Handle Large Number of Web Elements

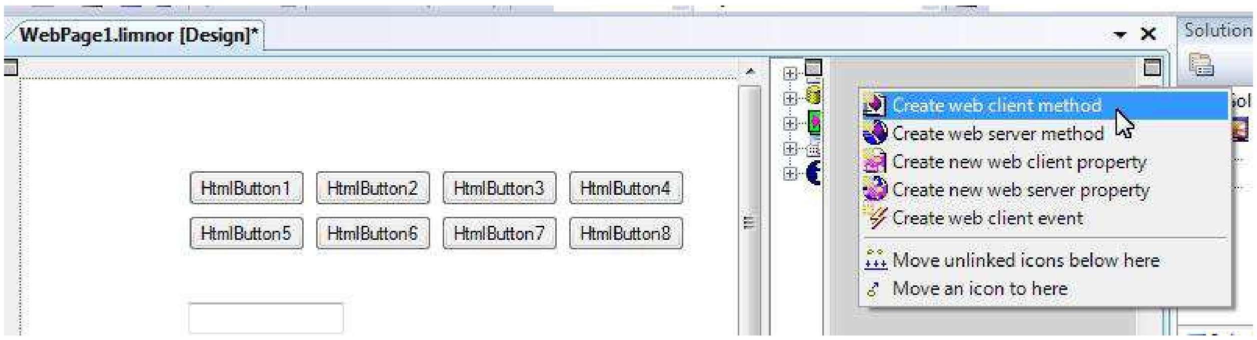
Contents

Operate on Many Elements

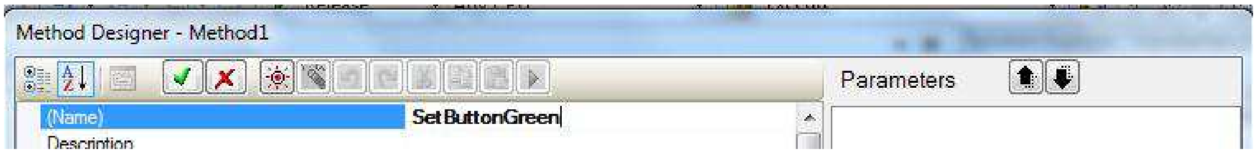
Suppose we have many buttons on a web page. Suppose we want to set the background color of all the buttons to green. If we create one action to do it for each button then we have to create many actions. It will be tedious and likely to make mistakes. A better approach is to just use one action to act on all the buttons.

Every web element has a "getElementsByTagName" method which returns an array of child elements. We may use a "loop" action to perform a set of actions to all the array items. For how to handle arrays and collections, see http://www.limnor.com/support/UseArrayAndCollections.pdf

We may use the technique presented in http://www.limnor.com/support/UseArrayAndCollections.pdf to set all buttons' background color to green. We'll create a method to do it:

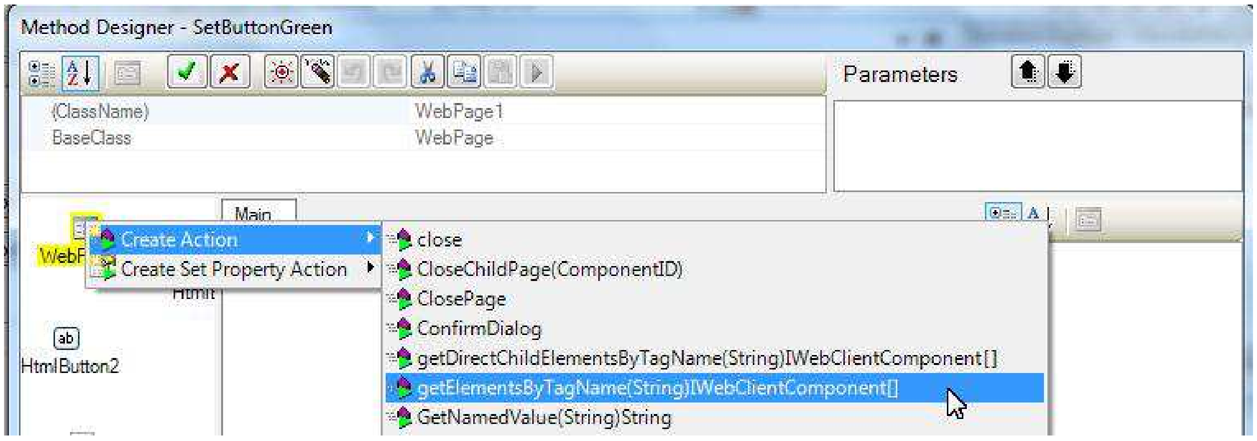


The Method Editor appears. We change its name to “SetButtonsGreen”:

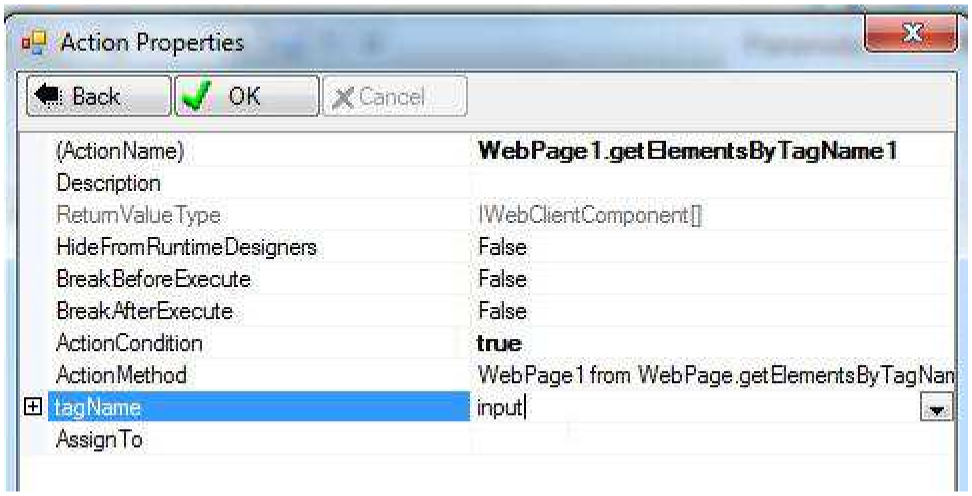


Collect elements into an array

Create a “getElementsByTagName” action on the web page because all elements are child elements of the web page:

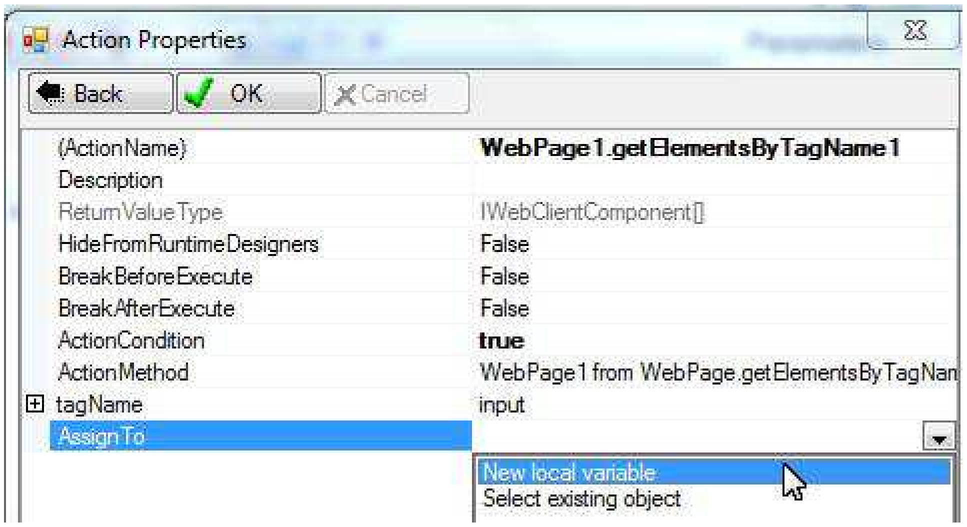

Method Designer - SetButtonGreen
Parameters
(ClassName) WebPage1
BaseClass WebPage
Main
Create Action
Create Set Property Action
HtmlButton2
close
CloseChildPage(ComponentID)
ClosePage
ConfirmDialog
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String

In this sample, the tag name for buttons is “input”. So set “tagName” to “input”:



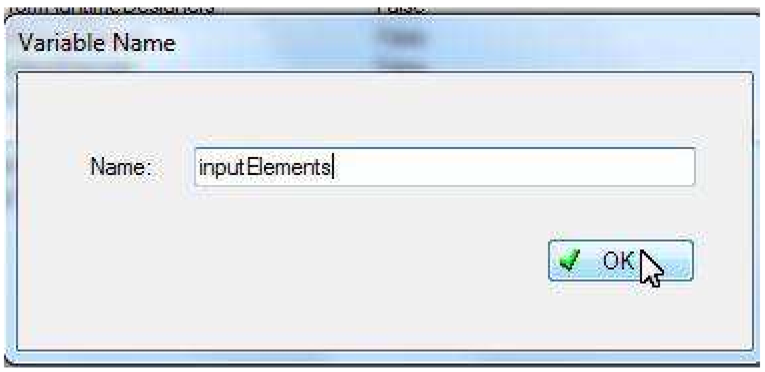
(ActionName)	WebPage1.getElementsByTagName1
Description	
ReturnValueType	IWebClientComponent[]
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	WebPage1 from WebPage.getElementsByTagNan
tagName	input
AssignTo	

