Receive Emails

Contents

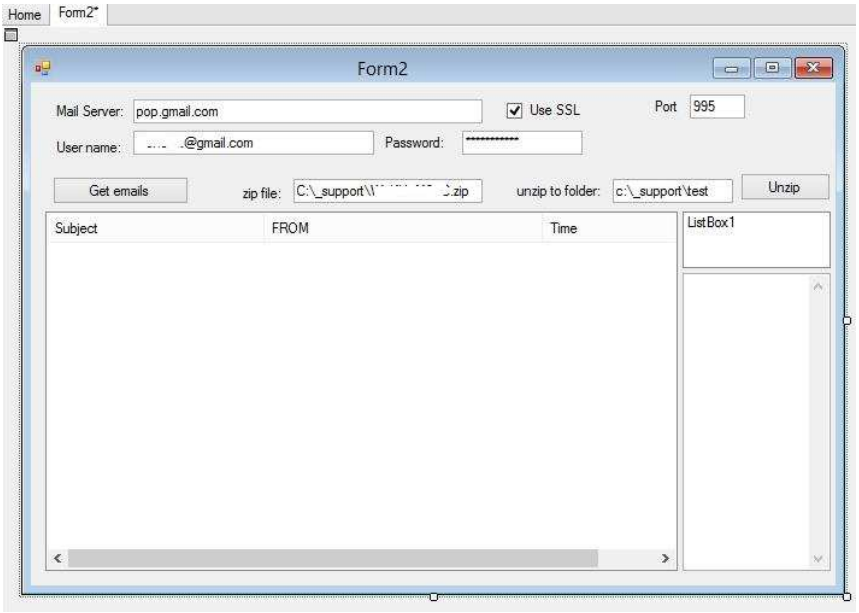
Introduction

This sample demonstrates following programming skills

- Use third-party software libraries. This sample uses an email receiver libraries downloaded from http://hpop.sourceforge.net/
- Process items in a collection or array and access single item
- Unzip files
- Create methods

UI Design

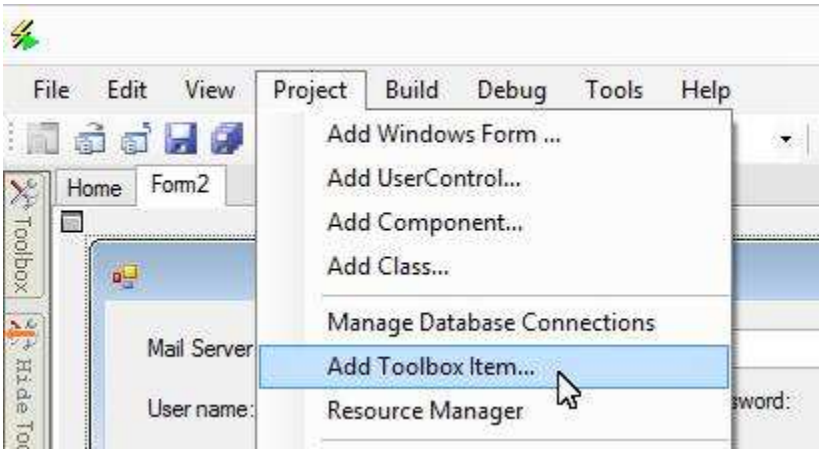
The sample is a Windows Form application project. It uses a list view to show an email list for making email selection. It uses a list box to show email attachments for a selected email. It uses a text box to show email body of a selected email. It uses two splitters to arrange display of those email parts.



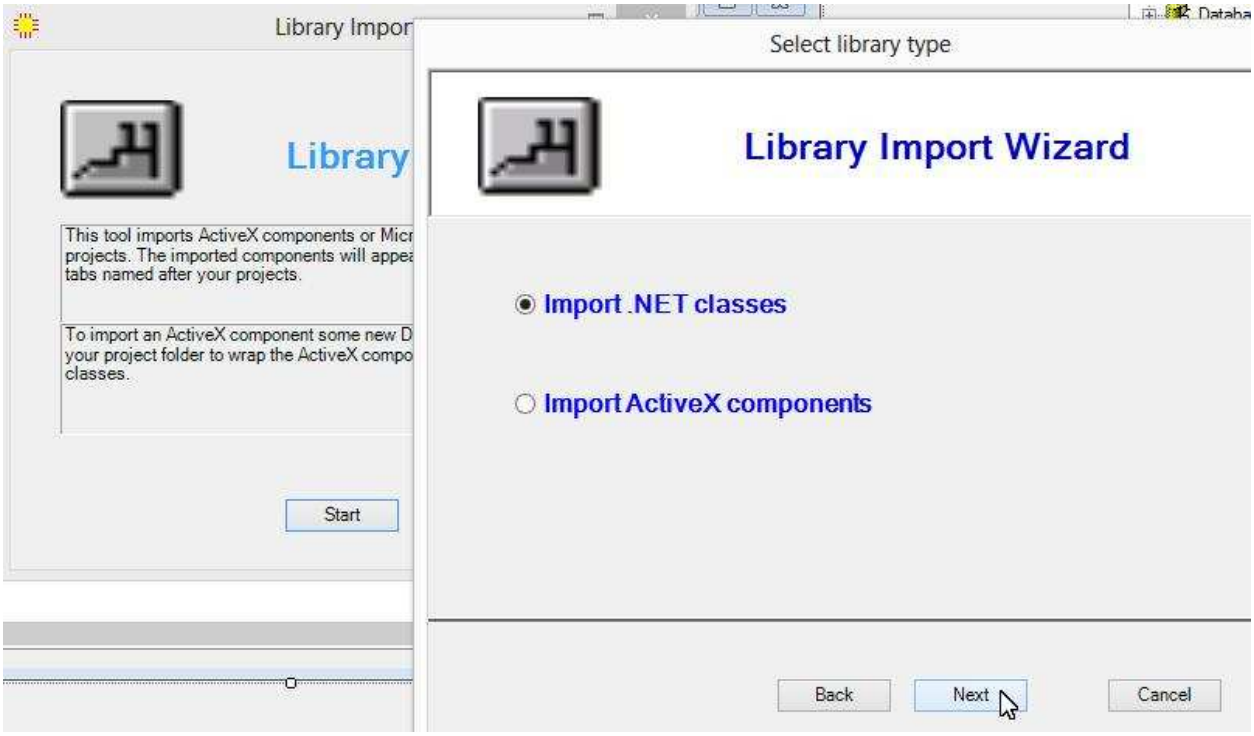
Add Mail Receiver

For this sample, a mail receiver component was downloaded from http://hpop.sourceforge.net/. The component is in file OpenPop.DLL and named Pop3Client. Follow steps below to add the component to a form.

1. Choose menu “Project|Add Toolbox Item…”

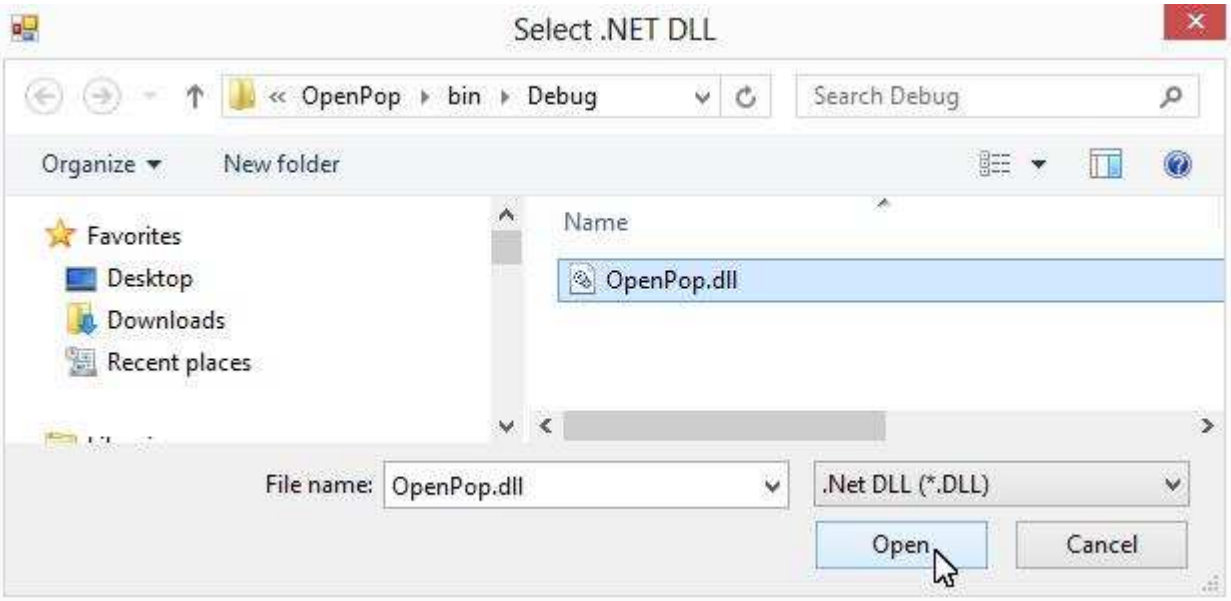


2. Choose “Import .Net classes” and click Next

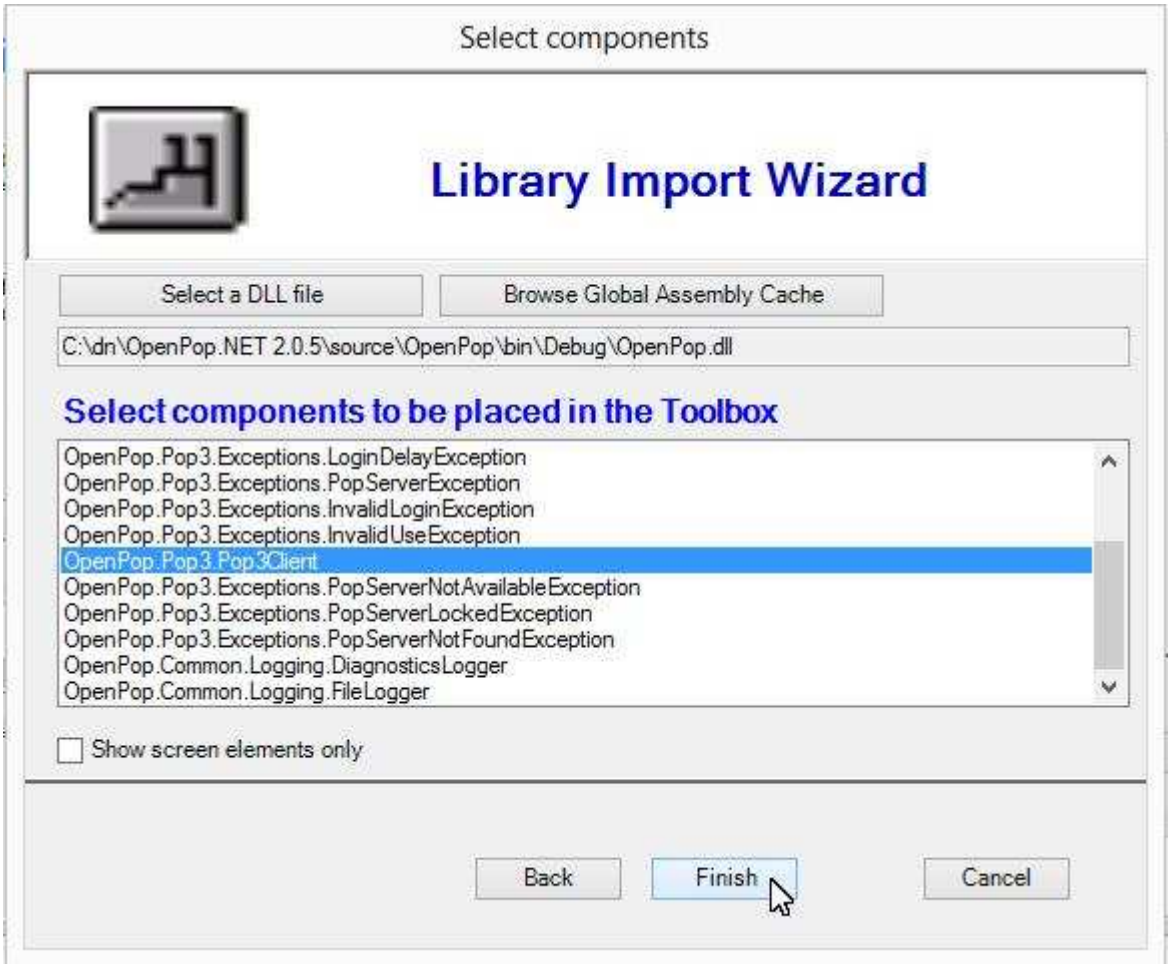


3. Choose “Select a DLL file”

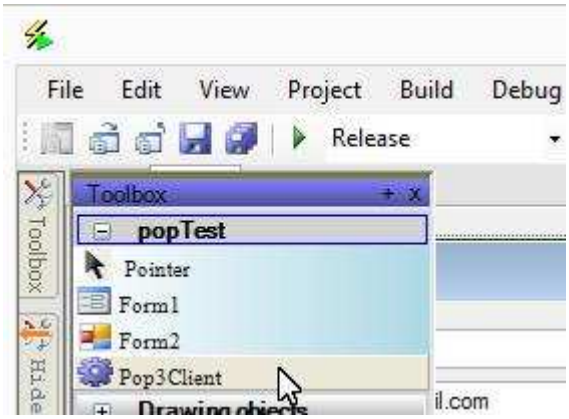
4. Select software library DLL. For this sample, the DLL is OpenPop.DLL



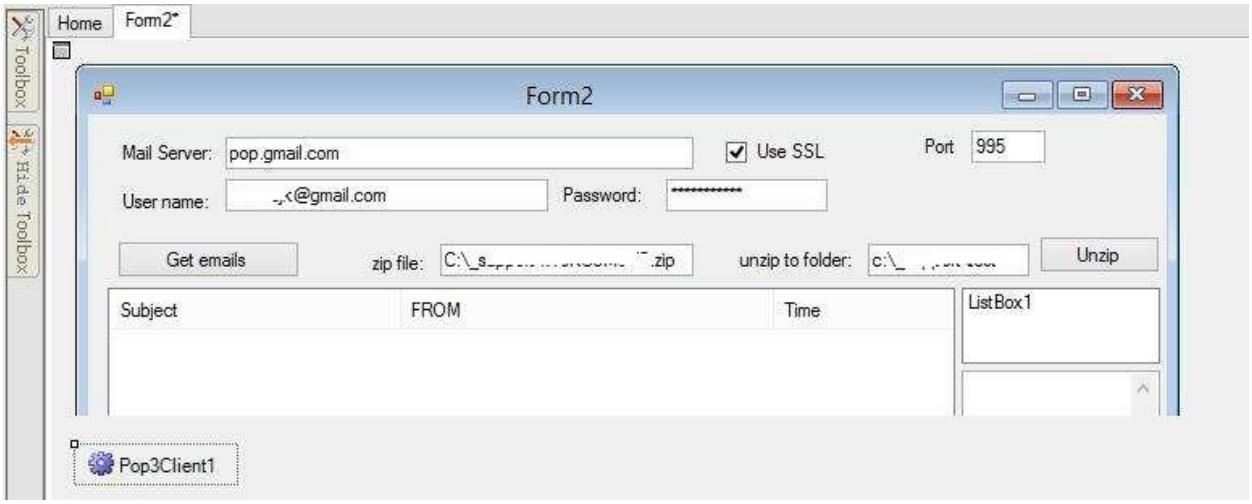
5. Classes in the DLL are displayed. Choose Pop3Client



6. The component Pop3Client appears in the Toolbox. **Drag** it and **drop** it to the form (you must use a drag/drop operation):



7. An instance of Pop3Client appears on the form



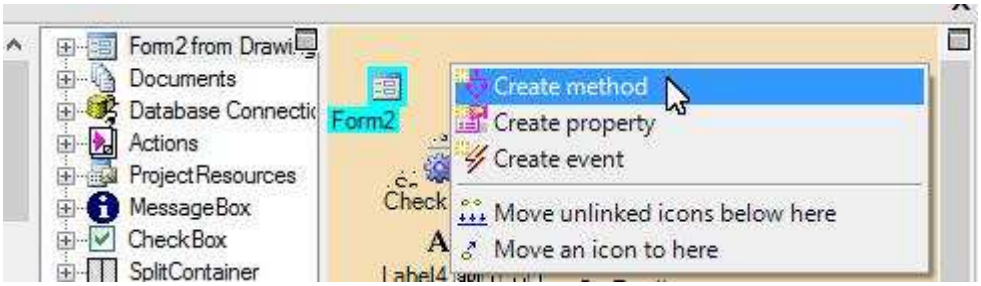
Once an instance of Pop3Client appears on the form, we can use its properties, methods and events for our programming. The functionality of Pop3Client is mostly expressed in its methods. This sample shows how to use its methods.

Create Mail Receiving Method

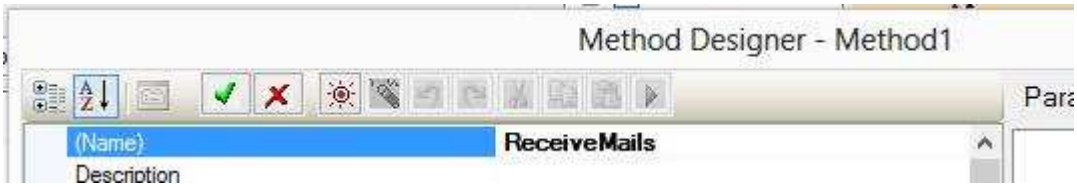
Based on a sample provided by http://hpop.sourceforge.net/ in C#, this sample shows equivalent visual programming in Limnor Studio. A new method is created to connect to a POP server and receive emails.

Create new method

1. Create a new method

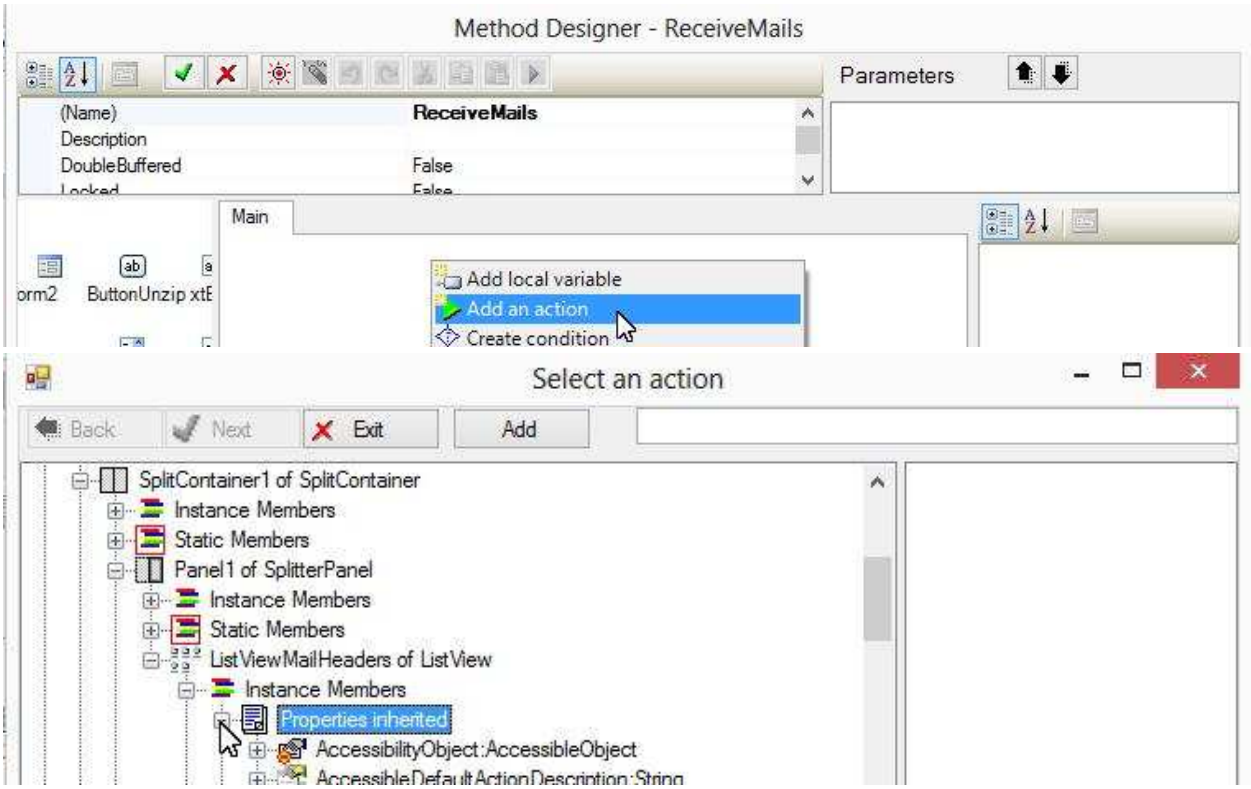


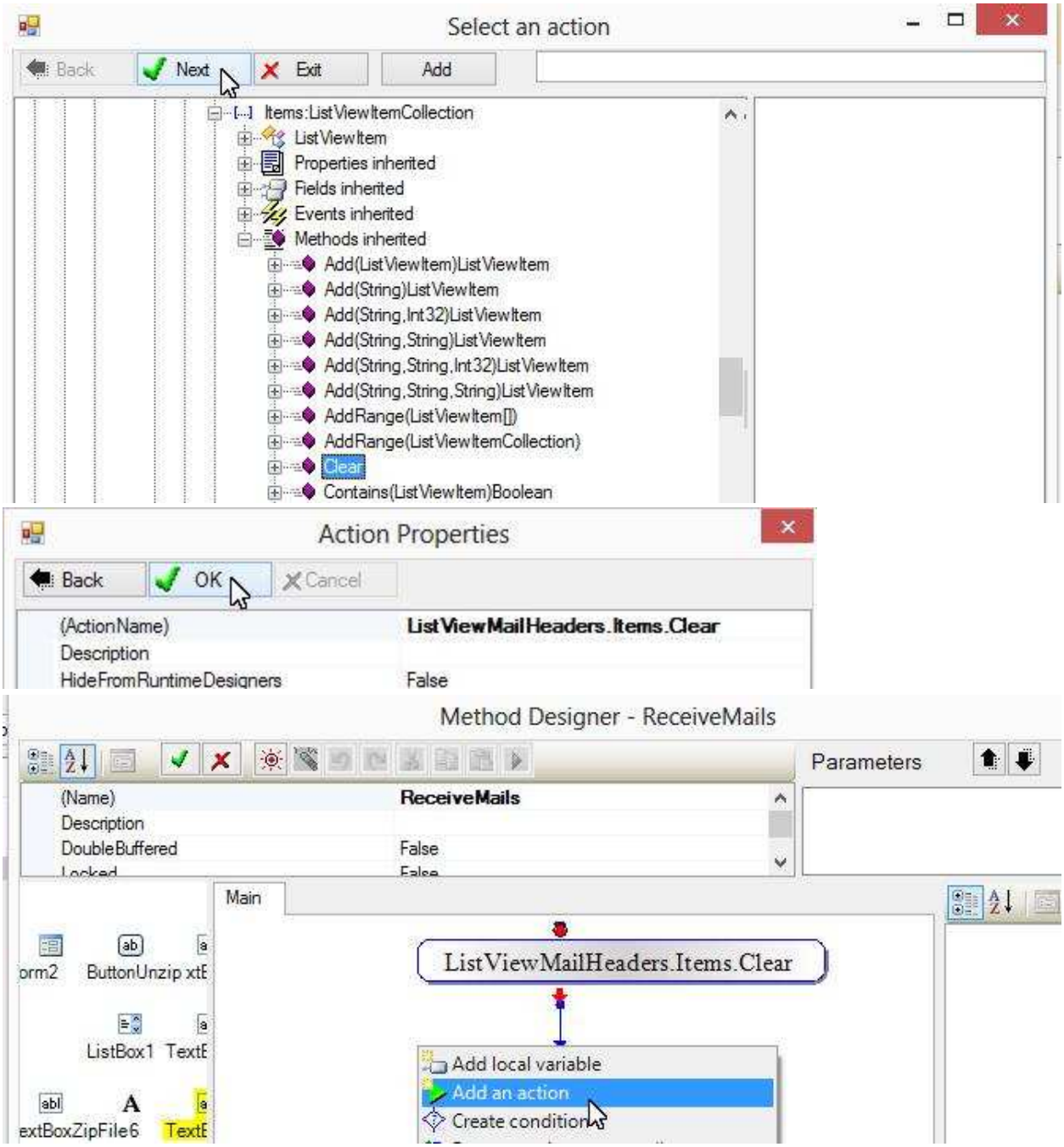
2. Give the new method a meaningful name:

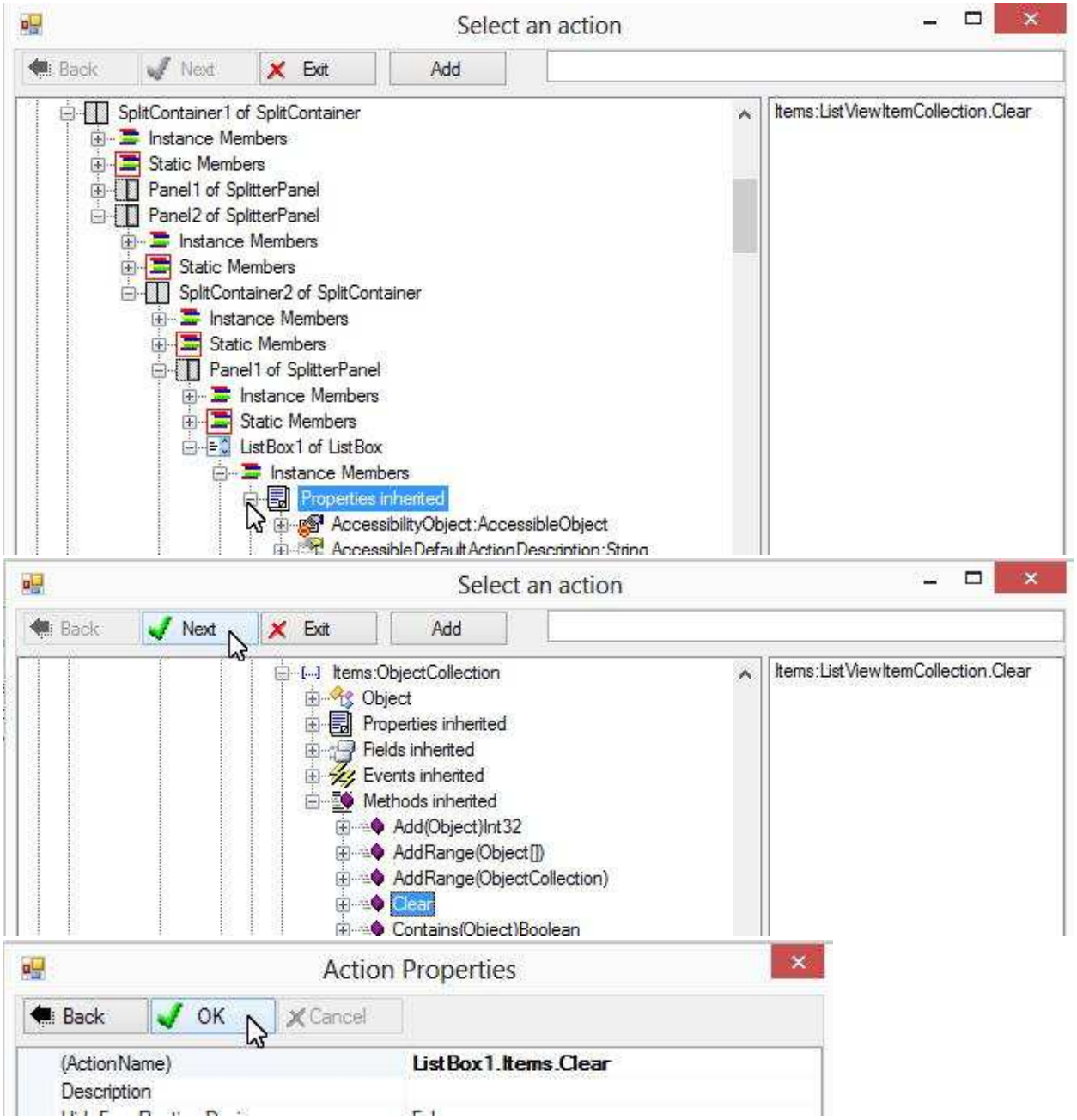


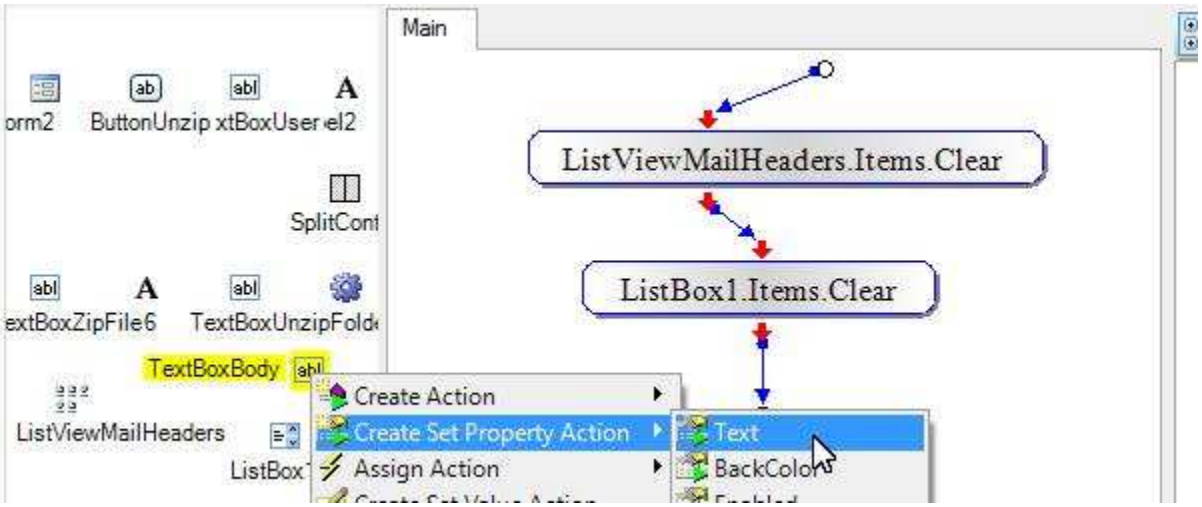
Clear email contents display

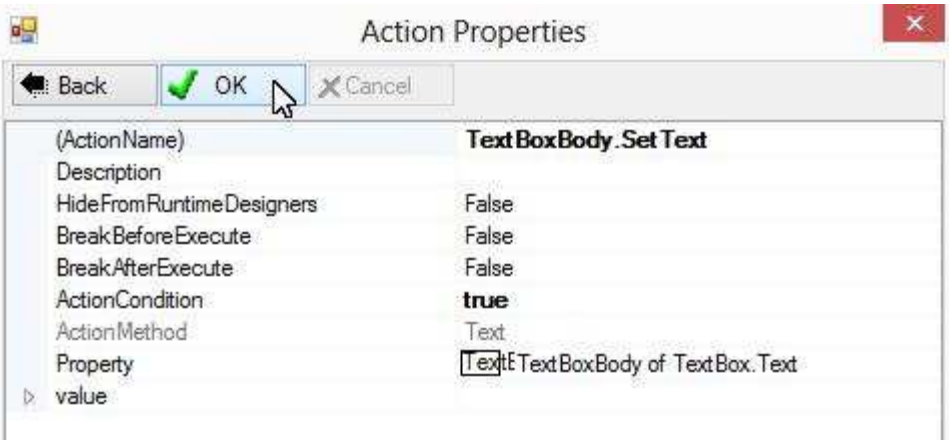
3. Create actions to remove list view items, list box items and email body



