplayMember (none)
Dock None
DrawMode Normal
DropDownHeight
DropDownStyle
DropDownWidth
Enabled True

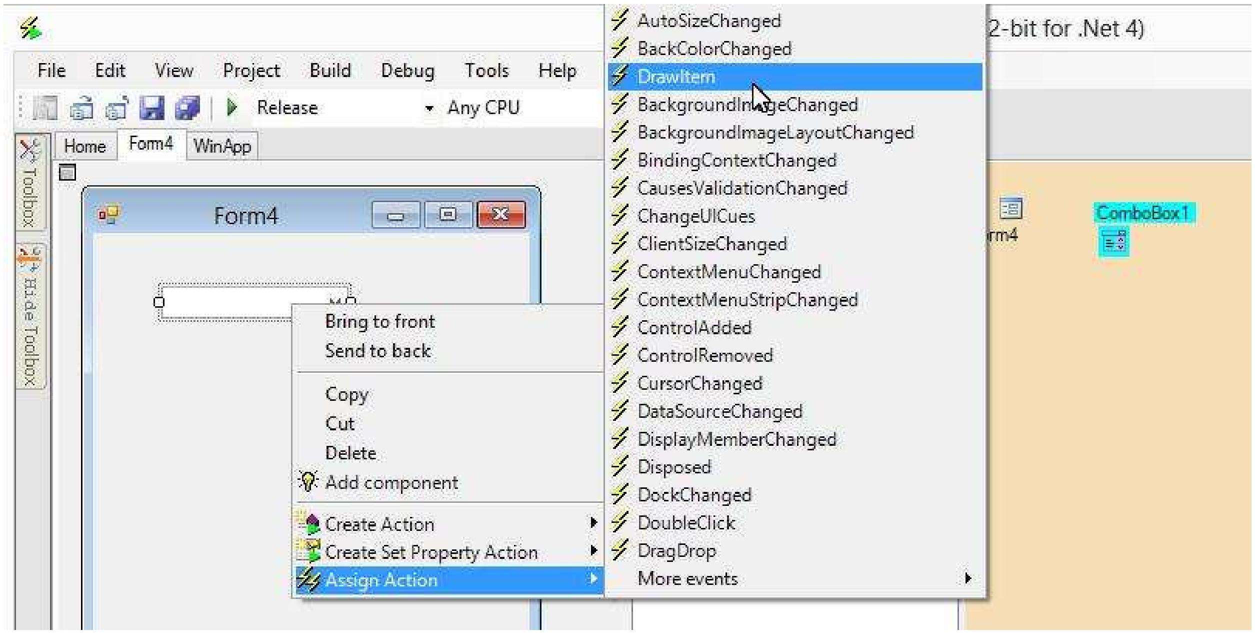
Run the project again. This time no items are displayed because we take over the item drawing task but we do nothing to draw items:

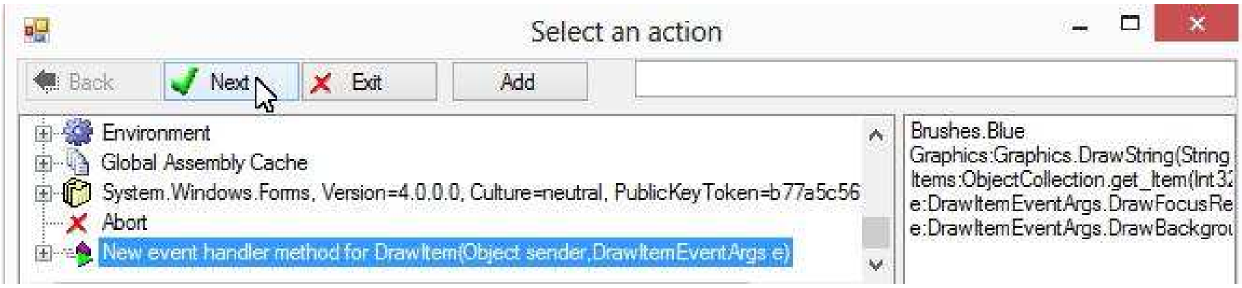


Handle DrawItem event

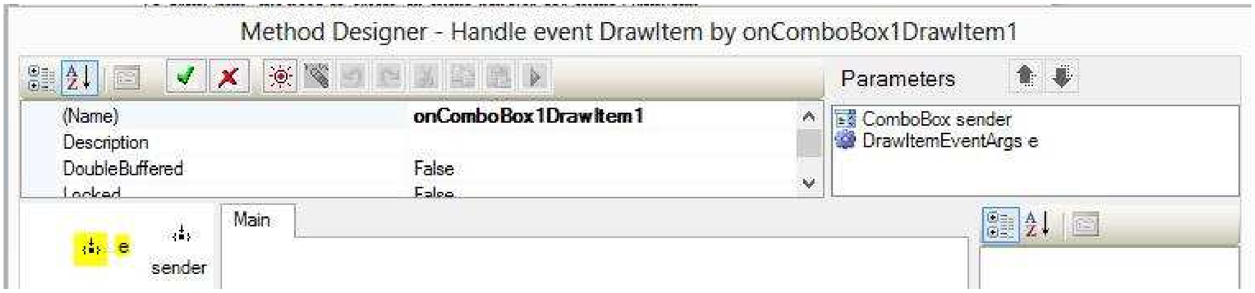
Create event handler method

To draw item, we need to create an event handler for event DrawItem.




Select an action
Back
Next
Exit
Add
Environment
Global Assembly Cache
System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c56
Abort
New event handler method for DrawItem(Object sender,DrawItemEventArgs e)
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int3
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou

A Method Editor appears for adding event handling actions.

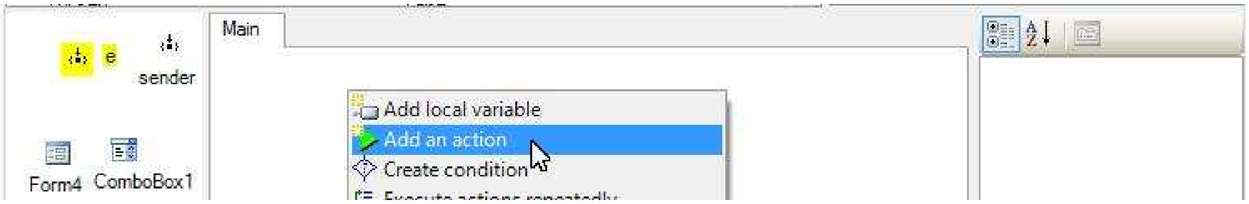

Method Designer - Handle event DrawItem by onComboBox1DrawItem1

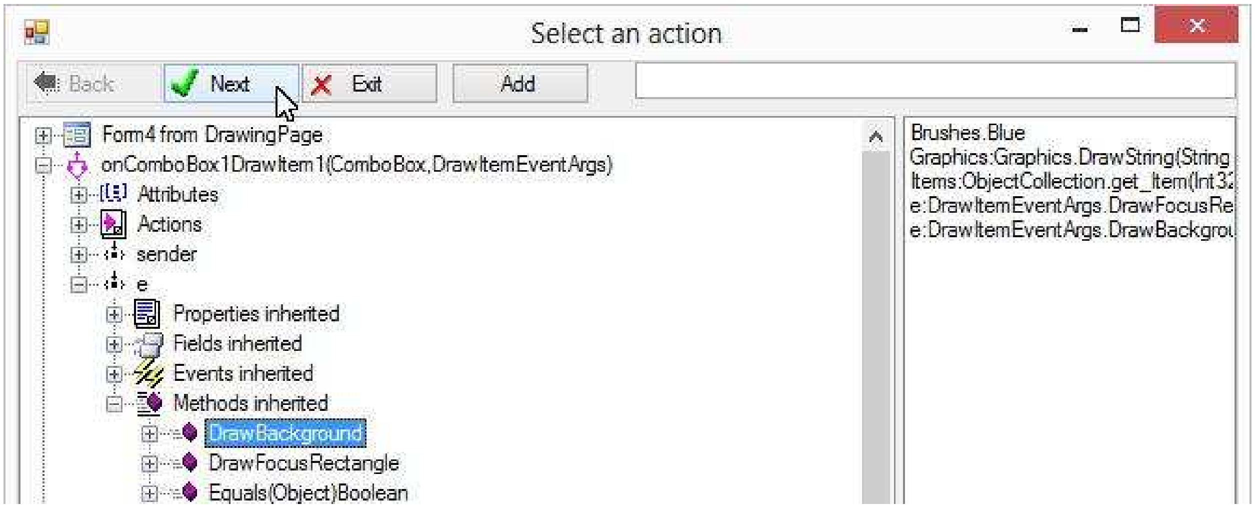
(Name)	onComboBox1DrawItem1
Description	
DoubleBuffered	False
Locked	False

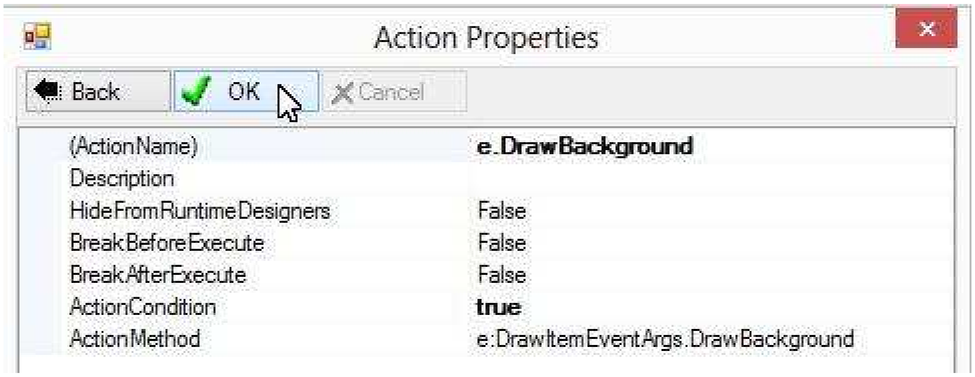
Note that the event parameter e contains important properties and methods for making drawings.

Draw item background

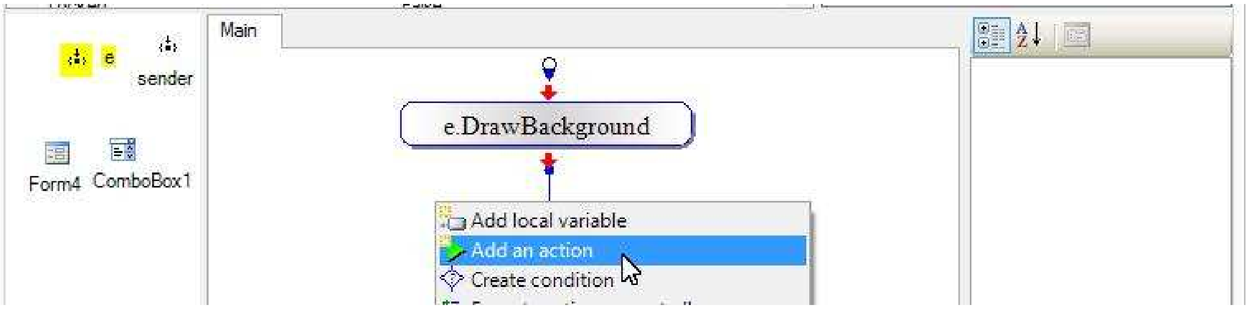
Parameter e provides a DrawBackGround method and a DrawFocusRectangle method. We may execute these two methods first before making other drawing.

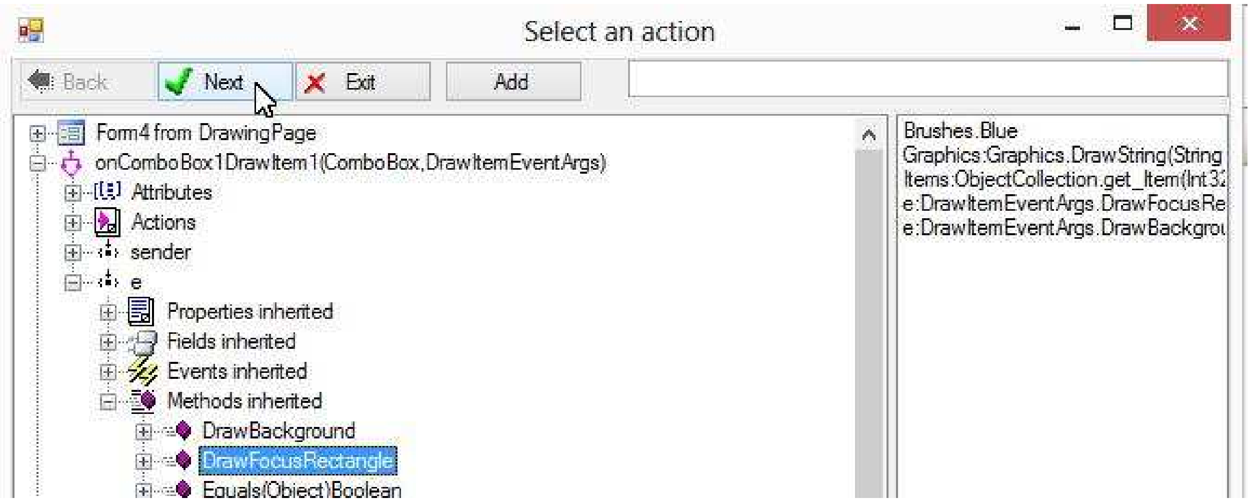


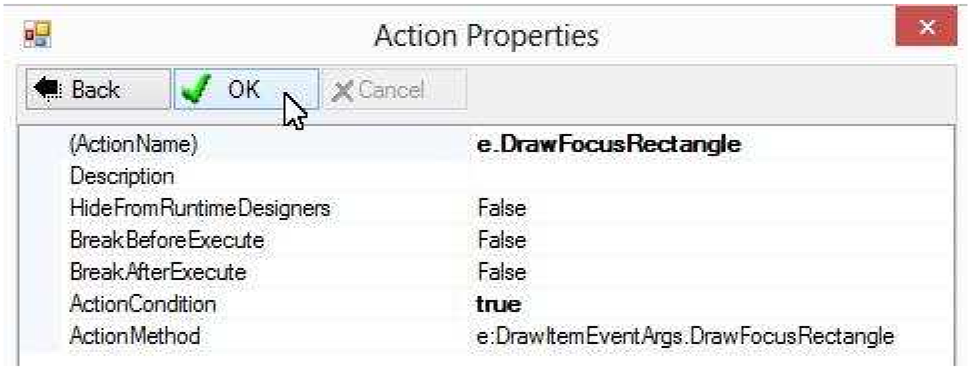




(ActionName)	**e.DrawBackground**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	e:DrawItemEventArgs.DrawBackground