Select “New local variable” to create a new array to get the action result:



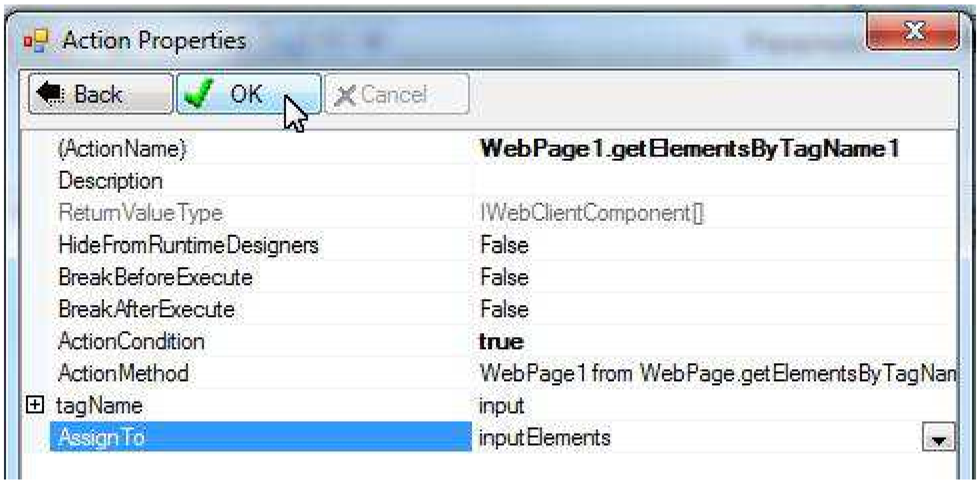
(ActionName)	WebPage1.getElementsByTagName1
Description	
ReturnValueType	IWebClientComponent[]
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	WebPage1 from WebPage.getElementsByTagNan
tagName	input
AssignTo	

Give the variable a name:



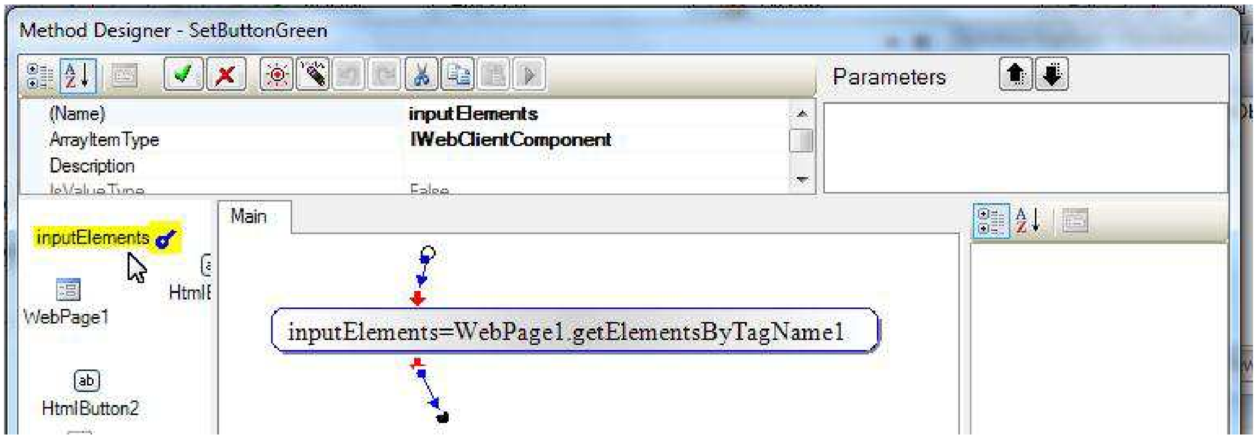
Name: inputElements

Click OK.



(ActionName)	WebPage1.getElementsByTagName1
Description	
ReturnValueType	IWebClientComponent[]
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	WebPage1 from WebPage.getElementsByTagNan
tagName	input
AssignTo	inputElements

The action appears in the Action Pane. The variable, inputElements, appears in the Variable Pane.



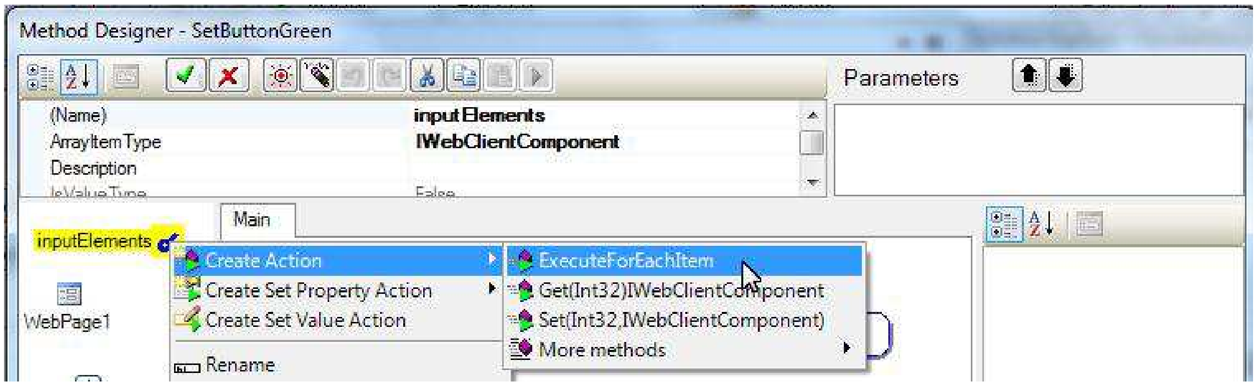
(Name)	inputElements
ArrayItemType	IWebClientComponent
Description	
IsValueType	False

inputElements=WebPage1.getElementsByTagName1

Go through array elements

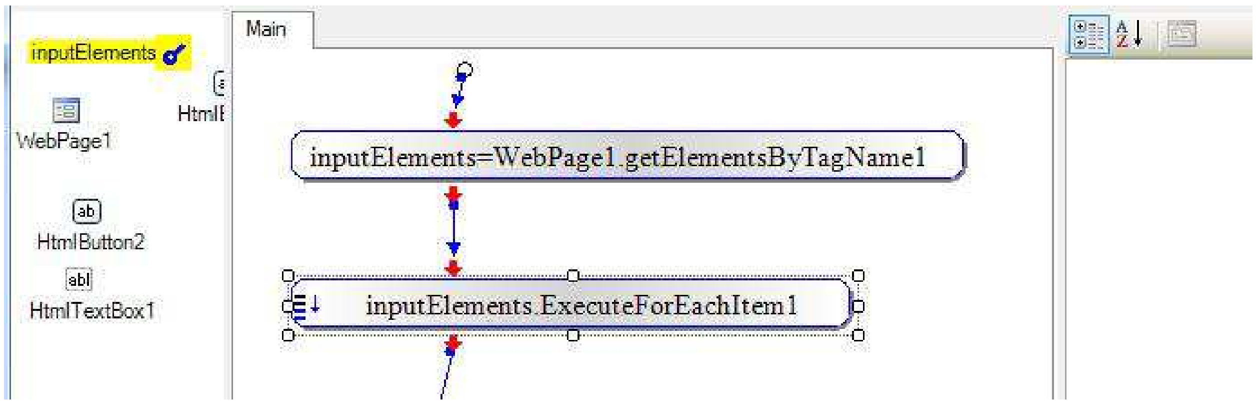
See http://www.limnor.com/support/UseArrayAndCollections.pdf for different techniques of going through array elements. Here we create a “for each” action to do it.

Right-click inputElements; choose

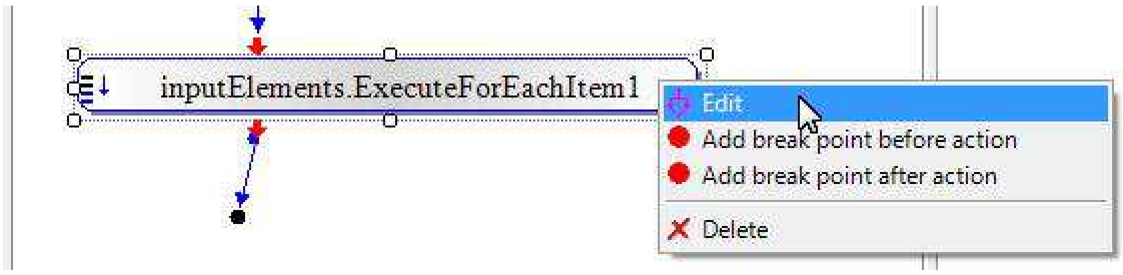

Method Designer - SetButtonGreen
Parameters
inputElements
WebPage1
Main
Create Action
Create Set Property Action
Create Set Value Action
Rename
ExecuteForEachItem
Get(Int32)IWebClientComponent
Set(Int32,IWebClientComponent)
More methods

(Name)	inputElements
ArrayItemType	IWebClientComponent
Description	
IsValueType	False