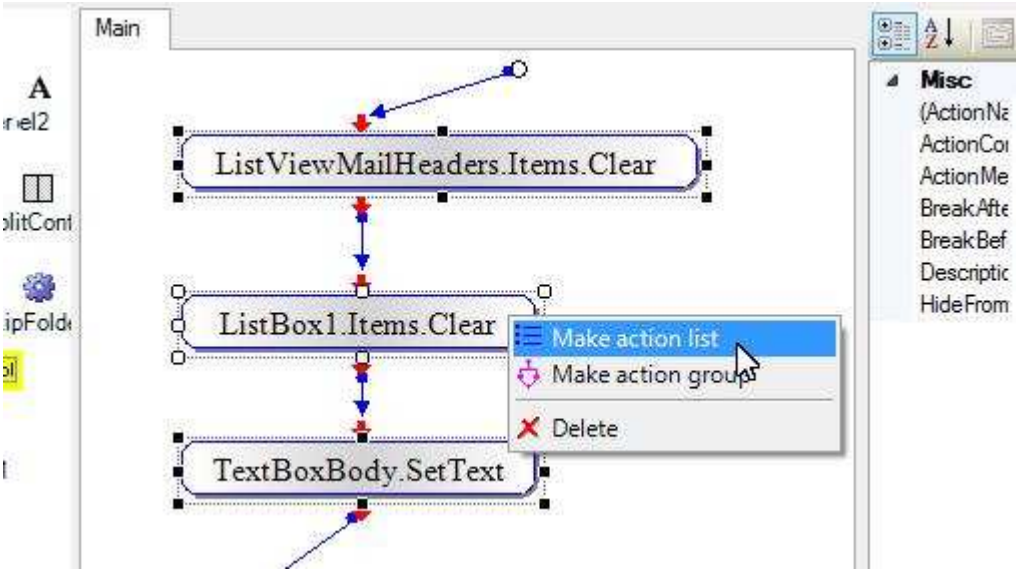

Select an action
Back
Next
Exit
Add
Items:ListViewItemCollection
ListViewItem
Properties inherited
Fields inherited
Events inherited
Methods inherited
Add(ListViewItem)ListViewItem
Add(String)ListViewItem
Add(String,Int32)ListViewItem
Add(String,String)ListViewItem
Add(String,String,Int32)ListViewItem
Add(String,String,String)ListViewItem
AddRange(ListViewItem[])
AddRange(ListViewItemCollection)
Clear
Contains(ListViewItem)Boolean
Action Properties
Back
OK
Cancel
(ActionName)
ListViewMailHeaders.Items.Clear
Description
HideFromRuntimeDesigners
False
Method Designer - ReceiveMails
Parameters
(Name)
ReceiveMails
Description
DoubleBuffered
False
Main
ListViewMailHeaders.Items.Clear
Add local variable
Add an action
Create condition


Select an action
Back
Next
Exit
Add
SplitContainer1 of SplitContainer
Instance Members
Static Members
Panel1 of SplitterPanel
Panel2 of SplitterPanel
Instance Members
Static Members
SplitContainer2 of SplitContainer
Instance Members
Static Members
Panel1 of SplitterPanel
Instance Members
Static Members
ListBox1 of ListBox
Instance Members
Properties inherited
AccessibilityObject:AccessibleObject
Items:ListViewItemCollection.Clear
Select an action
Back
Next
Exit
Add
Items:ObjectCollection
Object
Properties inherited
Fields inherited
Events inherited
Methods inherited
Add(Object)Int32
AddRange(Object[])
AddRange(ObjectCollection)
Clear
Contains(Object)Boolean
Items:ListViewItemCollection.Clear
Action Properties
Back
OK
Cancel
(ActionName)
ListBox1.Items.Clear
Description



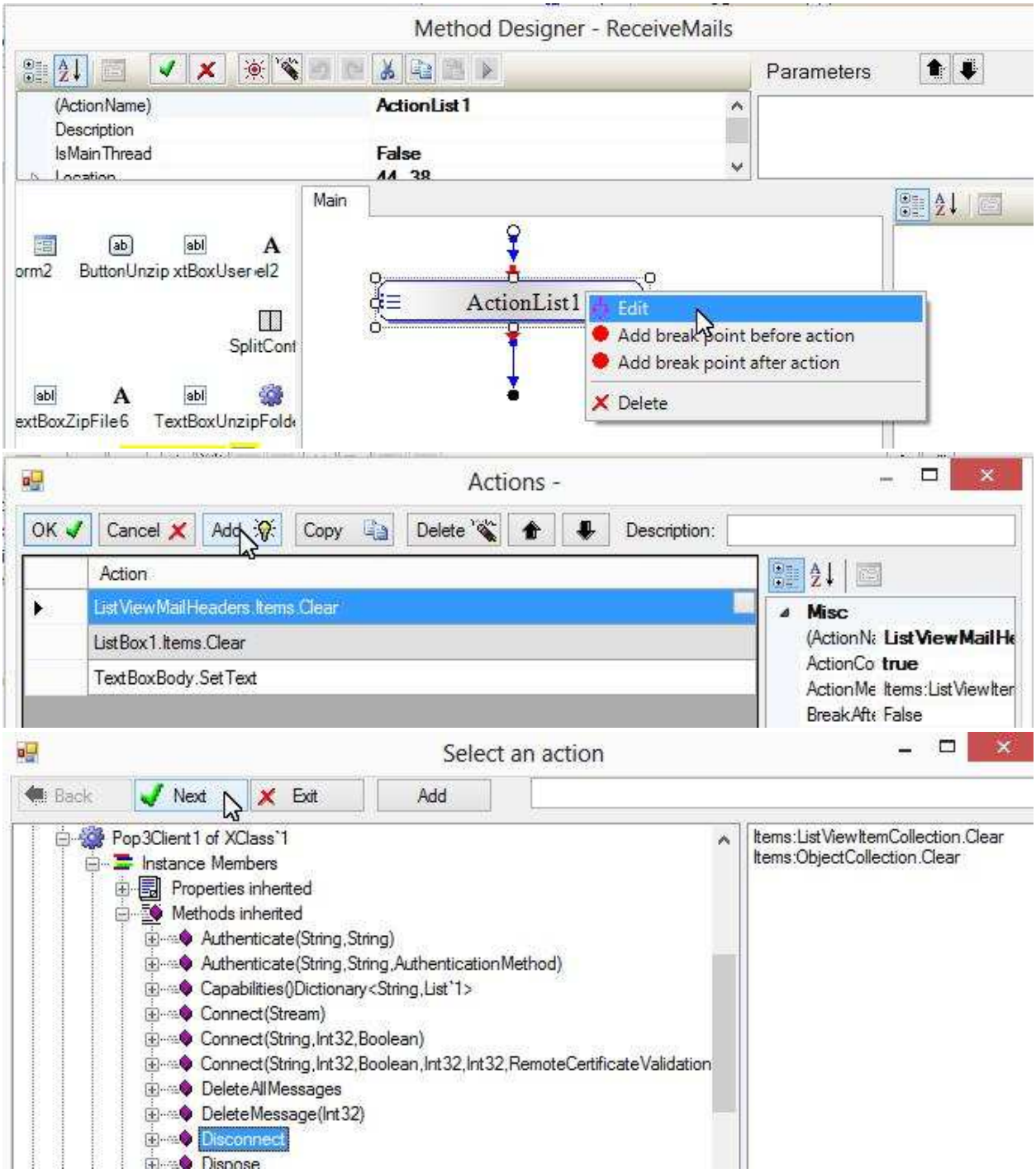


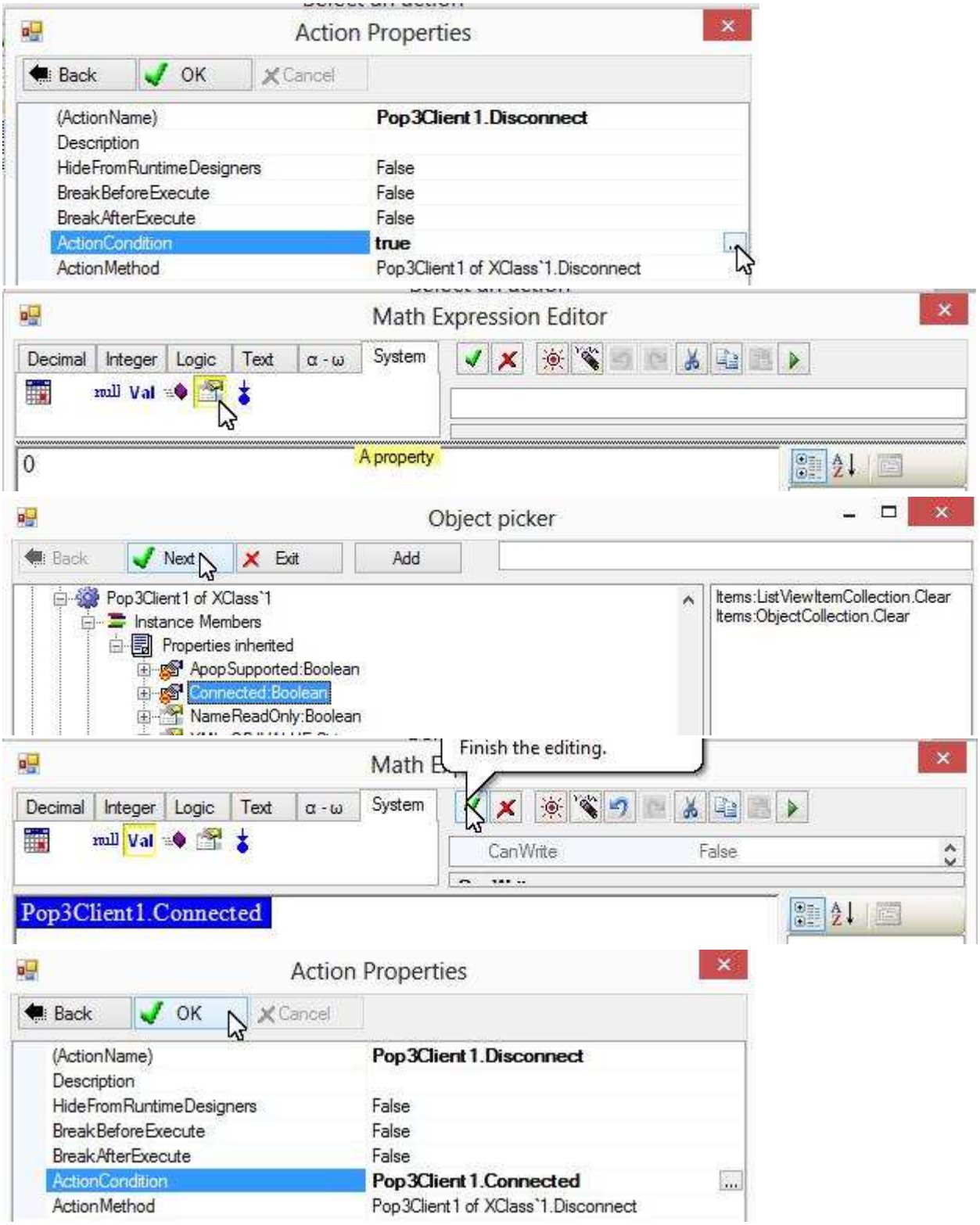
To save screen space, we may group the 3 new actions into one single action by selecting them and right-click the selection and choose “Make action list”:



Disconnect existing connection

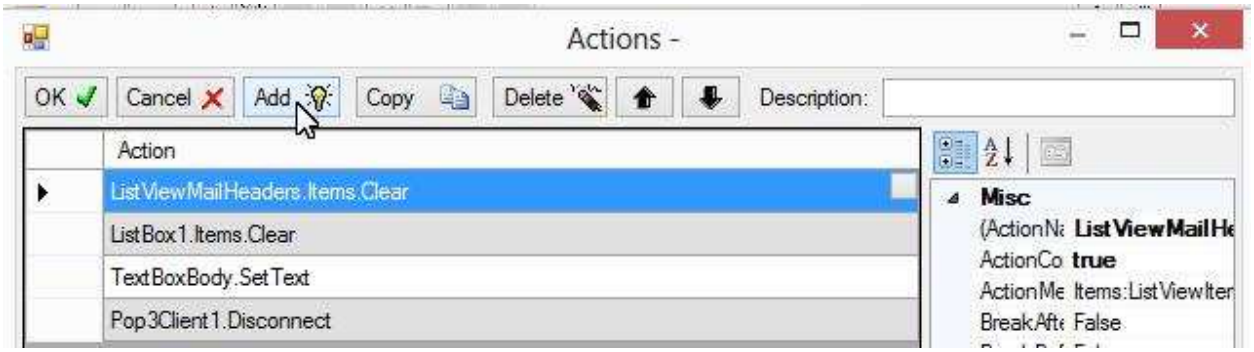
4. Before making a new connection to an email server, add an action to the action list to disconnect if Pop3Client component is still connected

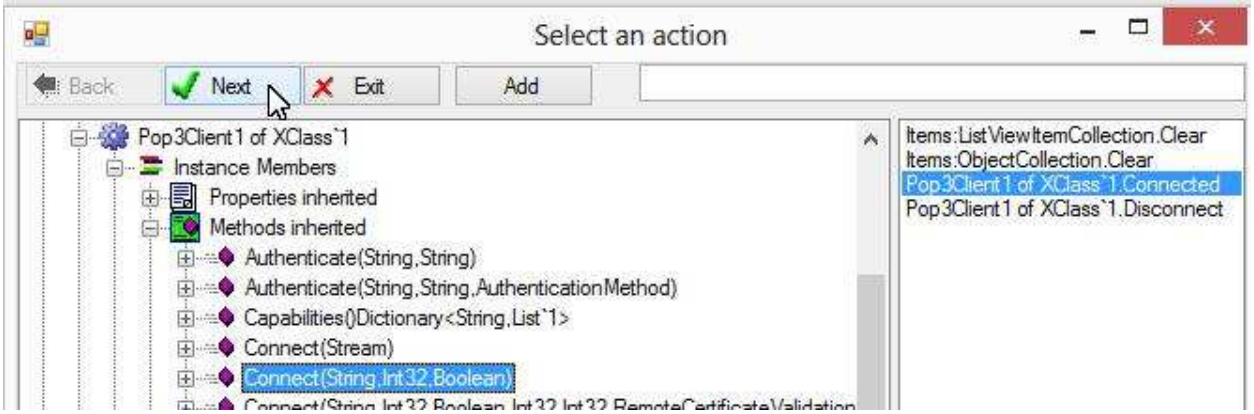




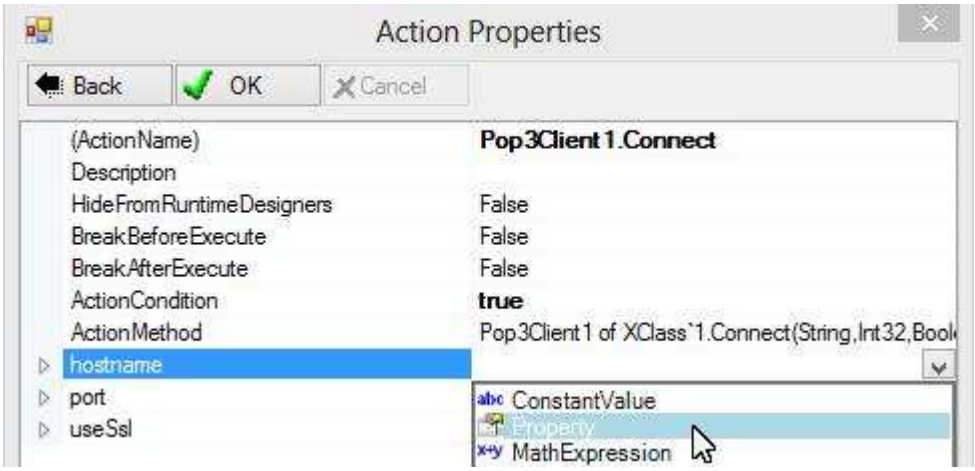
Make email connection

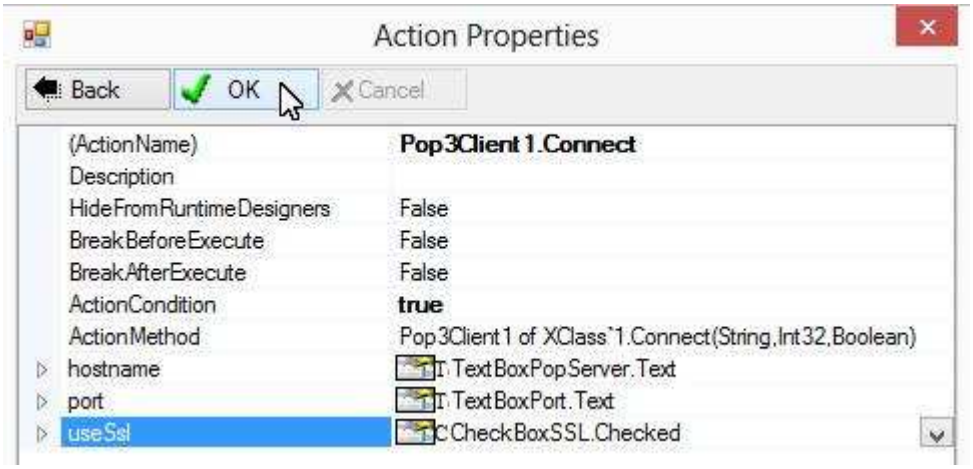
5. Add an action to connect to an email server



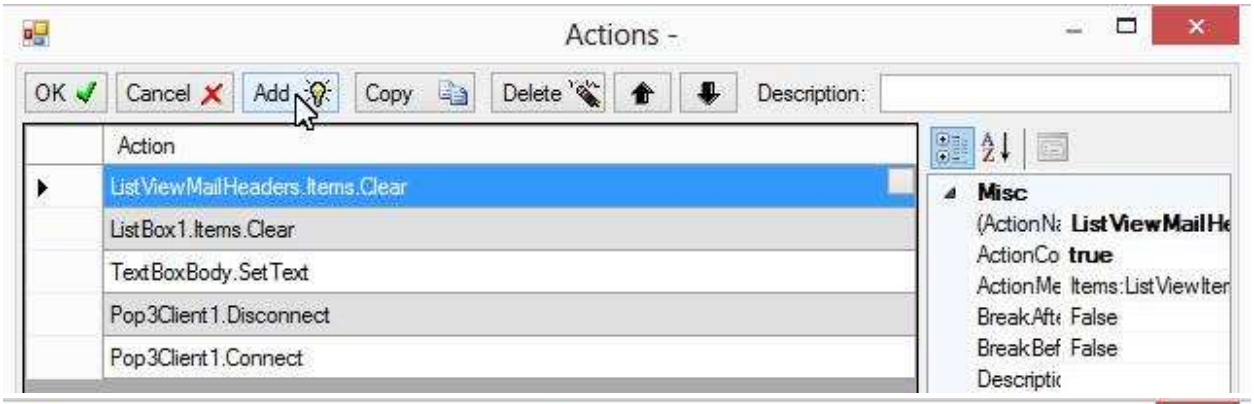


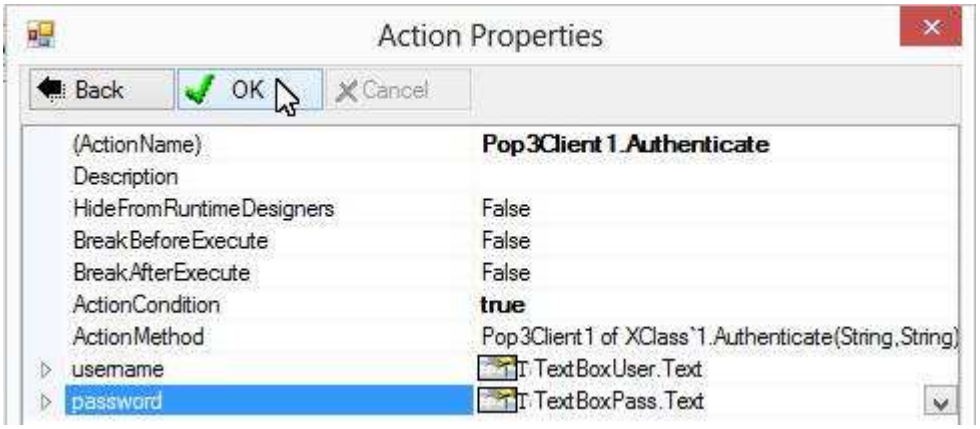
Use Text property of text boxes and SSL checkbox to provide values for action parameters:



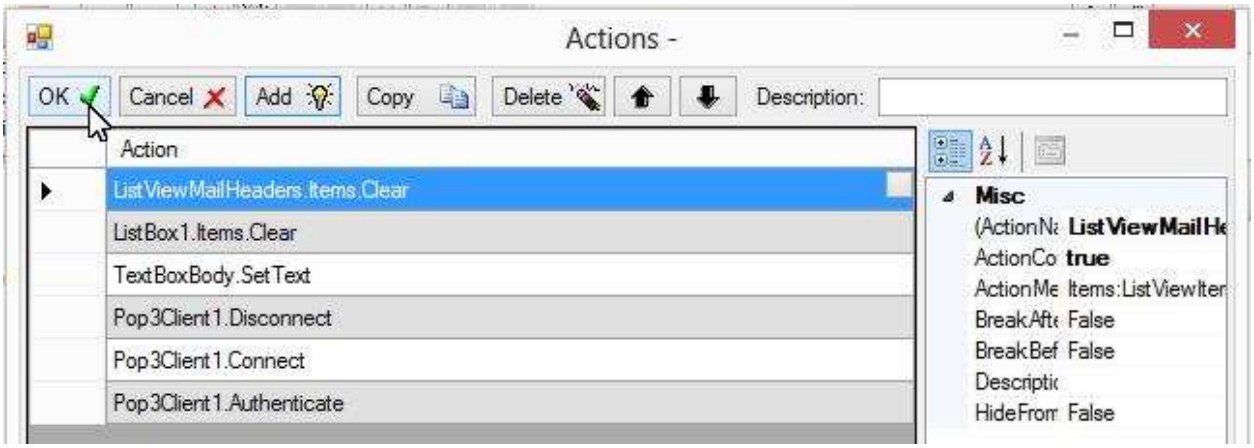


6. Add an action to authenticate the user account:



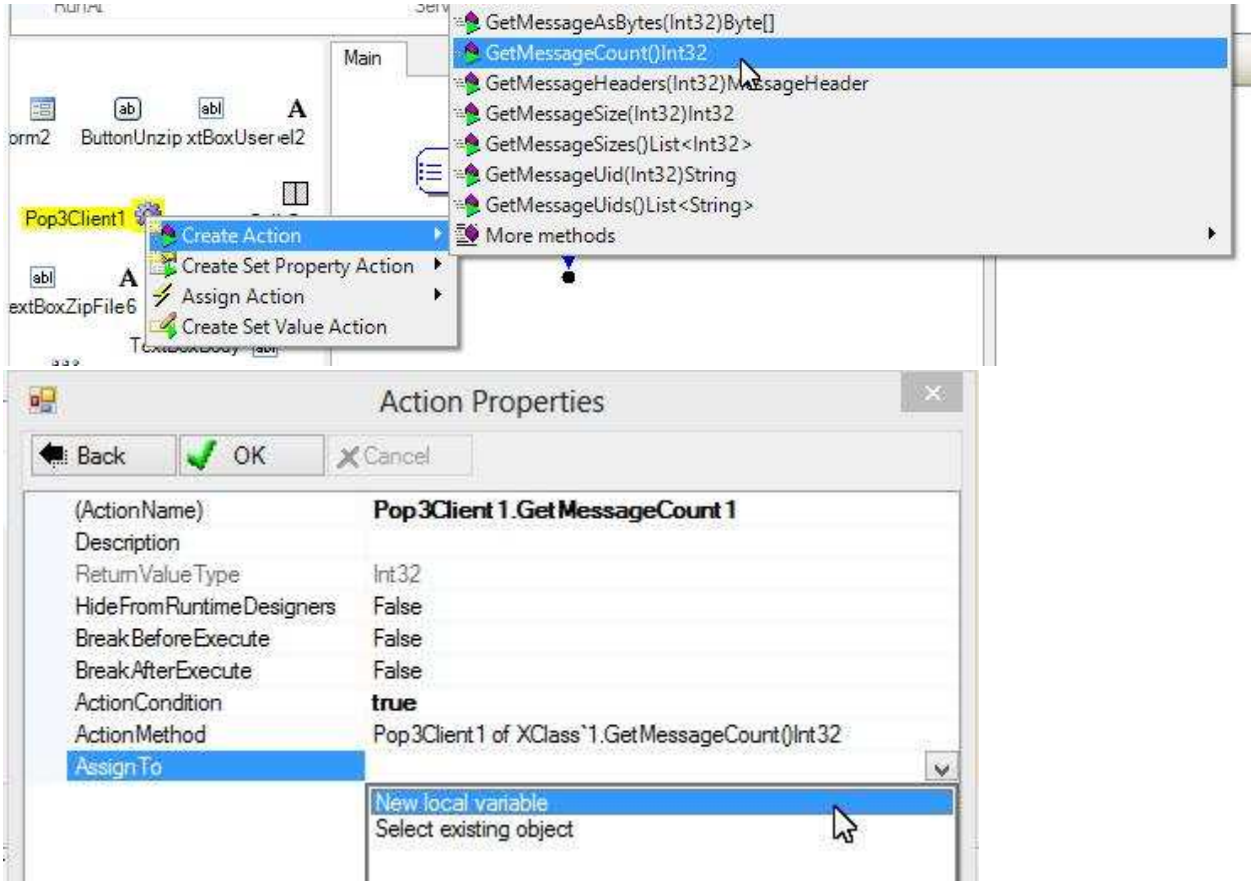


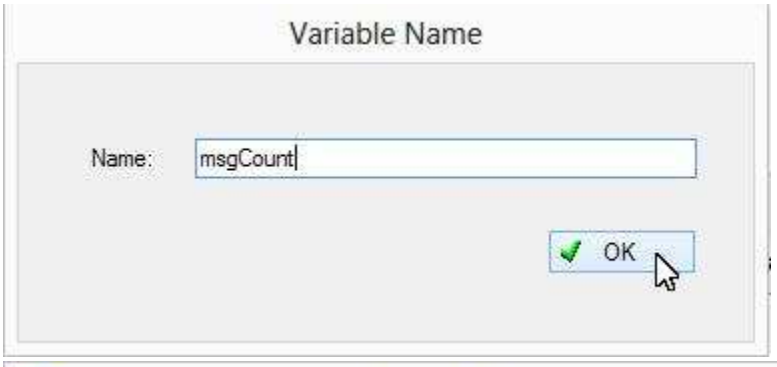
Finish editing the action list:

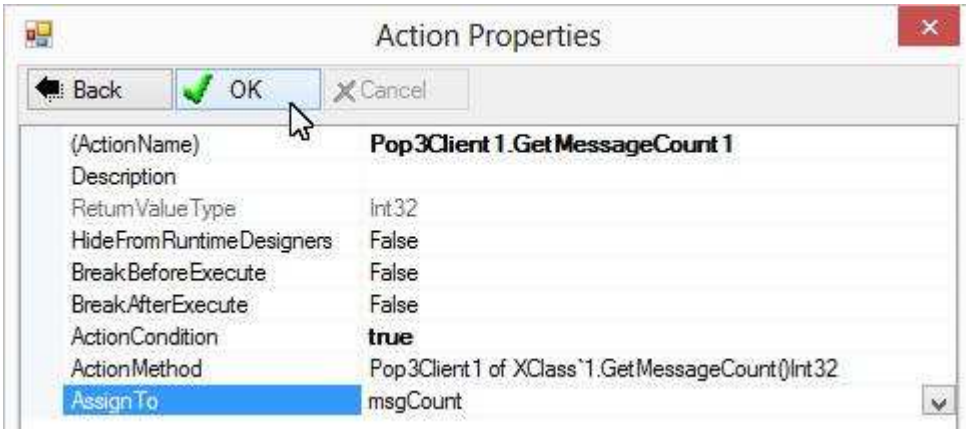


Get Email Count

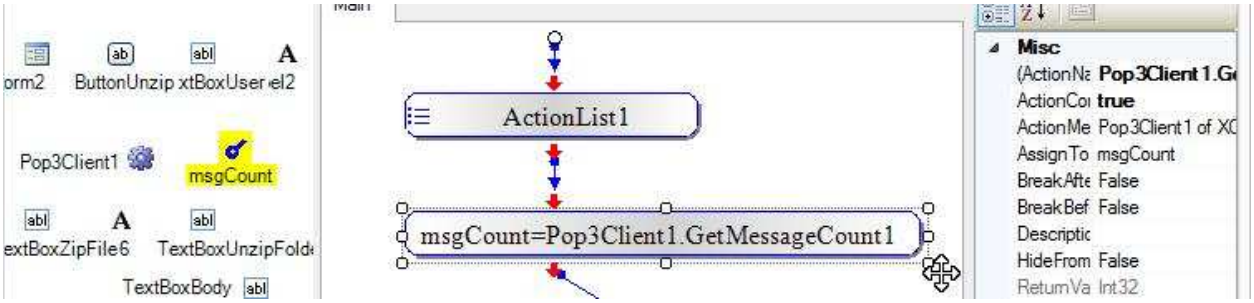
7. Get email count:





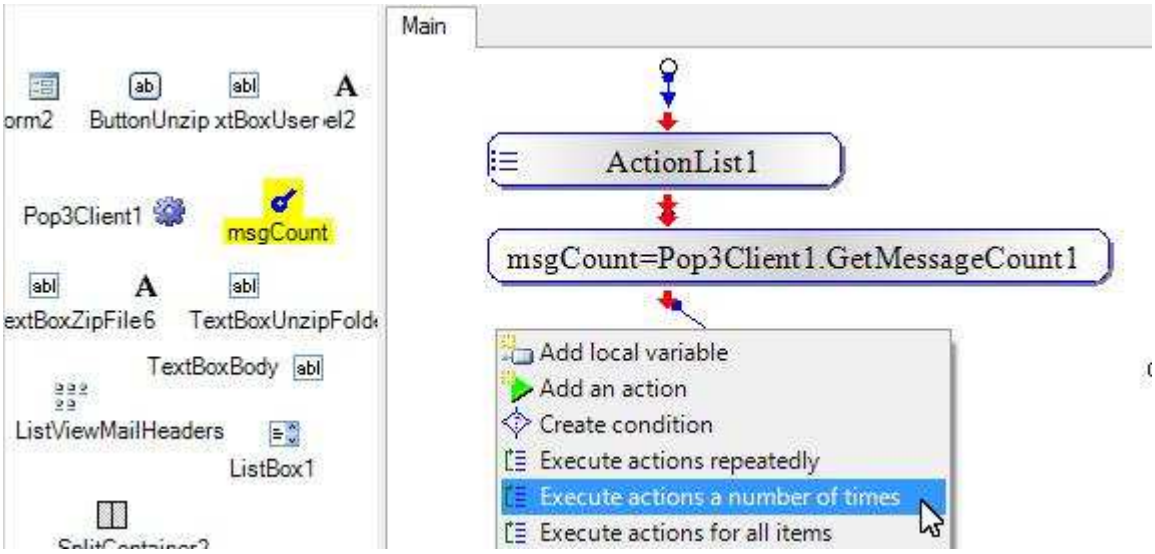


Link the new action to the action list:

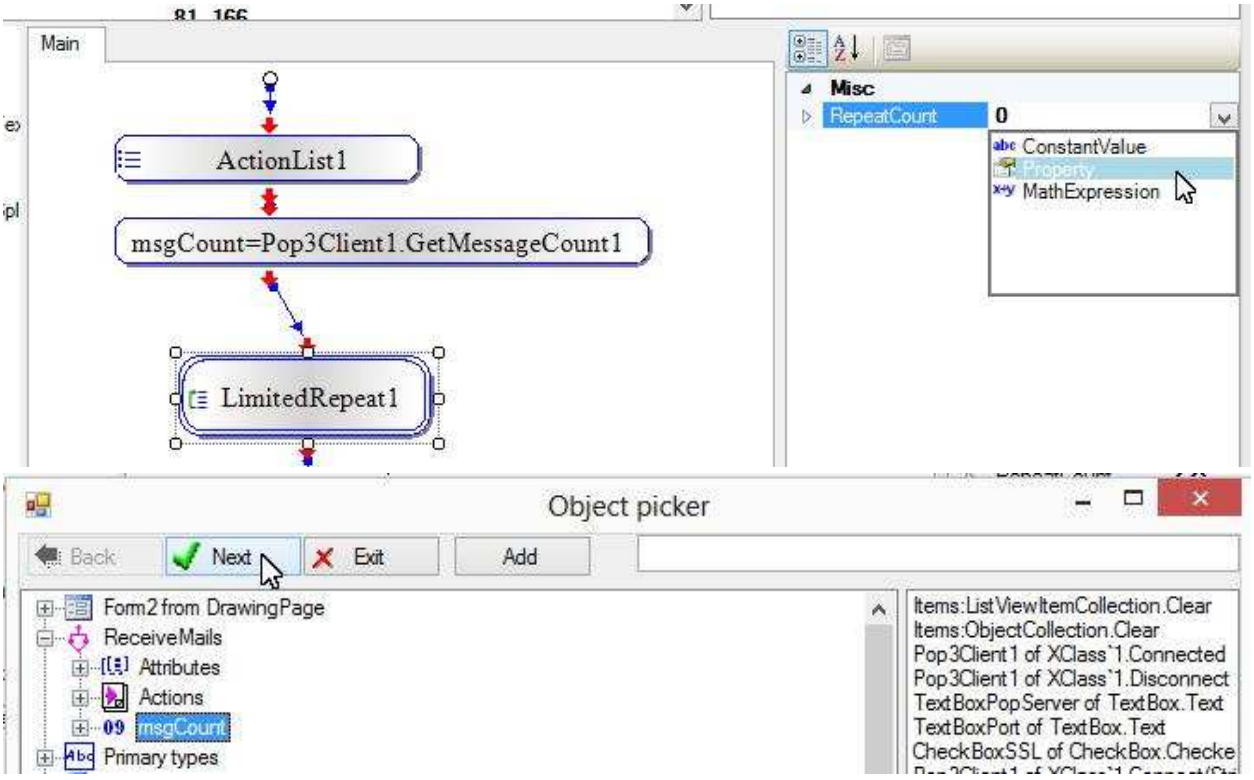


Fetch email one by one

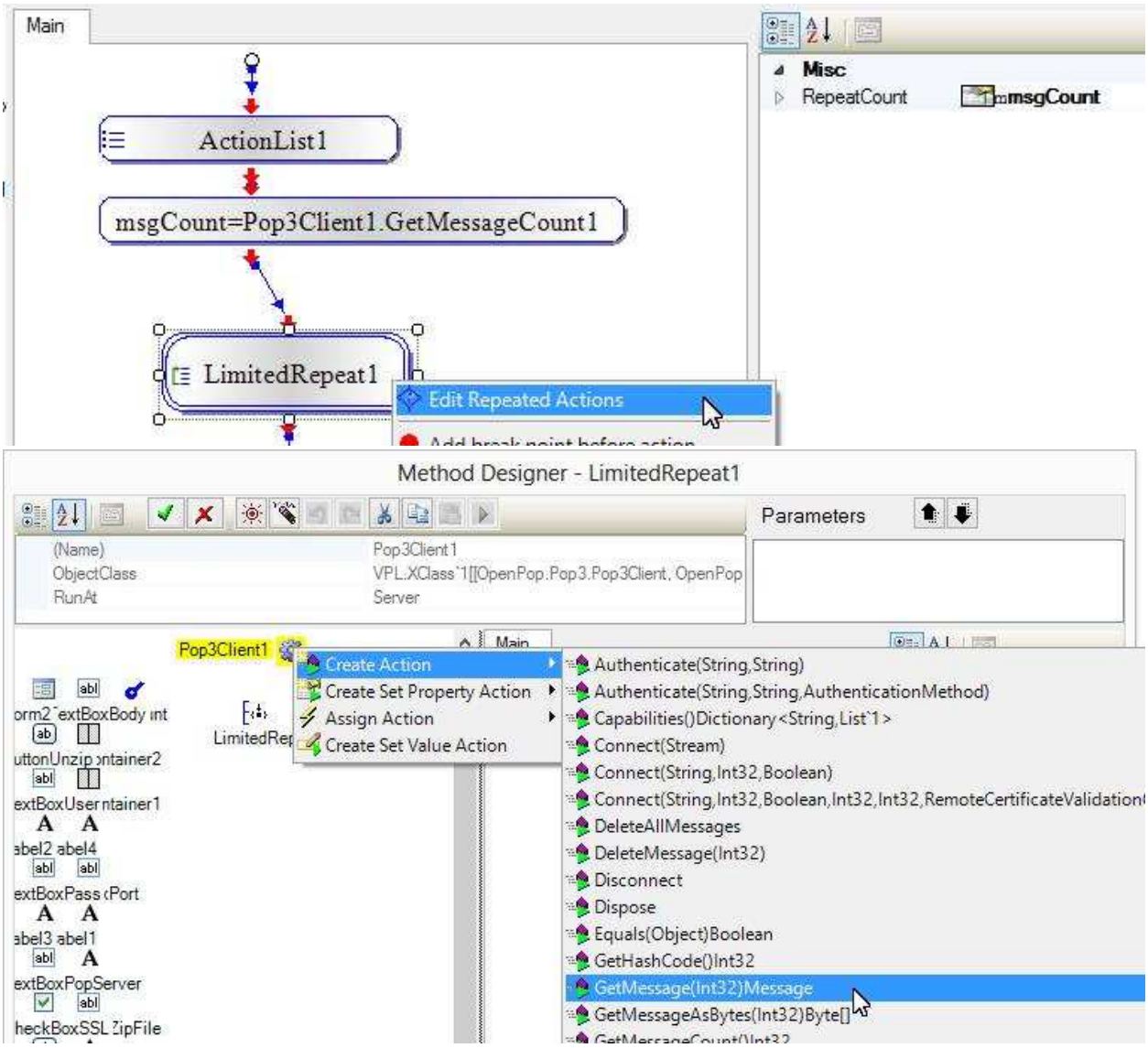
8. Use a loop action to download every email from the email server



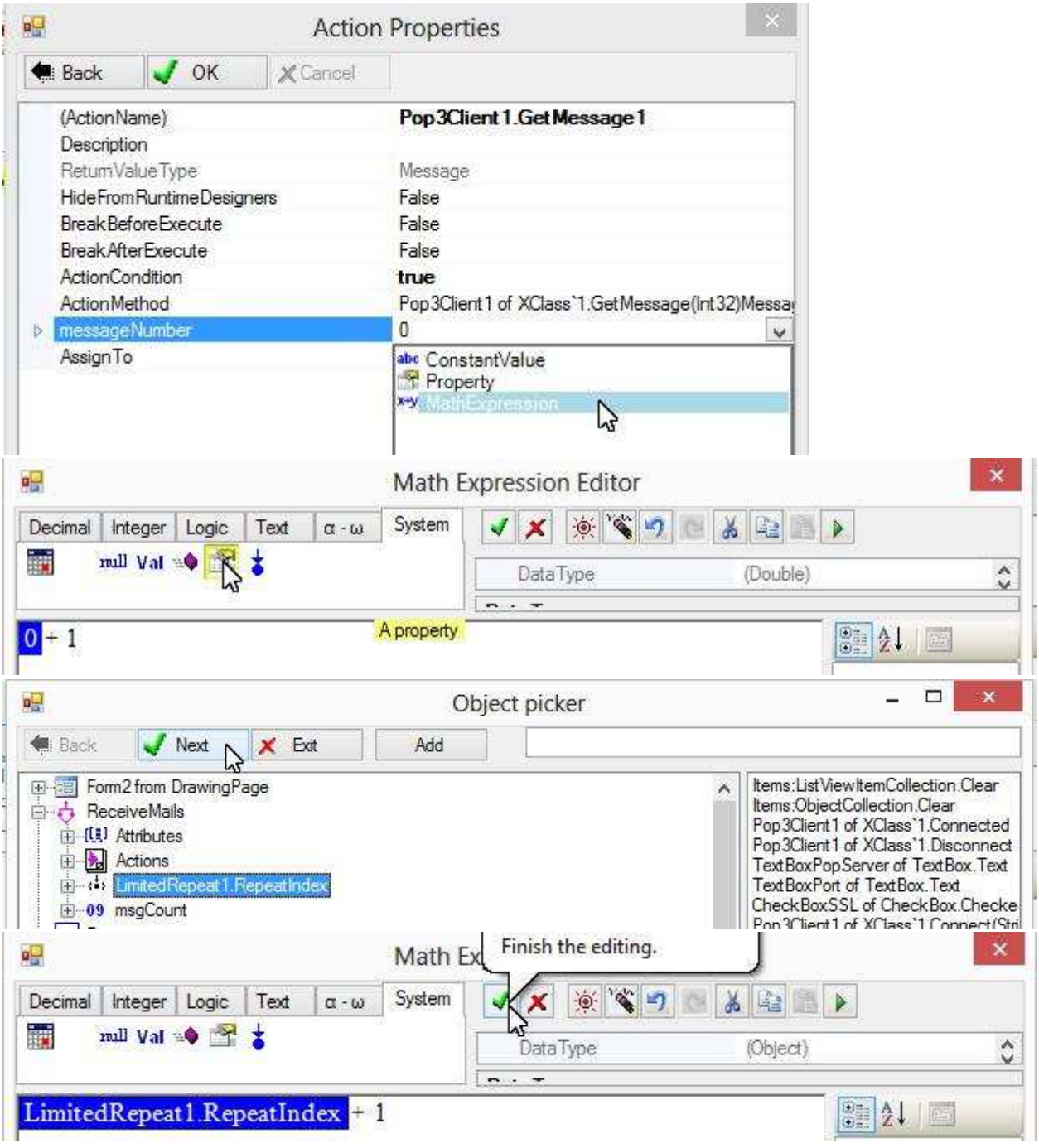
Set loop count to the message count:

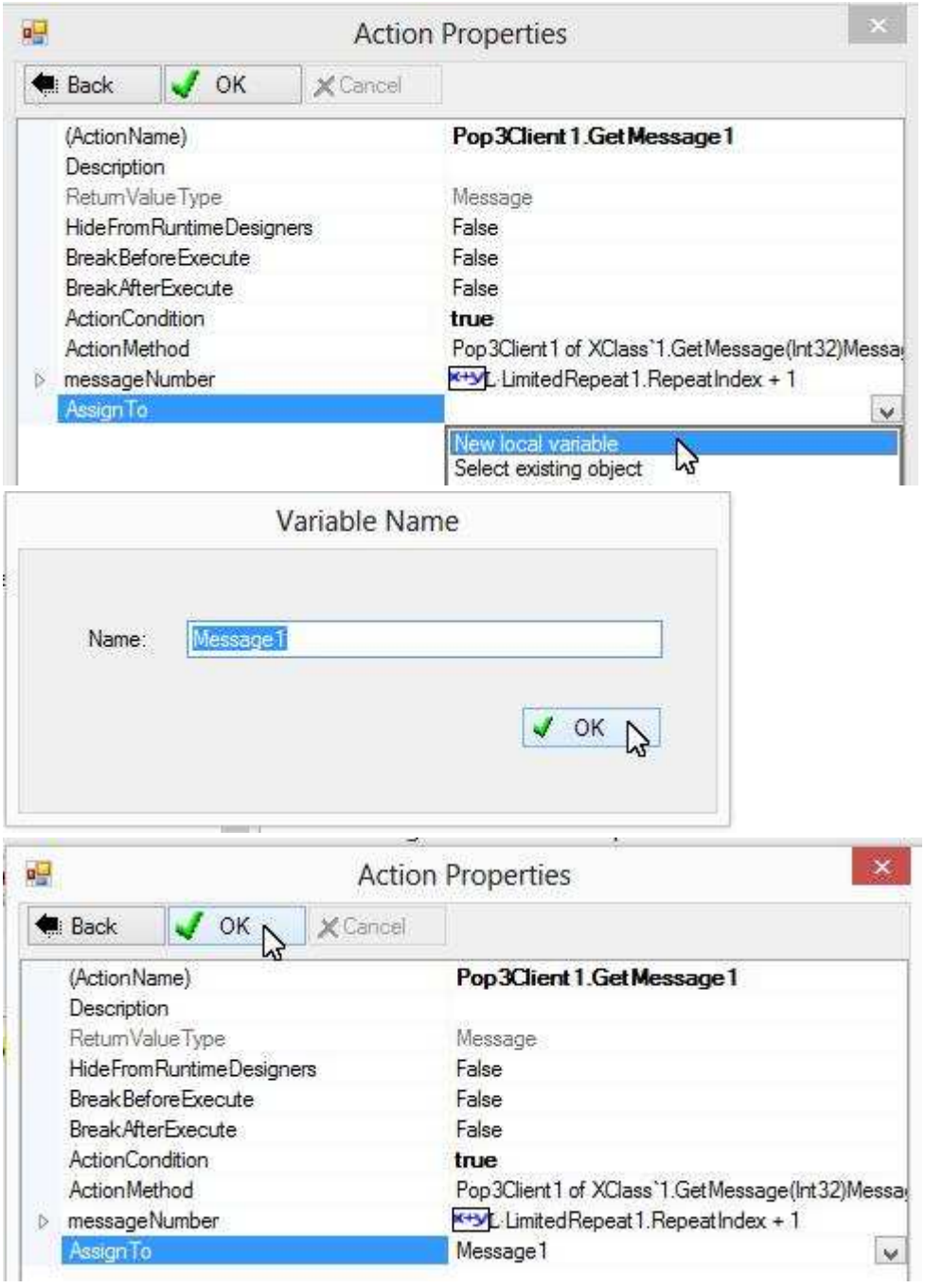


Add email retrieving actions to the loop action:



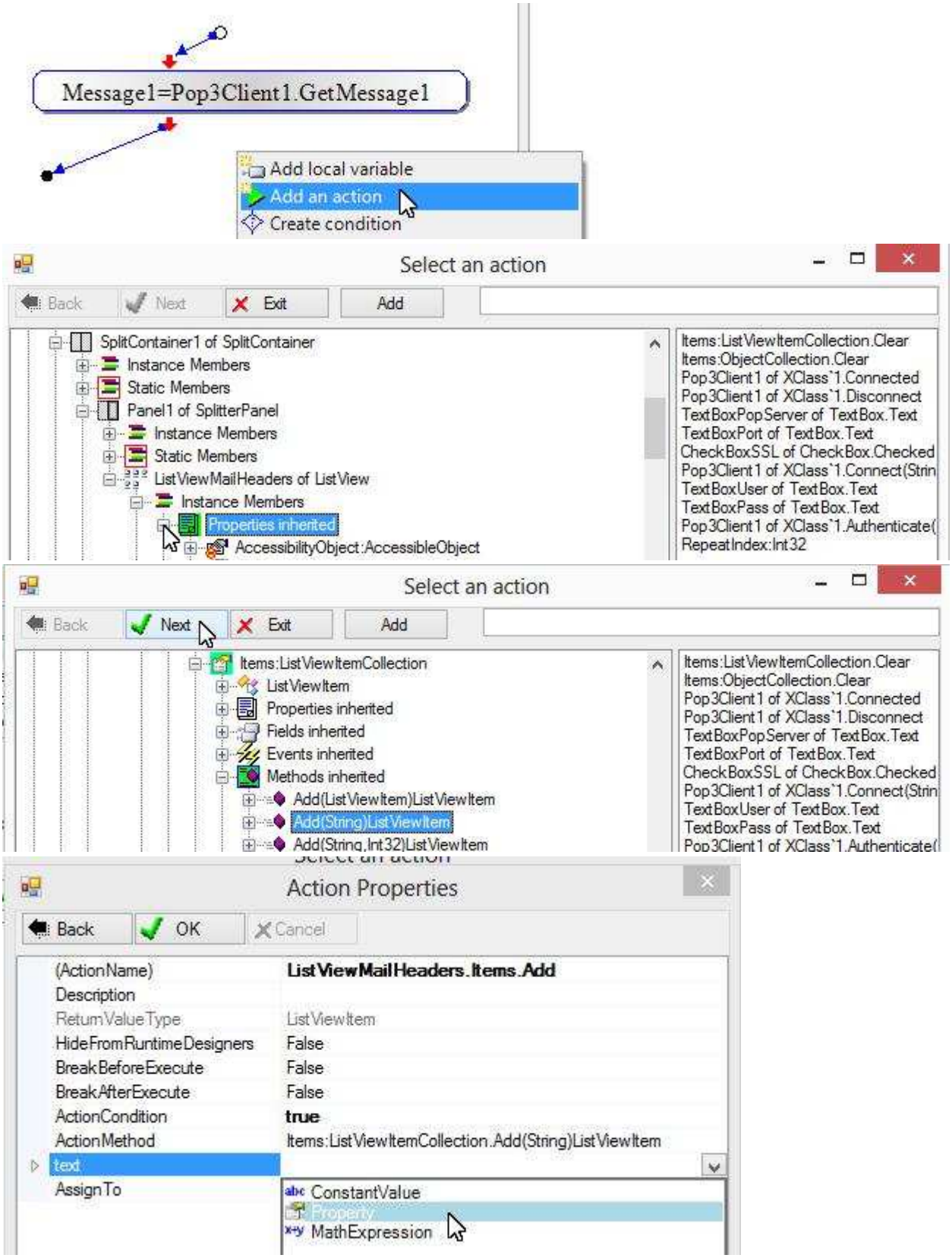
GetMessage requires an email index which starts from 1 to the email count. The repeat index of the loop action is from 0 to email count minus 1. So, we need to pass repeat index + 1 to GetMessage:

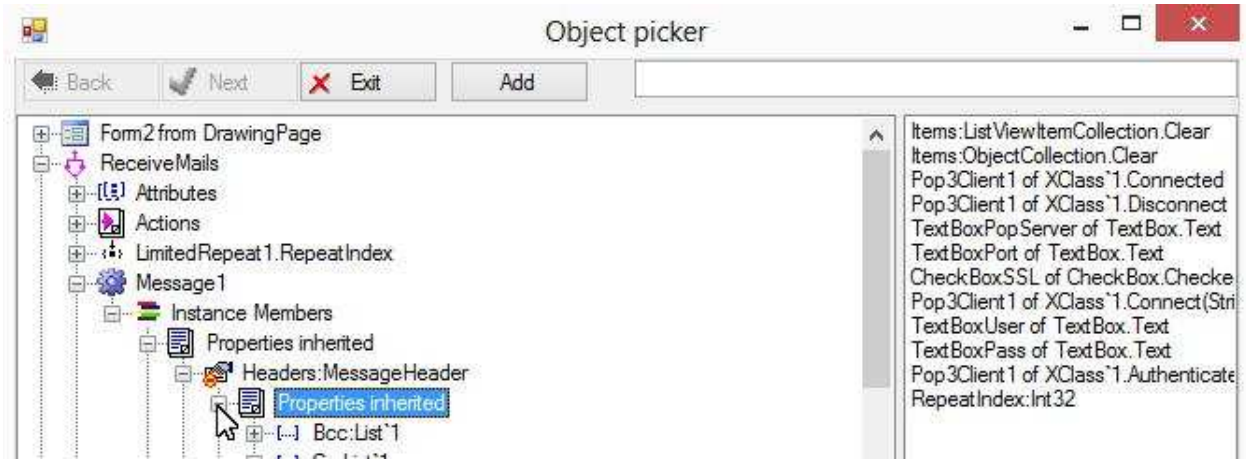

Action Properties
Back
OK
Cancel
(ActionName)
Pop3Client1.GetMessage1
Description
ReturnValueType
Message
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Pop3Client1 of XClass`1.GetMessage(Int32)Messa
messageNumber
0
AssignTo
ConstantValue
Property
MathExpression
Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
null
Val
DataType
(Double)
0 + 1
A property
Object picker
Back
Next
Exit
Add
Form2 from DrawingPage
ReceiveMails
Attributes
Actions
LimitedRepeat1.RepeatIndex
msgCount
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checke
Math Ex
Finish the editing.
Decimal
Integer
Logic
Text
α - ω
System
null
Val
DataType
(Object)
LimitedRepeat1.RepeatIndex + 1



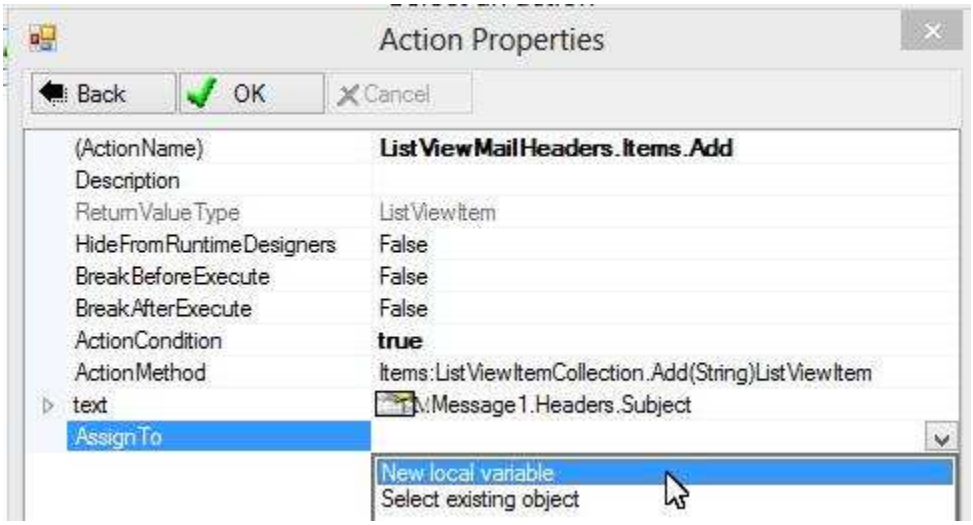
Show email object on List View

9. Use email subject to add a list view item:

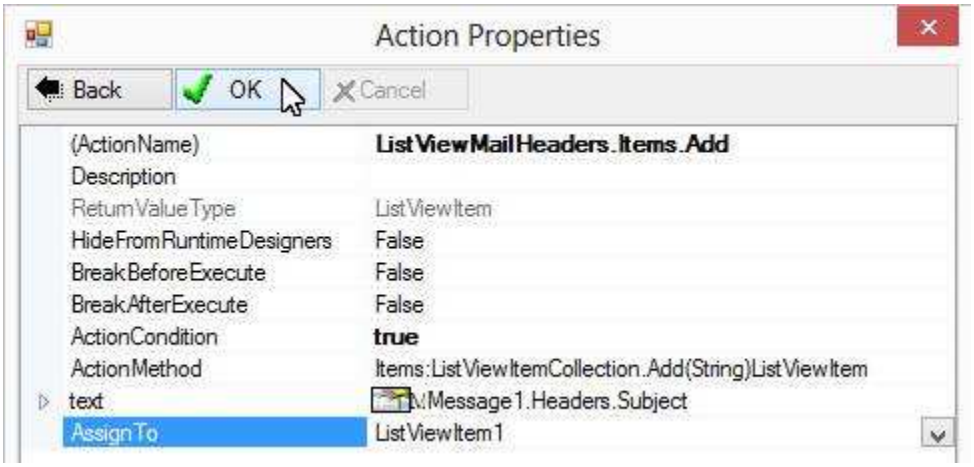

Message1=Pop3Client1.GetMessage1
Add local variable
Add an action
Create condition
Select an action
Back
Next
Exit
Add
SplitContainer1 of SplitContainer
Instance Members
Static Members
Panel1 of SplitterPanel
Instance Members
Static Members
ListViewMailHeaders of ListView
Instance Members
Properties inherited
AccessibilityObject:AccessibleObject
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checked
Pop3Client1 of XClass`1.Connect(Strin
TextBoxUser of TextBox.Text
TextBoxPass of TextBox.Text
Pop3Client1 of XClass`1.Authenticate(
RepeatIndex:Int32
Select an action
Back
Next
Exit
Add
Items:ListViewItemCollection
ListViewItem
Properties inherited
Fields inherited
Events inherited
Methods inherited
Add(ListViewItem):ListViewItem
Add(String):ListViewItem
Add(String,Int32):ListViewItem
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checked
Pop3Client1 of XClass`1.Connect(Strin
TextBoxUser of TextBox.Text
TextBoxPass of TextBox.Text
Pop3Client1 of XClass`1.Authenticate(
Action Properties
Back
OK
Cancel
(ActionName)
ListViewMailHeaders.Items.Add
Description
ReturnValueType
ListViewItem
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Items:ListViewItemCollection.Add(String):ListViewItem
text
AssignTo
ConstantValue
Property
MathExpression


Object picker
Back
Next
Exit
Add
Form2 from DrawingPage
ReceiveMails
Attributes
Actions
LimitedRepeat1.RepeatIndex
Message1
Instance Members
Properties inherited
Headers:MessageHeader
Properties inherited
Bcc:List`1
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checke
Pop3Client1 of XClass`1.Connect(Stri
TextBoxUser of TextBox.Text
TextBoxPass of TextBox.Text
Pop3Client1 of XClass`1.Authenticate
RepeatIndex:Int32

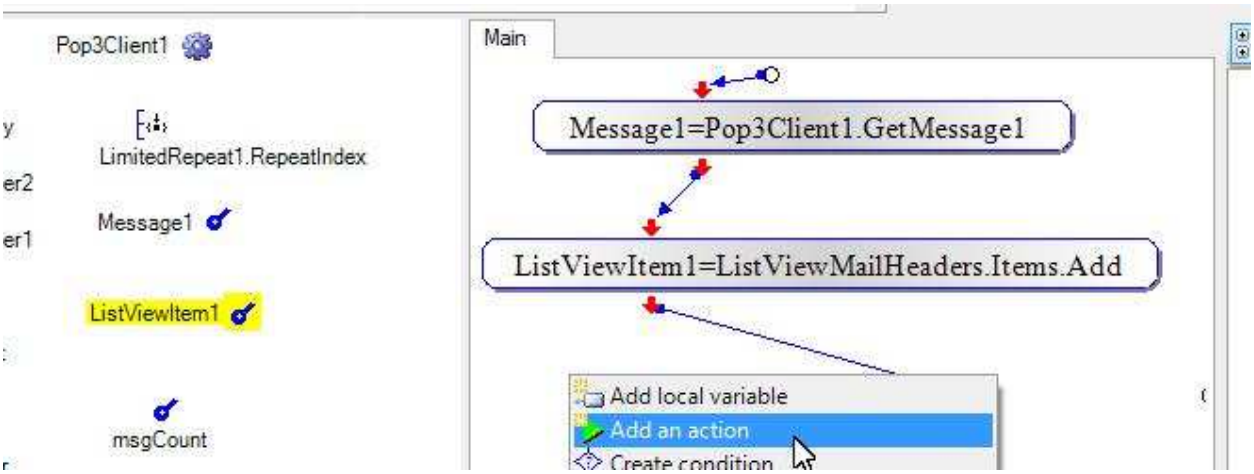

Object picker
Back
Next
Exit
Add
Sender:RfcMailAddress
Subject:String
To:List`1
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected

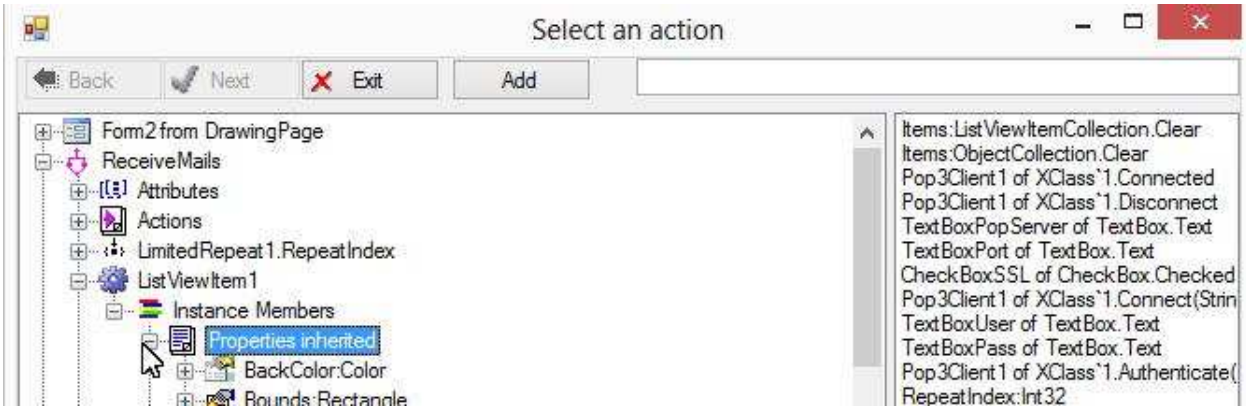

Action Properties
Back
OK
Cancel
(ActionName)
ListViewMailHeaders.Items.Add
Description
ReturnValueType
ListViewItem
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Items:ListViewItemCollection.Add(String)ListViewItem
text
Message1.Headers.Subject
AssignTo
New local variable
Select existing object

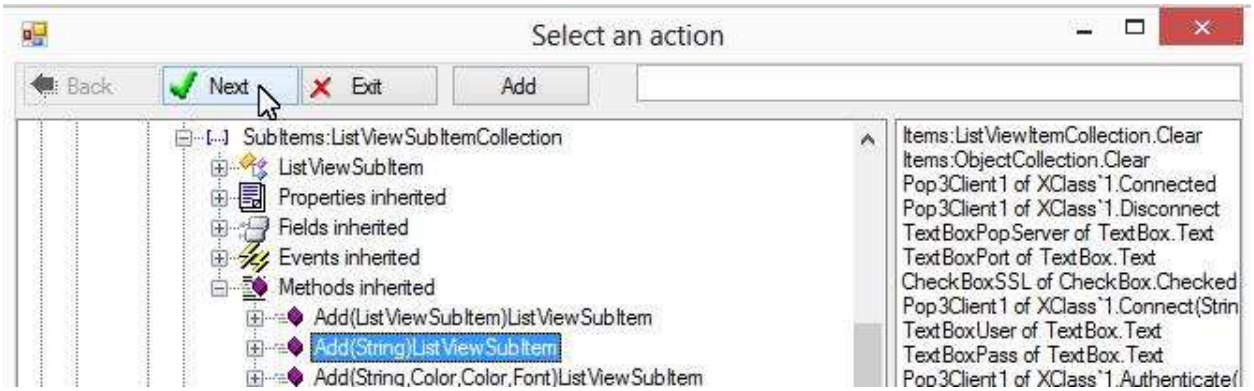

Variable Name
Name:
ListViewItem1
OK

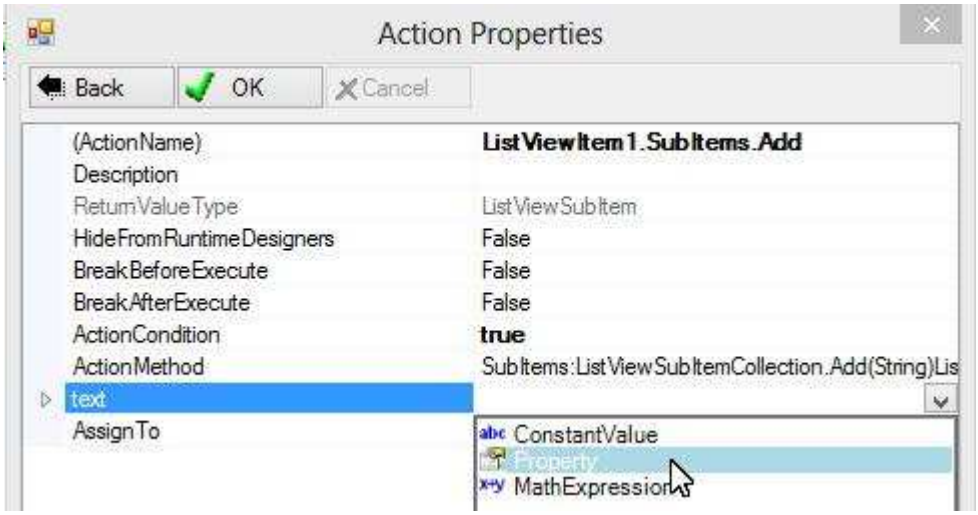


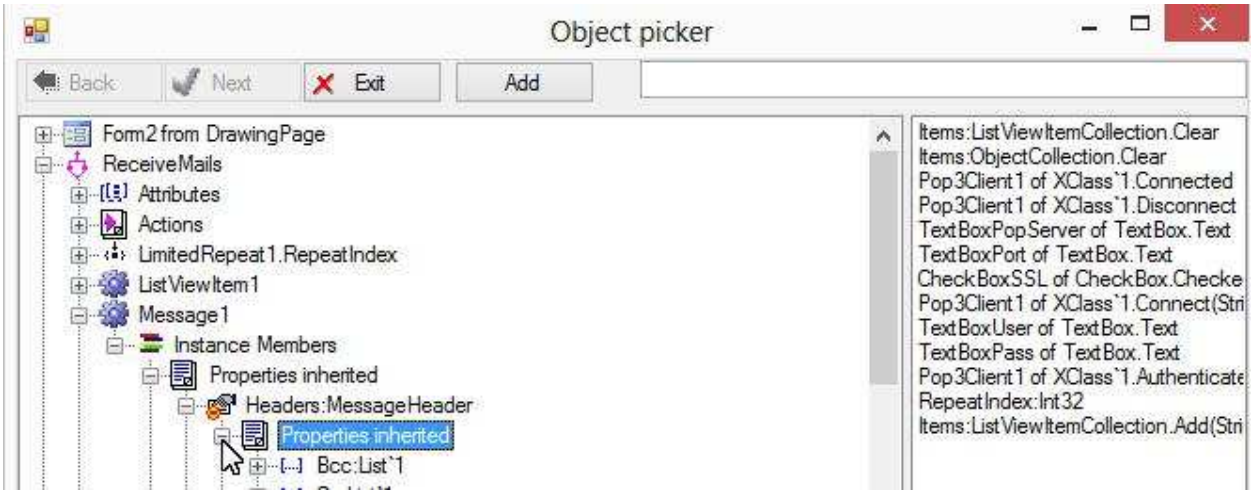
10. Add sub items to the new list view item