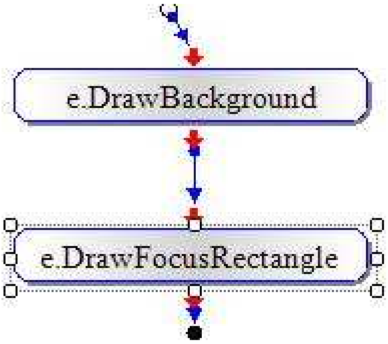






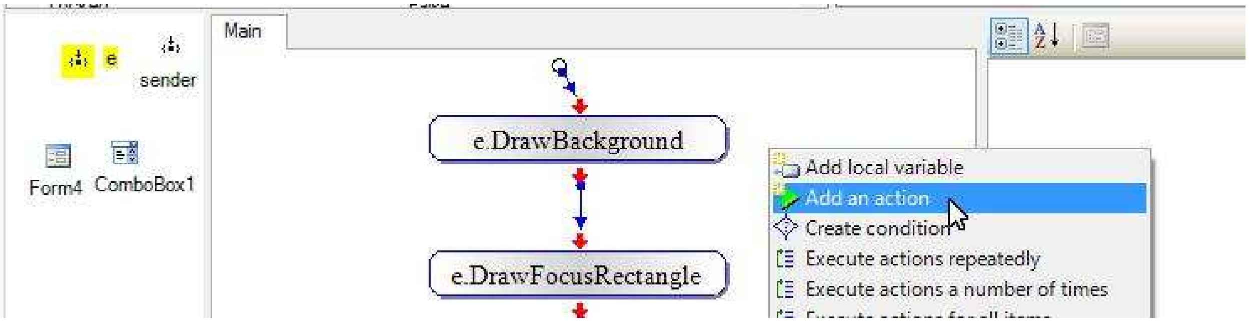
(ActionName)	**e.DrawFocusRectangle**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	e:DrawItemEventArgs.DrawFocusRectangle

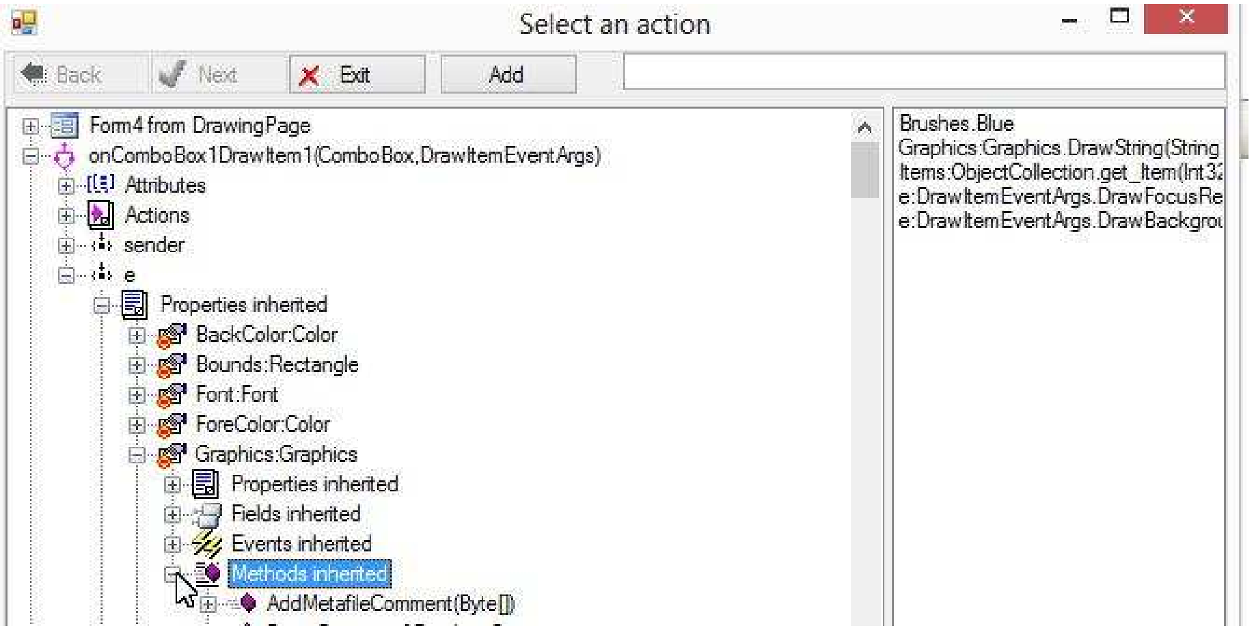
Link the two actions together.

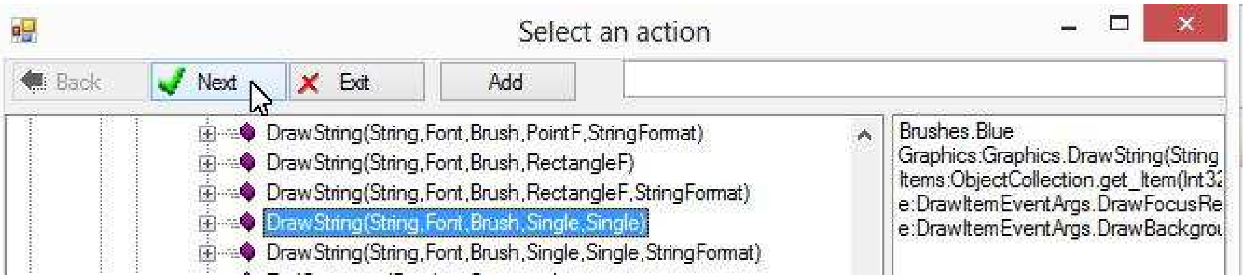


Draw item

Parameter e has a property, Graphics, which provides rich drawing capabilities. For example, draw images, draw lines, draw texts, draw ellipses, draw rectangles, etc. Here we make a very simple drawing: just draw the current list item as a text.

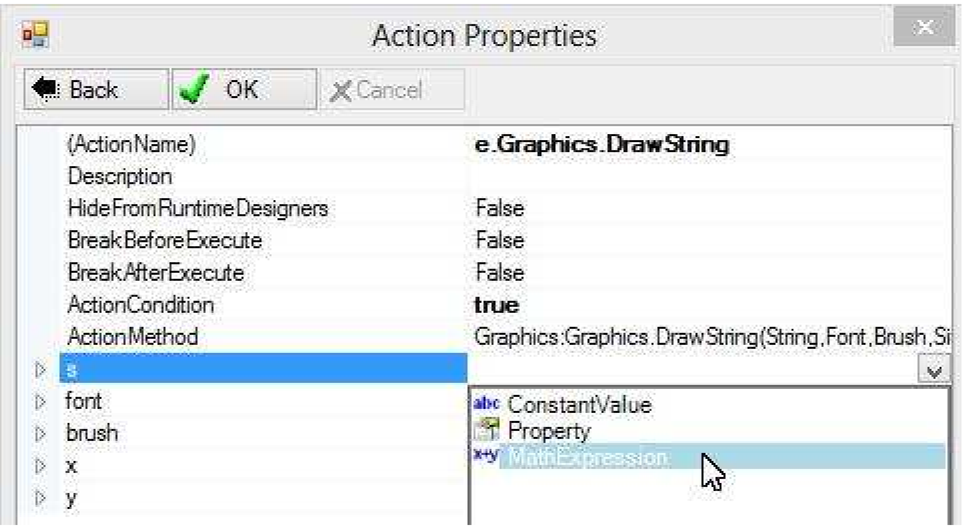





Select an action
Back
Next
Exit
Add
DrawString(String,Font,Brush,PointF,StringFormat)
DrawString(String,Font,Brush,RectangleF)
DrawString(String,Font,Brush,RectangleF,StringFormat)
DrawString(String,Font,Brush,Single,Single)
DrawString(String,Font,Brush,Single,Single,StringFormat)
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou

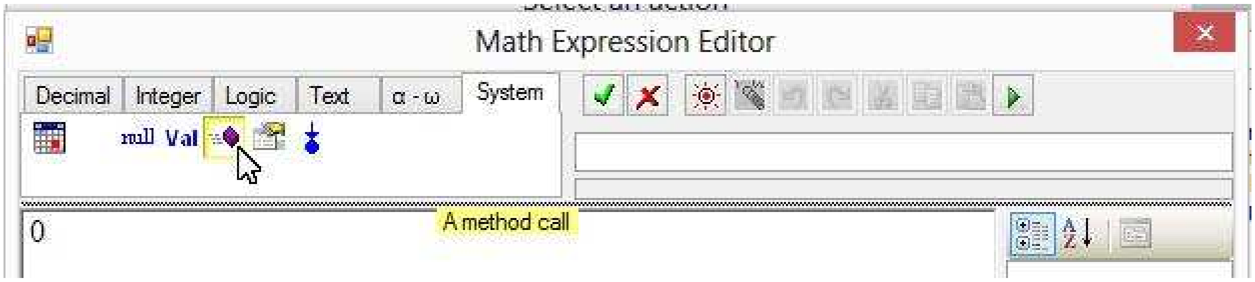
Get current item

Parameter “s” is the text to draw. We want to use the current list item. We may use method get_item of Items property of the list box to get the current list item. So, select “Math Expression” for parameter “s”:



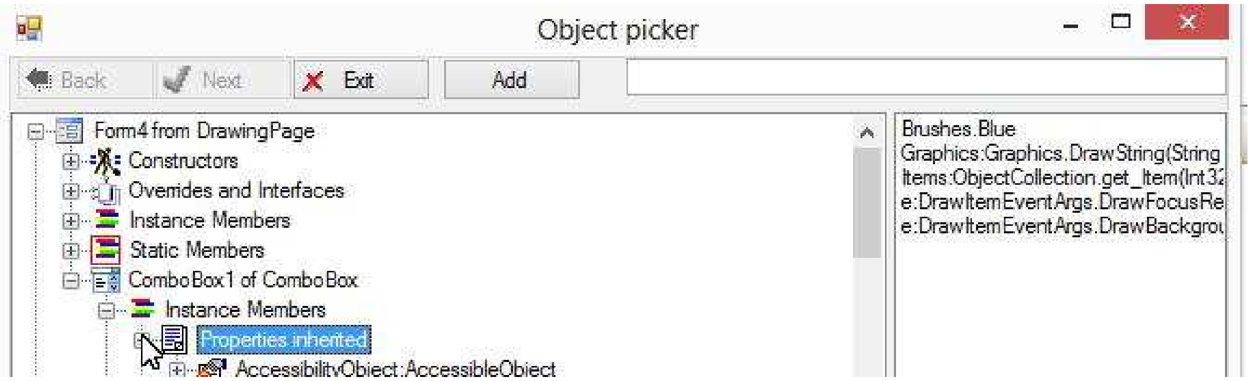
(ActionName)	e.Graphics.DrawString
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush,Si
s	
font	
brush	
x	
y	

Click the Method icon:



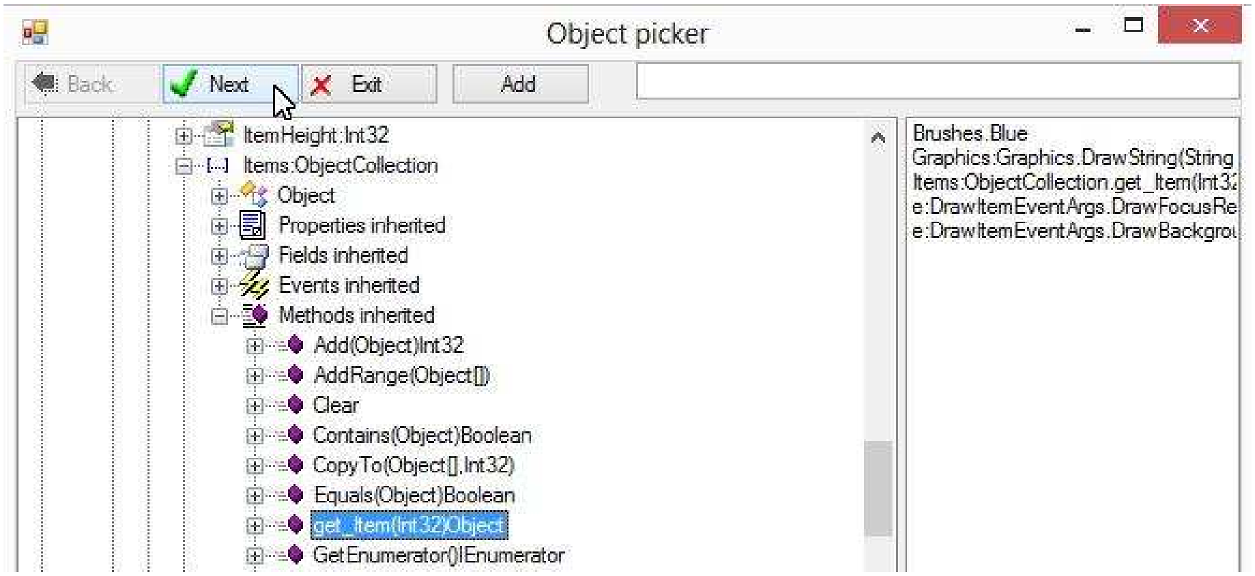
0

Select get_item method of the Items property of the list box:

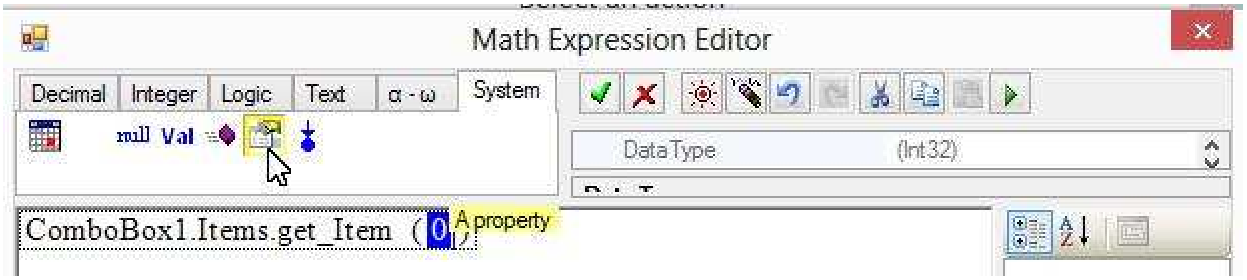


Form4 from DrawingPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
ComboBox1 of ComboBox
Instance Members
Properties inherited
AccessibilityObject:AccessibleObject

Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou

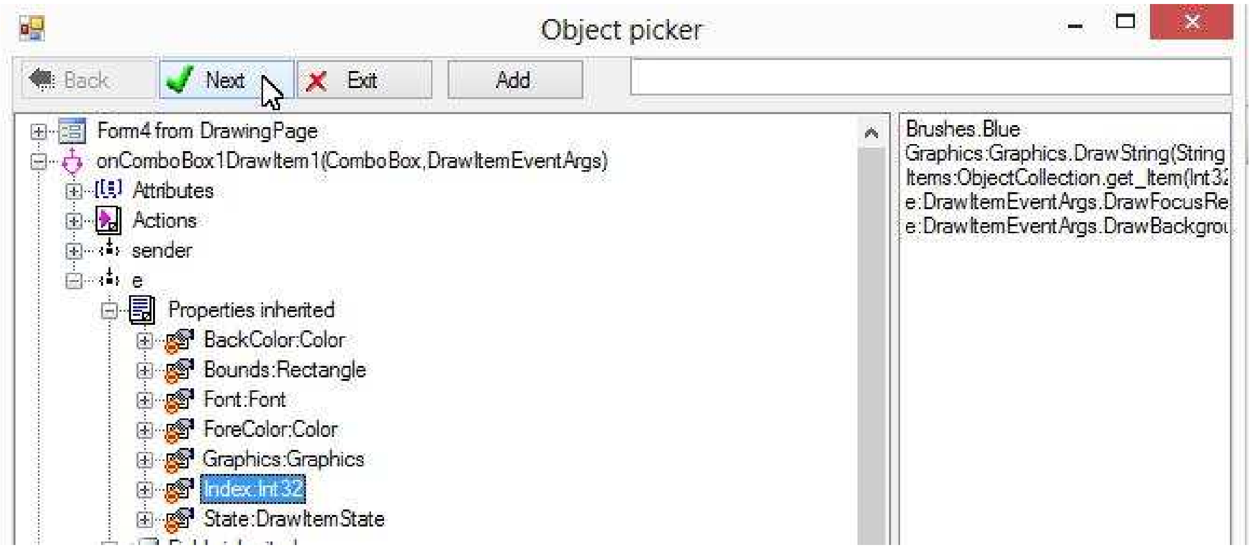


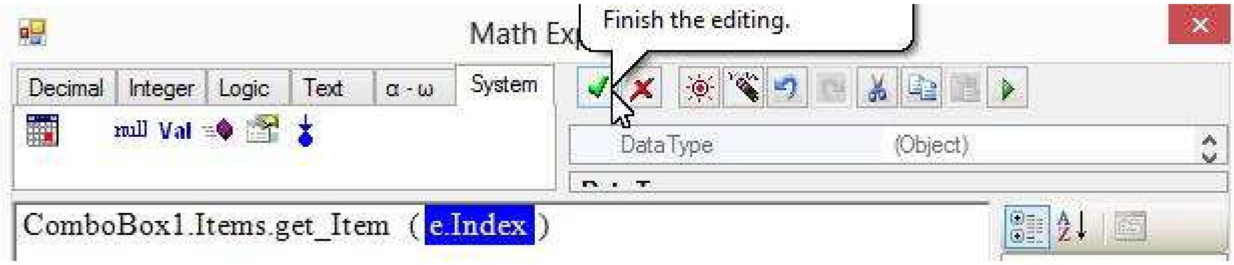
Method get_item requires an index parameter to specify the index of item. Parameter e has an Index property indicating the index of the current item. Click the property icon:



ComboBox1.Items.get_Item (0)

Select Index property of e:



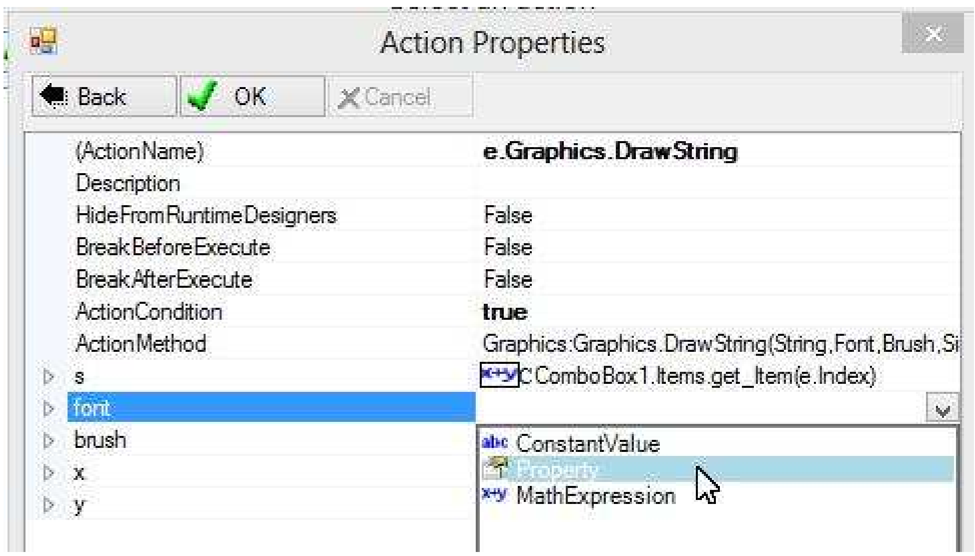


Data Type (Object)

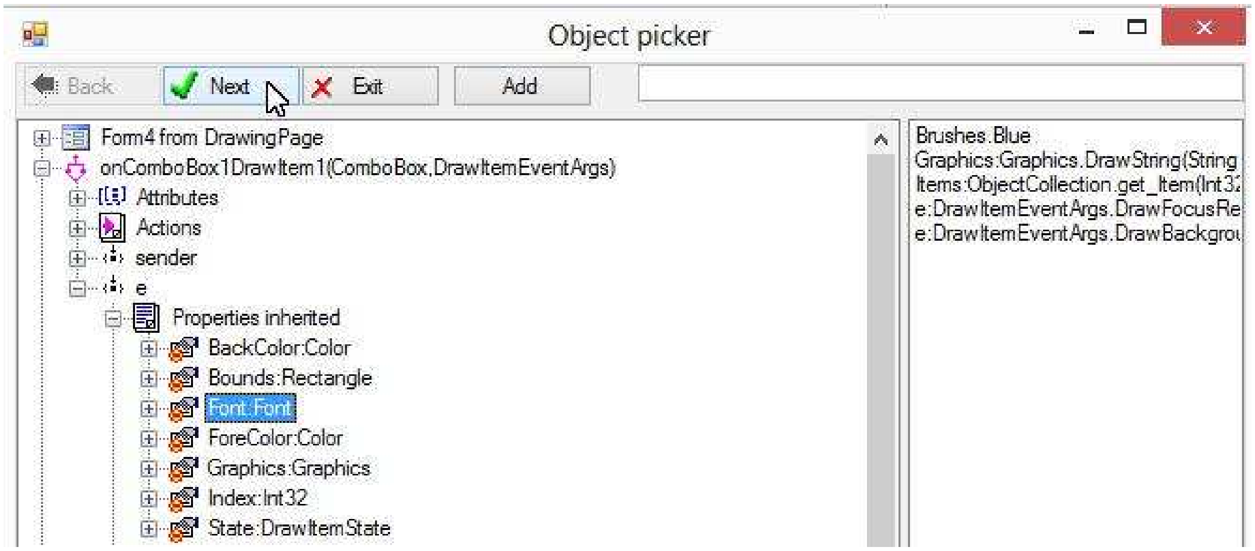
ComboBox1.Items.get_Item (e.Index)

Specify font

We simply use the current font:



(ActionName)	e.Graphics.DrawString
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush,Si
s	ComboBox1.Items.get_Item(e.Index)
font	
brush	
x	
y	

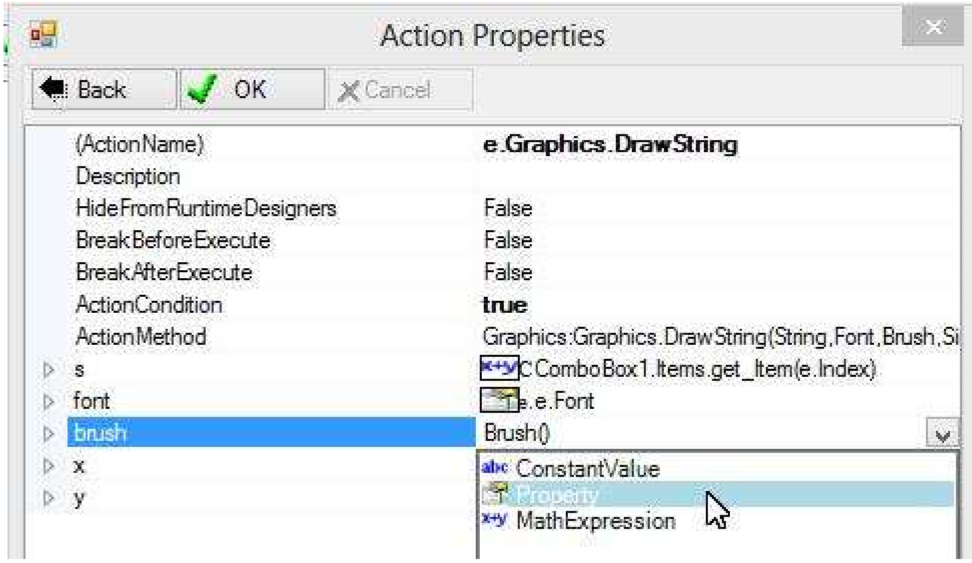


```
Form4 from DrawingPage
onComboBox1DrawItem1(ComboBox,DrawItemEventArgs)
    Attributes
    Actions
    sender
    e
        Properties inherited
            BackColor:Color
            Bounds:Rectangle
            Font:Font
            ForeColor:Color
            Graphics:Graphics
            Index:Int32
            State:DrawItemState
```

```
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou
```

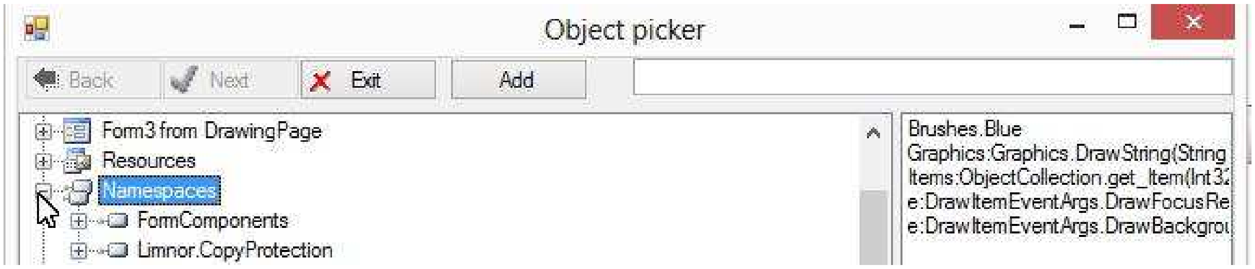
Specify drawing brush

You may create a brush to achieve desired text effects. For simplicity, we may use a brush provided by .Net Framework:

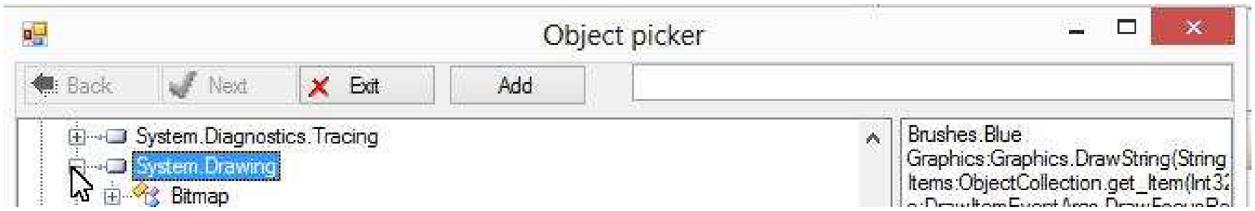


(ActionName)	e.Graphics.DrawString
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush,Si
s	CComboBox1.Items.get_Item(e.Index)
font	e.e.Font
brush	Brush()
x	
y	

Expand System.Drawing namespace:

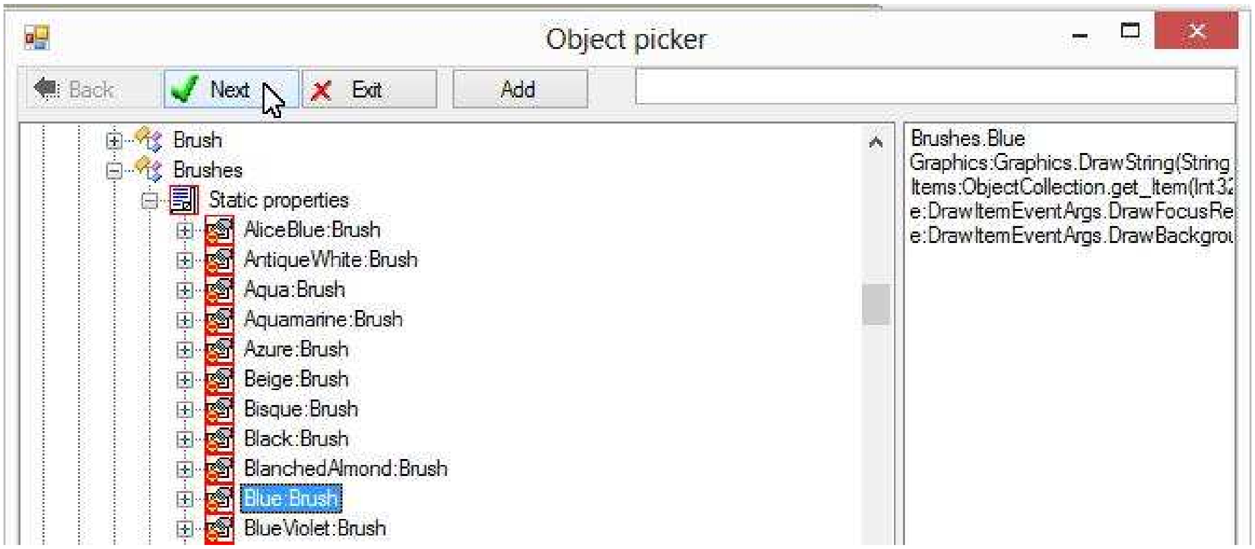


Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou



Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32

Select a prebuilt brush under Brushes:

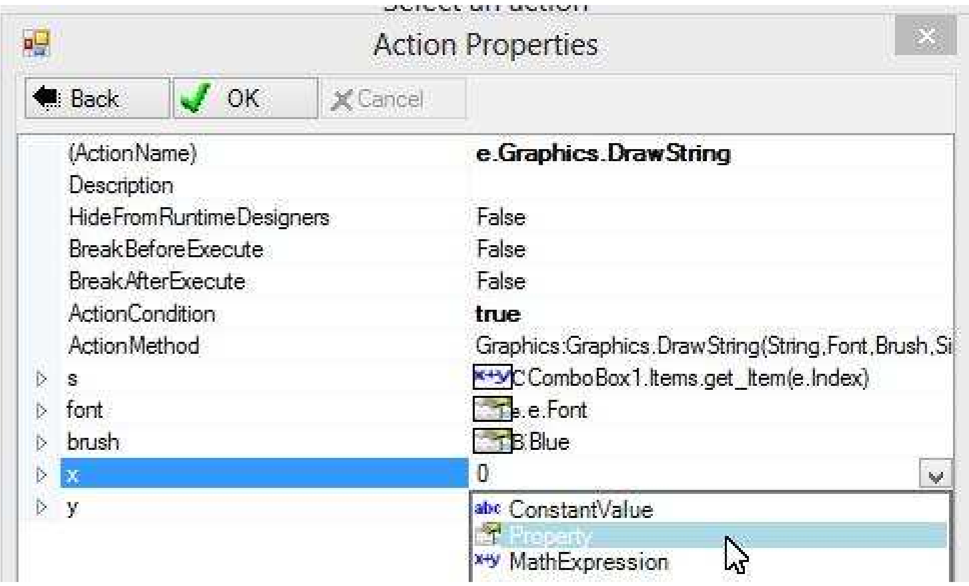


Brush
Brushes
Static properties
AliceBlue:Brush
AntiqueWhite:Brush
Aqua:Brush
Aquamarine:Brush
Azure:Brush
Beige:Brush
Bisque:Brush
Black:Brush
BlanchedAlmond:Brush
Blue:Brush
BlueViolet:Brush

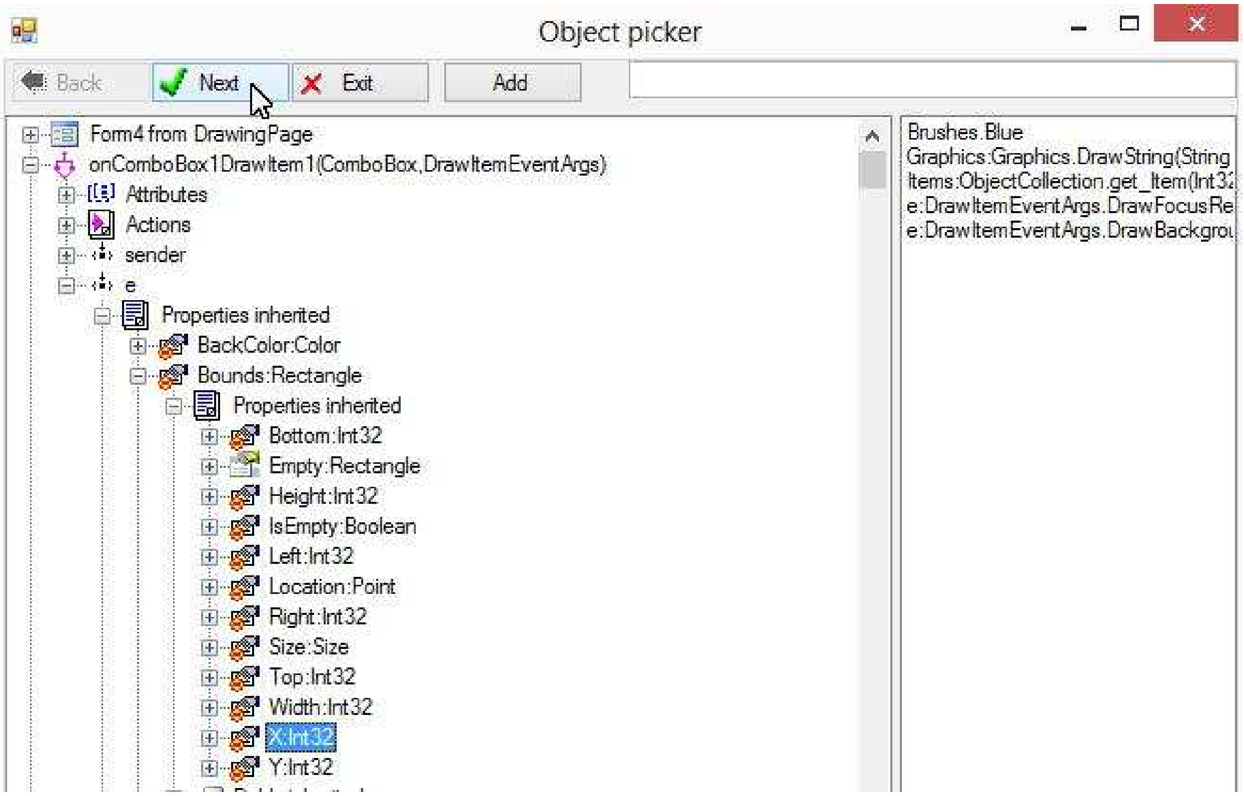
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou

Specify drawing location

Bounds property of parameter e indicates the rectangle area to draw the item. We may use this property to calculate drawing location. For simplicity, we just draw from top left corner of the item area:

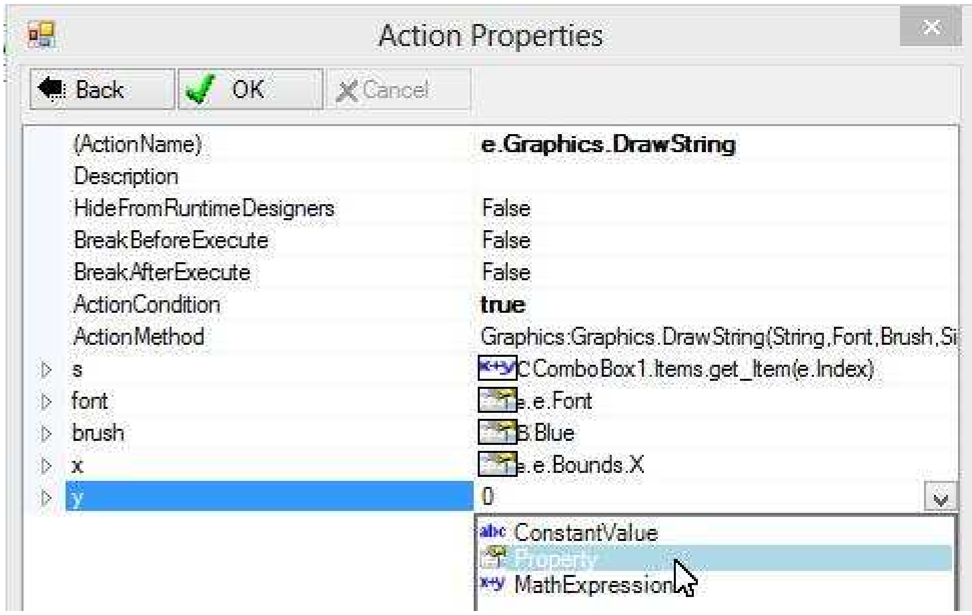


(ActionName)	e.Graphics.DrawString
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush,Si
s	C:ComboBox1.Items.get_Item(e.Index)
font	e.e.Font
brush	B:Blue
x	0
y	

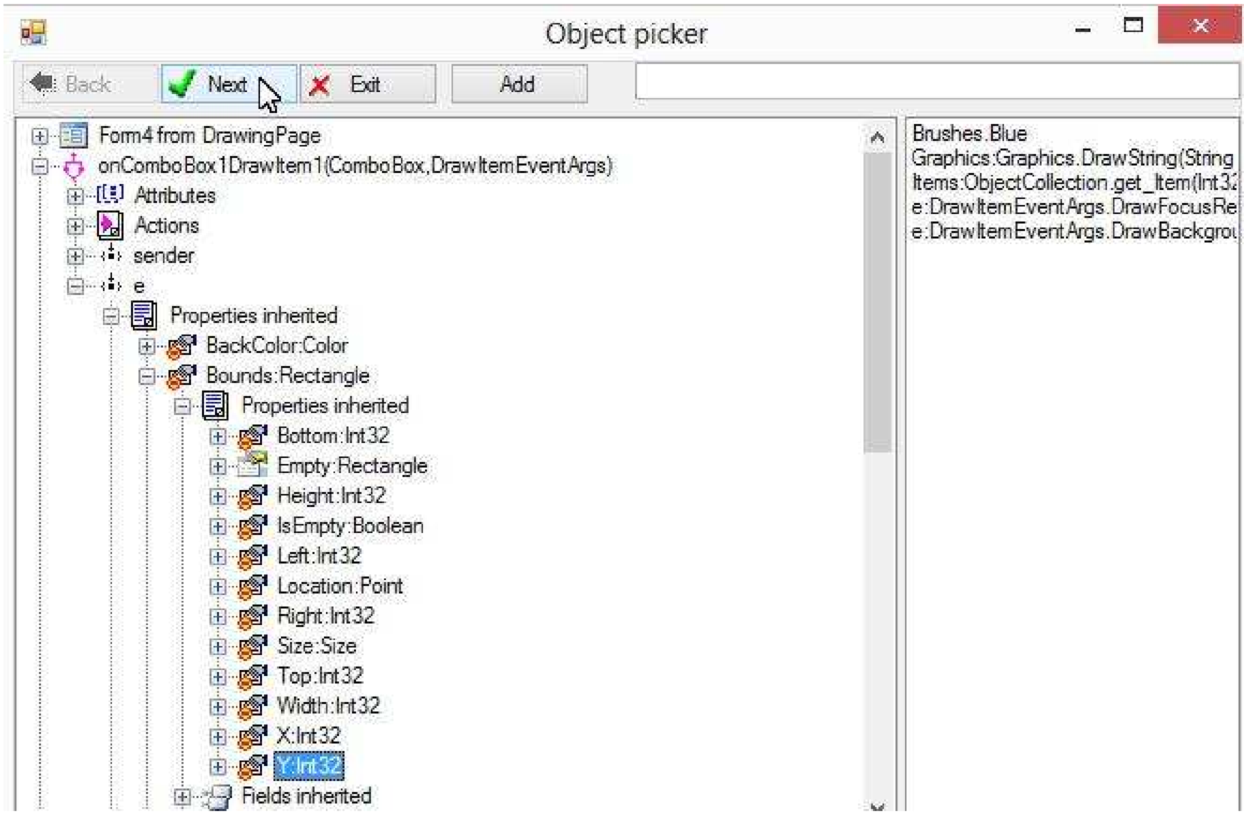


```
Form4 from DrawingPage
onComboBox1DrawItem1(ComboBox,DrawItemEventArgs)
    Attributes
    Actions
    sender
    e
        Properties inherited
            BackColor:Color
            Bounds:Rectangle
                Properties inherited
                    Bottom:Int32
                    Empty:Rectangle
                    Height:Int32
                    IsEmpty:Boolean
                    Left:Int32
                    Location:Point
                    Right:Int32
                    Size:Size
                    Top:Int32
                    Width:Int32
                    X:Int32
                    Y:Int32
```

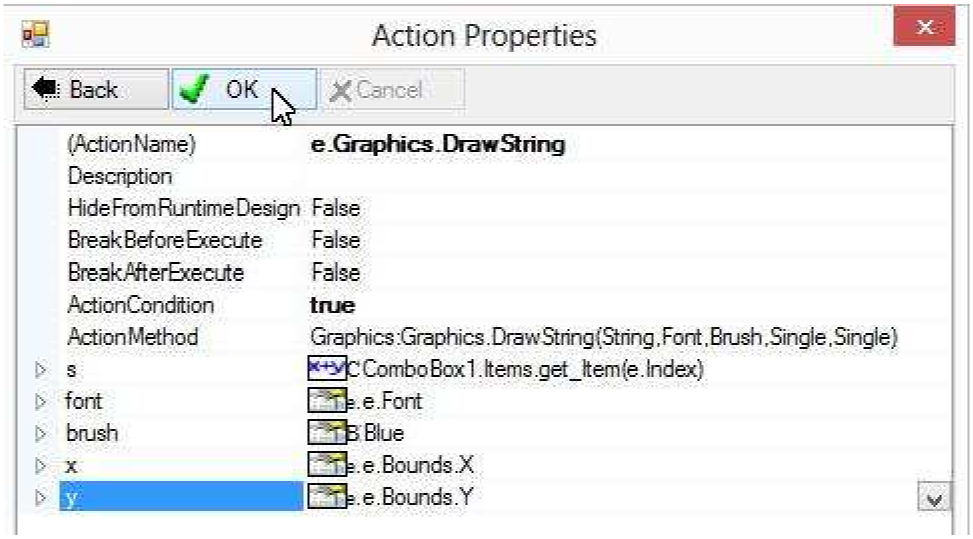
```
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou
```



(ActionName)	**e.Graphics.DrawString**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush,Si
s	C.ComboBox1.Items.get_Item(e.Index)
font	e.e.Font
brush	B.Blue
x	e.e.Bounds.X
y	0