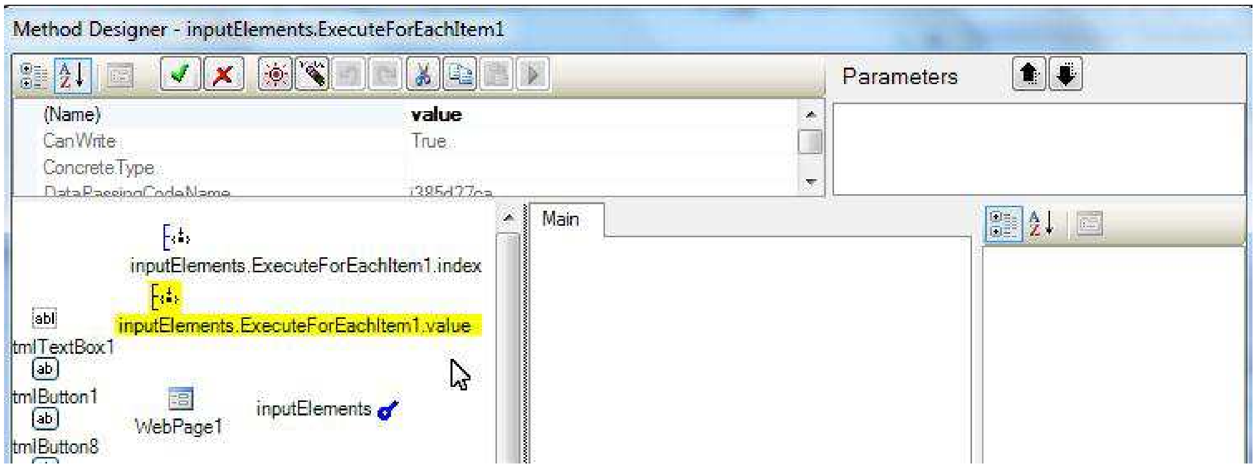
Link the “for each” action to the last action:


inputElements
WebPage1
HtmlButton2
HtmlTextBox1
Main
inputElements=WebPage1.getElementsByTagName1
inputElements.ExecuteForEachItem1

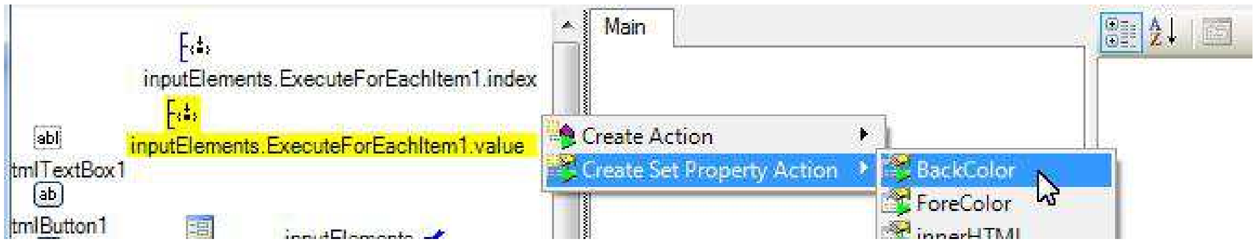
To apply actions to each array item, right-click the “for each” action; choose “Edit”:


inputElements.ExecuteForEachItem1
Edit
Add break point before action
Add break point after action
Delete

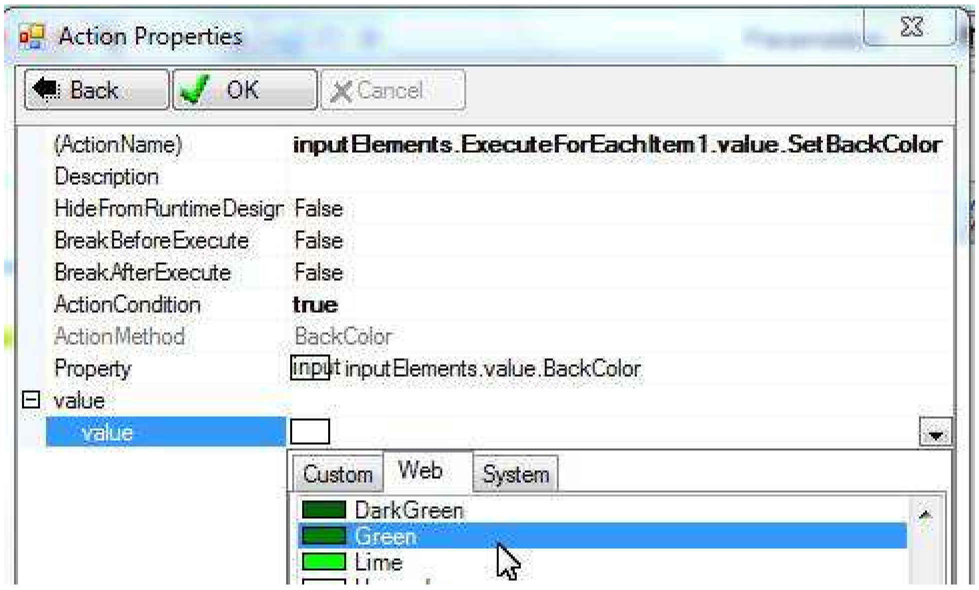
A new Method Editor appears. Note that a variable, inputElements.ExecuteForEachItem1.value, represents the array item to be processed:


Method Designer - inputElements.ExecuteForEachItem1
Parameters
(Name) value
CanWrite True
ConcreteType
Main
inputElements.ExecuteForEachItem1.index
inputElements.ExecuteForEachItem1.value
tmlTextBox1
tmlButton1
tmlButton8
WebPage1
inputElements

We want to the background color of the array item to green. Right-click the variable; choose “Create Set Property Action”; choose “BackColor” property:

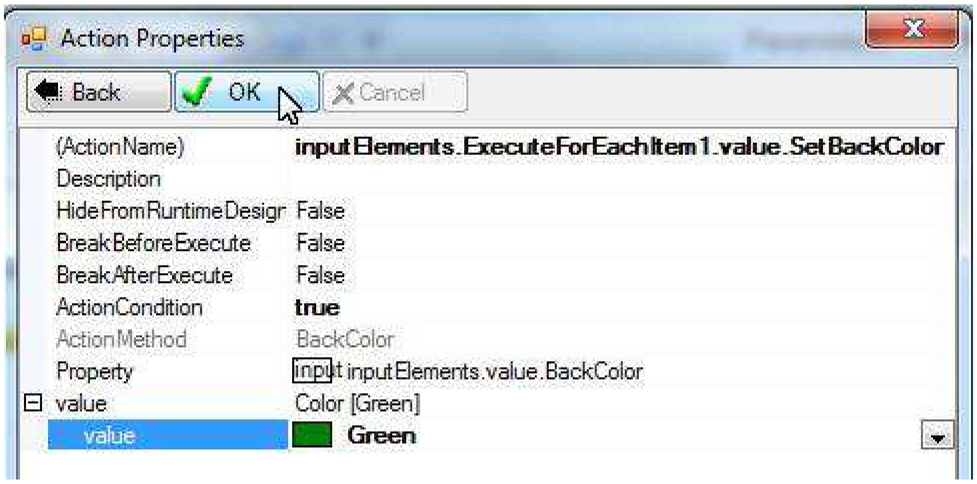

inputElements.ExecuteForEachItem1.index
inputElements.ExecuteForEachItem1.value
tmlTextBox1
tmlButton1
Main
Create Action
Create Set Property Action
BackColor
ForeColor

Set the "value" of the action to green:

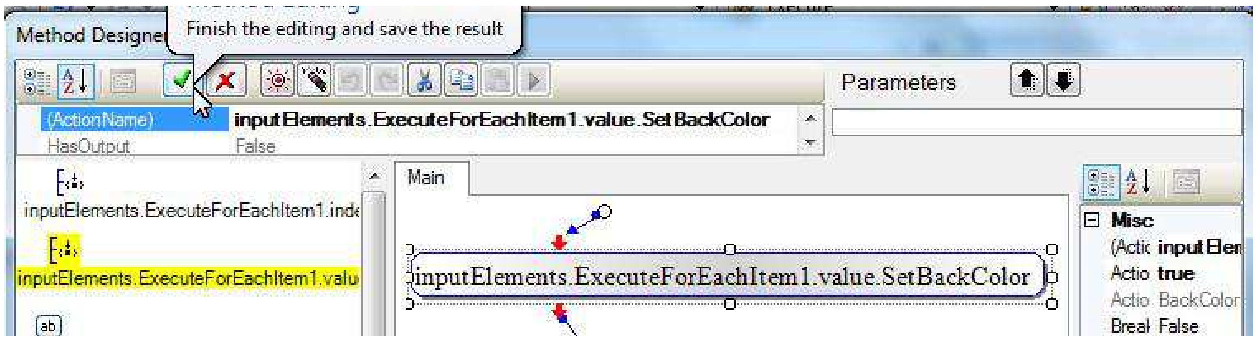


(ActionName)	**inputElements.ExecuteForEachItem1.value.SetBackColor**
Description	
HideFromRuntimeDesigr	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	BackColor
Property	input inputElements.value.BackColor
value	
value	

Click OK. The action appears in the Action Pane:

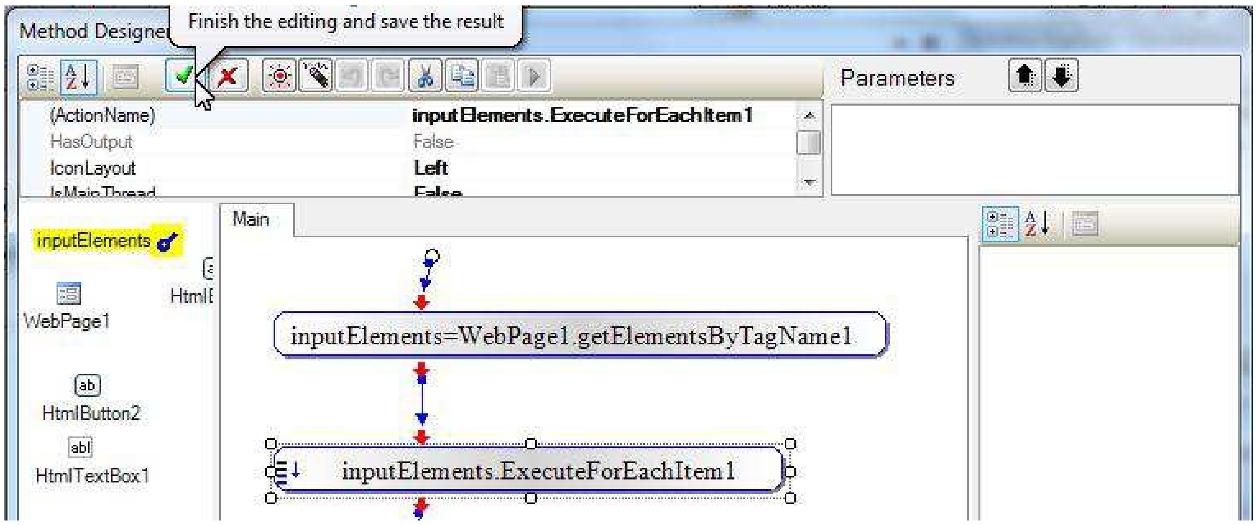


(ActionName)	**inputElements.ExecuteForEachItem1.value.SetBackColor**
Description	
HideFromRuntimeDesigr	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	BackColor
Property	input inputElements.value.BackColor
value	Color [Green]
value	🟩 **Green**

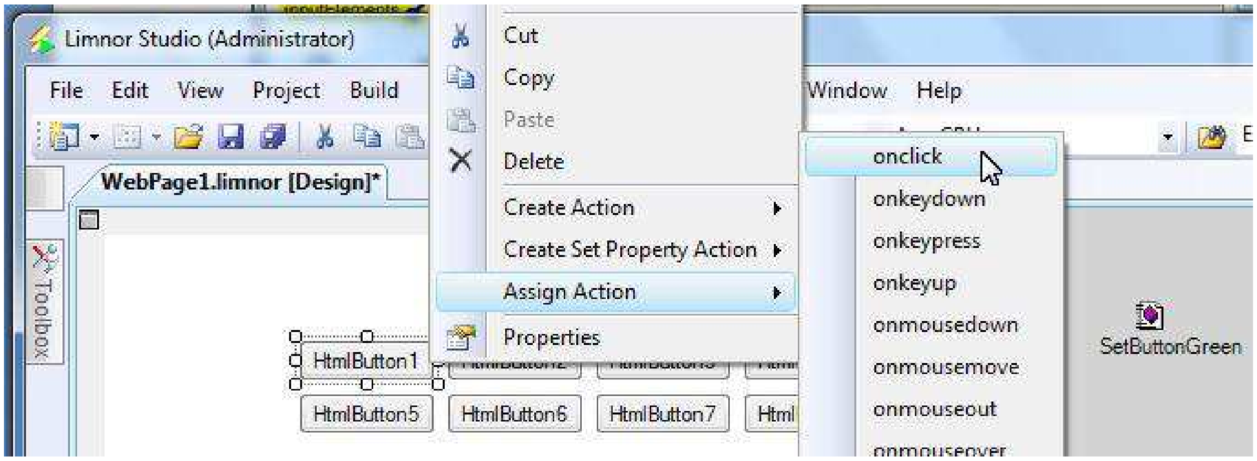

Method Designer
Finish the editing and save the result
Parameters
(ActionName)
inputElements.ExecuteForEachItem1.value.SetBackColor
HasOutput
False
inputElements.ExecuteForEachItem1.inde
inputElements.ExecuteForEachItem1.valu
Main
inputElements.ExecuteForEachItem1.value.SetBackColor
Misc
(Actic inputElem
Actio true
Actio BackColor
Breal False

Test

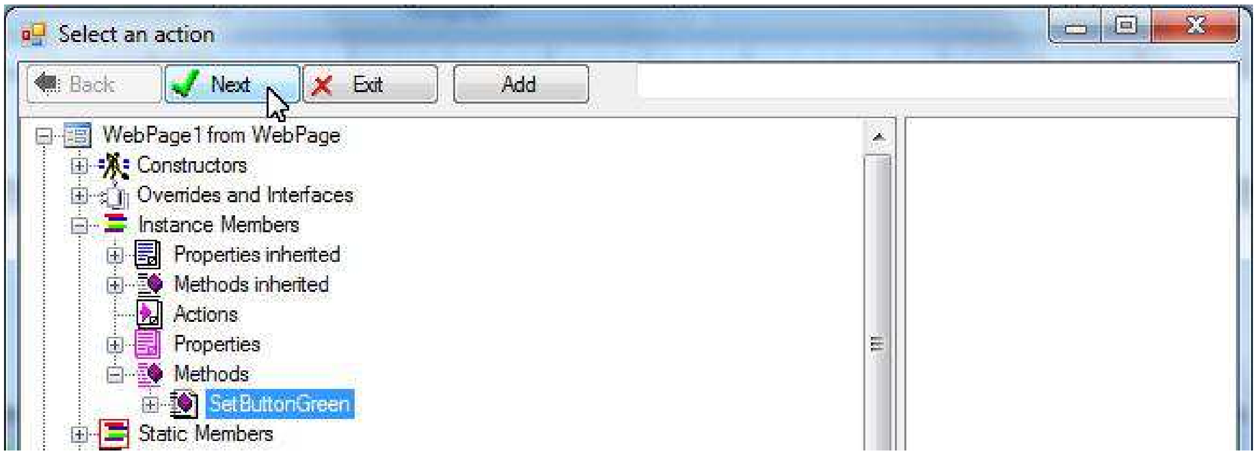
Let’s finish method editing and test our programming.


Method Designer
Finish the editing and save the result
Parameters
(ActionName)
inputElements.ExecuteForEachItem1
HasOutput
False
IconLayout
Left
IsMainThread
False
Main
inputElements
WebPage1
HtmlI
HtmlButton2
HtmlTextBox1
inputElements=WebPage1.getElementsByTagName1
inputElements.ExecuteForEachItem1

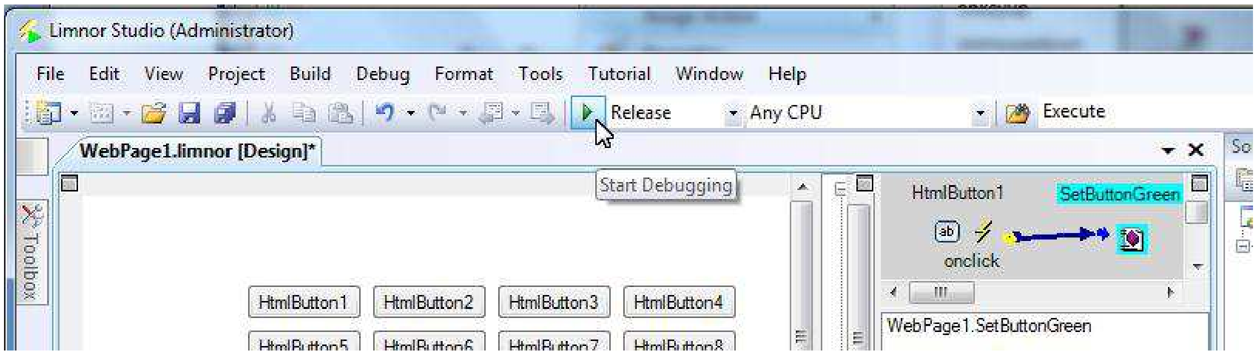
To test the above method, we execute it when clicking the first button:



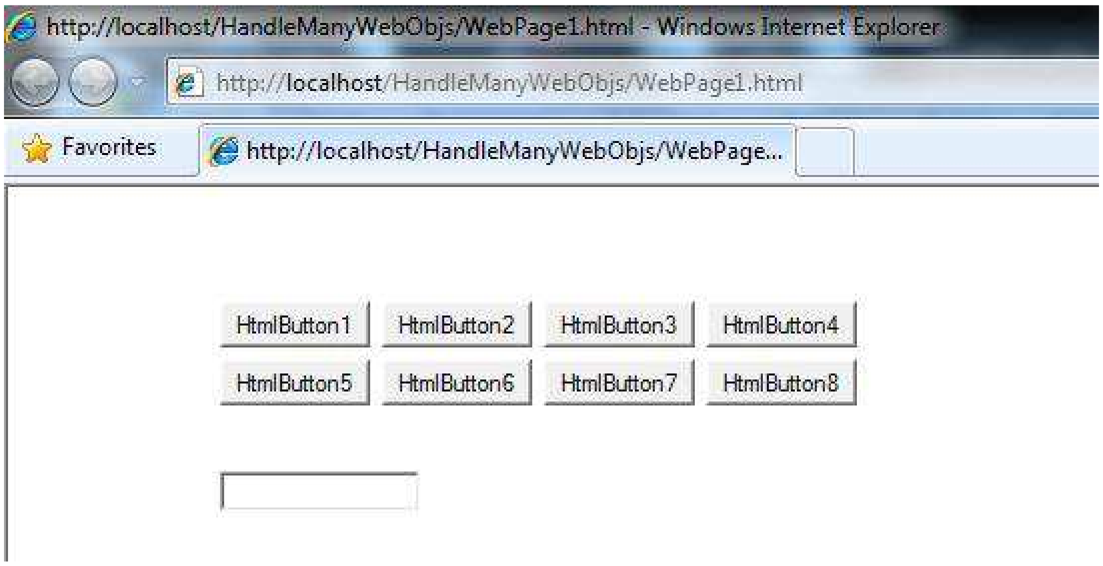
Select method SetButtonsGreen; click Next:



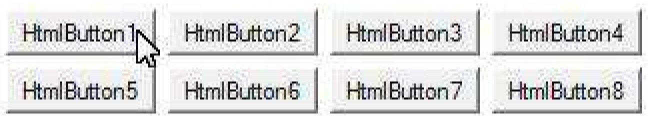
Click OK. An action is created and assigned to the button. Click Run button to test the web page:



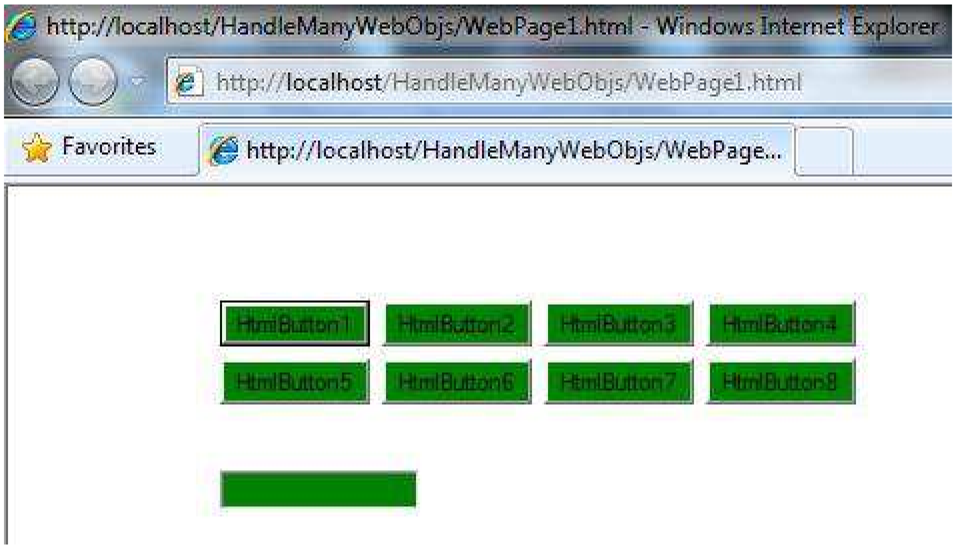
The page appears:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

Click the first button to execute the method we just created:



All buttons' background color changed to green as we expected:

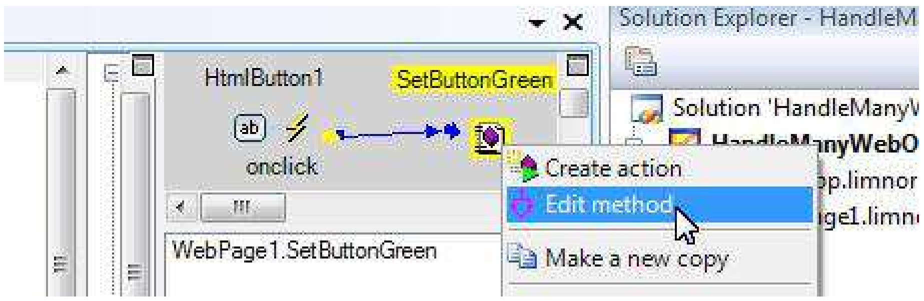

http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

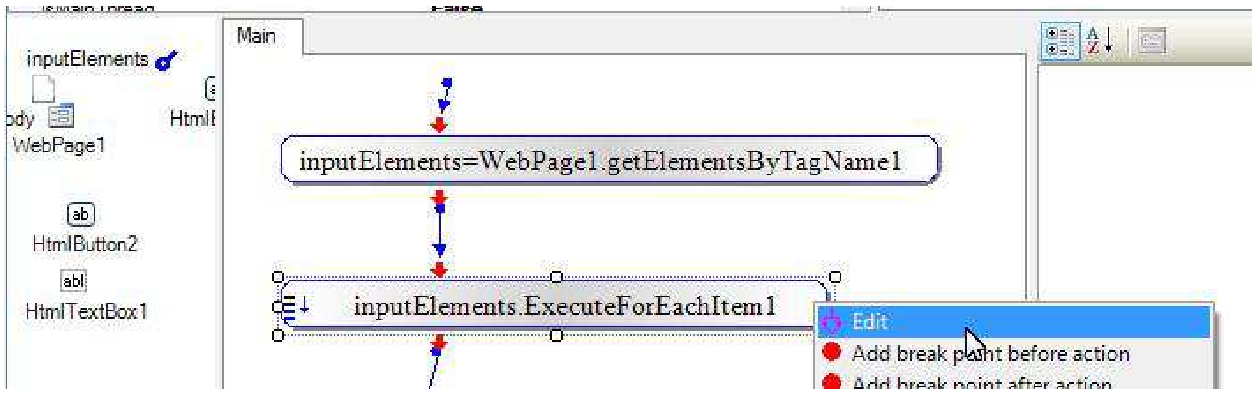
A bug and its solution

Note that clicking the first button not only changes background color of all the buttons, it also changes the background color of a text box. This is a bug of our method.

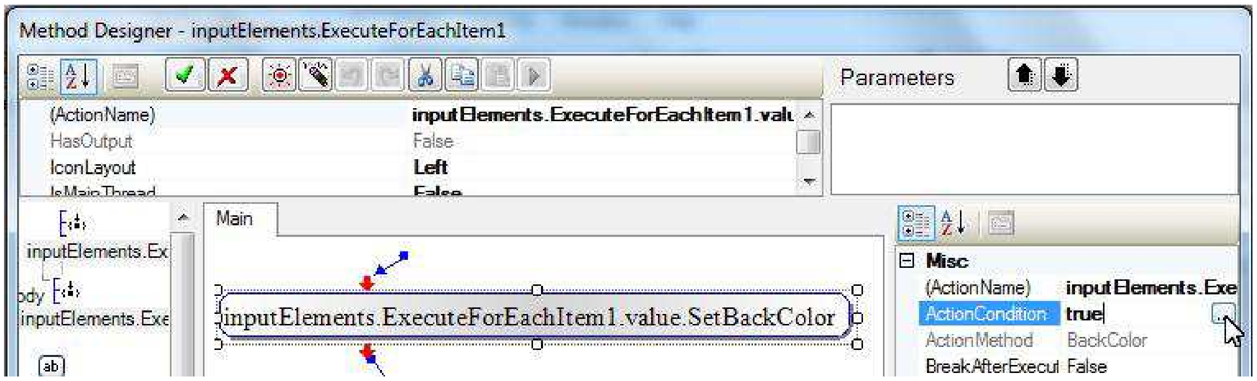
The bug occurs because the tag name for text boxes is also "input". So, our action of "getElementsByTagName" not only returns buttons, it also returns text boxes. To distinguish buttons from text boxes, we need to use "type" property. See http://www.w3schools.com/tags/tag_input.asp for "type" definitions. For a button, its "type" property is "button".

Let's add a condition to the action of setting background color so that it only execute if "type" is "button".





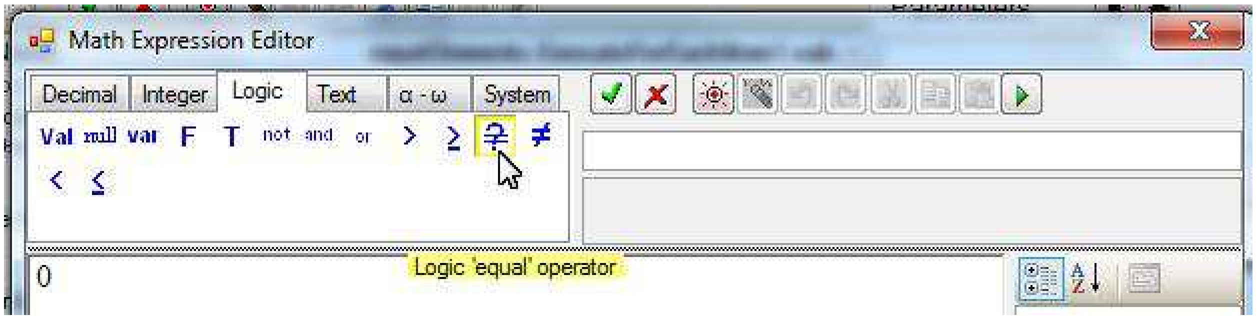
Main
inputElements
WebPage1
HtmlButton2
HtmlTextBox1
inputElements=WebPage1.getElementsByTagName1
inputElements.ExecuteForEachItem1
Edit
Add break point before action
Add break point after action

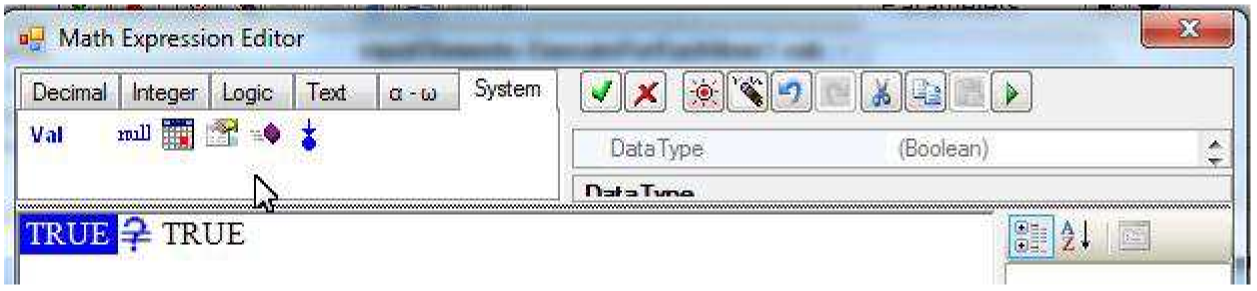


Method Designer - inputElements.ExecuteForEachItem1
Parameters
Main
inputElements.ExecuteForEachItem1.value.SetBackColor
Misc