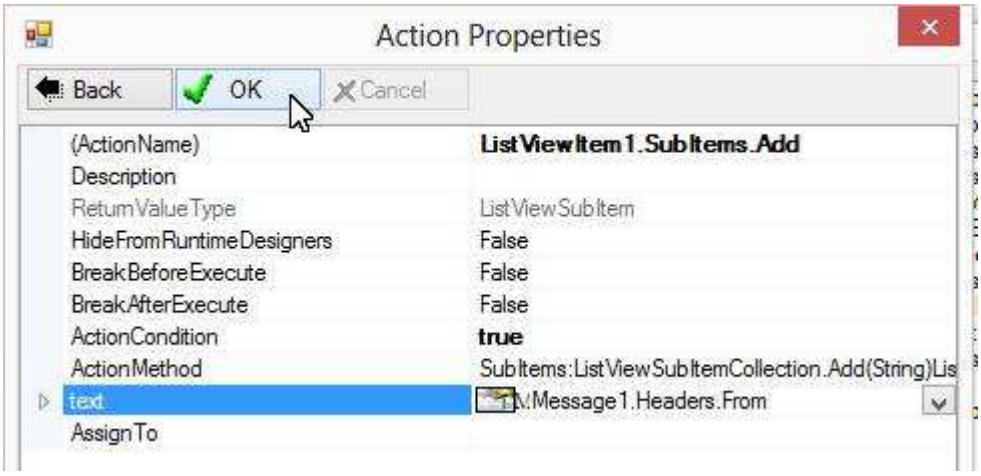


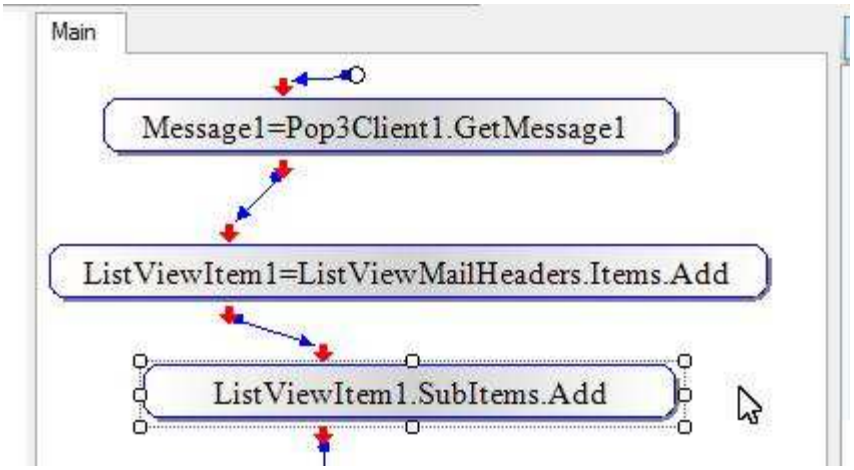

Select an action
Back
Next
Exit
Add
SubItems:ListViewSubItemCollection
ListViewSubItem
Properties inherited
Fields inherited
Events inherited
Methods inherited
Add(ListViewSubItem)ListViewSubItem
Add(String)ListViewSubItem
Add(String,Color,Color,Font)ListViewSubItem
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checked
Pop3Client1 of XClass`1.Connect(Strin
TextBoxUser of TextBox.Text
TextBoxPass of TextBox.Text
Pop3Client1 of XClass`1.Authenticate(


Action Properties
Back
OK
Cancel
(ActionName)
ListViewItem1.SubItems.Add
Description
ReturnValueType
ListViewSubItem
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
SubItems:ListViewSubItemCollection.Add(String)Lis
text
AssignTo
ConstantValue
Property
MathExpression

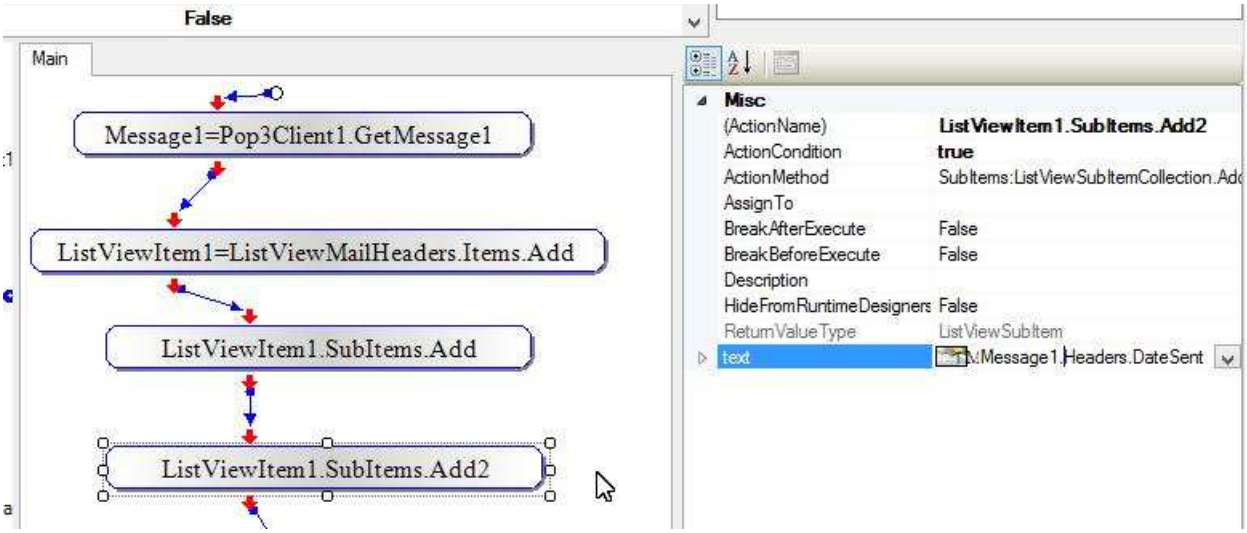

Object picker
Back
Next
Exit
Add
Form2 from DrawingPage
ReceiveMails
Attributes
Actions
LimitedRepeat1.RepeatIndex
ListViewItem1
Message1
Instance Members
Properties inherited
Headers:MessageHeader
Properties inherited
Bcc:List`1
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checke
Pop3Client1 of XClass`1.Connect(Stri
TextBoxUser of TextBox.Text
TextBoxPass of TextBox.Text
Pop3Client1 of XClass`1.Authenticate
RepeatIndex:Int32
Items:ListViewItemCollection.Add(Stri


Object picker
Back
Next
Exit
Add
DispositionNotificationTo:List`1
From:RfcMailAddress
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected



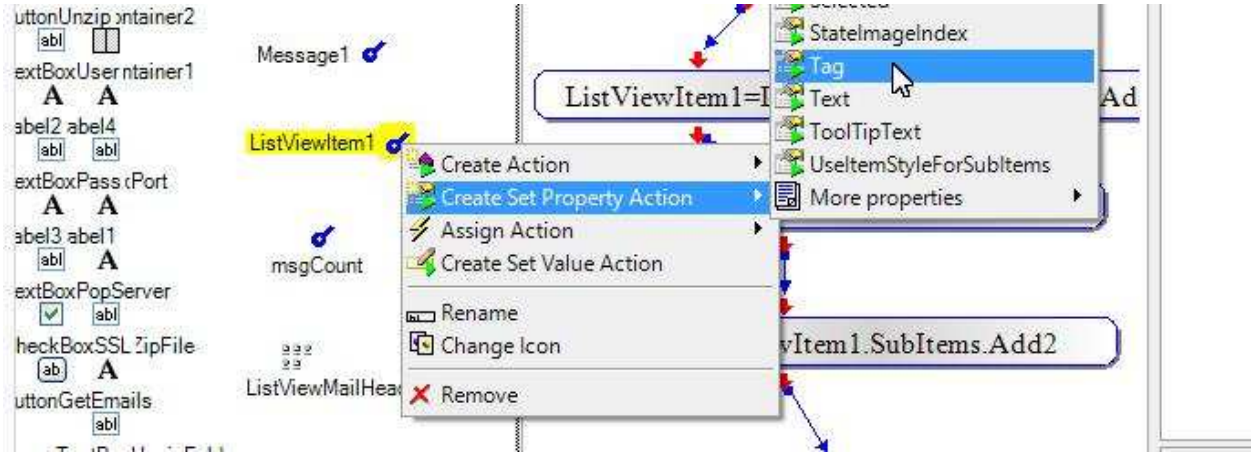


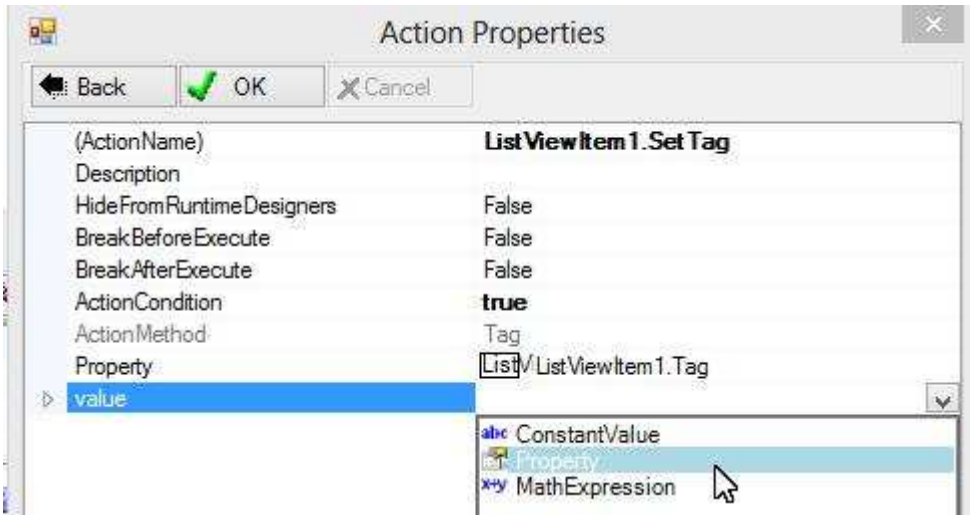
11. In the same way, add "Sent time" to the list view item as a sub item:



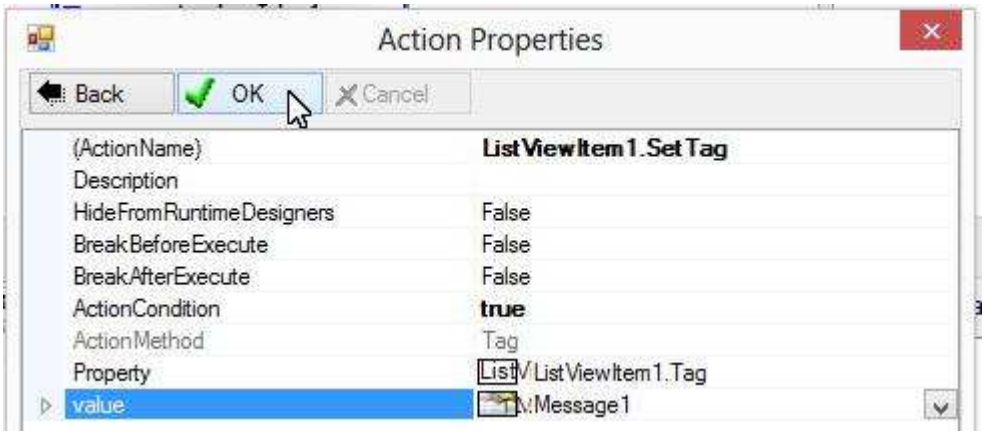
Remember email object associated with list view item

12. A list view item has a Tag property. We may set it to the email object so that when a list view item is selected we know which email object to work on

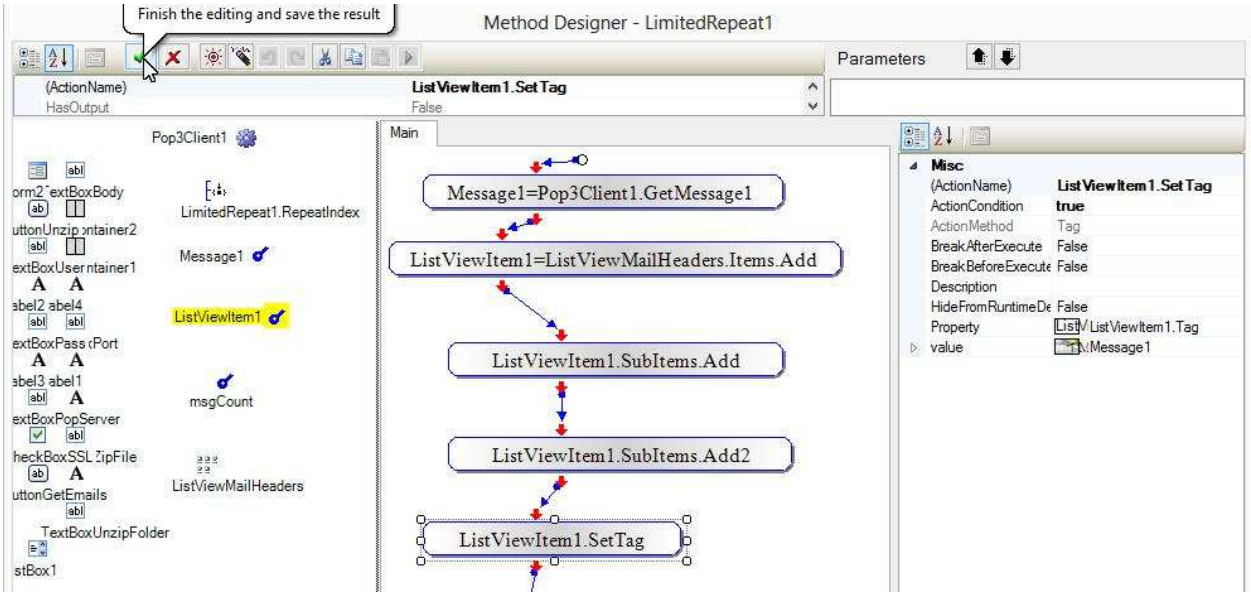

Message1
ListViewItem1
msgCount
ListViewMailHea
Create Action
Create Set Property Action
Assign Action
Create Set Value Action
Rename
Change Icon
Remove
StateImageIndex
Tag
Text
ToolTipText
UseItemStyleForSubItems
More properties
ListViewItem1=I
vItem1.SubItems.Add2


Action Properties
Back
OK
Cancel
(ActionName) ListViewItem1.SetTag
Description
HideFromRuntimeDesigners False
BreakBeforeExecute False
BreakAfterExecute False
ActionCondition true
ActionMethod Tag
Property ListV ListViewItem1.Tag
value
ConstantValue
Property
MathExpression

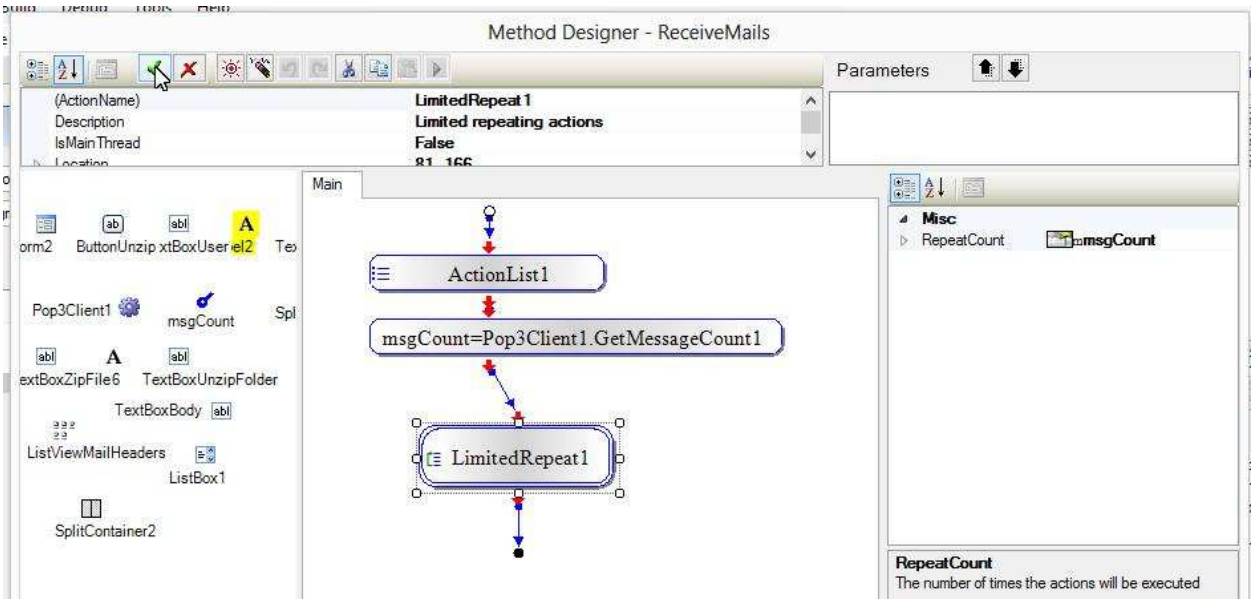

Object picker
Back
Next
Exit
Add
Form2 from DrawingPage
ReceiveMails
Attributes
Actions
LimitedRepeat1.RepeatIndex
ListViewItem1
Message1
msgCount
Items:ListViewItemCollection.Clear
Items:ObjectCollection.Clear
Pop3Client1 of XClass`1.Connected
Pop3Client1 of XClass`1.Disconnect
TextBoxPopServer of TextBox.Text
TextBoxPort of TextBox.Text
CheckBoxSSL of CheckBox.Checke
Pop3Client1 of XClass`1.Connect(Stri
TextBoxUser of TextBox.Text
TextBoxPass of TextBox.Text



These are all the actions we need to fetch an email:

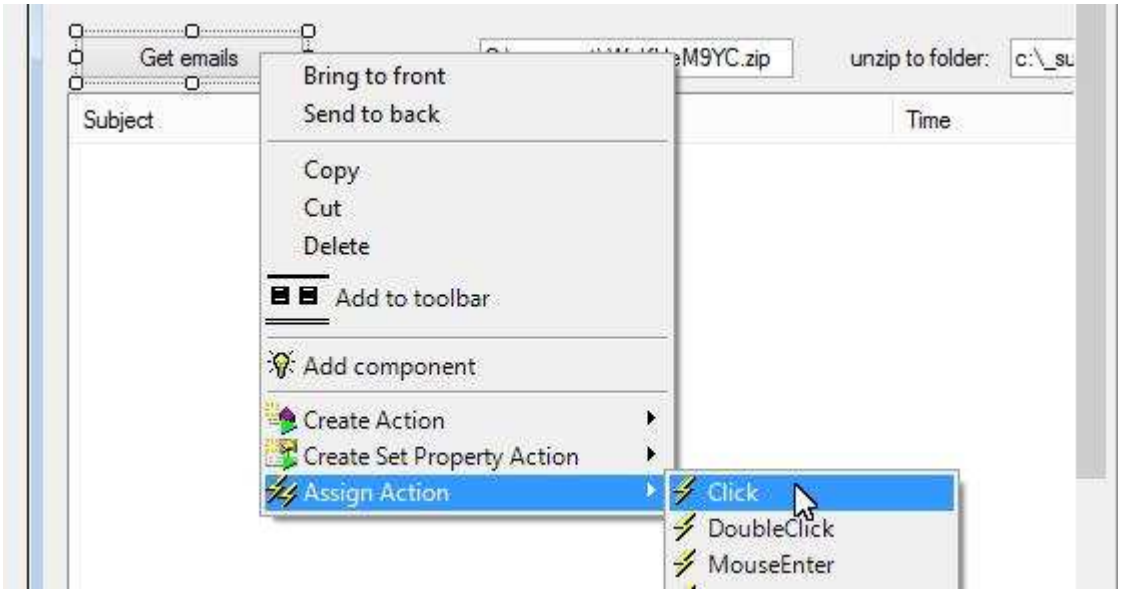


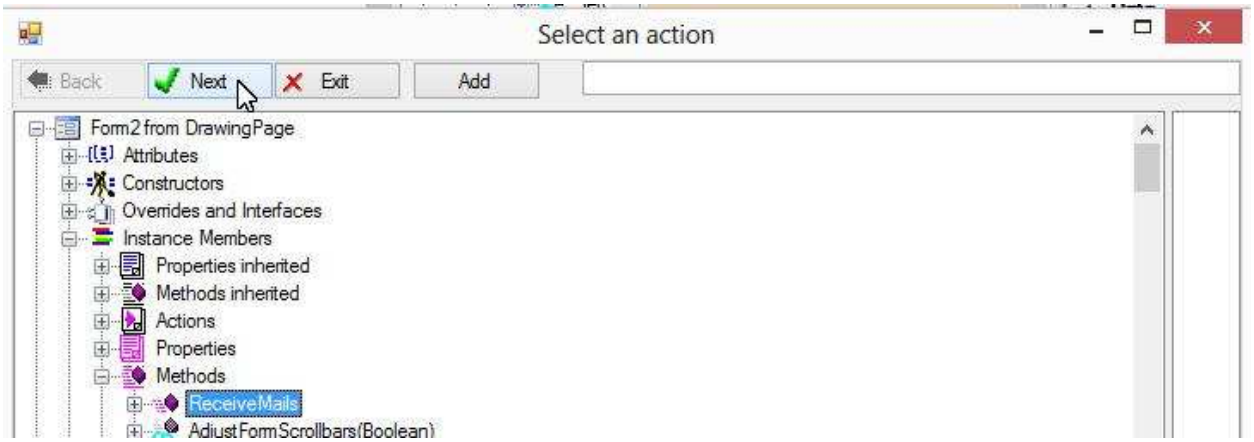
That is all we need for the new method:

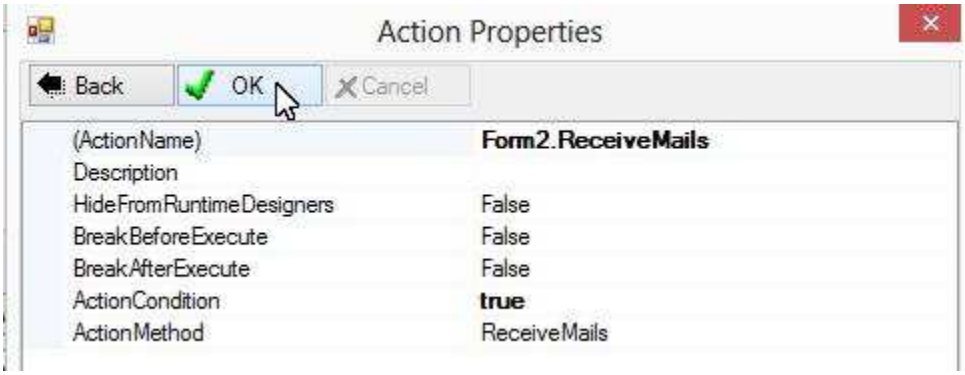


Invoke the Method

We use a button to invoke the above method.



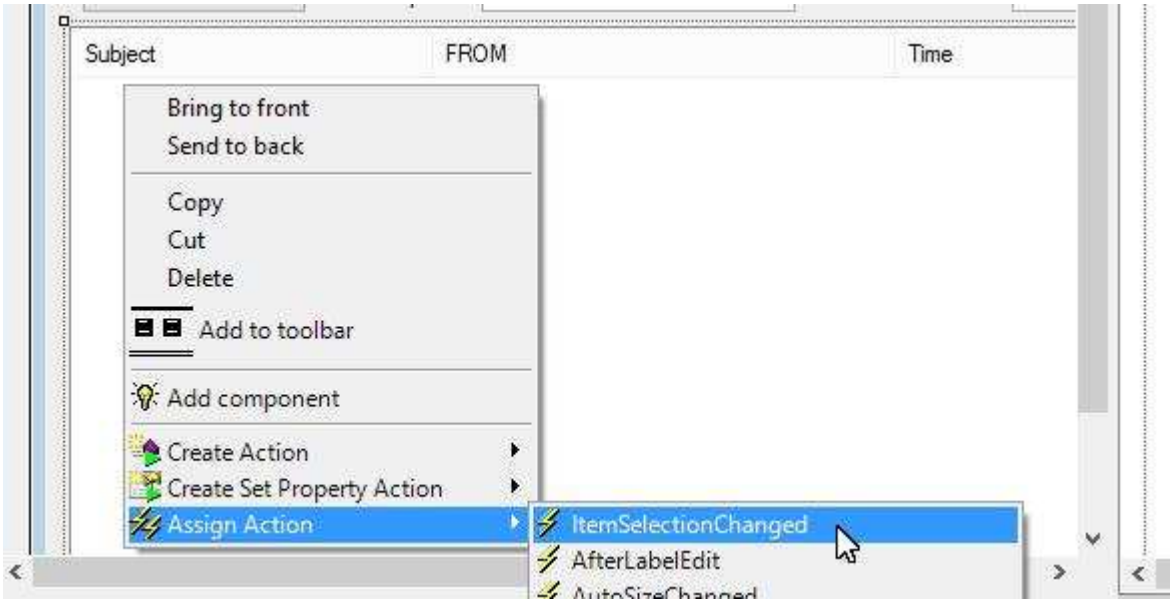


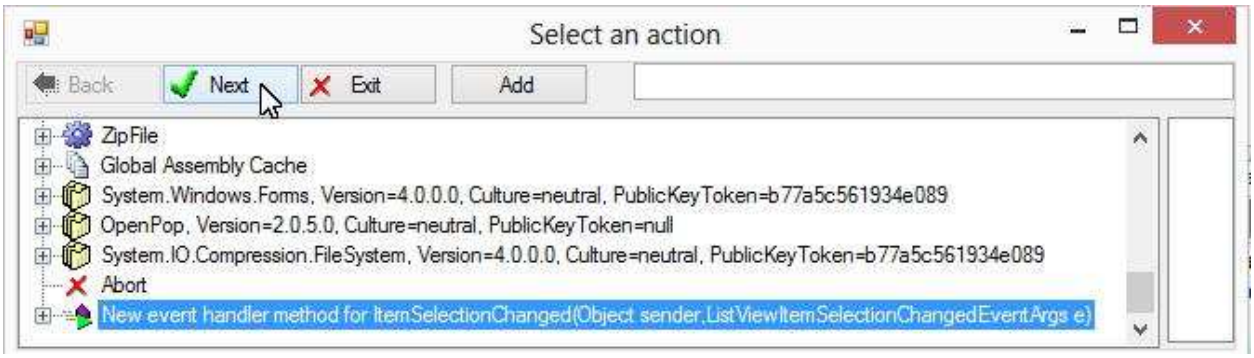


The action is created and assigned to the button:

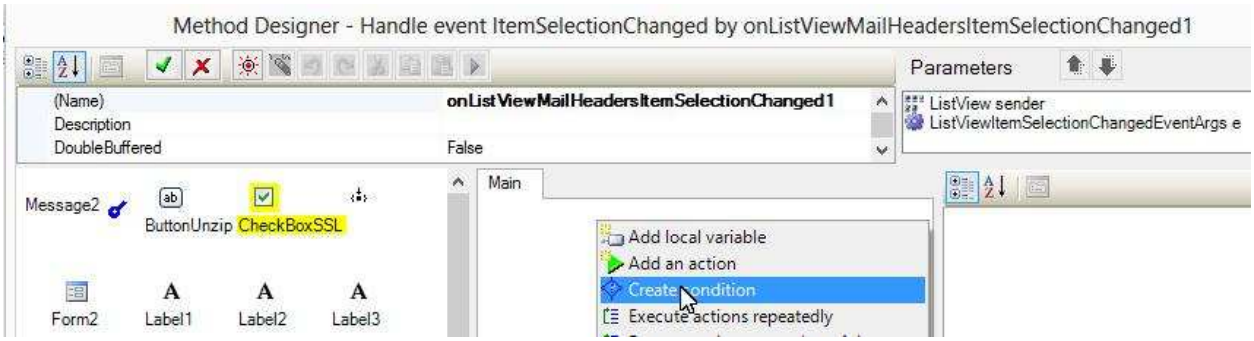
Show Email Contents

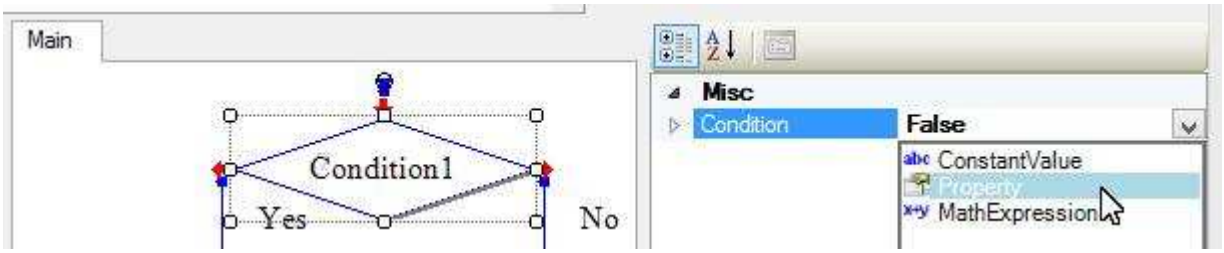
On selecting a list view item, we want to show email body and save attached files. Create an event handler method to do it:

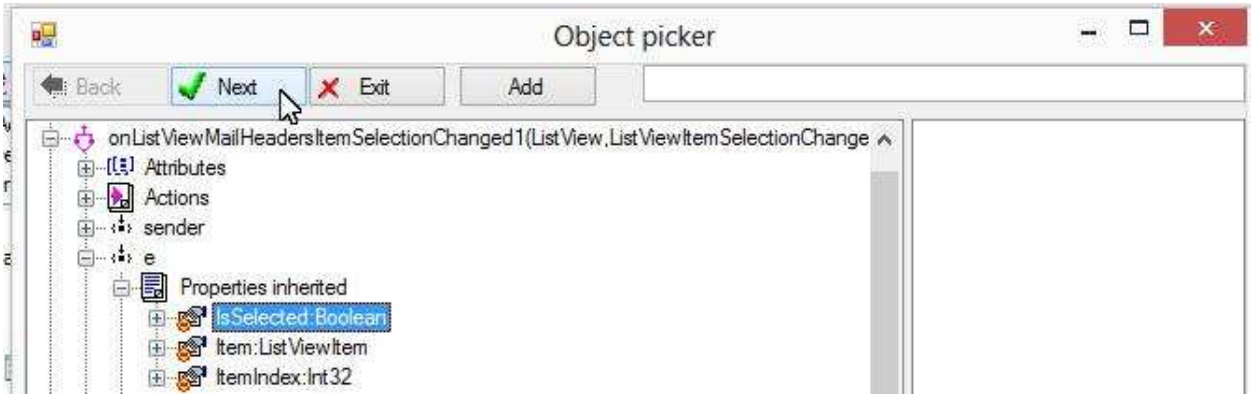




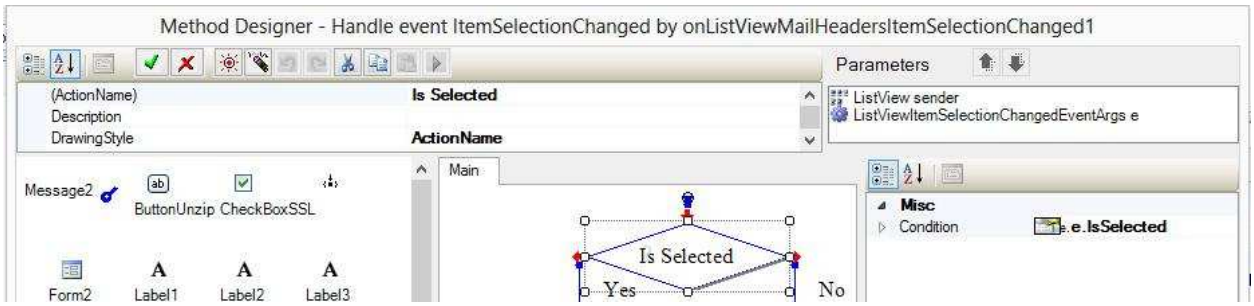
A method editor appears for adding event handling actions. First, add a Condition to check that if the list view item is in “selected” state:







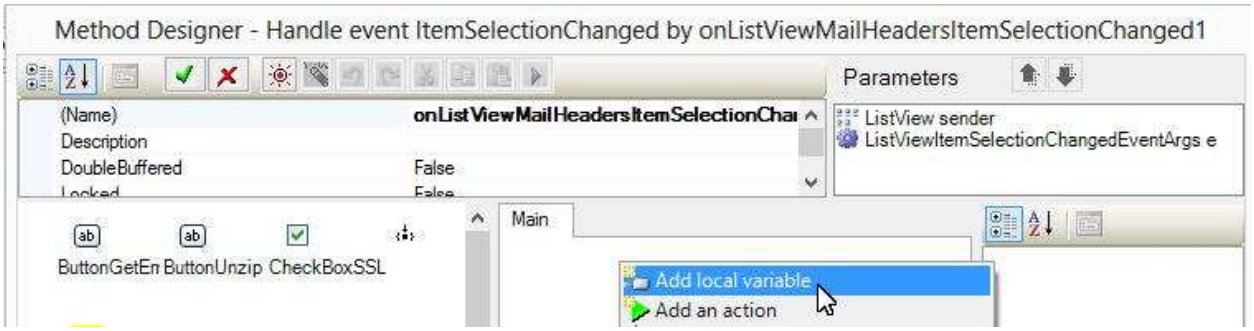
Change its name to “Is selected”:

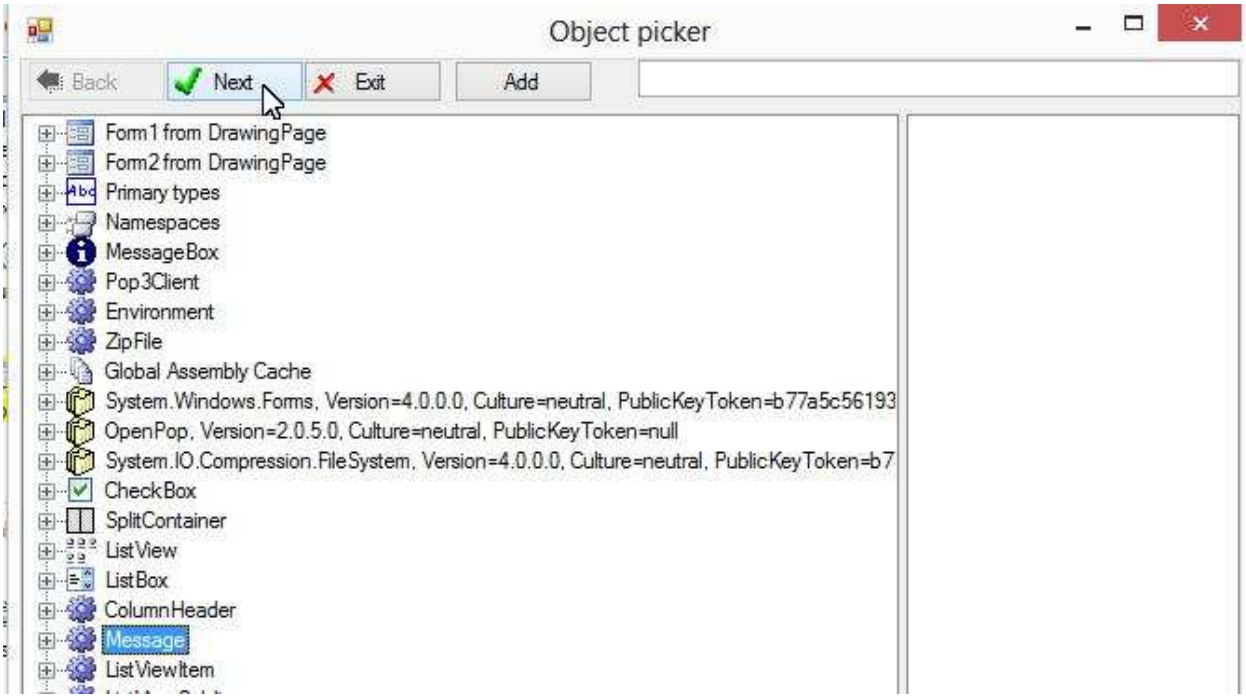


Retrieve email object

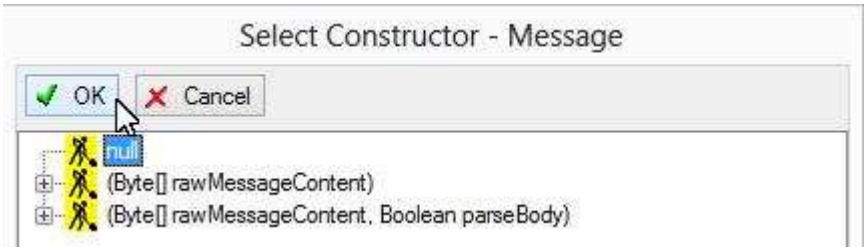
We know that the Tag property of a list view item is an email object. To work on the email object, create a variable to get back the email object from the Tag property.