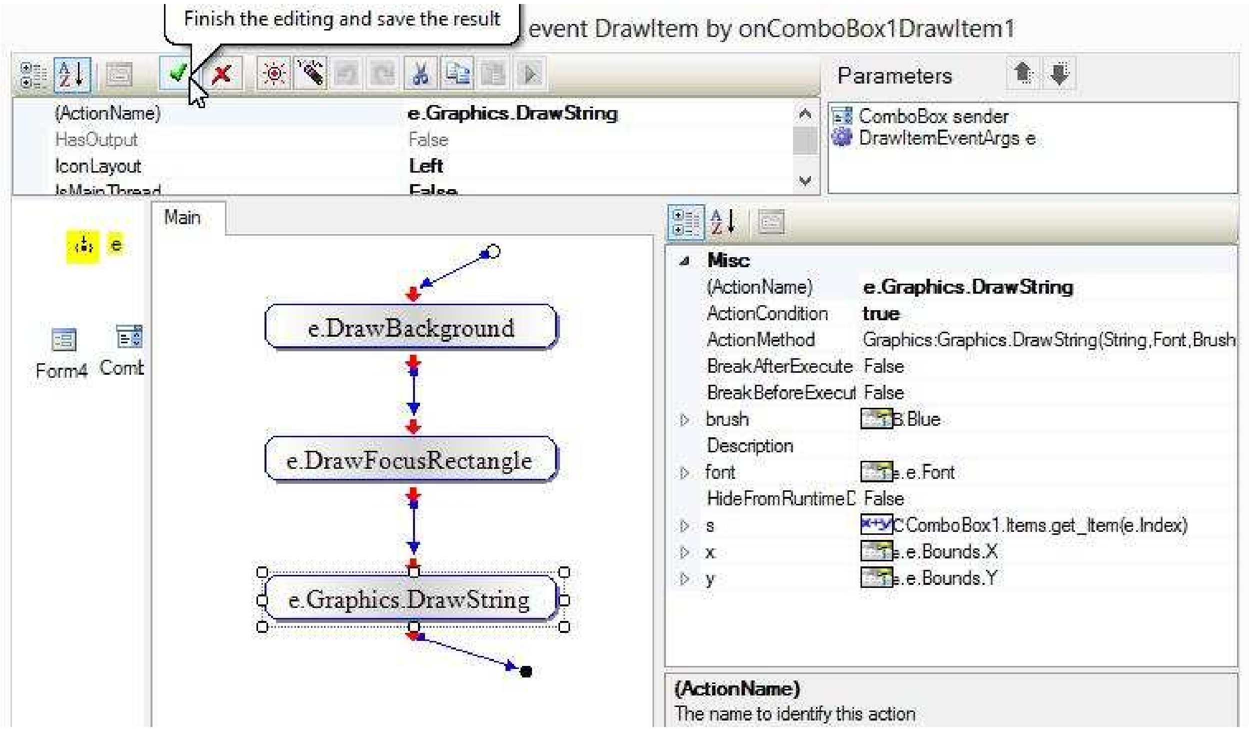


Click OK to finish creating the action:



(ActionName)	**e.Graphics.DrawString**
Description	
HideFromRuntimeDesign	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush,Single,Single)
s	CComboBox1.Items.get_Item(e.Index)
font	e.e.Font
brush	B.Blue
x	e.e.Bounds.X
y	e.e.Bounds.Y

Link the action to the last action. That is all for this event handler:



(ActionName)	e.Graphics.DrawString
HasOutput	False
IconLayout	Left
IsMainThread	False

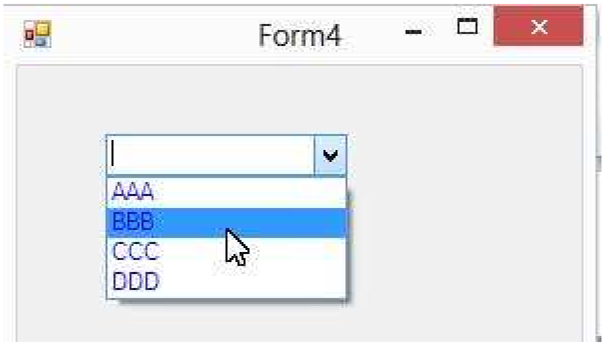
Misc

(ActionName)	e.Graphics.DrawString
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawString(String,Font,Brush
BreakAfterExecute	False
BreakBeforeExecut	False
brush	B.Blue
Description	
font	e.e.Font
HideFromRuntimeD	False
s	ComboBox1.Items.get_Item(e.Index)
x	e.e.Bounds.X
y	e.e.Bounds.Y

(ActionName)
The name to identify this action

Test

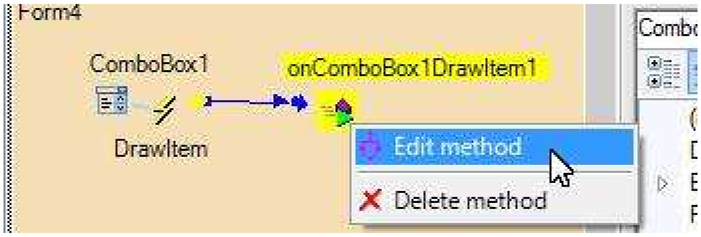
Run the project. This time the items are displayed using the brush we selected:



Draw image

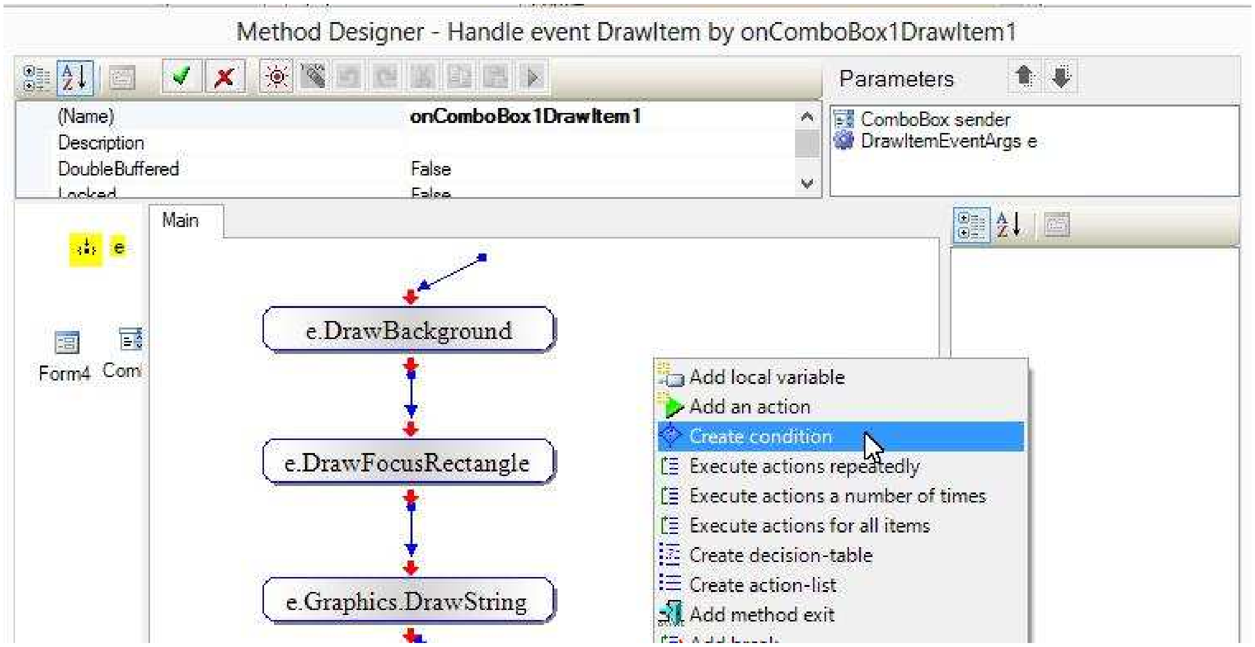
Suppose we created two image resources. One image is for selected state and the other for unselected state of the item. For creating resources, see http://www.limnor.com/support/Limnor%20Studio%20-%20User%20Guide%20-%20Resource%20Manager.pdf

Edit the event handler method to add image drawings:



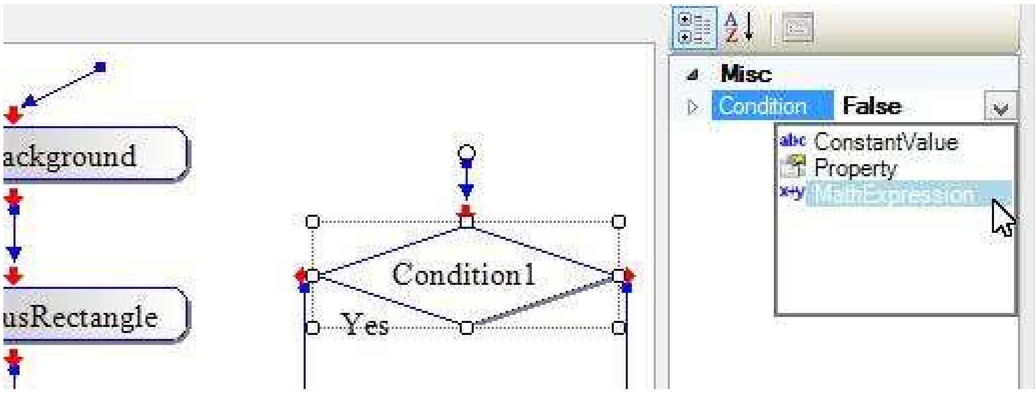
Check item state

Insert a Condition action to check item state:

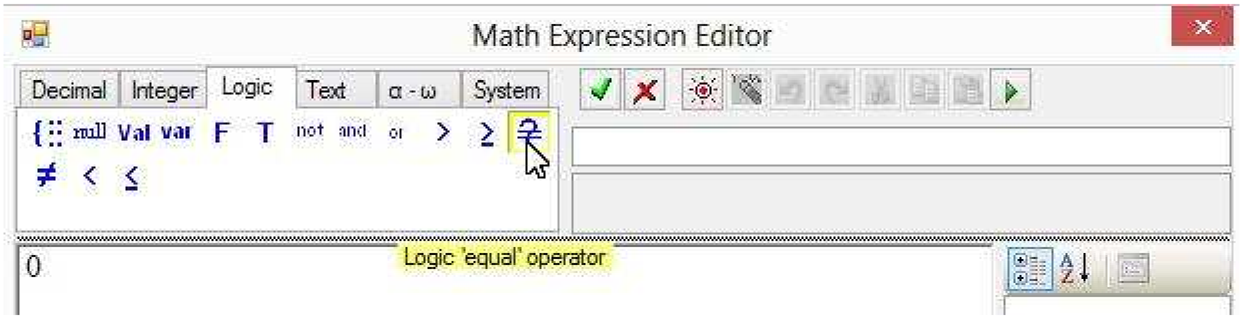


(Name)	onComboBox1DrawItem1
Description	
DoubleBuffered	False
Locked	False

Use an expression to check item state:



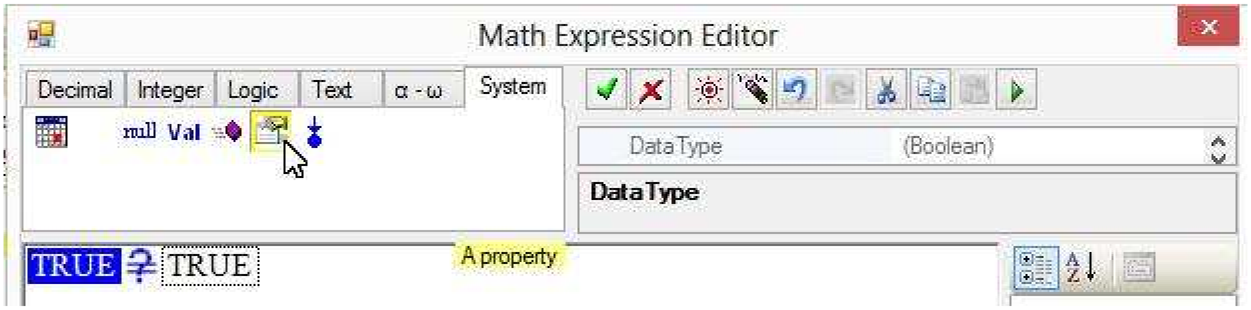
Use equal-checking:



0

Logic 'equal' operator

Select item state:

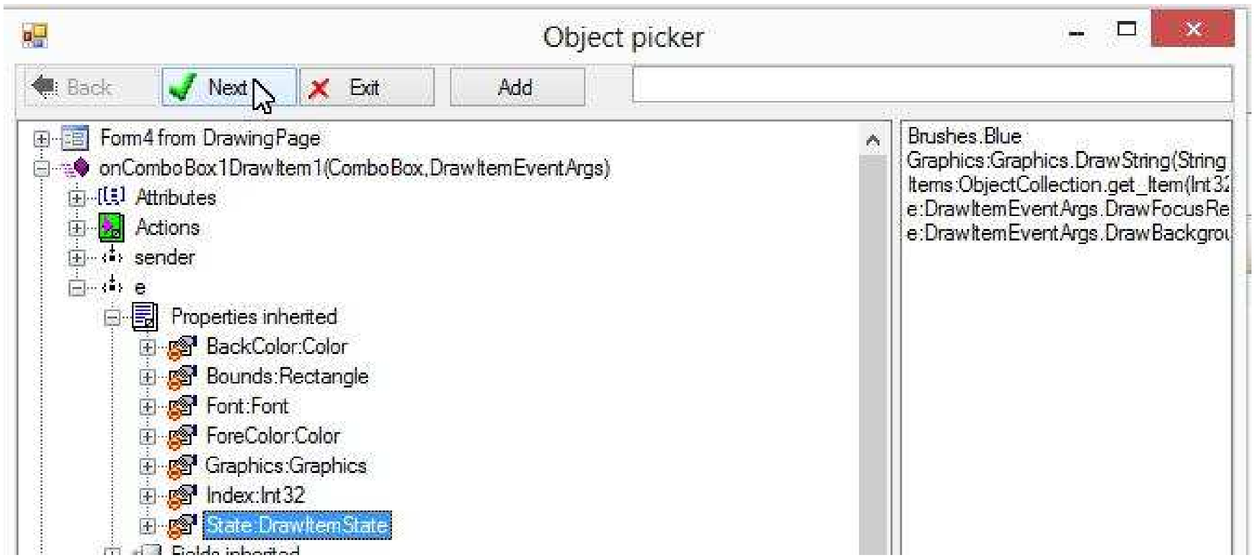


DataType (Boolean)