(ActionName)	inputElements.ExecuteForEachItem1.valu
HasOutput	False
IconLayout	Left

(ActionName)	inputElements.Exe
ActionCondition	true
ActionMethod	BackColor
BreakAfterExecut	False



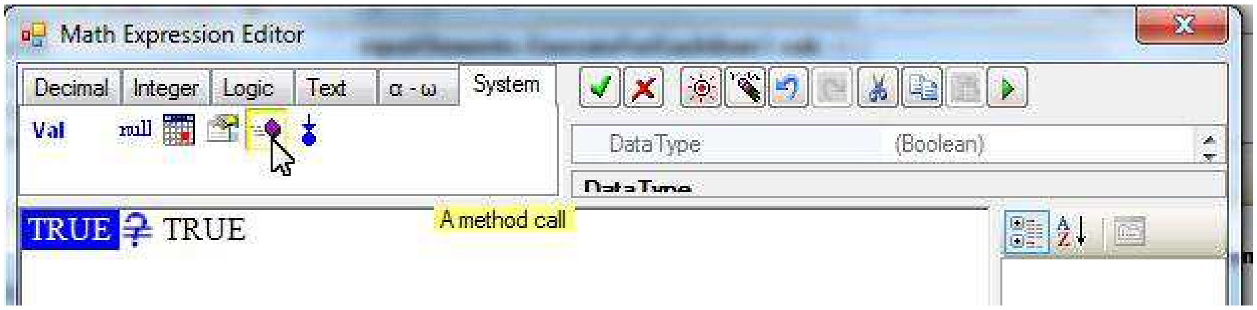


DataType	(Boolean)

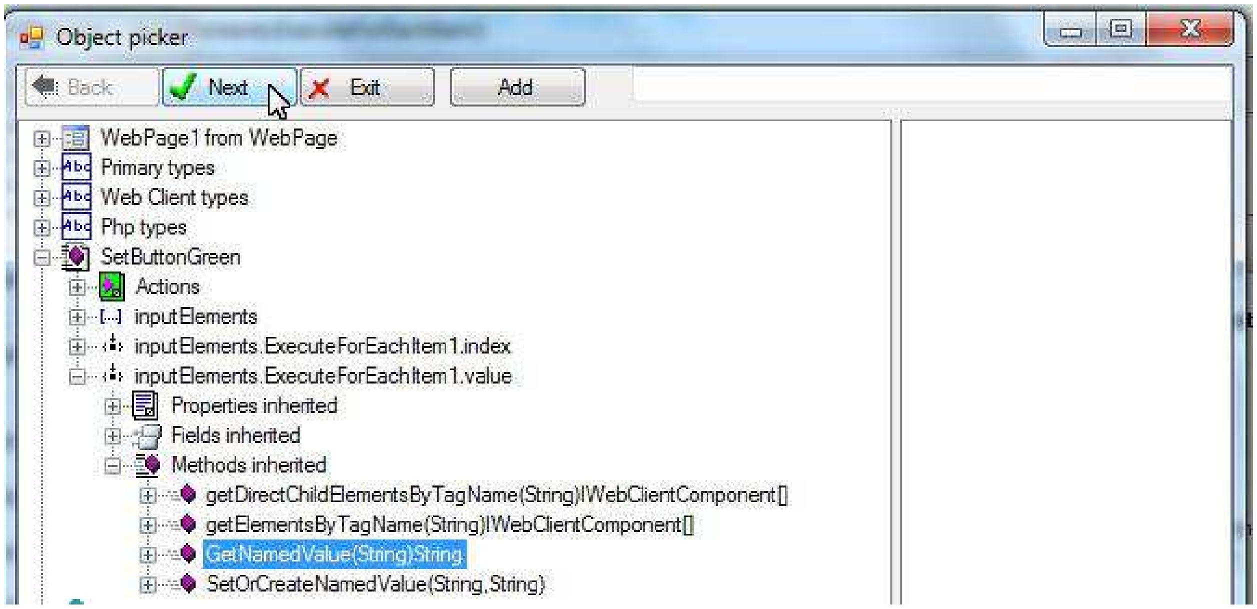
If we click the Property to get the “type” property of the array item then we will discover that the array item does not show a “type” property. This is because the array item represents a generic HTML element. Some kinds of HTML element do not have a “type” property.

All HTML elements have a “**GetNamedValue**” method. We may use it to get any property of any elements.

Click the Method icon and select GetNamedValue method of the array item:

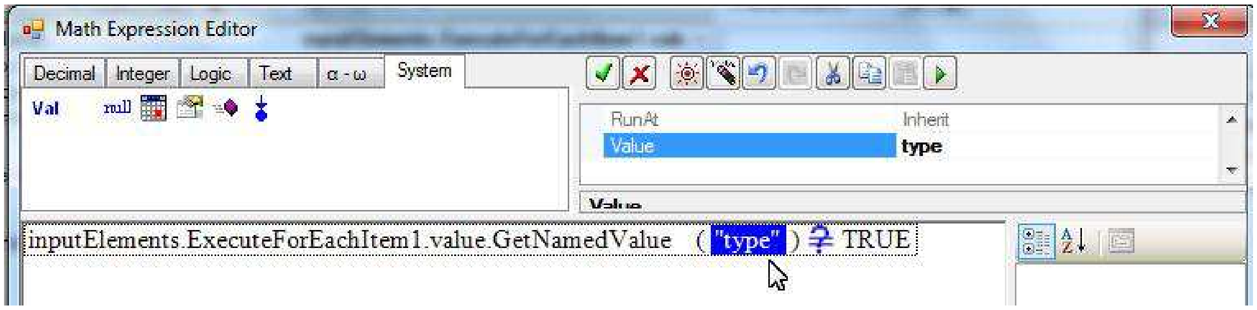


DataType (Boolean)



```
WebPage1 from WebPage
Primary types
Web Client types
Php types
SetButtonGreen
  Actions
  inputElements
  inputElements.ExecuteForEachItem1.index
  inputElements.ExecuteForEachItem1.value
    Properties inherited
    Fields inherited
    Methods inherited
      getDirectChildElementsByTagName(String)IWebClientComponent[]
      getElementsByTagName(String)IWebClientComponent[]
      GetNamedValue(String)String
      SetOrCreateNamedValue(String,String)
```

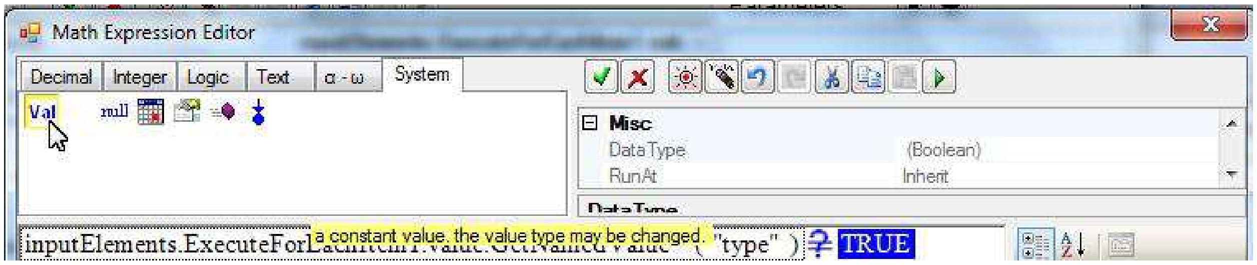
Type “type” for the method parameter:



RunAt Inherit
Value type

```
inputElements.ExecuteForEachItem1.value.GetNamedValue ( "type" ) TRUE
```

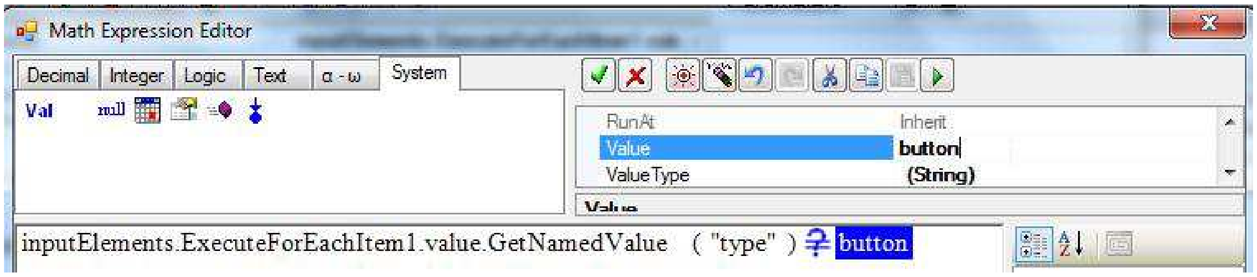
Select the value icon, “Val”, for the second expression item:



DataType (Boolean)
RunAt Inherit

inputElements.ExecuteForEachItem1.value.GetNamedValue ("type") TRUE

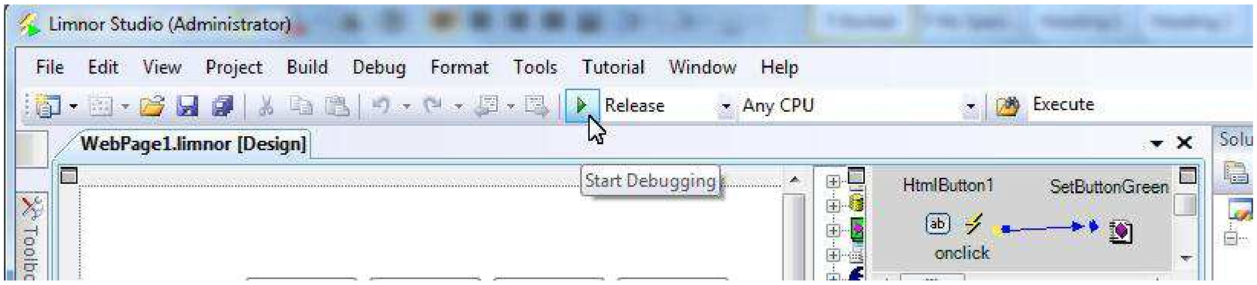
Type “button” for the value:



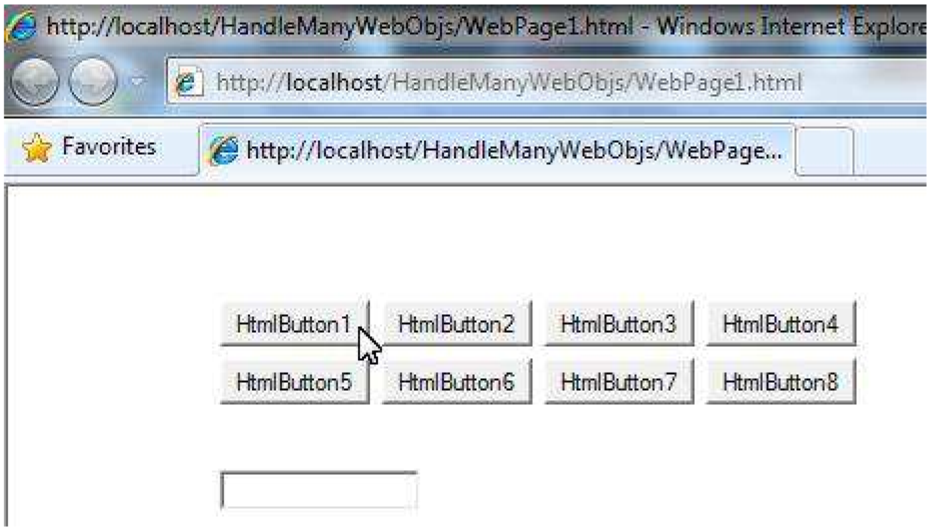
RunAt Inherit
Value button
ValueType (String)

inputElements.ExecuteForEachItem1.value.GetNamedValue ("type") button

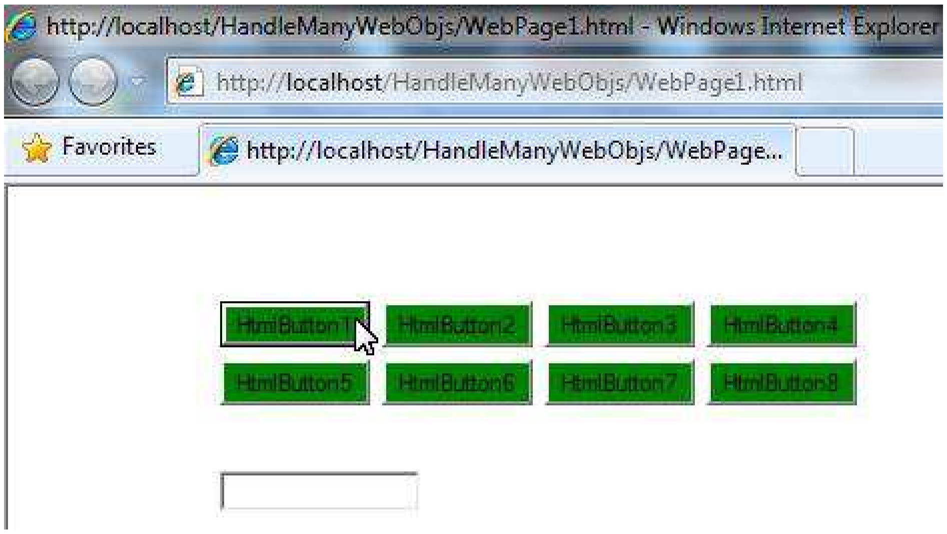
Finish the editing and test it again:



The web page appears. Click the first button:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

This time, all buttons' background color changes to green. The background of the text box is not changed. We fixed the bug.


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

Notes

From the above sample, we can see that using a "loop" action, one set of actions can be applied to many objects. Actually we may freely add as many buttons as we want and the method we created above will work without any changes.

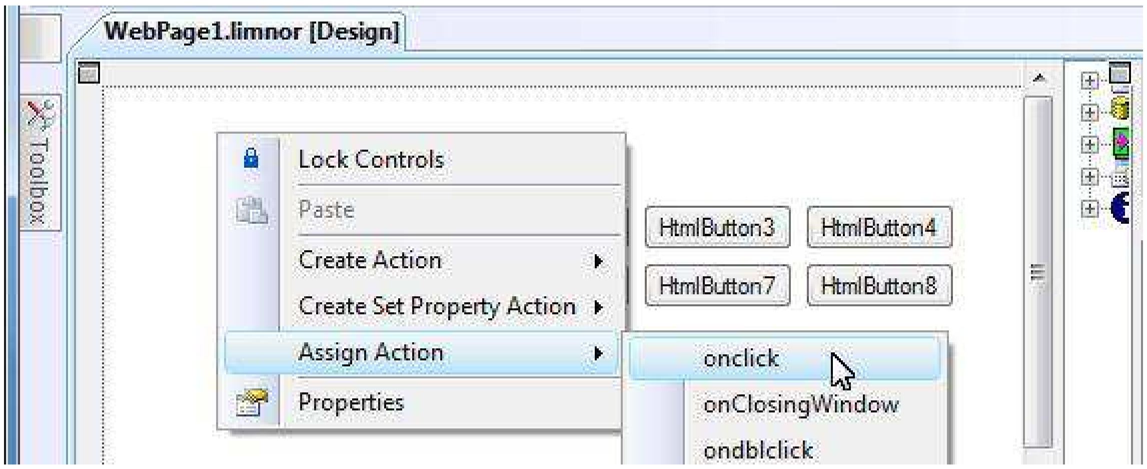
Handle events generated from many objects

Suppose we want to do such a programming: click a button, if the button's background color is green then change the button's background color to yellow; otherwise set the button's background color to green.

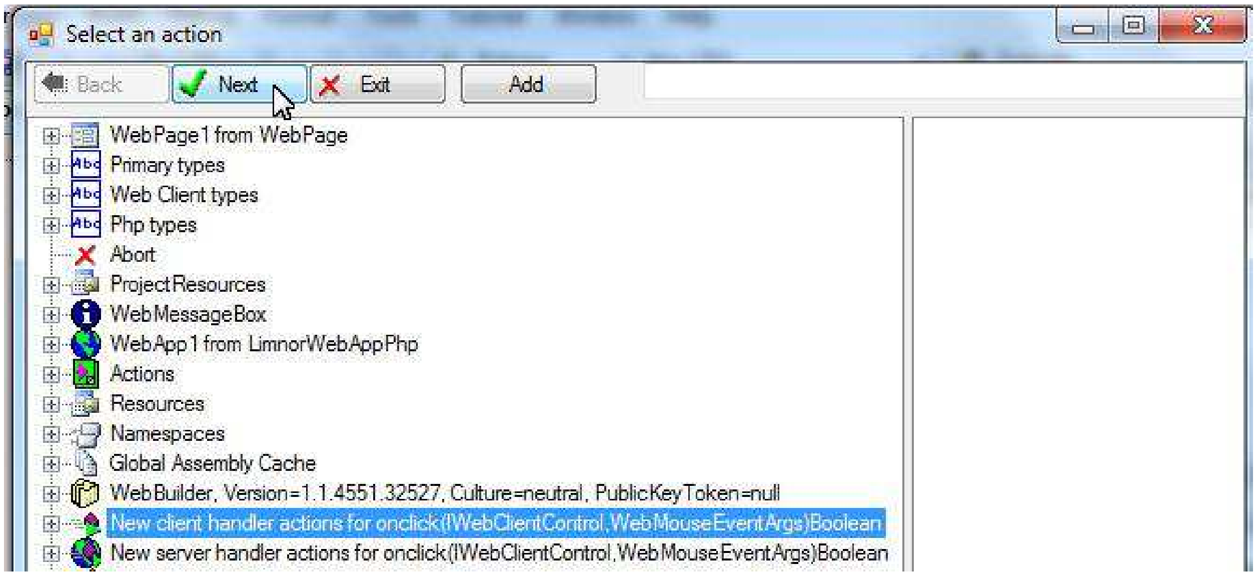
If we handle each button individually then we need to handle many events and create many actions. If we have many buttons to handle then the programming work is tedious and very likely to generate bugs.