1. Create an email message variable

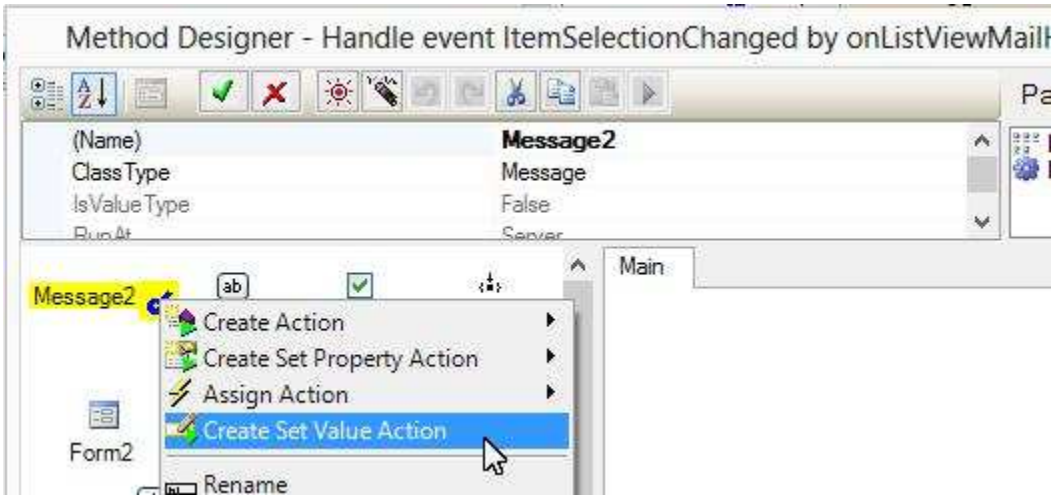




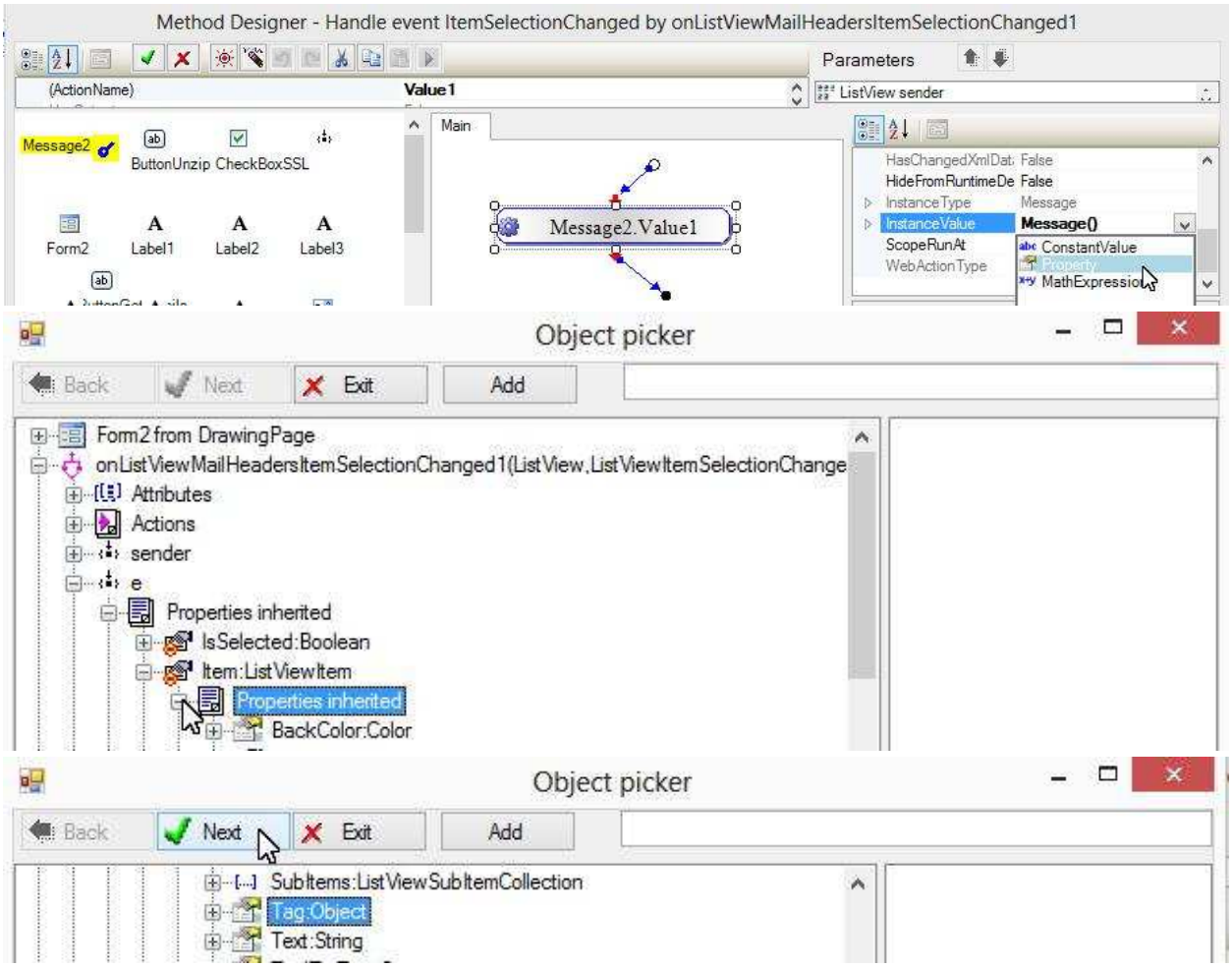
Choose “null” because we do not want to create a new object; we want to assign Tag property to the variable:

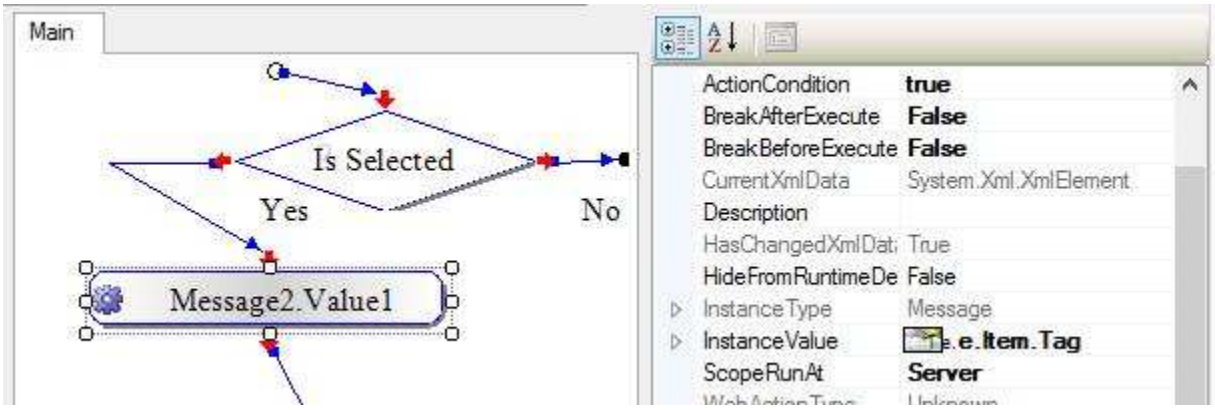


A new variable appears. Create an action to assign Tag property of the list view item to the variable:



A new action appears. Set its InstanceValue to the Tag property of the list view item:

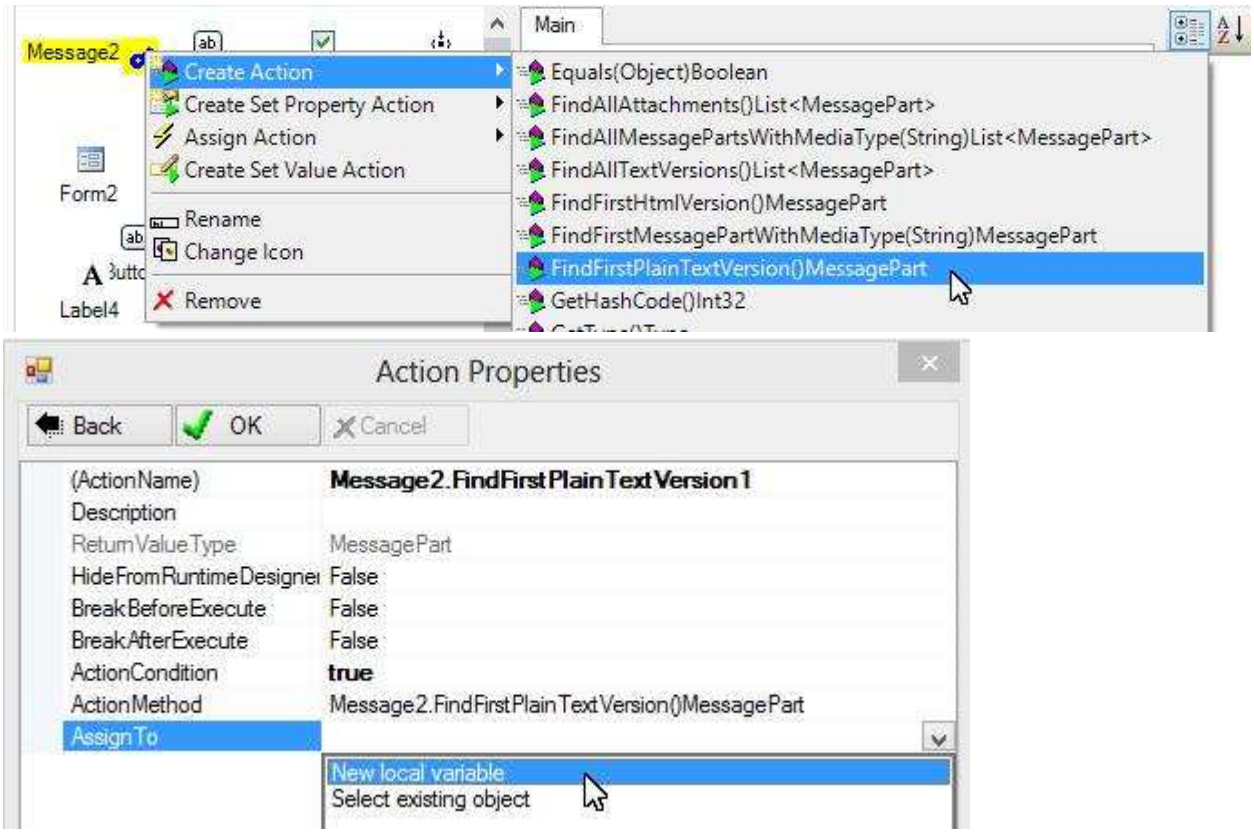


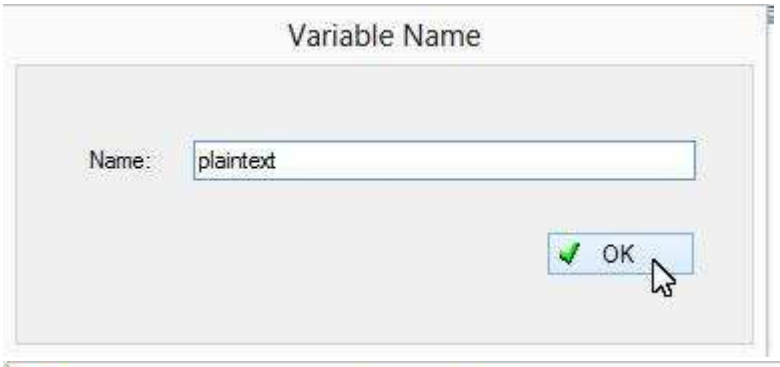


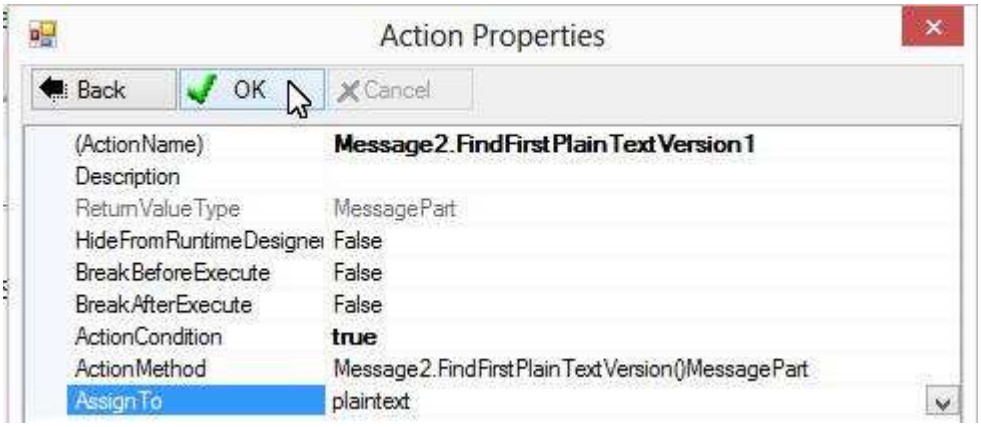
Get Email Text

These programming actions are translated from a C# sample provided by the OpenPop web site.

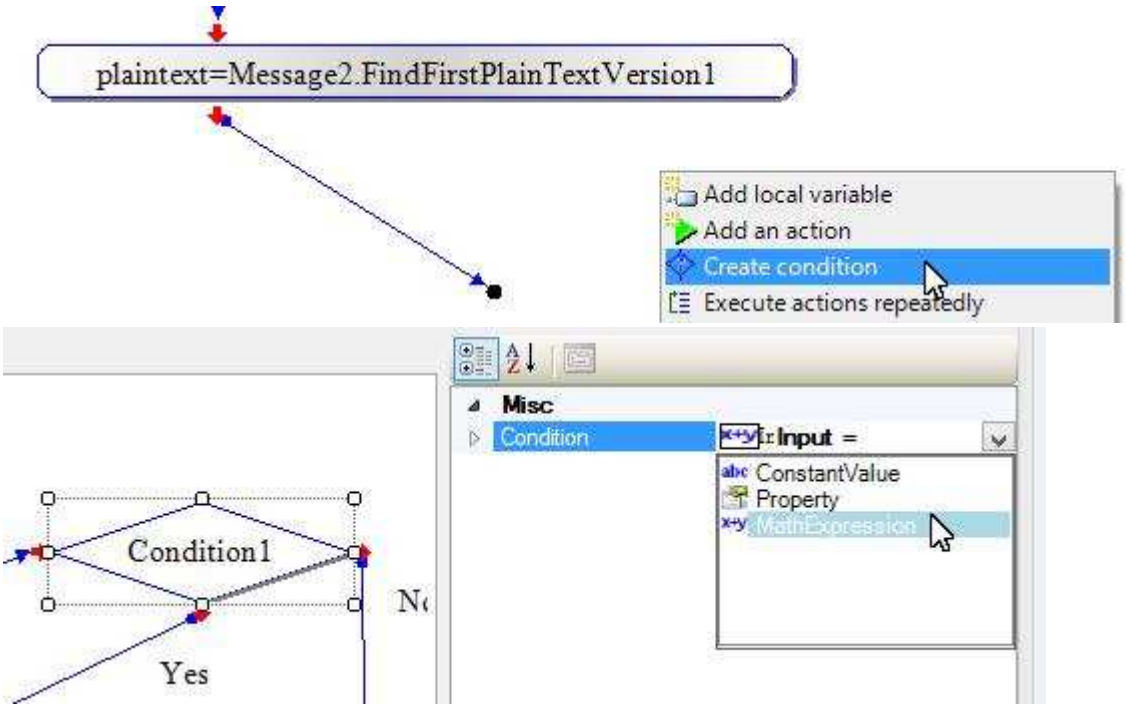
2. Get email body:

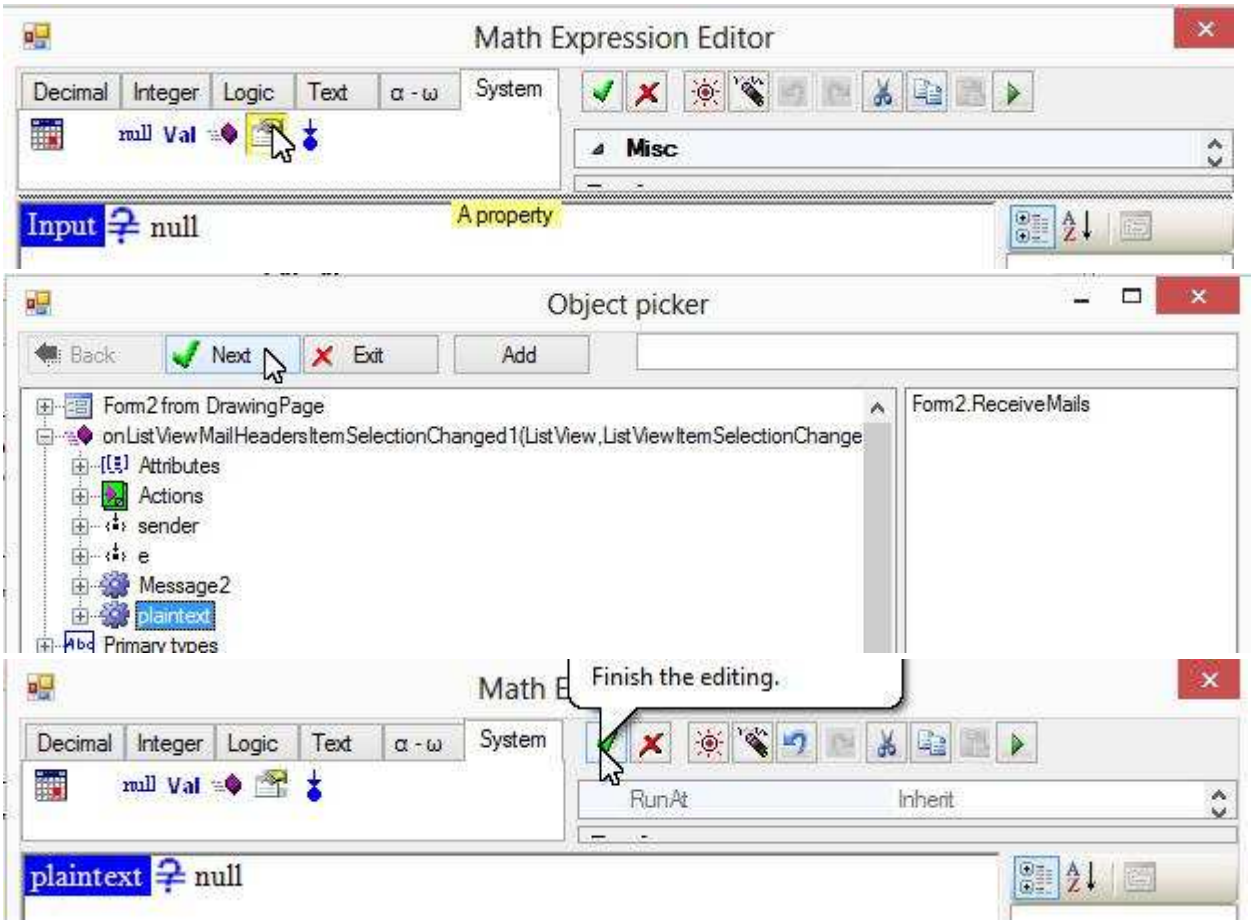




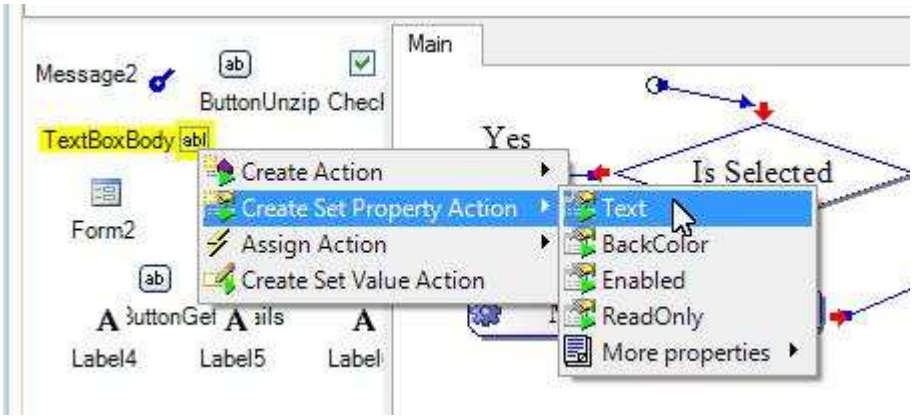


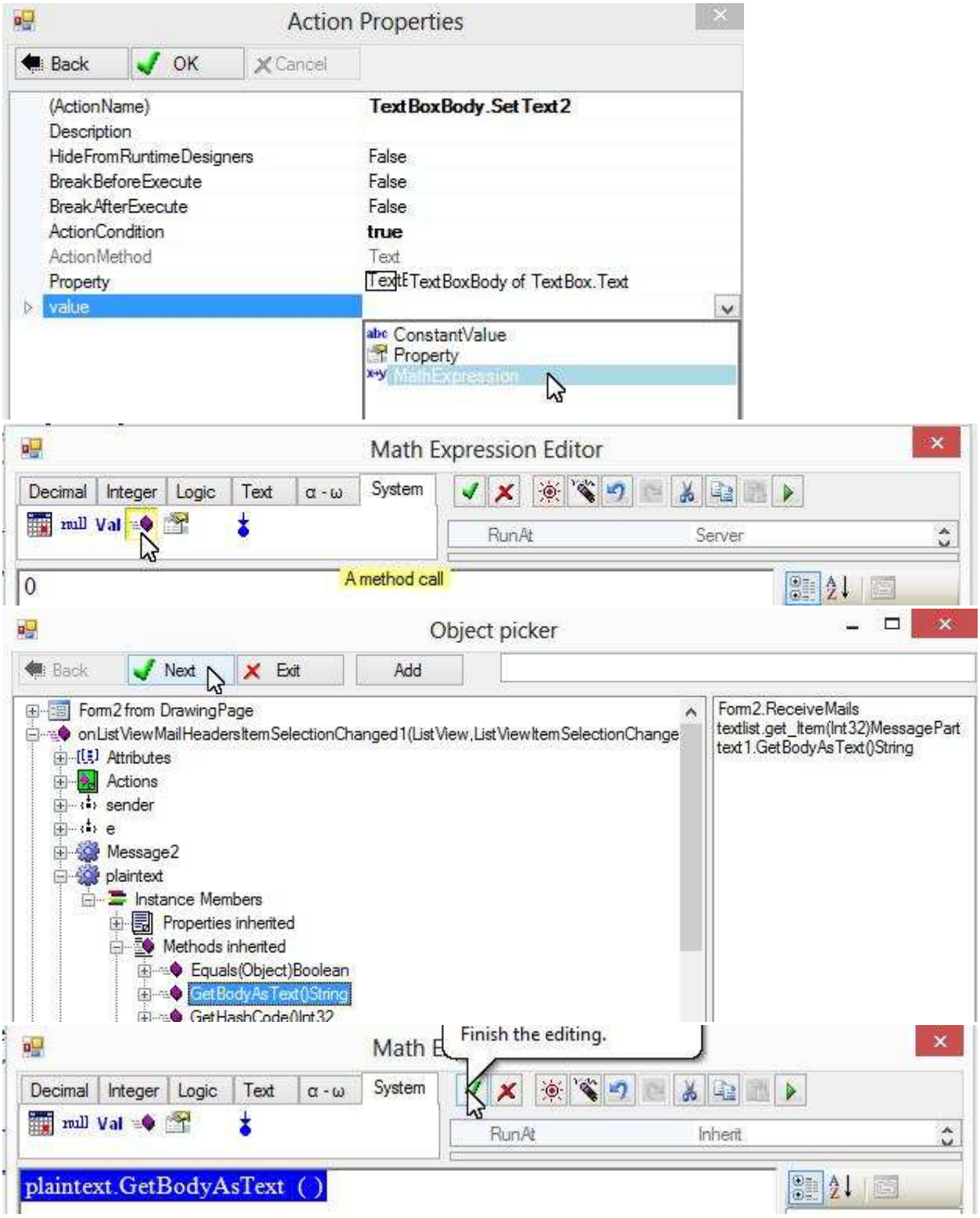
Check whether the return value is null to determine if plain text is found:

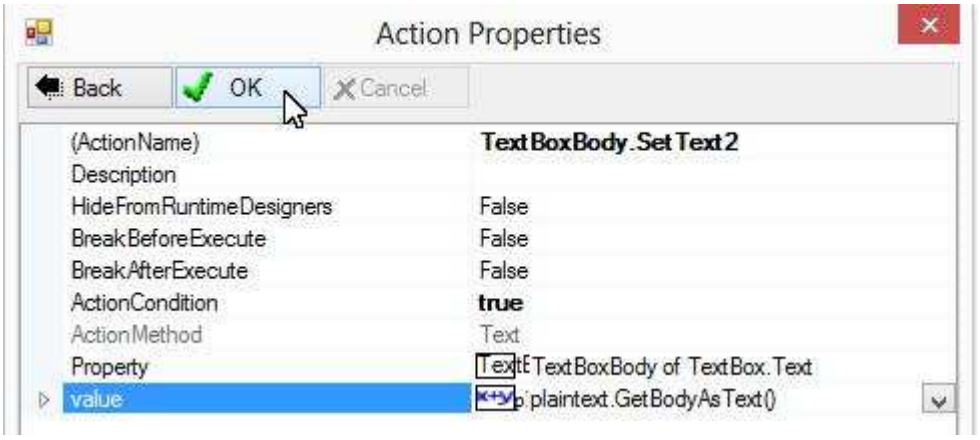




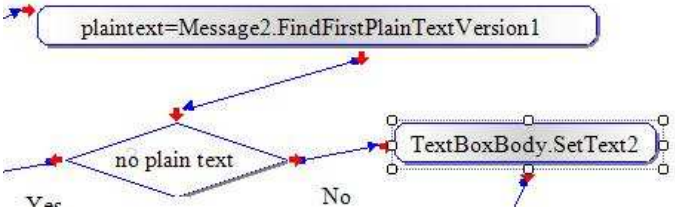
3. If plain text is found then show it in a text box for email body:

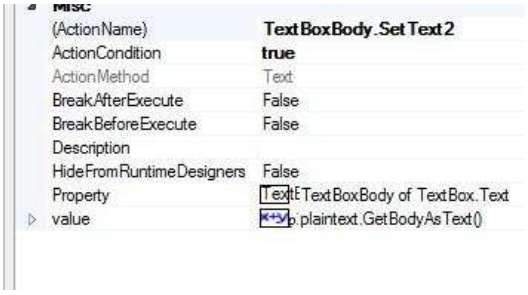



Action Properties
Back
OK
Cancel
(ActionName)
TextBoxBody.SetText2
Description
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Text
Property
TextTextBoxBody of TextBox.Text
value
ConstantValue
Property
MathExpression
Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
null
Val
RunAt
Server
0
A method call
Object picker
Back
Next
Exit
Add
Form2 from DrawingPage
onListViewMailHeadersItemSelectionChanged1(ListView,ListViewItemSelectionChange
Attributes
Actions
sender
e
Message2
plaintext
Instance Members
Properties inherited
Methods inherited
Equals(Object)Boolean
GetBodyAsText()String
GetHashCode()Int32
Form2.ReceiveMails
textlist.get_Item(Int32)MessagePart
text1.GetBodyAsText()String
Finish the editing.
Inherit
plaintext.GetBodyAsText ( )

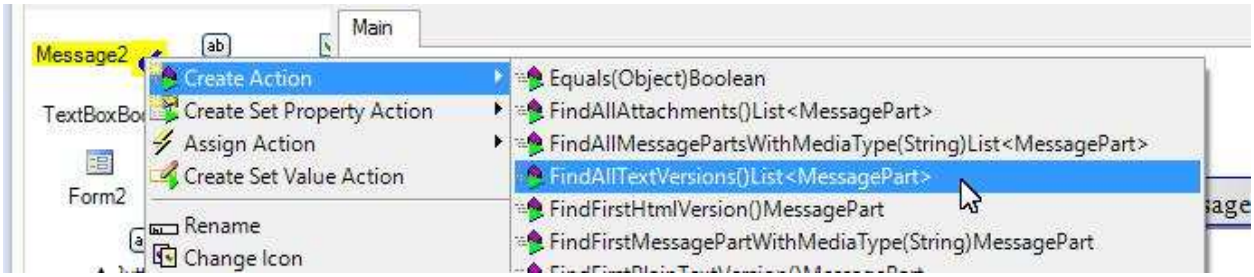


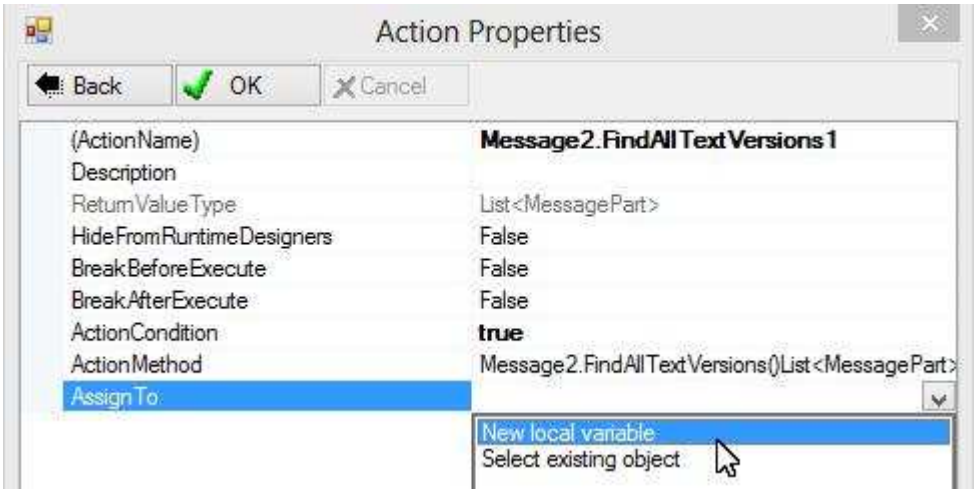
Link it to No:

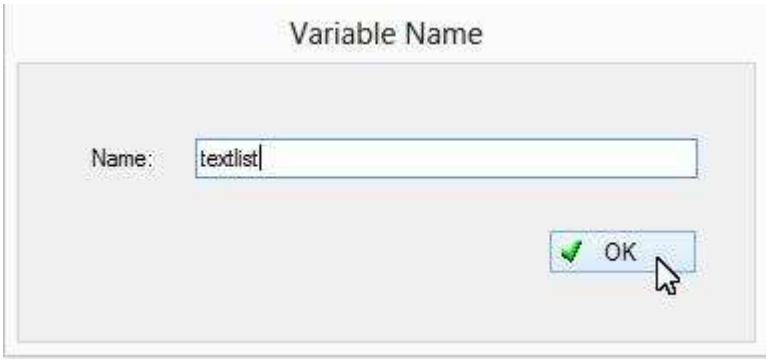


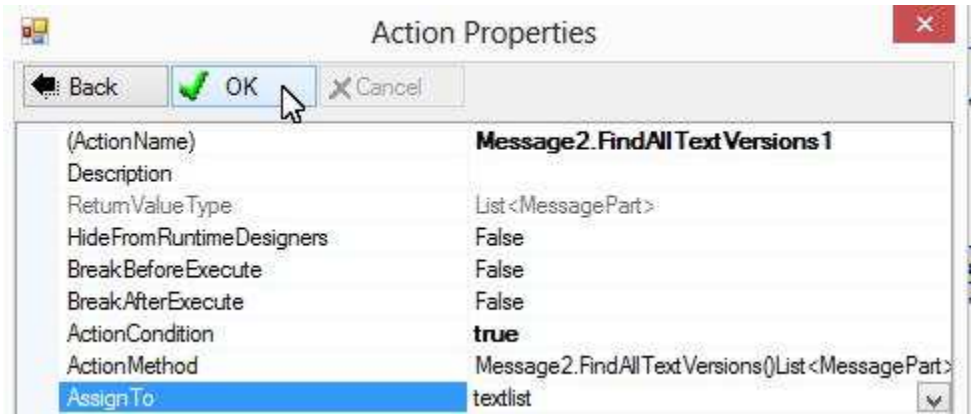


4. If plain text not found then try find any text:

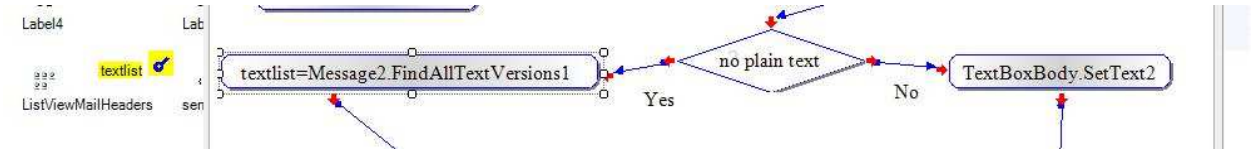




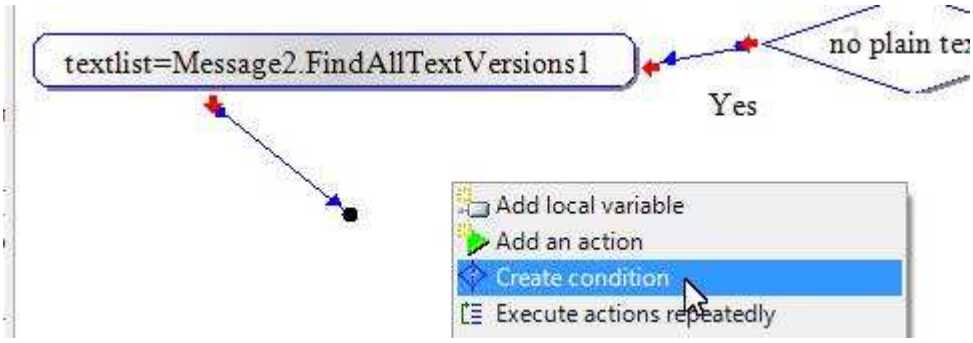


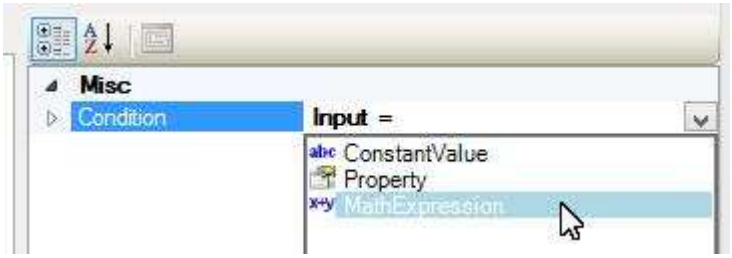


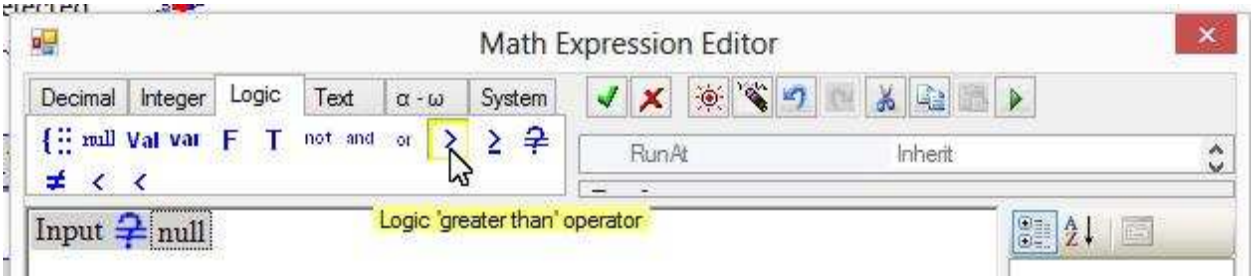
Link the new action to Yes:



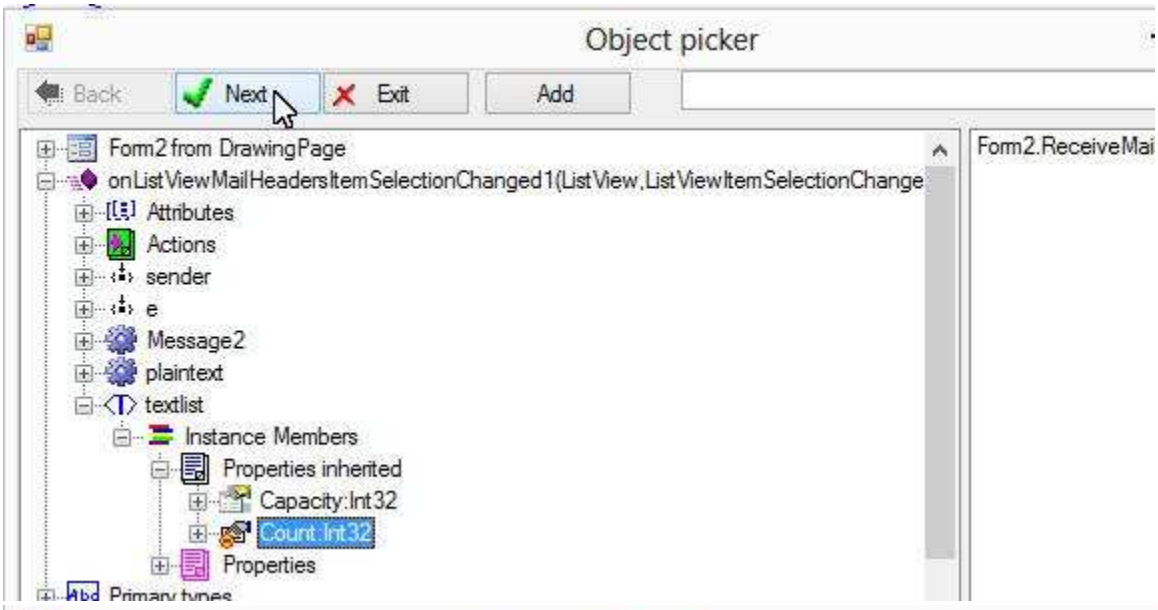
If textlist is not empty then use the first text as email text:

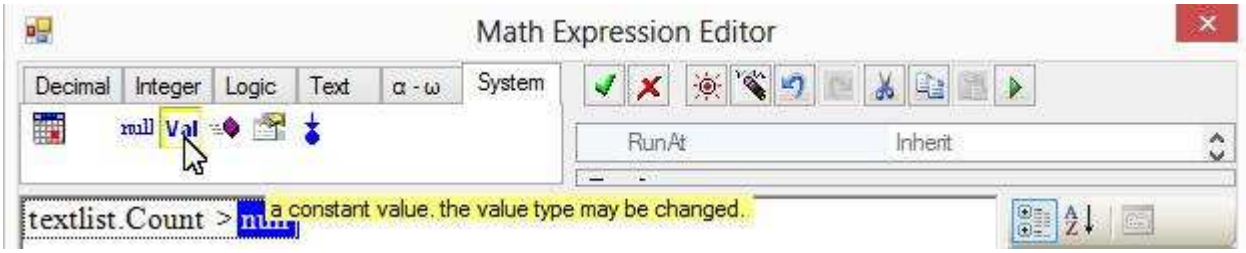


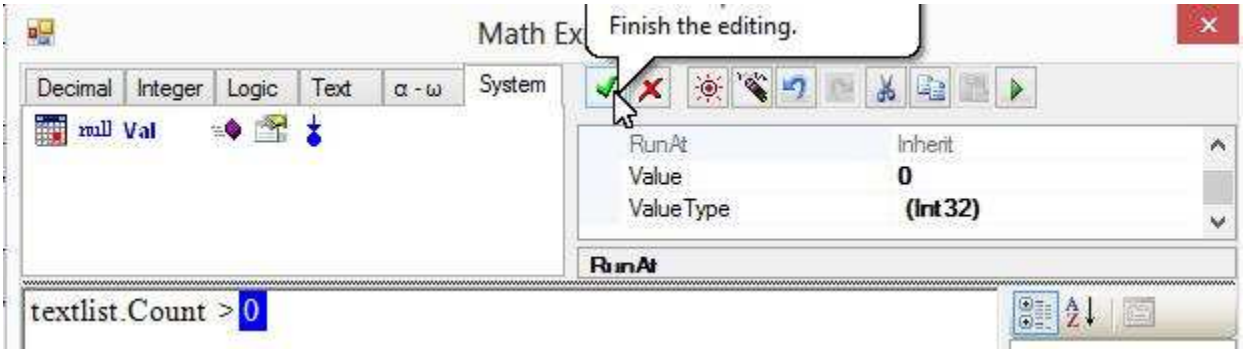



Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
null Val var F T not and or
RunAt
Inherit
Input null
Logic 'greater than' operator

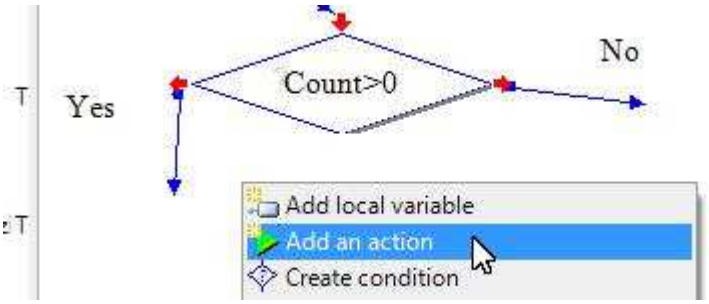

Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
null Val
RunAt
Inherit
Input > null
A property

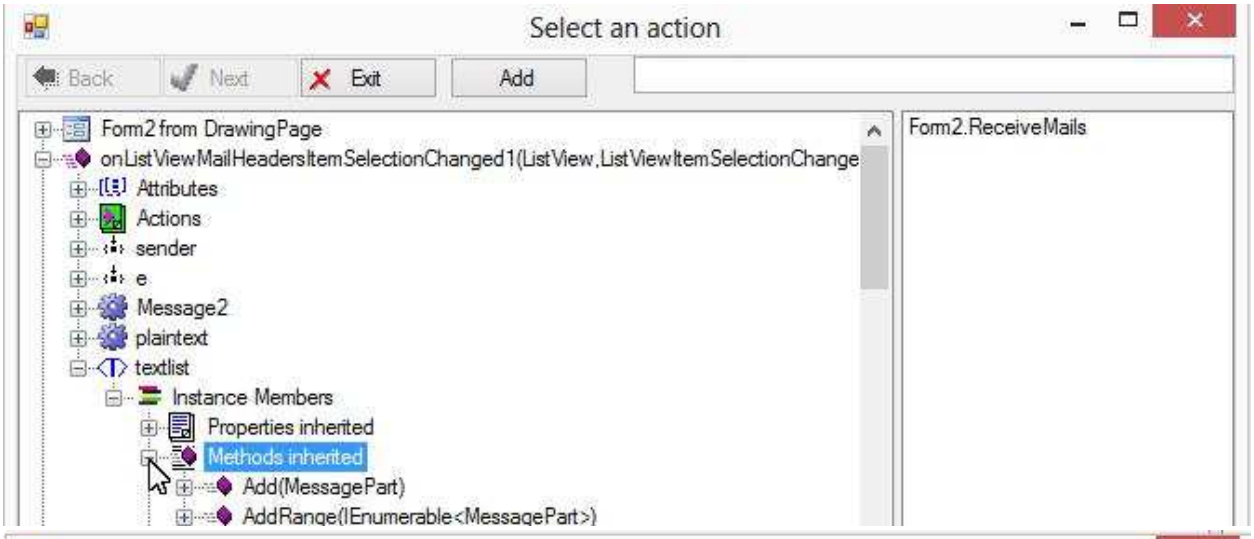

Object picker
Back
Next
Exit
Add
Form2 from DrawingPage
onListViewMailHeadersItemSelectionChanged1(ListView,ListViewItemSelectionChange
Attributes
Actions
sender
e
Message2
plaintext
textlist
Instance Members
Properties inherited
Capacity:Int32
Count:Int32
Properties
Form2.ReceiveMai

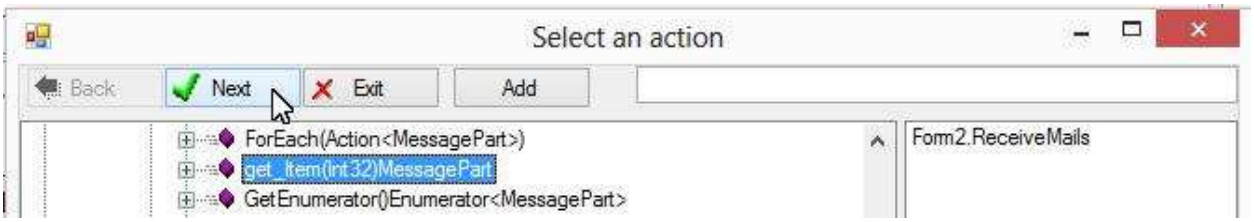

Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
null Val
RunAt
Inherit
textlist.Count > nu
a constant value. the value type may be changed.

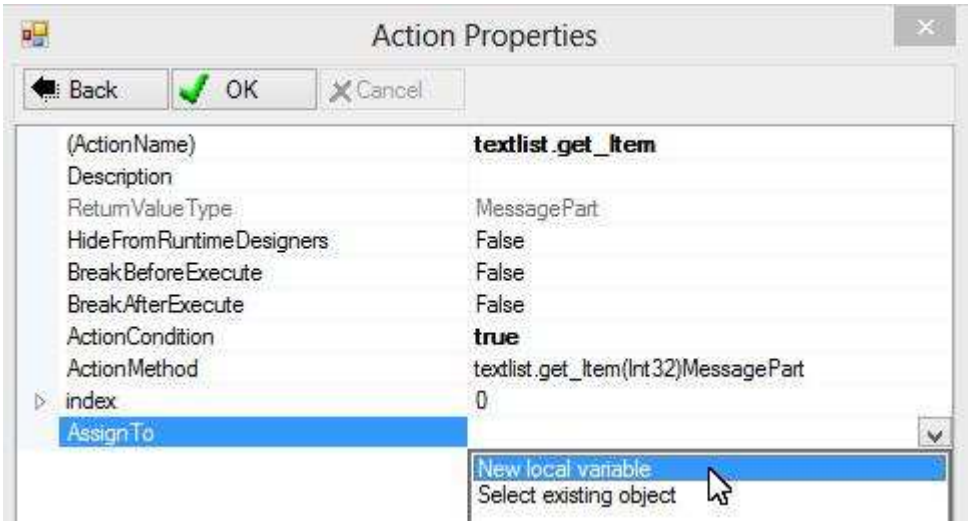


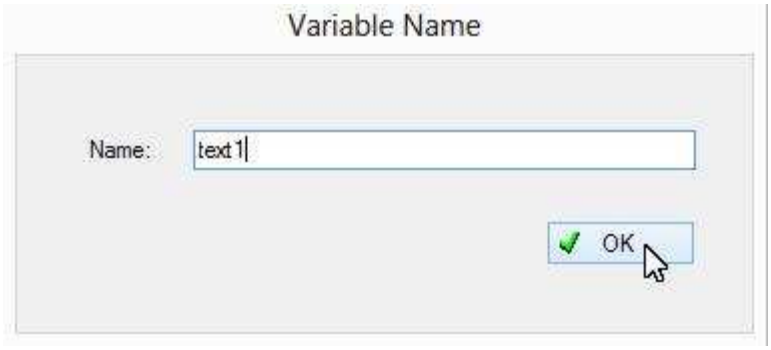
5. If the text list is not empty then get the first one:

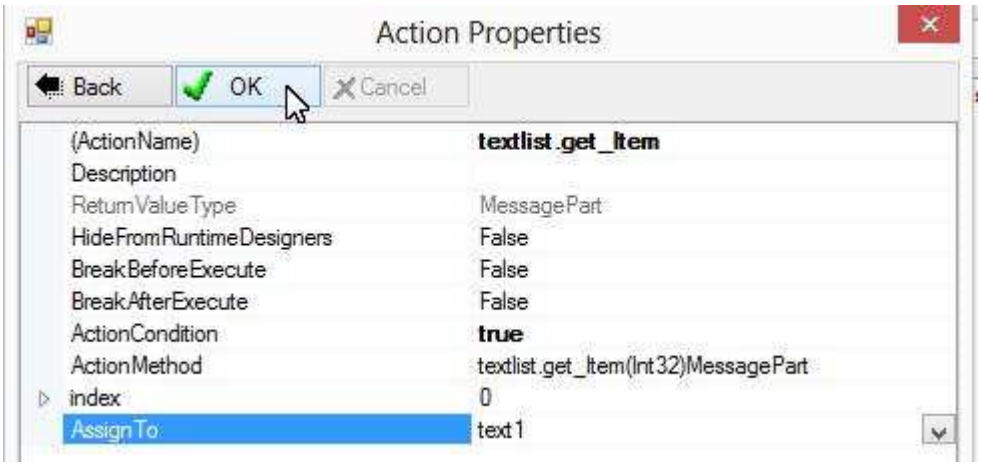




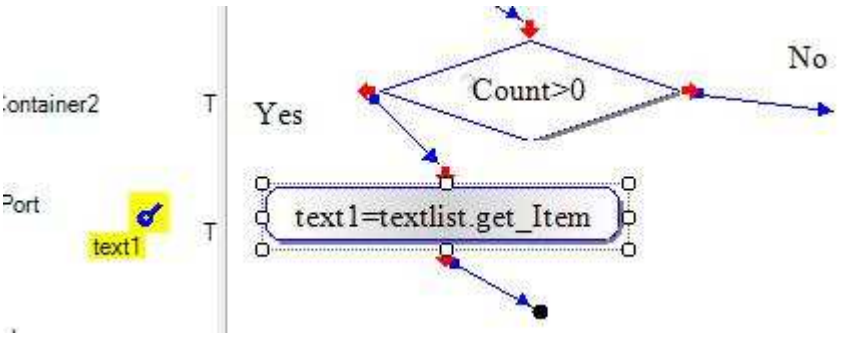




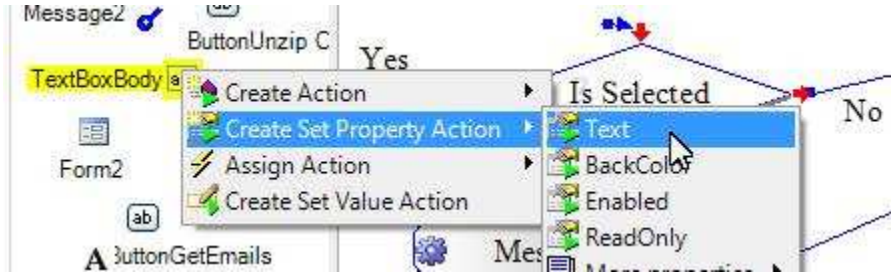


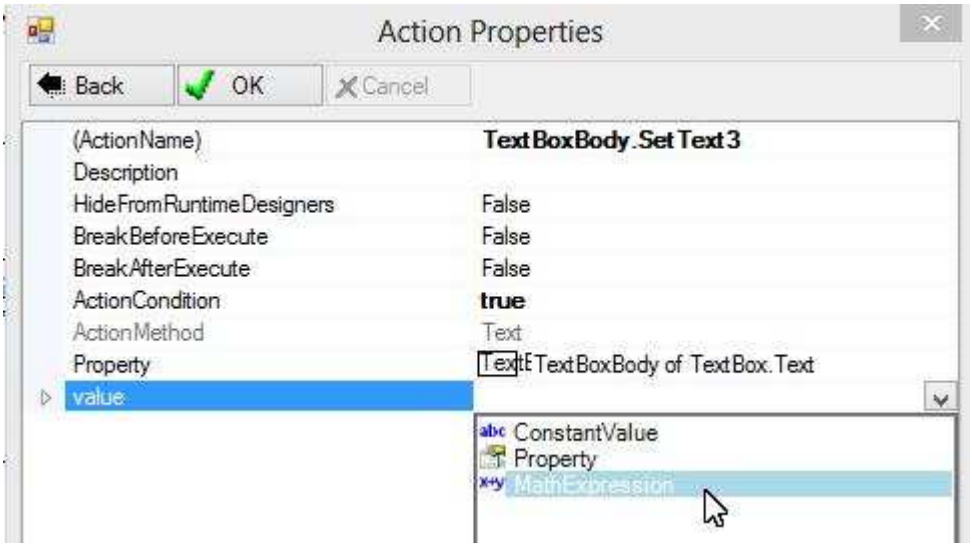


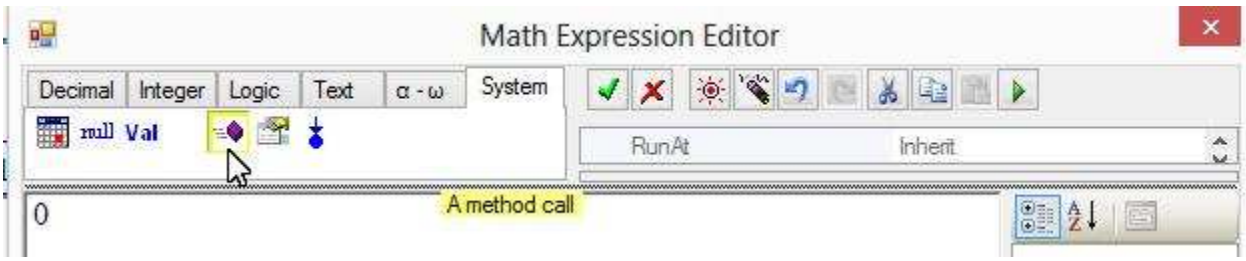
Link the new action to Yes:

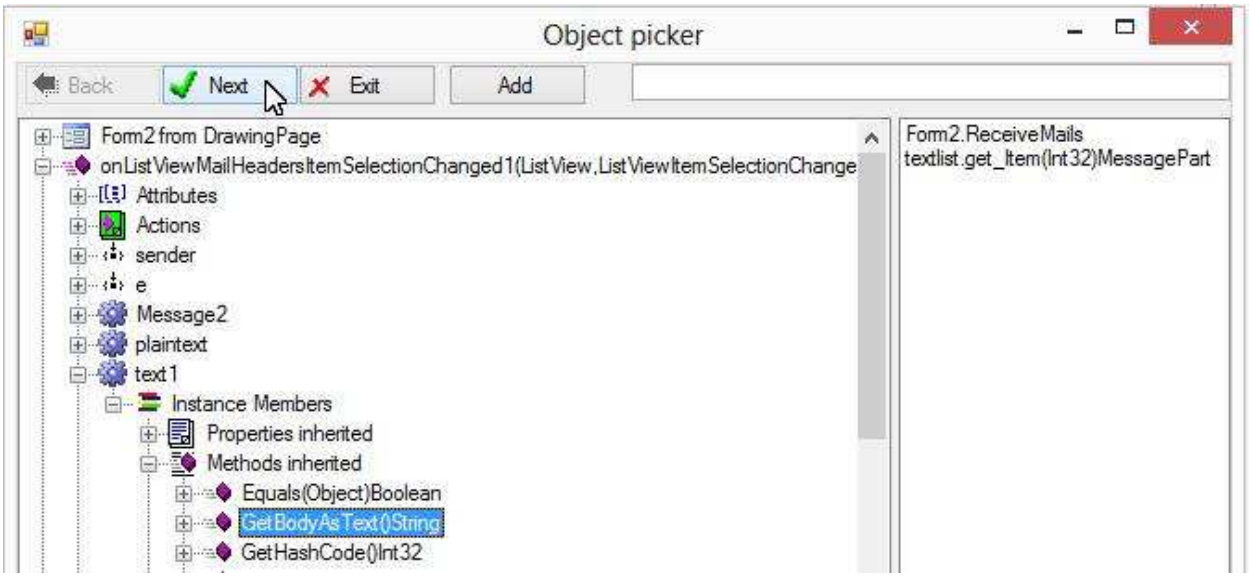


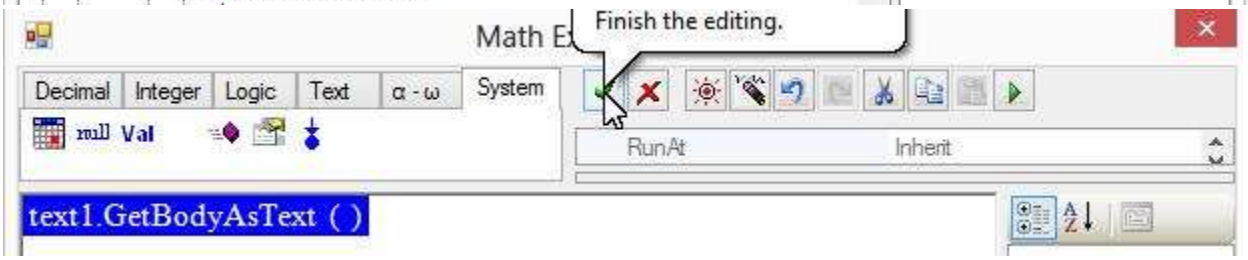
Show text1 on the text box for email body:

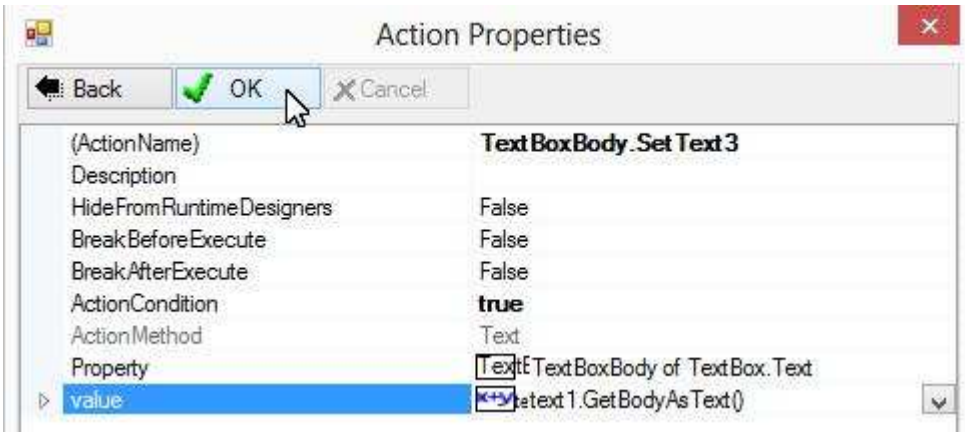




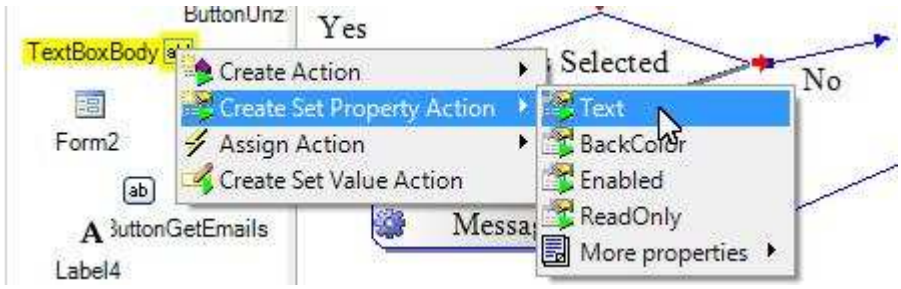


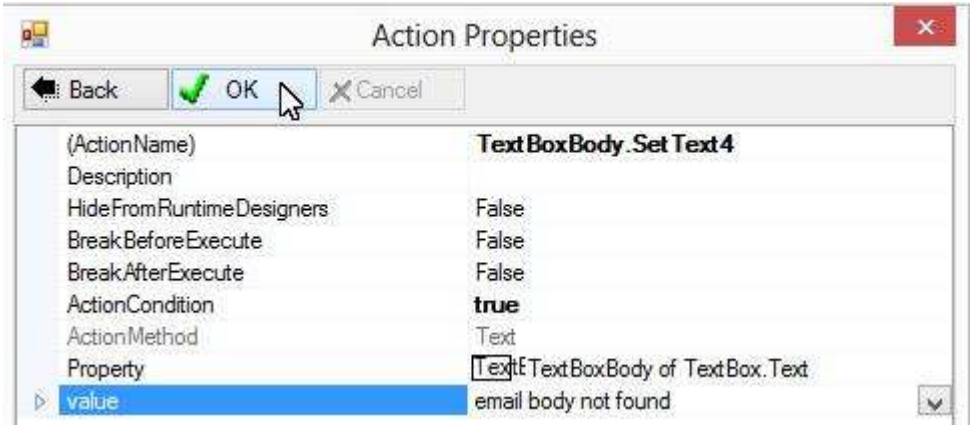




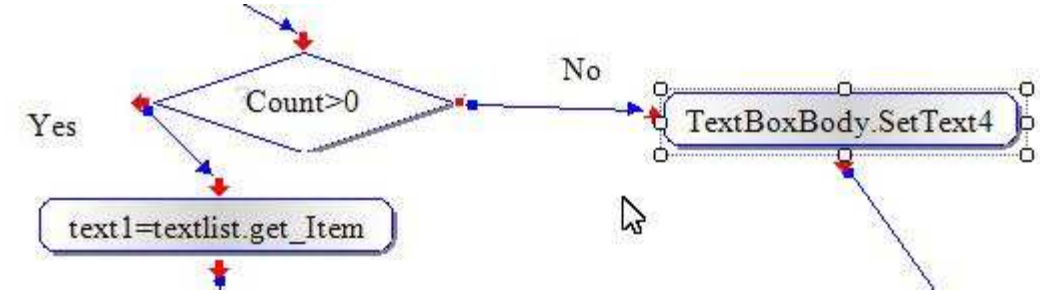


6. If the text list is empty then show a string “email body not found”:





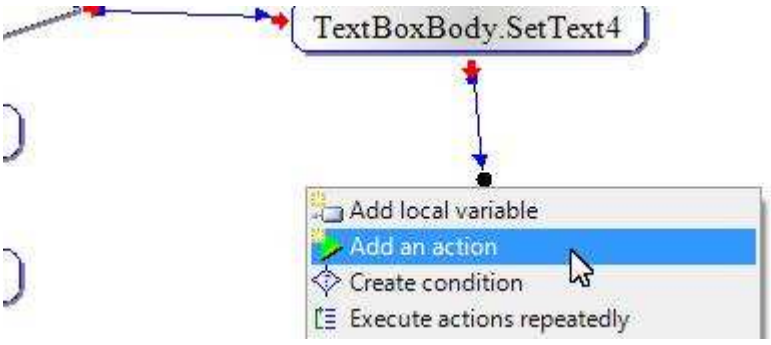
Link the new action to No:

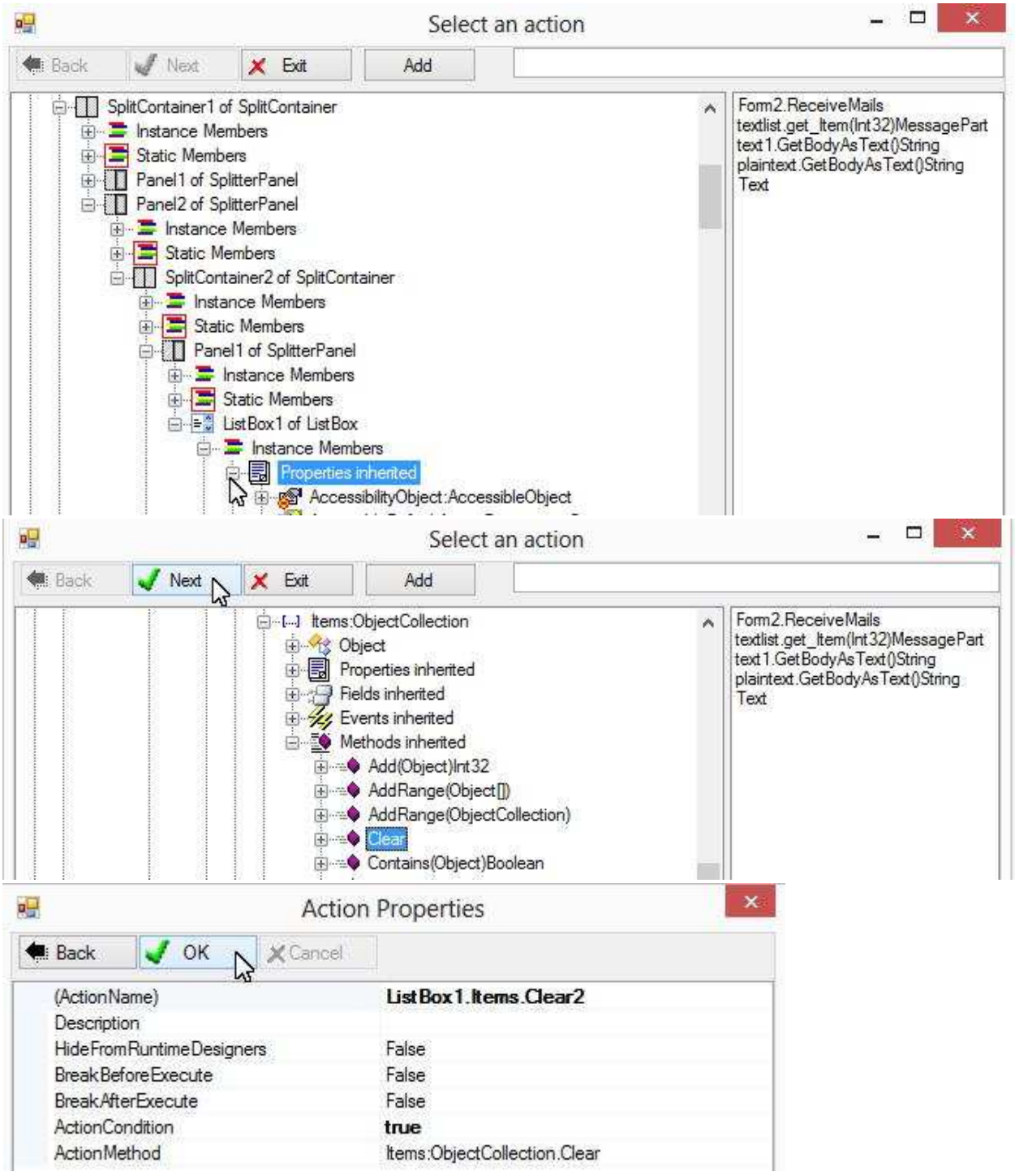


List and Save Attachments

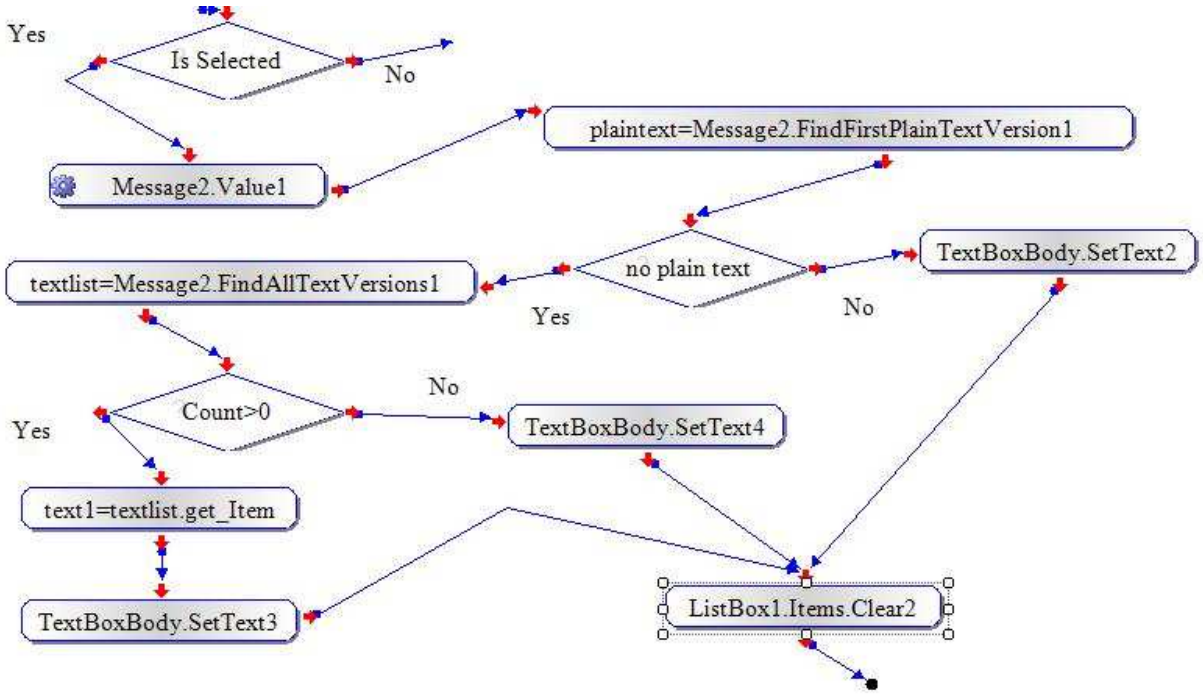
Clear Attachment List

7. Clear attachment list box first:



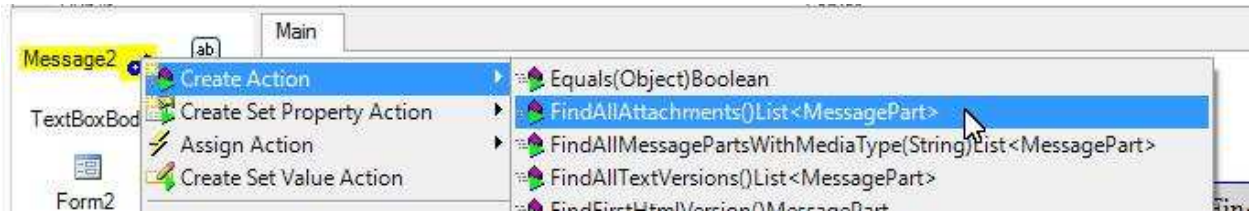


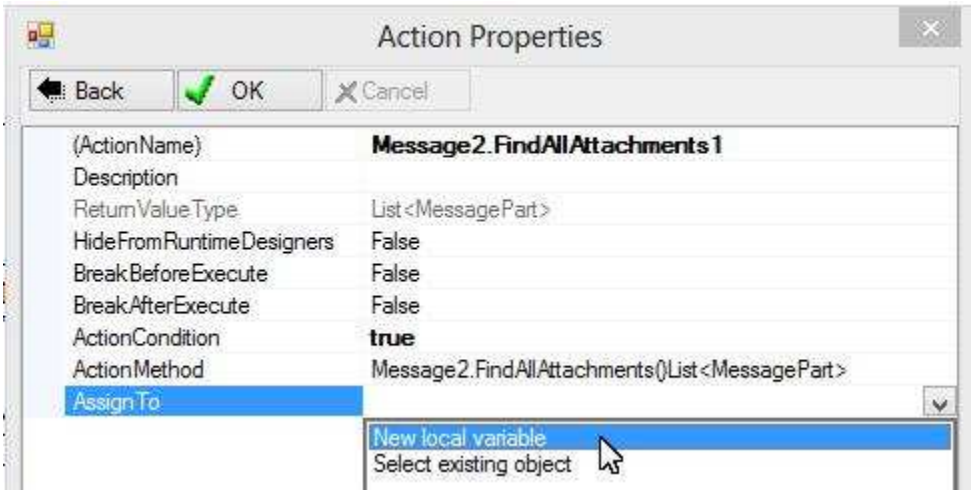
Link all 3 branches to the new action:

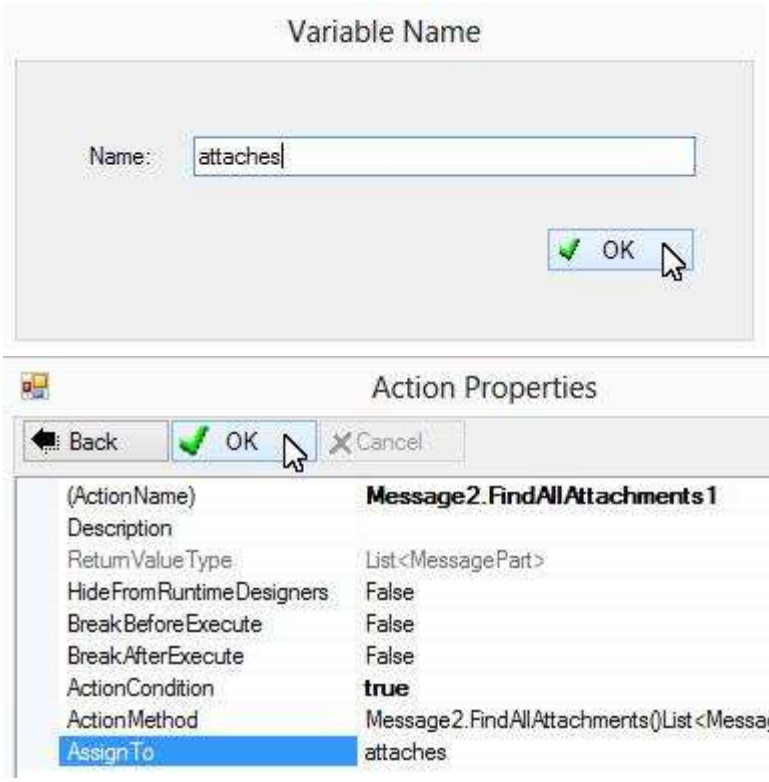


Get attachments

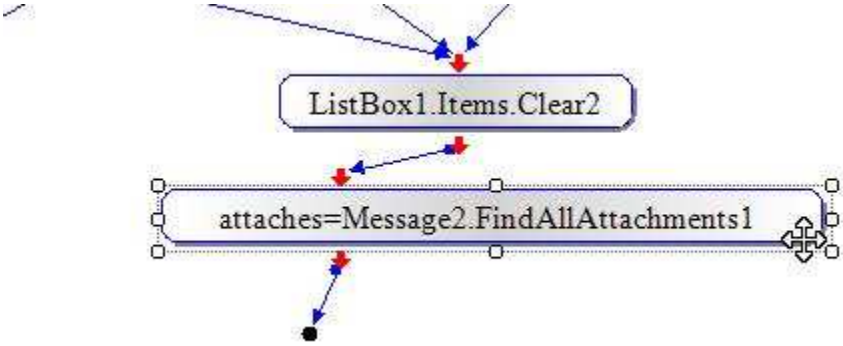
8. Create an action to get all attachments





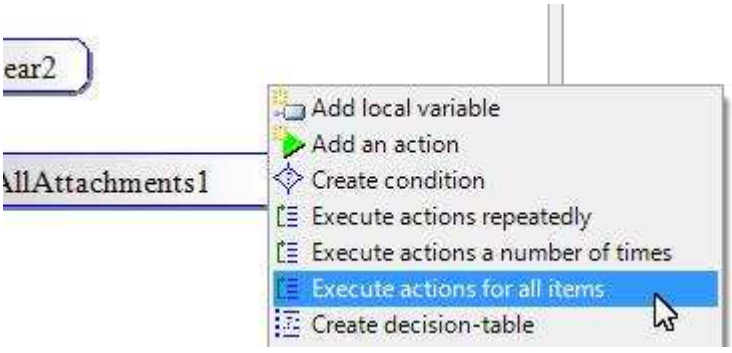


Link the new action:

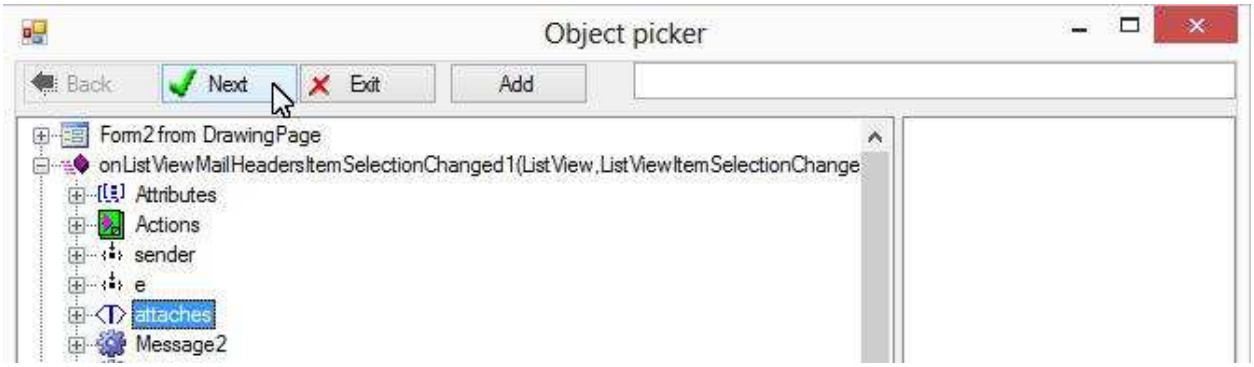


Process Each Attachment

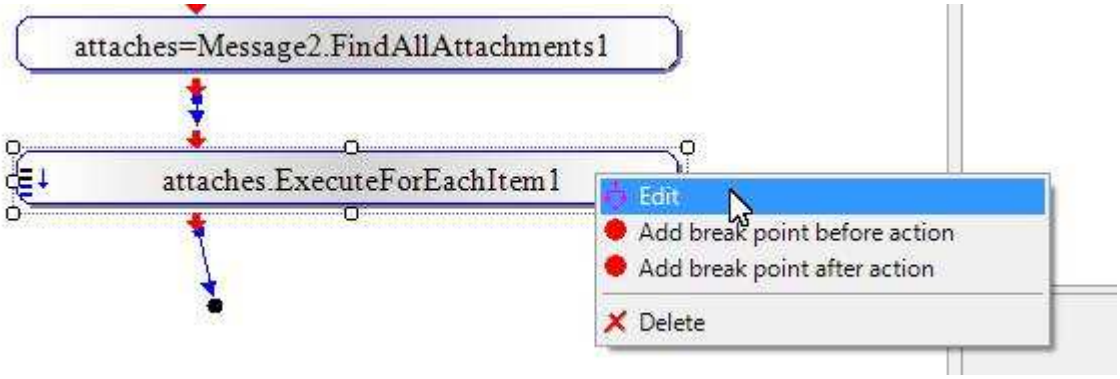
9. Use a loop action to process every attachment:



Select variable “attaches” for the new loop actions:

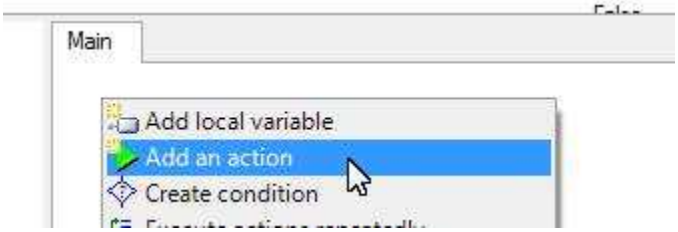


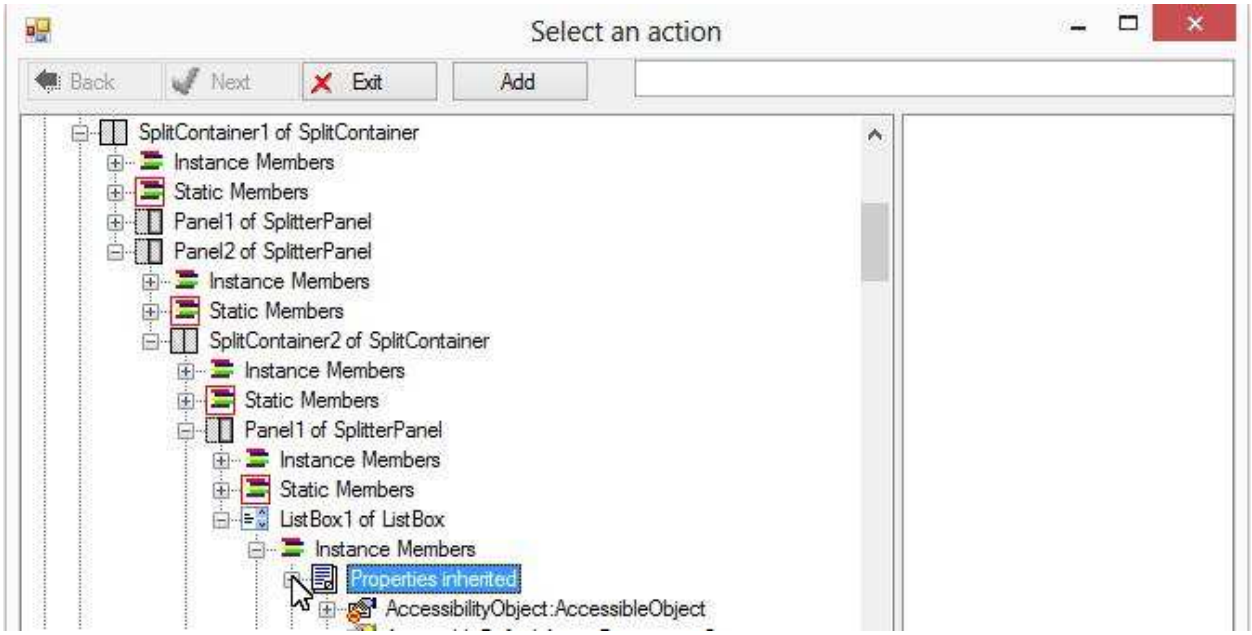
Edit the loop action to add actions to process each attachment:

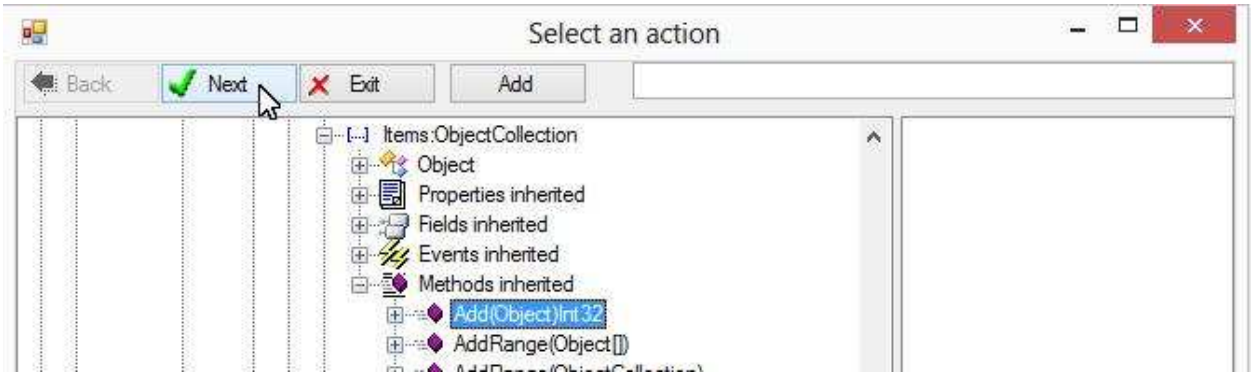


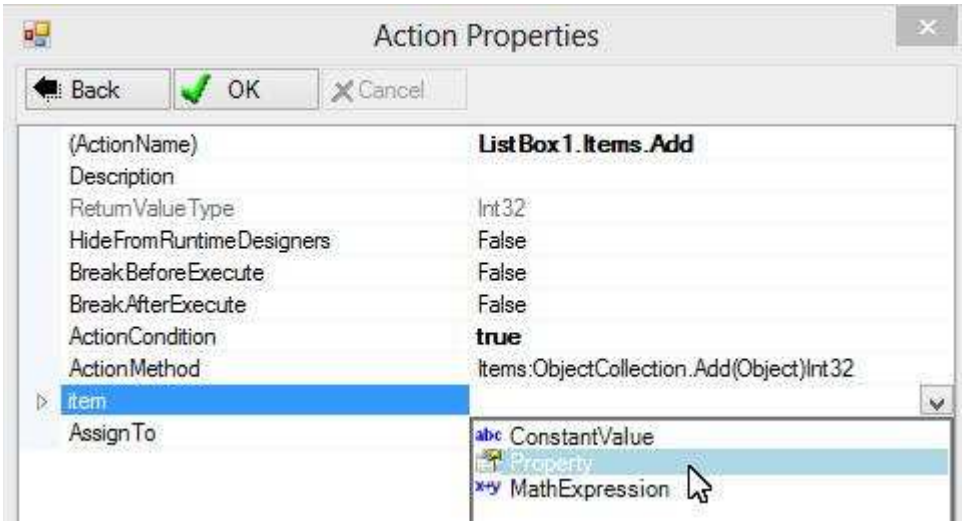
Show file name

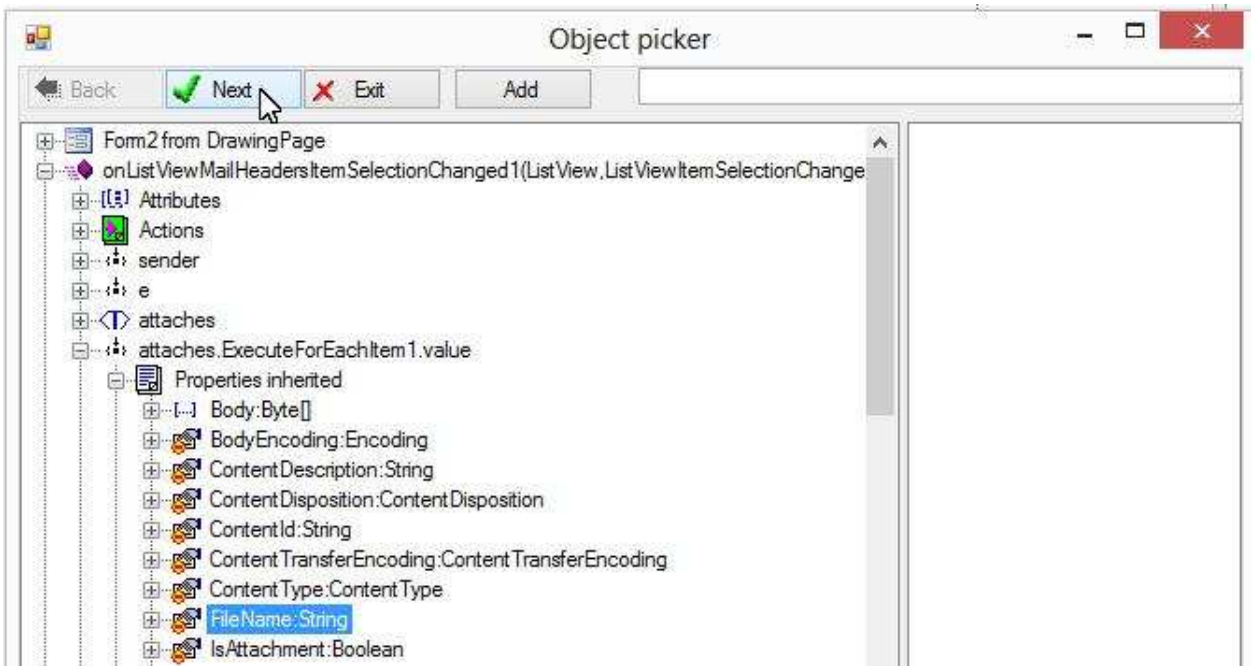
10. For this sample, we add file name of the attachment to a list box:

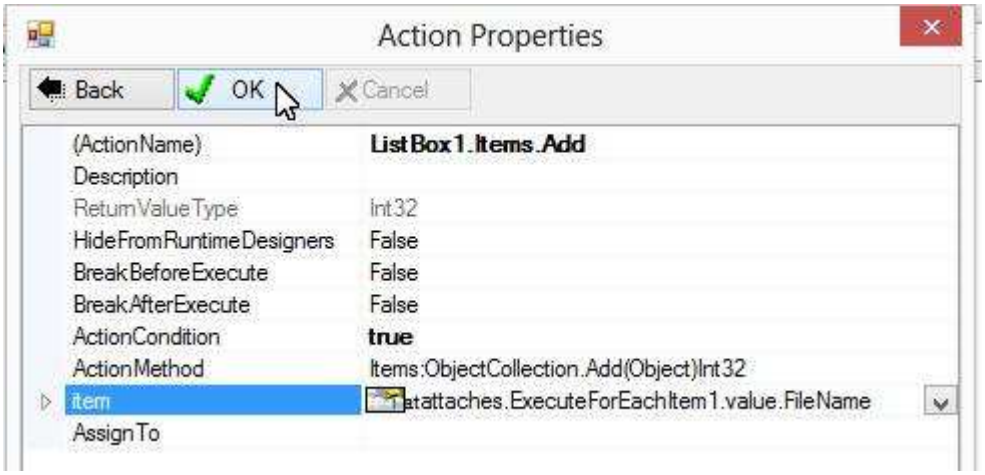



Select an action
Back
Next
Exit
Add
SplitContainer1 of SplitContainer
Instance Members
Static Members
Panel1 of SplitterPanel
Panel2 of SplitterPanel
Instance Members
Static Members
SplitContainer2 of SplitContainer
Instance Members
Static Members
Panel1 of SplitterPanel
Instance Members
Static Members
ListBox1 of ListBox
Instance Members
Properties inherited
AccessibilityObject:AccessibleObject


Select an action
Back
Next
Exit
Add
Items:ObjectCollection
Object
Properties inherited
Fields inherited
Events inherited
Methods inherited
Add(Object):Int32
AddRange(Object[])

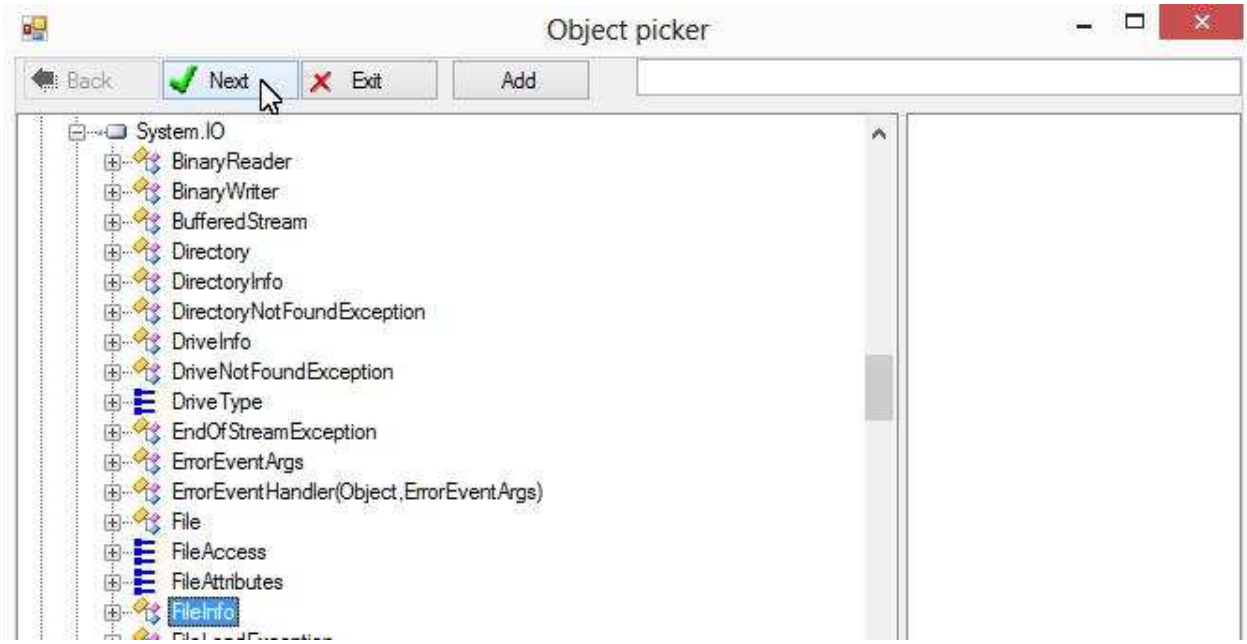

Action Properties
Back
OK
Cancel
(ActionName)
ListBox1.Items.Add
Description
ReturnValueType
Int32
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
Items:ObjectCollection.Add(Object):Int32
item
AssignTo
ConstantValue
Property
MathExpression

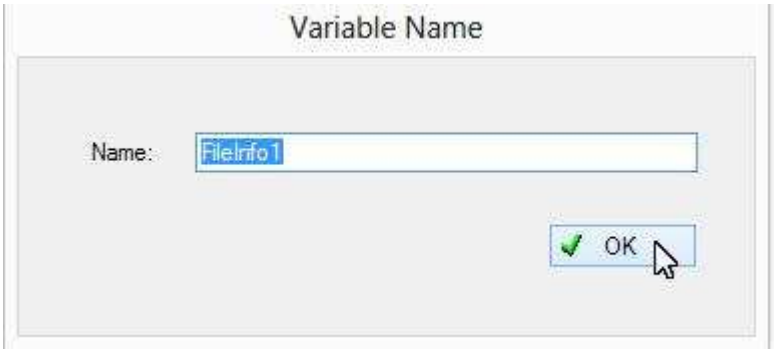




Save attachment to file

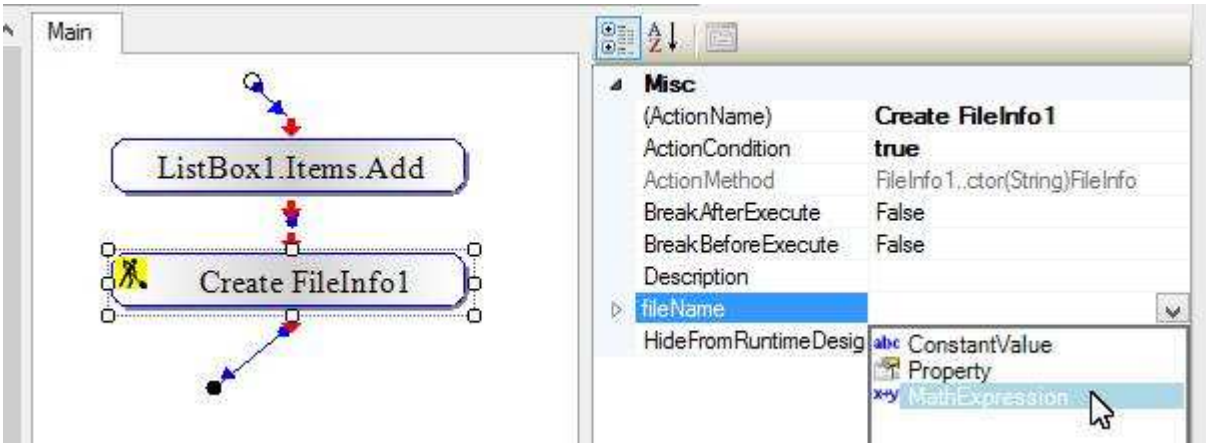
11. To save the attachment to file, OpenPop requires a FileInfo object. Let's create it:



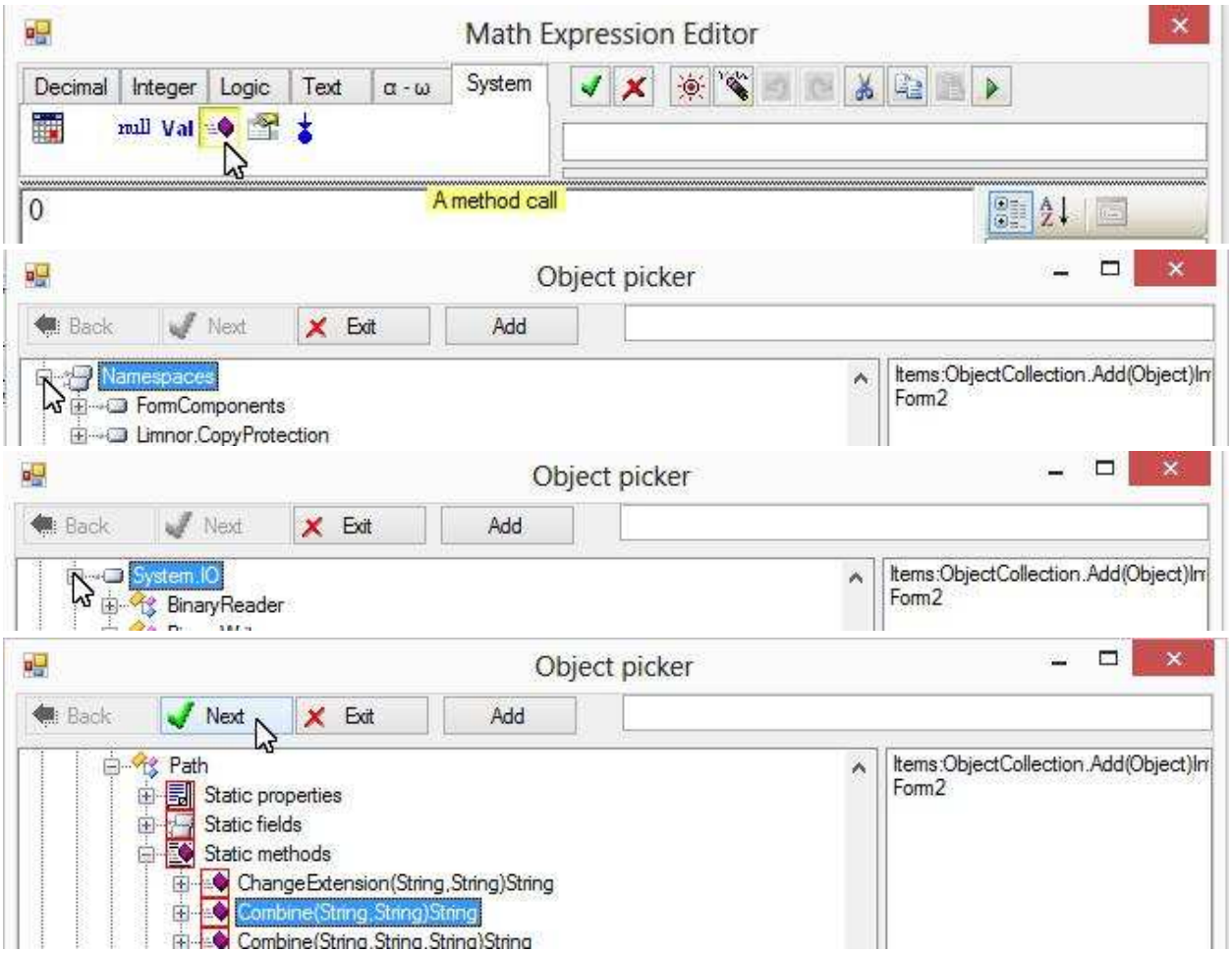


Choose a constructor which uses a file name:

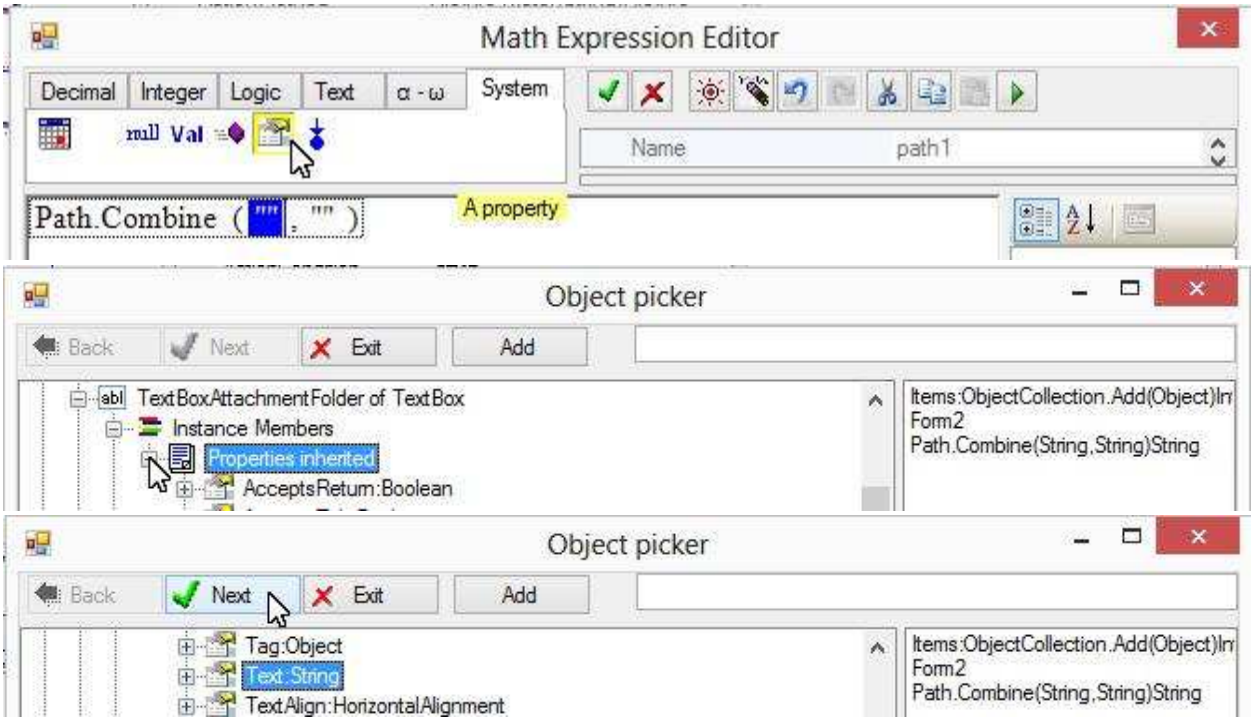
A constructor action appears. Set its fileName to use the file name of the attachment and use the folder specified by a text box:



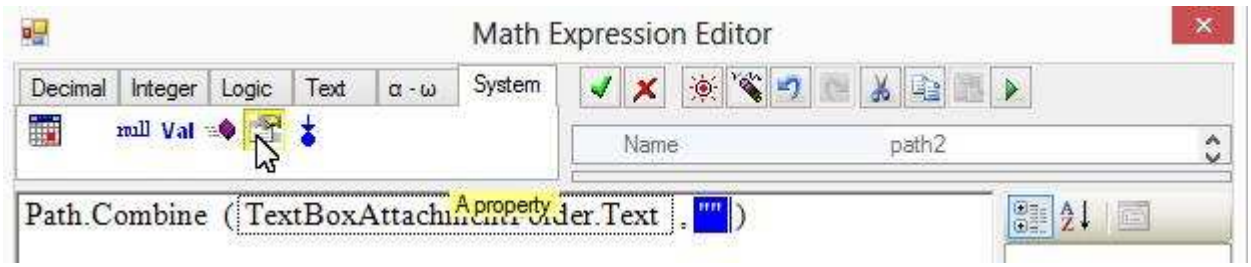
Use Combine method of Path class to form a full path of a file to save the attachment:

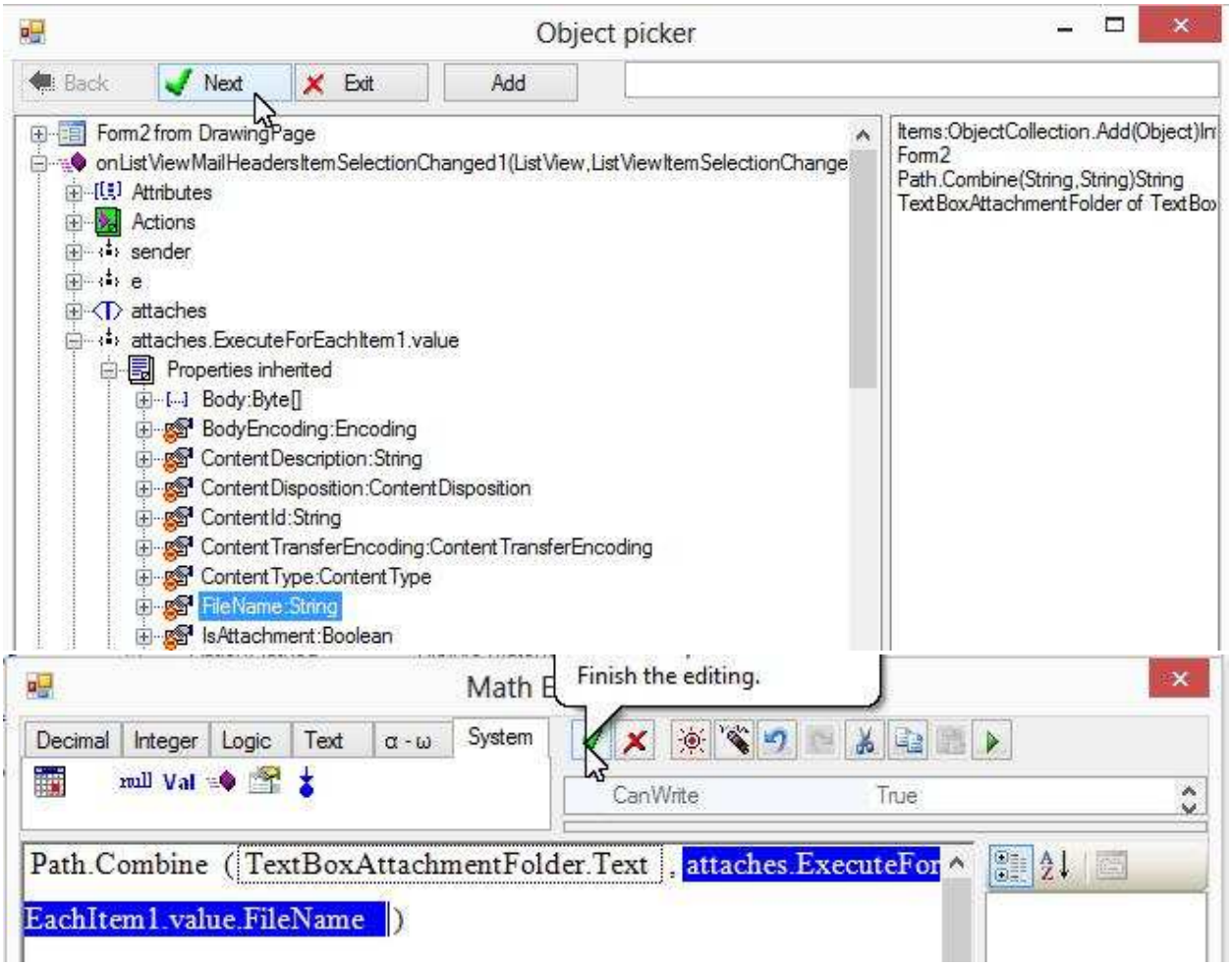


The first parameter of Combine is a folder:



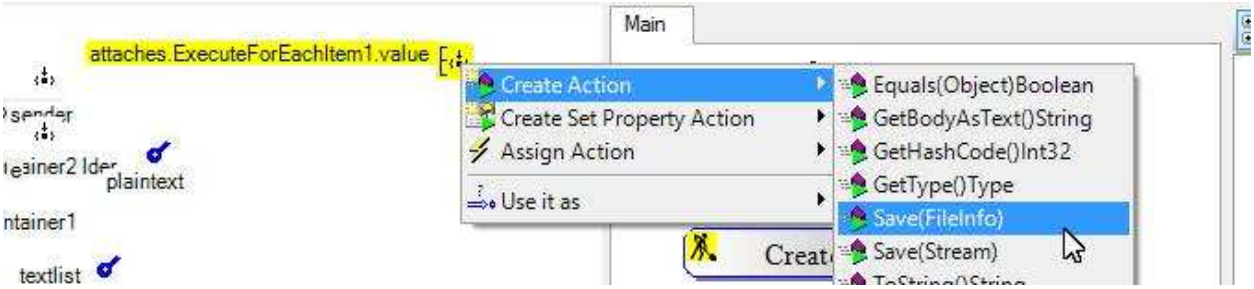
The second parameter of Combine is a file name, which we use file name of the attachment object:



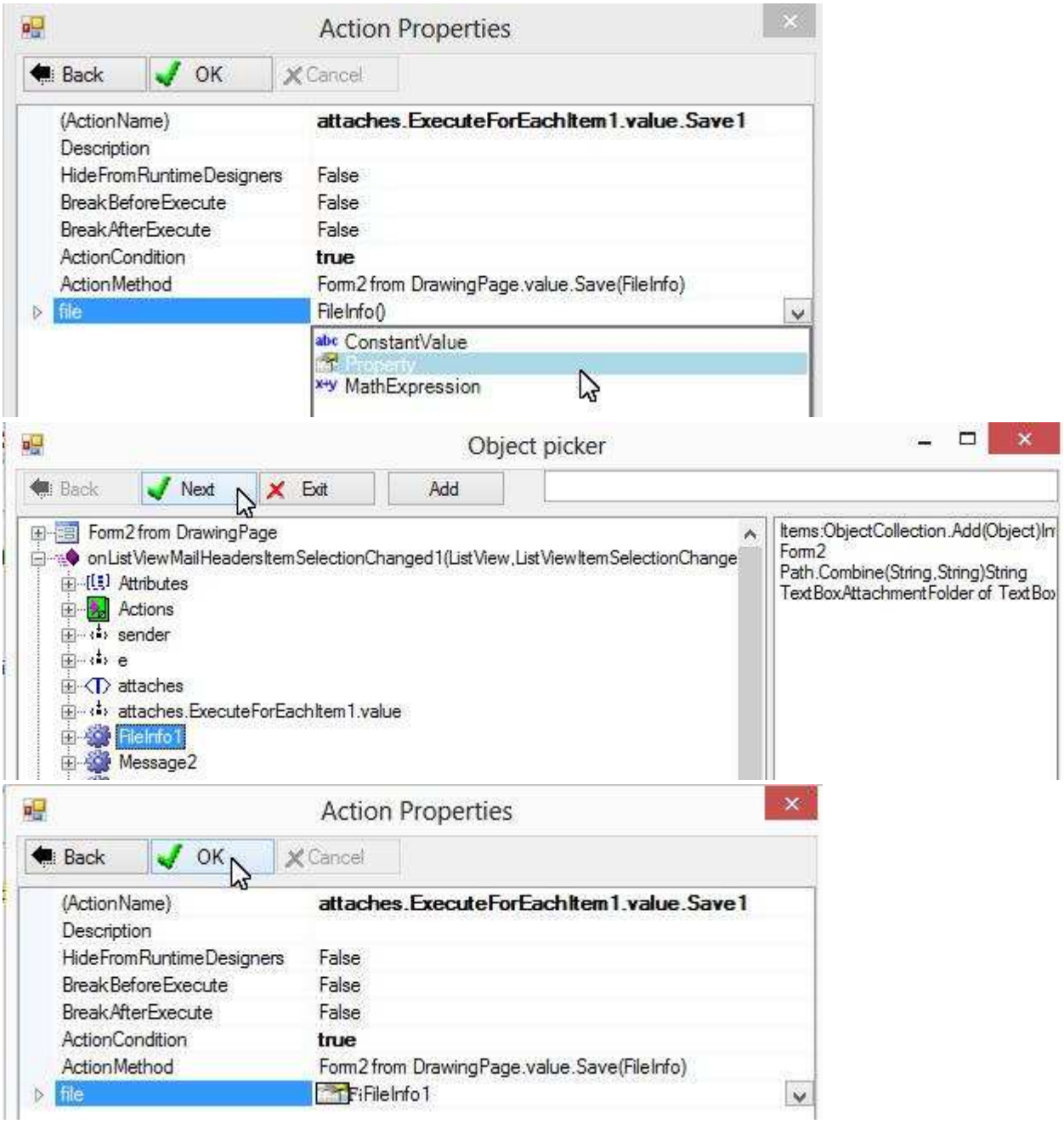


We thus create a FileInfo object required by OpenPop:

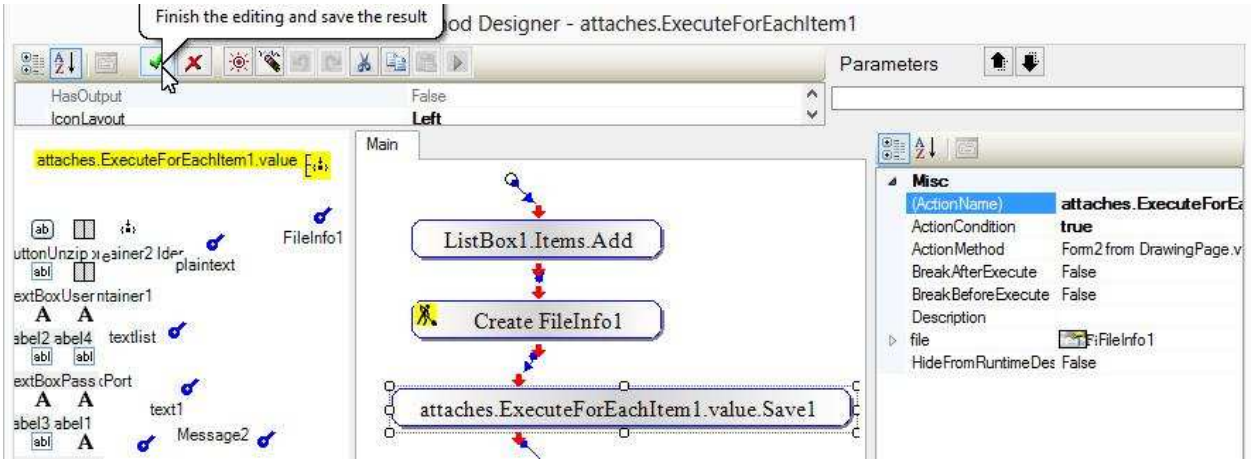
12. Save the attachment to file:



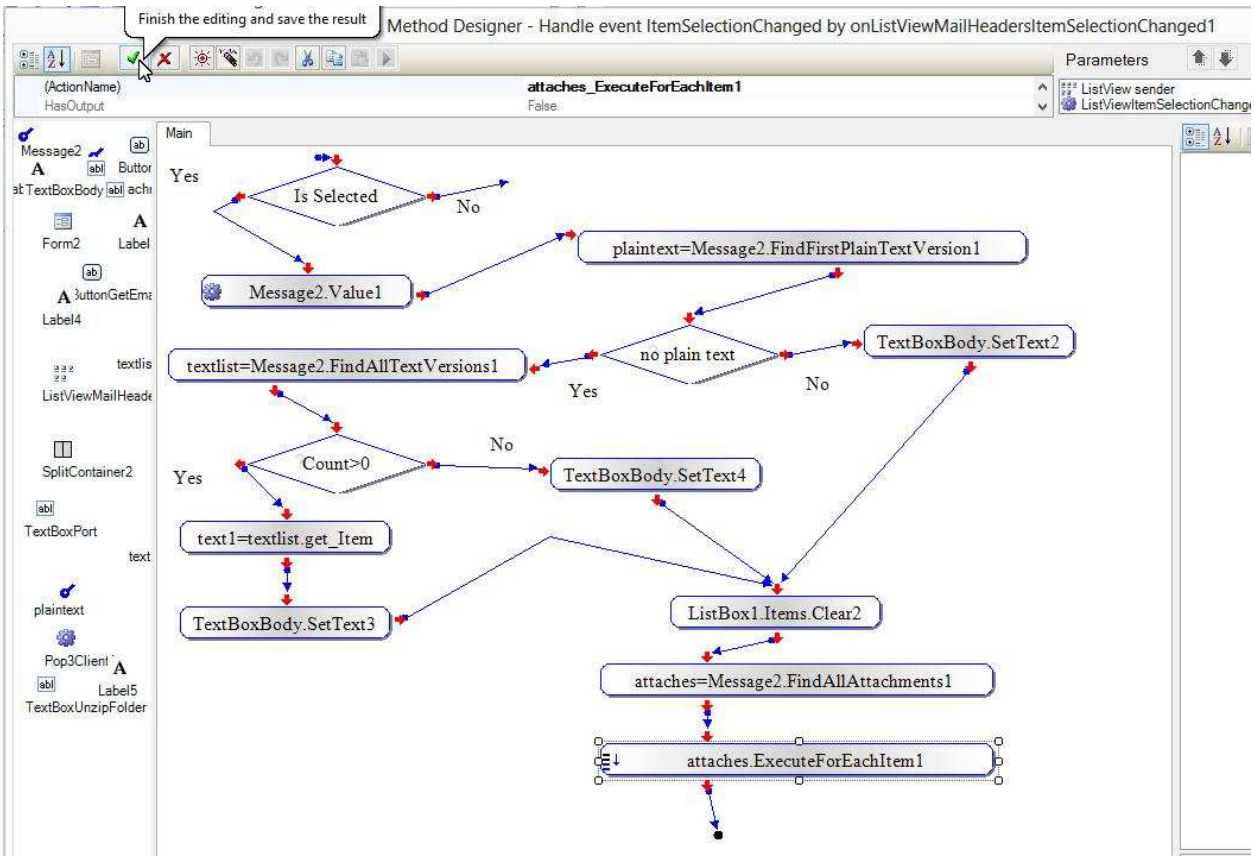
Use the FileInfo object we just created:



Link the new action. That is all we need to process an attachment:

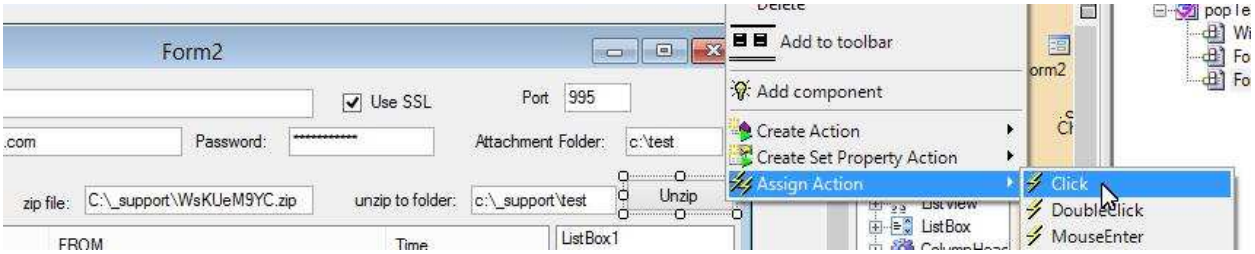


13. That is all we need to program to fetch an email:

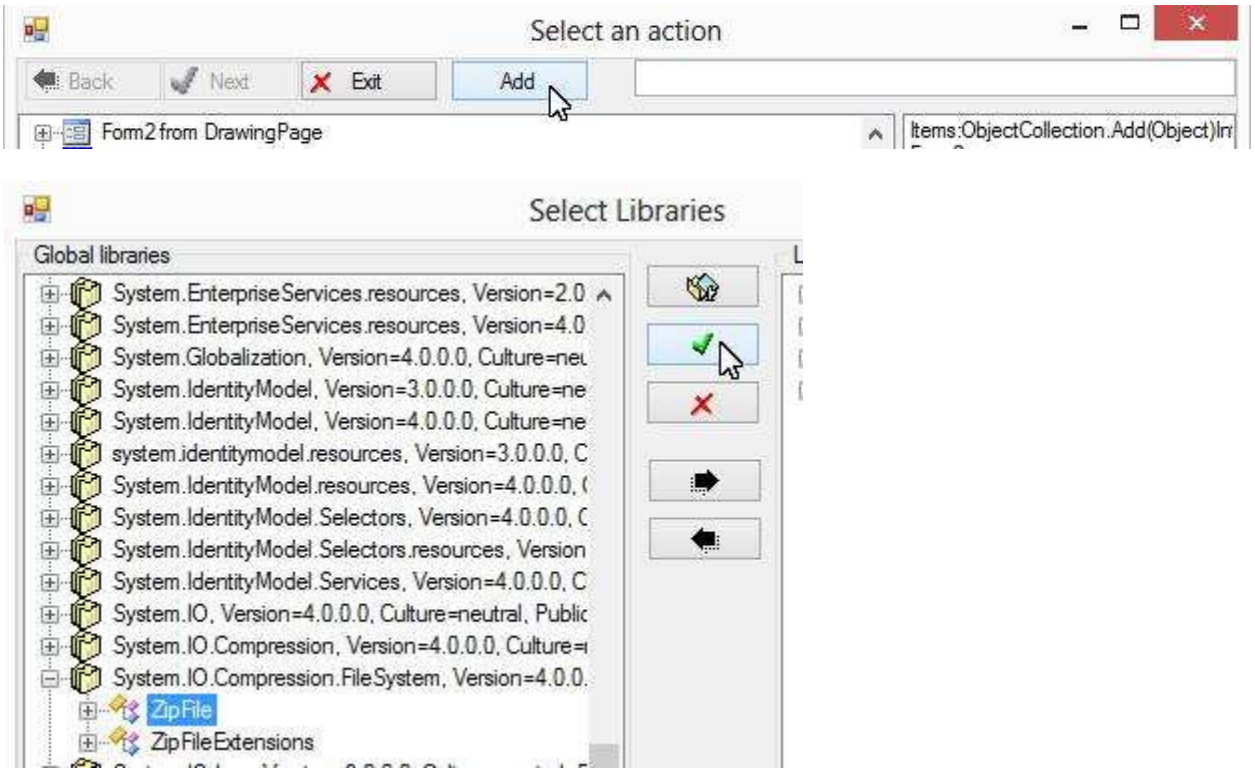


Unzip a file

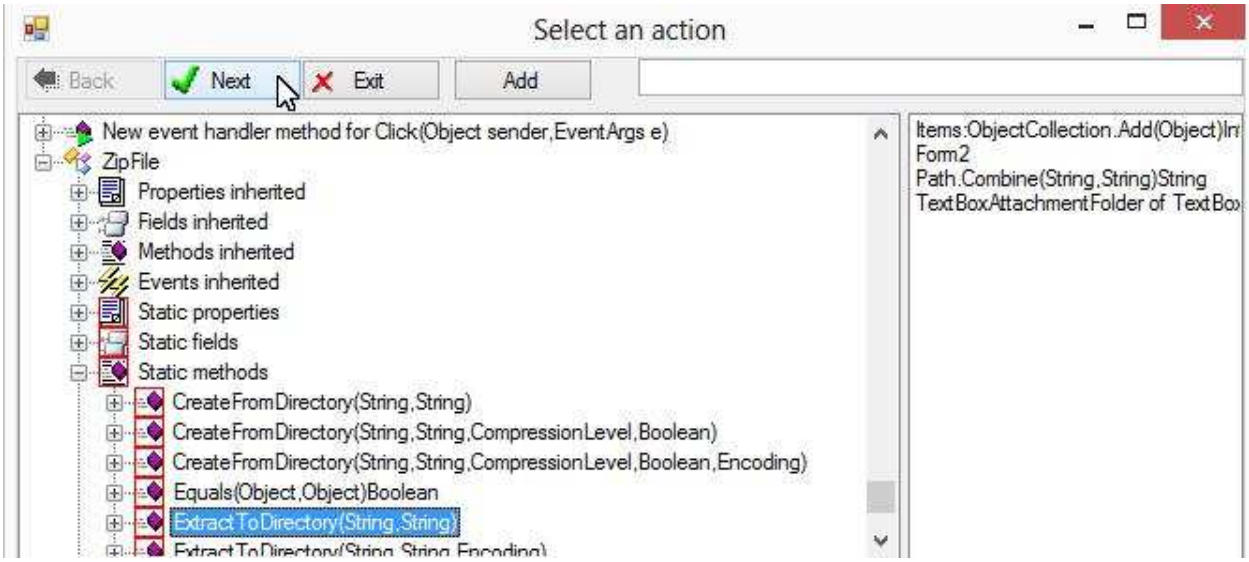
In some cases you may want to automatically unzip an attachment file which is a zip file. Microsoft .Net Framework 4.5 provides a ZipFile class to do it. Let's use an example to show how to use ZipFile class. Suppose we use a button to invoke an unzip action:



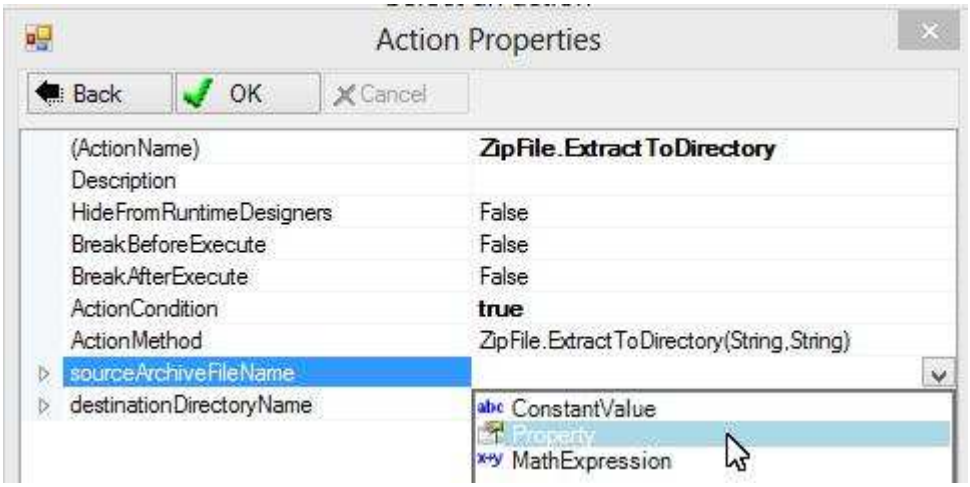
Locate the ZipFile class:

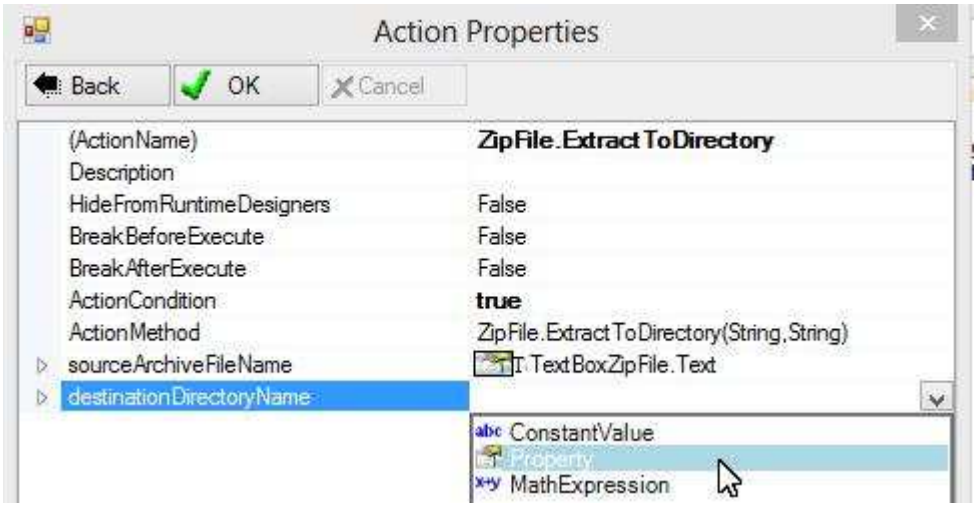


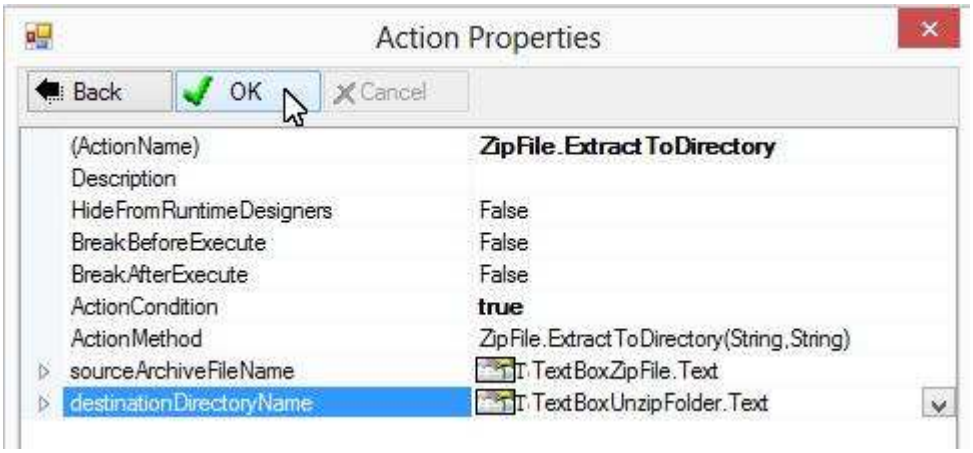
Select an ExtractToDirectory method of the ZipFile class:



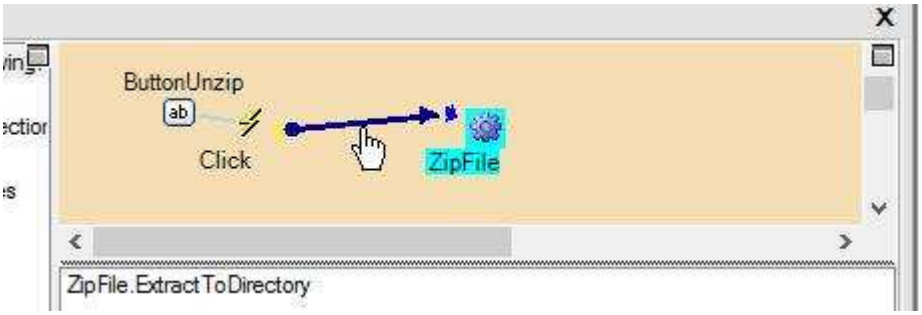
Note that since ExtractToDirectory is a static method, we do not need to create an instance of ZipFile class in order to use it. We provide zip file path and a folder to hold unzipped files. In this sample, both values are from text boxes:







The action is created and assigned to the button:



Feedback

Please send your feedback to support@limnor.com