DataType

TRUE ≟ TRUE

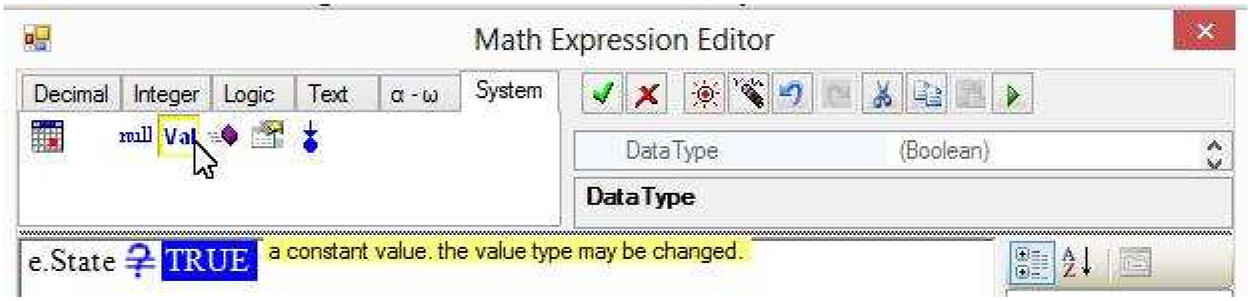
A property



```
Form4 from DrawingPage
onComboBox1DrawItem1(ComboBox,DrawItemEventArgs)
    Attributes
    Actions
    sender
    e
        Properties inherited
            BackColor:Color
            Bounds:Rectangle
            Font:Font
            ForeColor:Color
            Graphics:Graphics
            Index:Int32
            State:DrawItemState
        Fields inherited
```

```
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou
```

Select the second operant of the equal checking. Click “Val” icon:



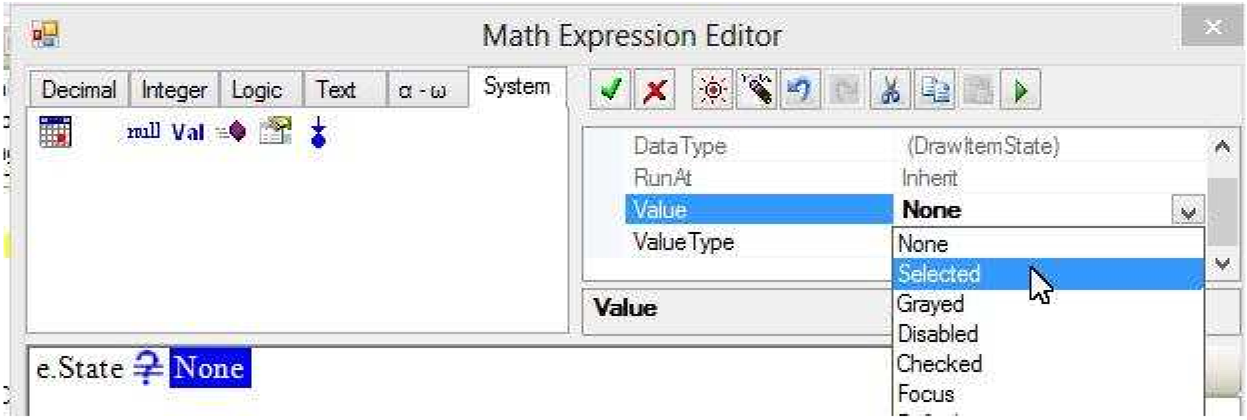
DataType (Boolean)

DataType

e.State ≟ TRUE

a constant value. the value type may be changed.

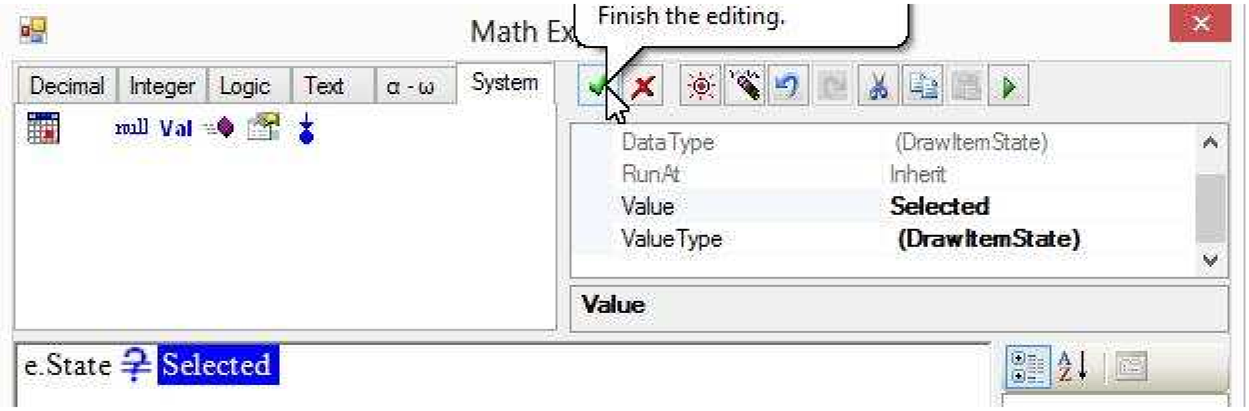
Select “Selected” state:



DataType	(DrawItemState)
RunAt	Inherit
Value	None
ValueType	

e.State ≟ None

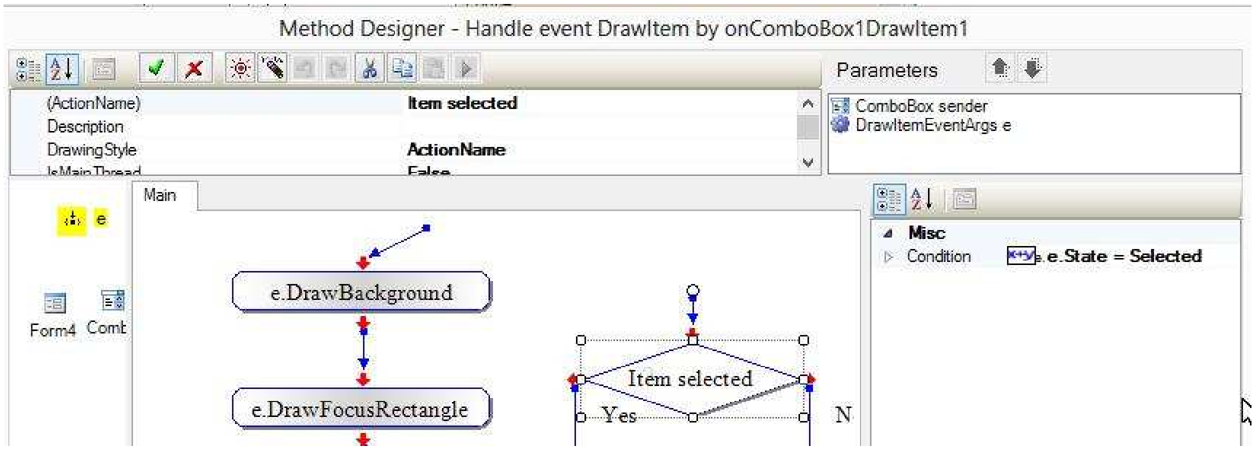
This is an expression for checking "Selected” state of the item:



DataType	(DrawItemState)
RunAt	Inherit
Value	Selected
ValueType	(DrawItemState)

e.State ≟ Selected

Rename the condition to “Item selected”:



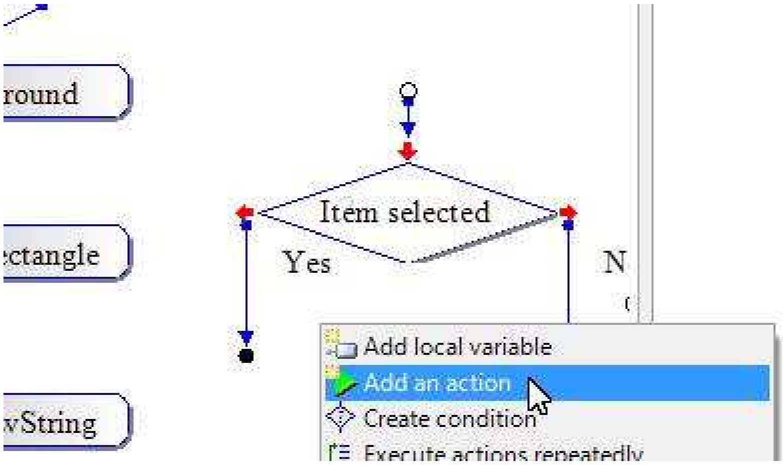
(ActionName)	Item selected
Description	
DrawingStyle	ActionName
IsMainThread	False

ComboBox sender
DrawItemEventArgs e

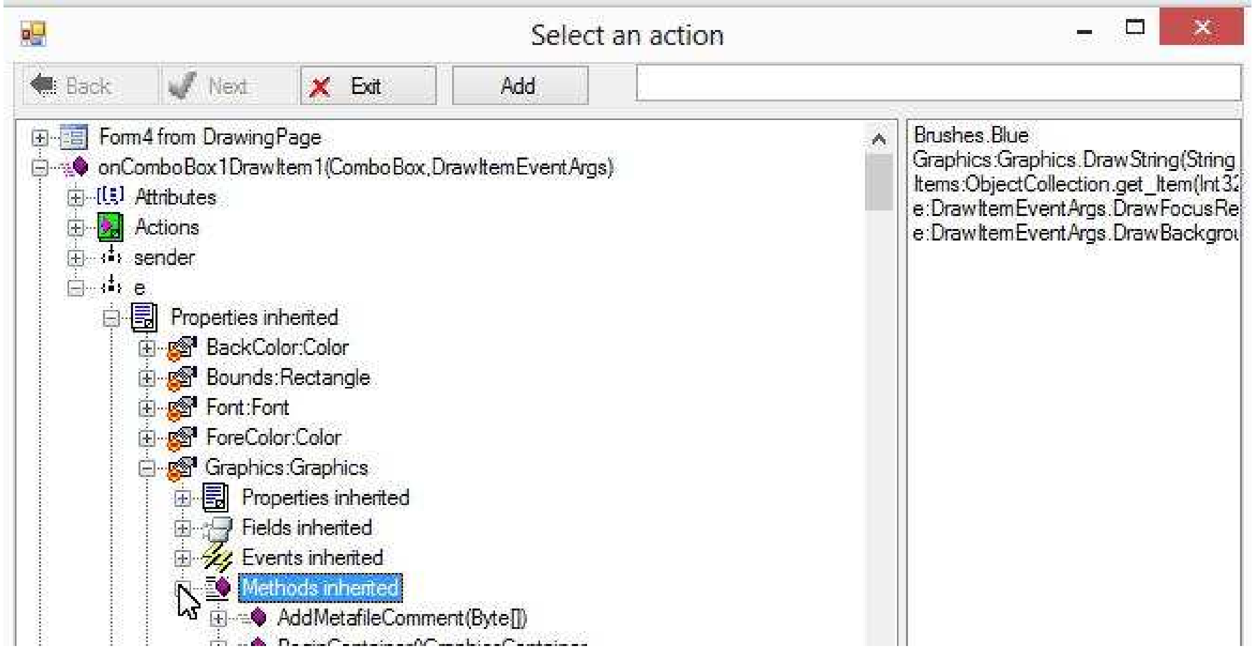
Misc
Condition e.State = Selected

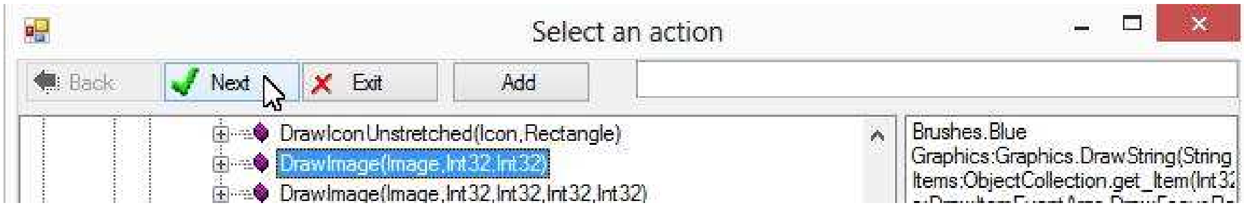
Draw image for selected item

Add a new action to draw image:

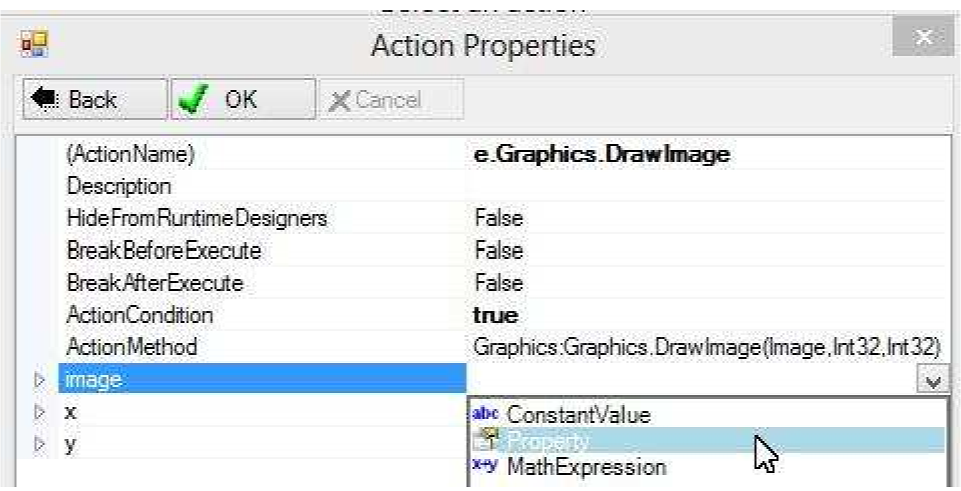


Select a DrawImage method of the Graphics property of parameter e:

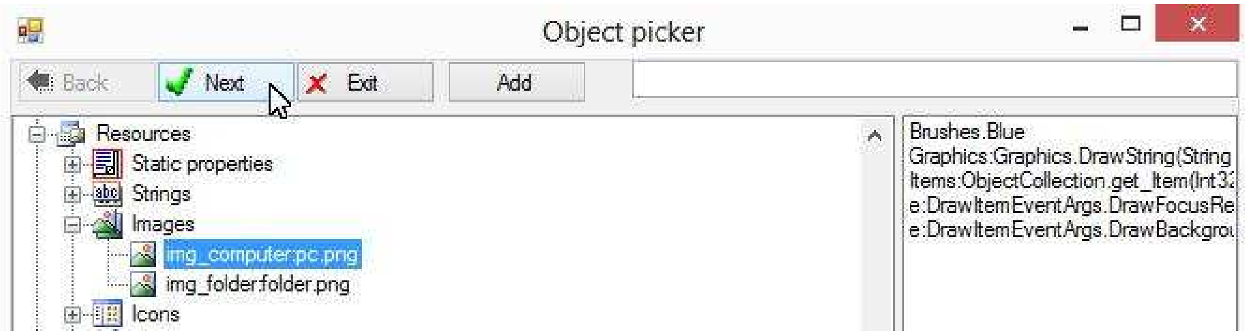



Select an action
Back
Next
Exit
Add
DrawIconUnstretched(Icon,Rectangle)
DrawImage(Image,Int32,Int32)
DrawImage(Image,Int32,Int32,Int32,Int32)
Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int32

Select an image resource for "image" of the action:

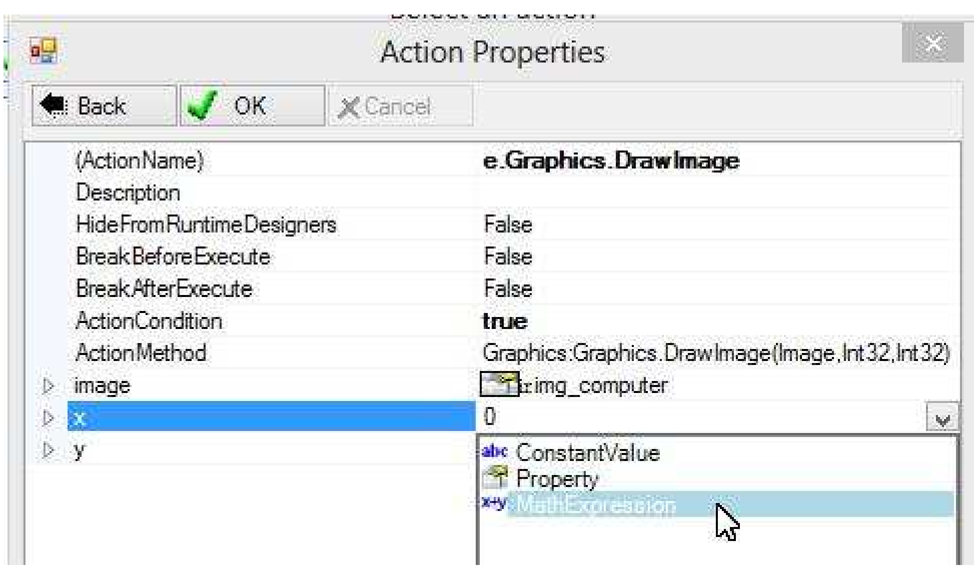


(ActionName)	e.Graphics.DrawImage
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawImage(Image,Int32,Int32)
image	
x	
y	

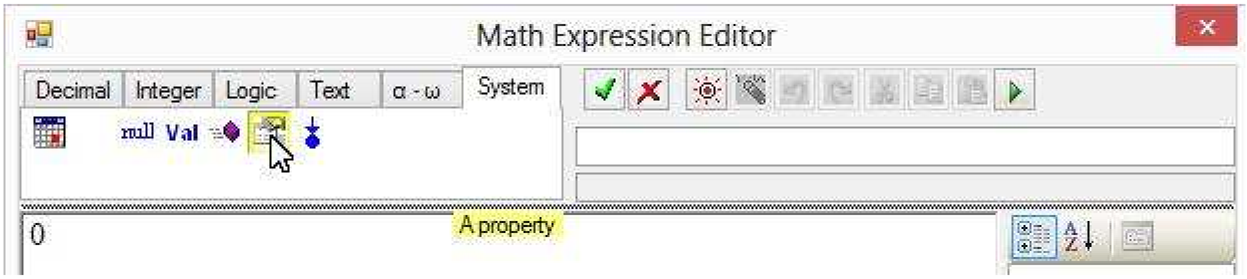


Brushes.Blue
Graphics:Graphics.DrawString(String
Items:ObjectCollection.get_Item(Int3:
e:DrawItemEventArgs.DrawFocusRe
e:DrawItemEventArgs.DrawBackgrou

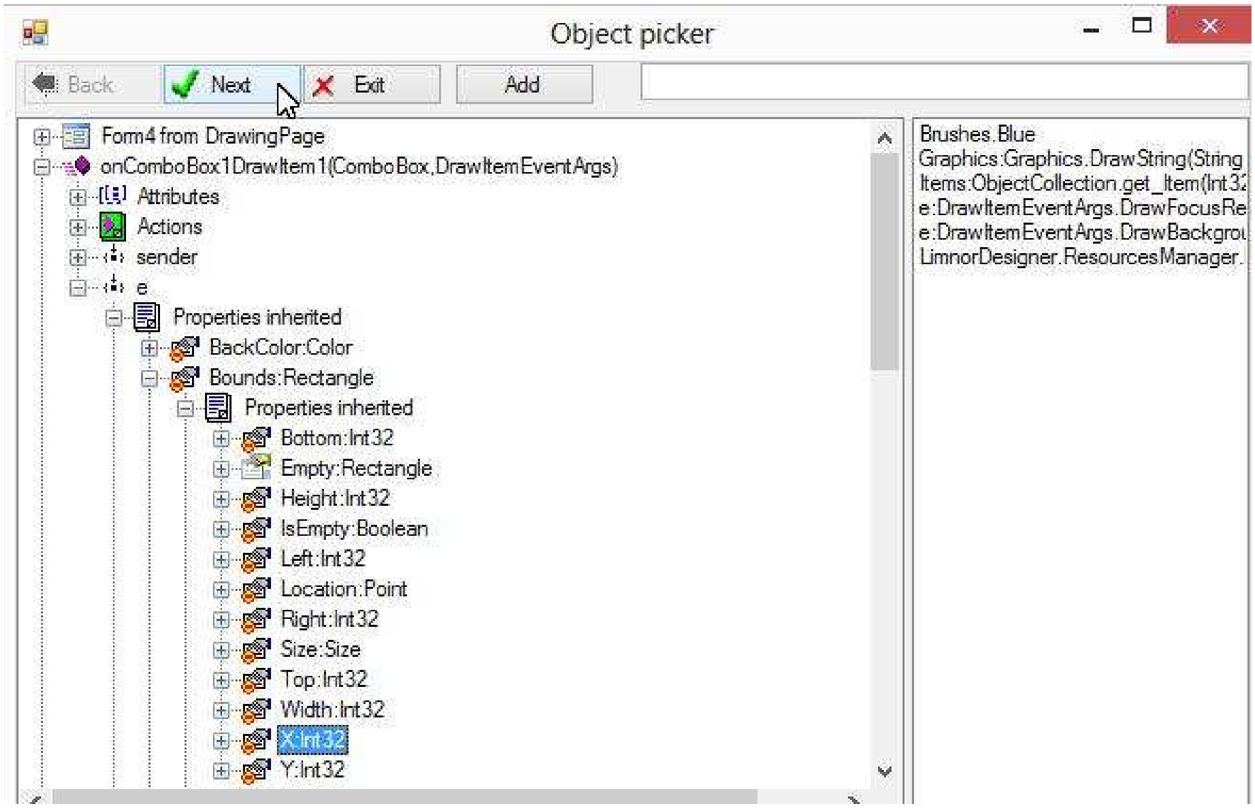
Draw image at location (1, 1) of the item area:

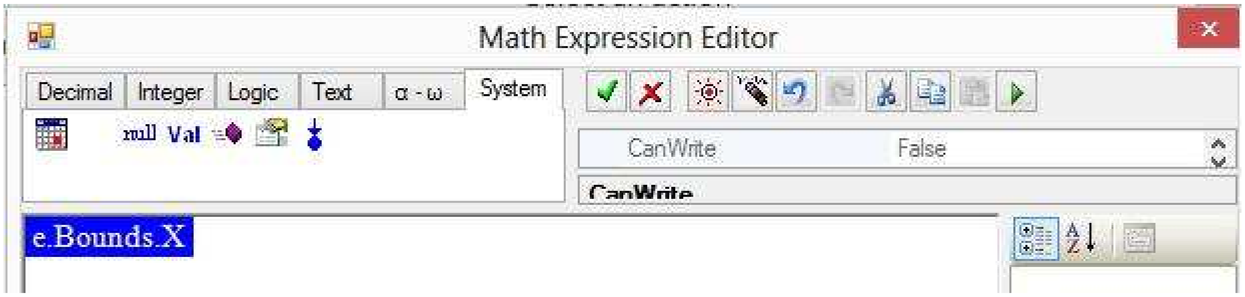


(ActionName)	e.Graphics.DrawImage
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	Graphics:Graphics.DrawImage(Image,Int32,Int32)
image	:img_computer
x	0
y	



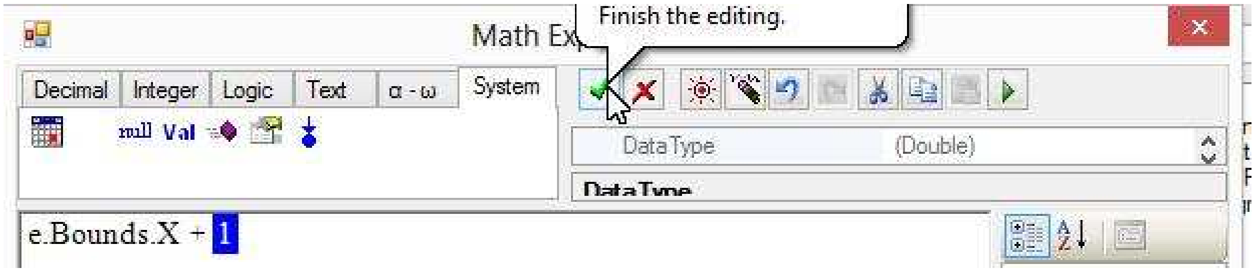
0





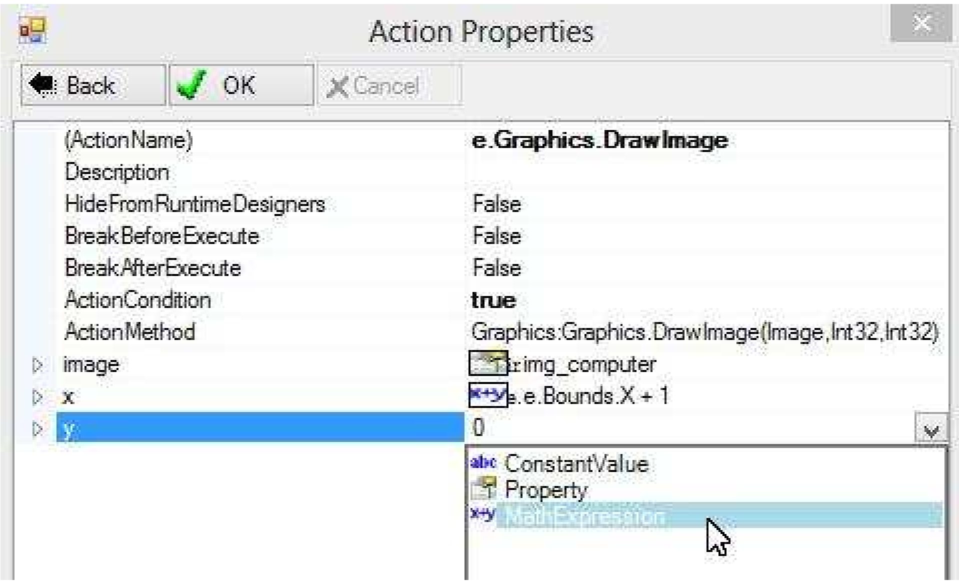
e.Bounds.X

Add 1 to it:

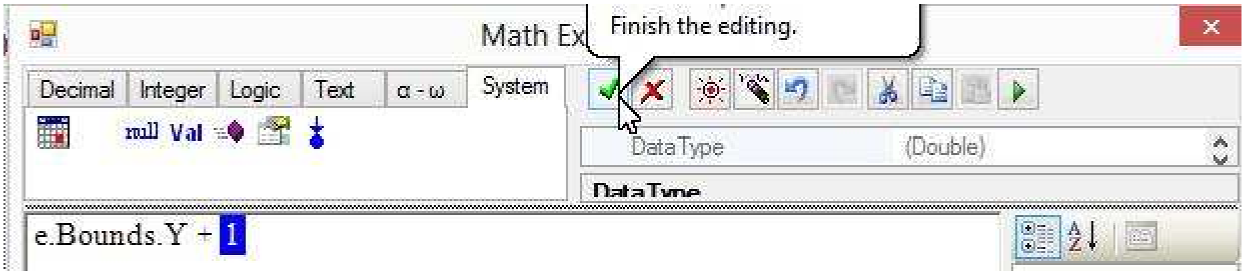


DataType	(Double)

e.Bounds.X + 1

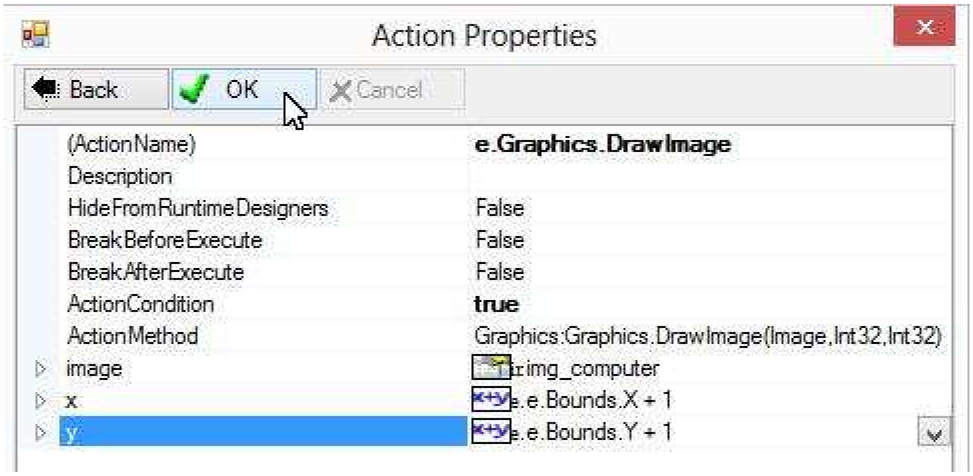


(ActionName)	**e.Graphics.DrawImage**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	Graphics:Graphics.DrawImage(Image,Int32,Int32)
▷ image	img_computer
▷ x	e.e.Bounds.X + 1
▷ y	0



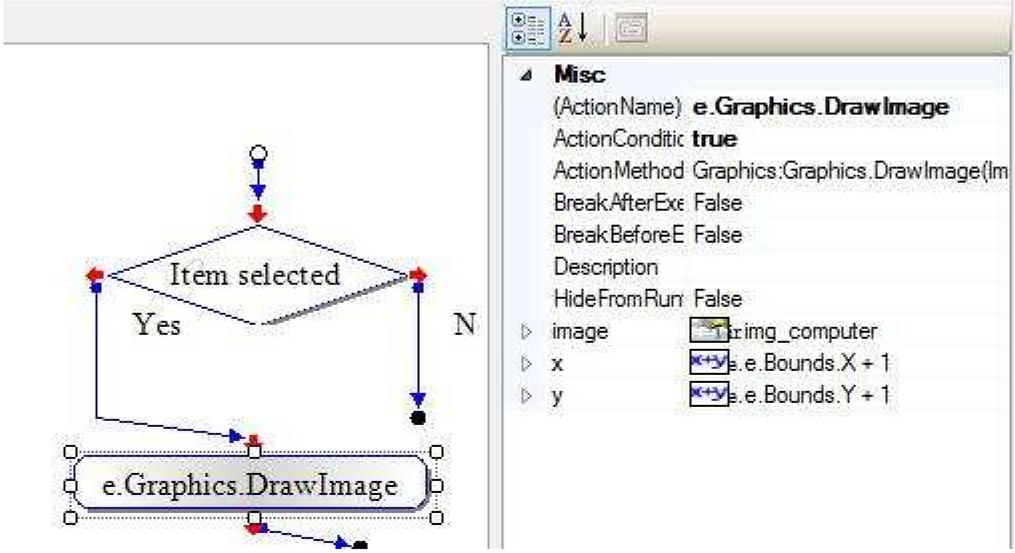
DataType	(Double)

e.Bounds.Y + 1



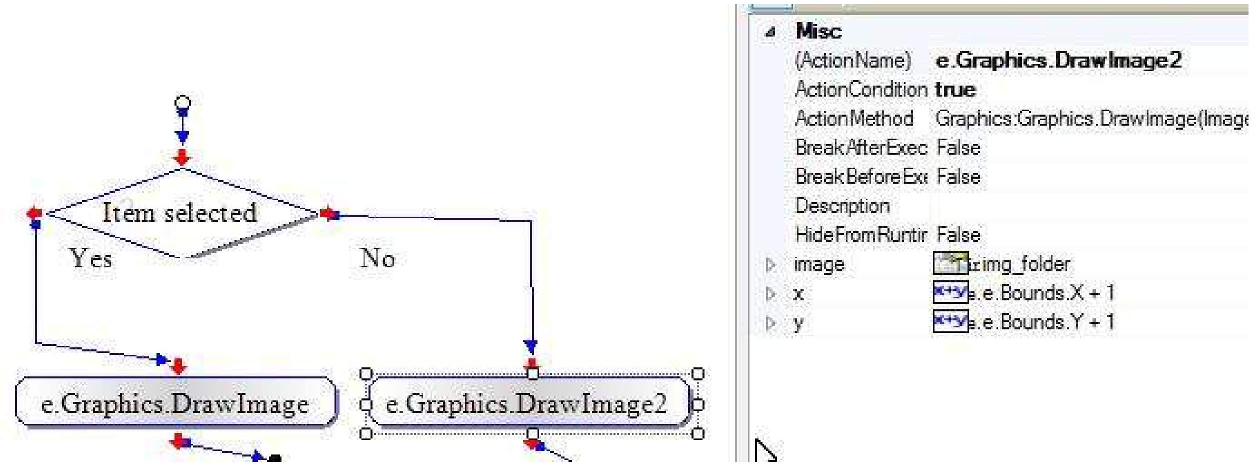
(ActionName)	**e.Graphics.DrawImage**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	Graphics:Graphics.DrawImage(Image,Int32,Int32)
▷ image	img_computer
▷ x	e.e.Bounds.X + 1
▷ y	e.e.Bounds.Y + 1

Link it to the Yes port of the condition action:



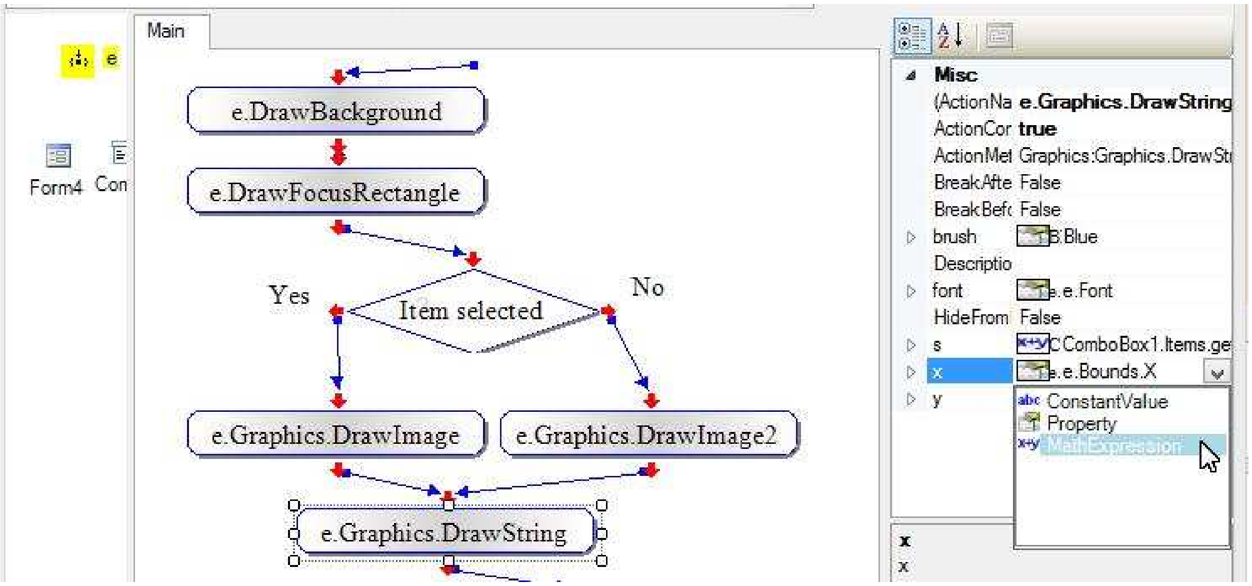
Misc	
(ActionName)	**e.Graphics.DrawImage**
ActionConditic	**true**
ActionMethod	Graphics:Graphics.DrawImage(Im
BreakAfterExe	False
BreakBeforeE	False
Description	
HideFromRun	False
image	img_computer
x	e.e.Bounds.X + 1
y	e.e.Bounds.Y + 1

Create another image drawing action to draw another image and link it to the No port of the condition action:

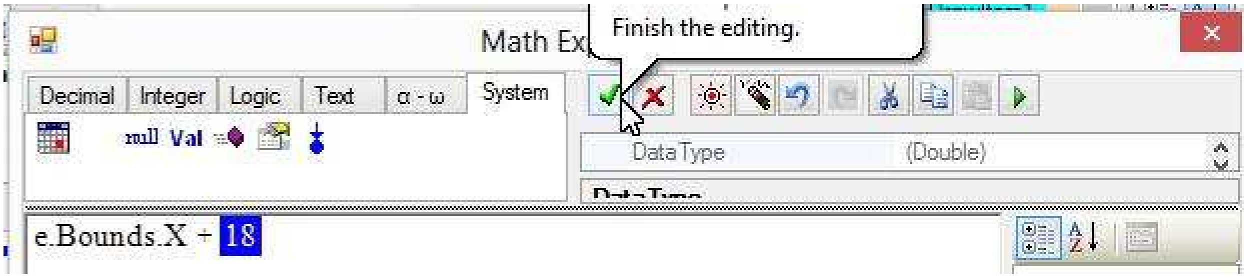


Misc	
(ActionName)	**e.Graphics.DrawImage2**
ActionCondition	**true**
ActionMethod	Graphics:Graphics.DrawImage(Image
BreakAfterExec	False
BreakBeforeExe	False
Description	
HideFromRuntir	False
image	img_folder
x	e.e.Bounds.X + 1
y	e.e.Bounds.Y + 1

Rearrange the actions to draw image first and draw text next. The image width is 16. We want to draw text after the image. So, we want to draw the text from 18 pixels:

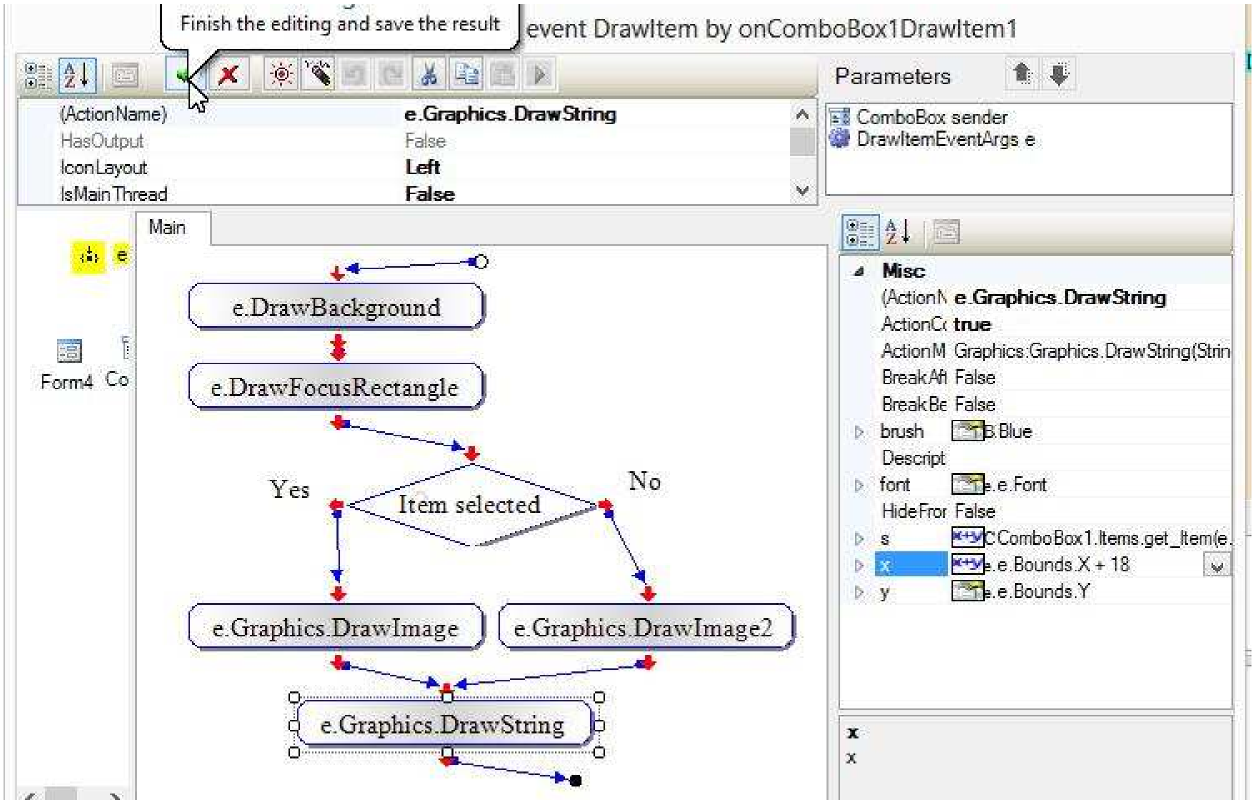


Misc
(ActionNa e.Graphics.DrawString
ActionCor true
ActionMet Graphics:Graphics.DrawStr
BreakAfte False
BreakBefc False
brush B.Blue
Descriptio
font e.e.Font
HideFrom False
s CComboBox1.Items.ge
x e.e.Bounds.X
y



DataType (Double)

```
e.Bounds.X + 18
```


Finish the editing and save the result

event DrawItem by onComboBox1DrawItem1

(ActionName)	e.Graphics.DrawString
HasOutput	False
IconLayout	Left
IsMainThread	False

Parameters

ComboBox sender
DrawItemEventArgs e

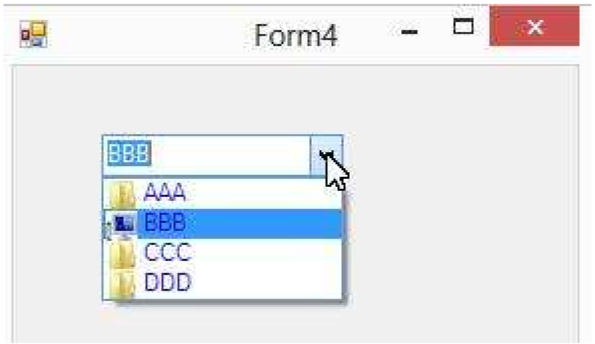
Misc

(ActionN	e.Graphics.DrawString
ActionCc	true
ActionM	Graphics:Graphics.DrawString(Strin
BreakAft	False
BreakBe	False
brush	B.Blue
Descript	
font	e.e.Font
HideFror	False
s	CComboBox1.Items.get_Item(e.
x	e.e.Bounds.X + 18
y	e.e.Bounds.Y

x
x

Test

Run the project. Images appear in the list:



Summary

By using owner draw, you have complete freedom of drawing list items. Set DrawMode property and then handle DrawItem event. Graphics property of event parameter e provides a very large number of drawing methods. Index property of e indicates the item to draw. State property of e indicates item selection state. Bounds property of e indicates the drawing area for the item.

The examples provided here are some simple drawings.

Feedback

Please send your feedback to support@limnor.com