Instead of handling each button, we may handle the element which is the parent of all the elements we want to handle. Keyboard and mouse events generated from one element can be handled in all levels of parents. See http://www.limnor.com/support/WebEventBubbling.pdf for details of using event bubbling.

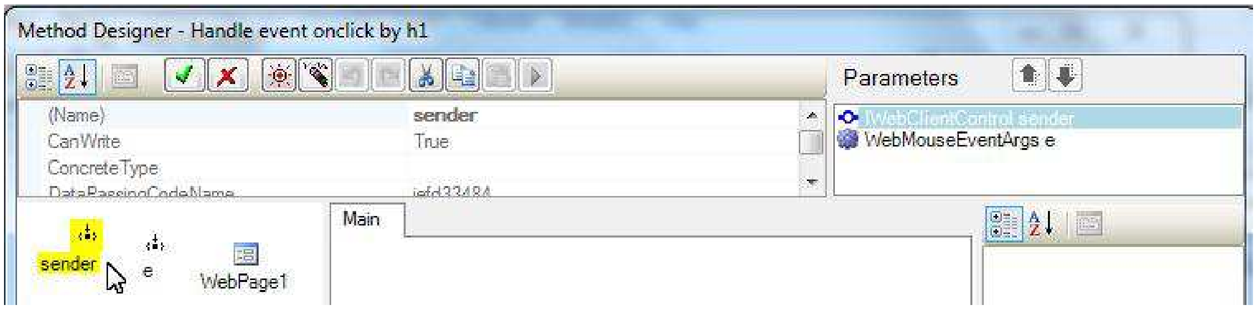
In this sample, the web page is the parent for all the buttons (actually a web page is a parent for all elements on the page). Instead of handling the click event of each button, we handle the click event of the page. The "sender" parameter is the element generating the event. Since we need to access the "sender" parameter, we need to create an event handling method. Right-click the web page; choose "Assign Action"; choose "onclick" event:



Select “New client event handler actions for onclick”:


Select an action
Back
Next
Exit
Add
WebPage1 from WebPage
Primary types
Web Client types
Php types
Abort
ProjectResources
WebMessageBox
WebApp1 from LimnorWebAppPhp
Actions
Resources
Namespaces
Global Assembly Cache
WebBuilder, Version=1.1.4551.32527, Culture=neutral, PublicKeyToken=null
New client handler actions for onclick(IWebClientControl,WebMouseEventArgs)Boolean
New server handler actions for onclick(IWebClientControl,WebMouseEventArgs)Boolean

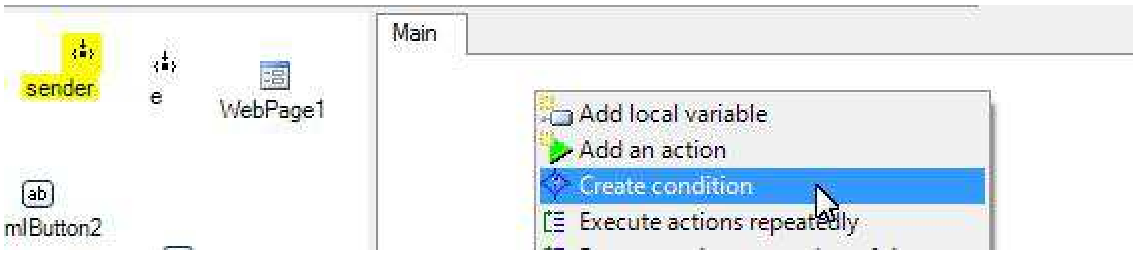
A Method Editor appears. Note that the “sender” parameter is the element generating the event.

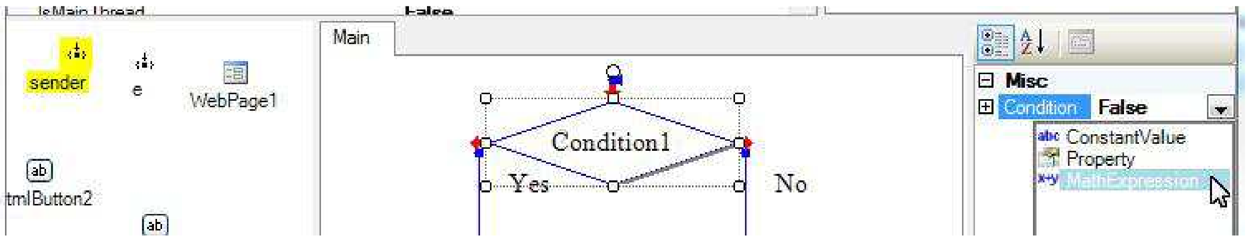


(Name)	sender
CanWrite	True
ConcreteType	
DataPassingCodeName	iefd33484

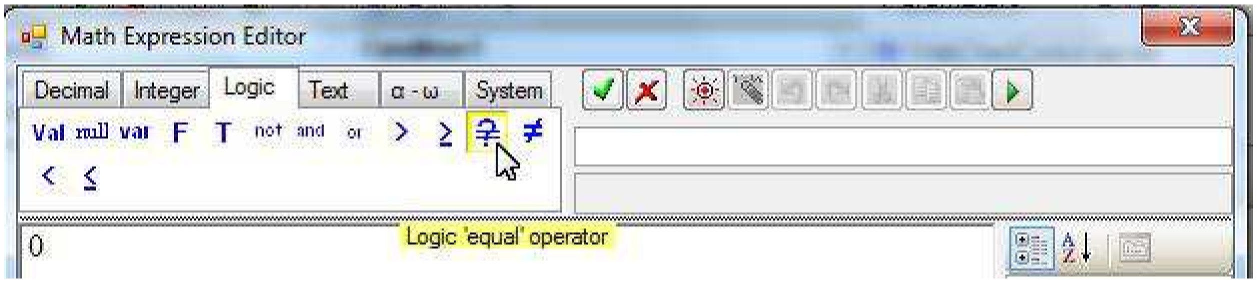
Condition for button

This event handler method will be executed when the page visitor clicks on any element on the page. We only want to handle button event. We already know that if the tagName is “input” and type is “button” then the sender is a button. Add a Condition action to check these conditions:

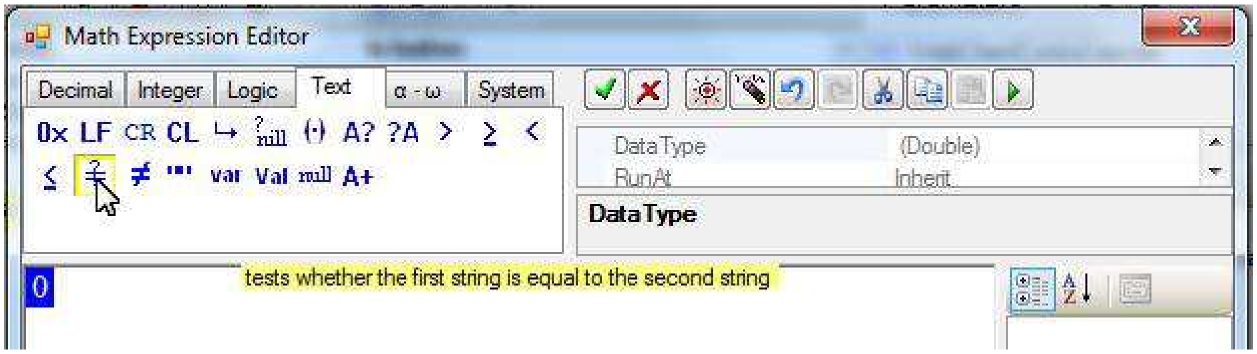




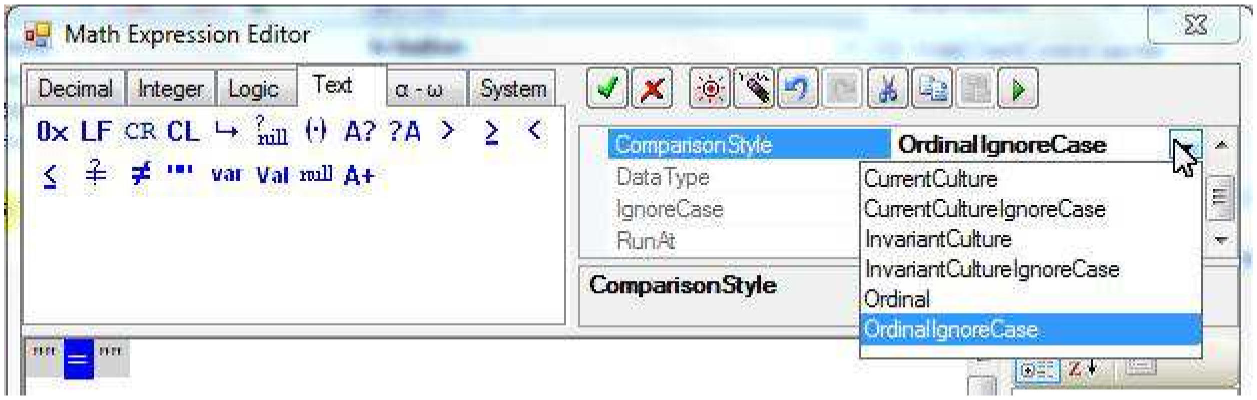
We want to check whether the tagName of the sender is “input”. Note that using =?= from Logic tab is **wrong**:



It is because that the tagName is case-insensitive. Different web browsers may return different cases. The above comparison operator is case-sensitive. We must use the =?= operator from the Text tab:



Note that this =?= operator allows you to specify comparison options:

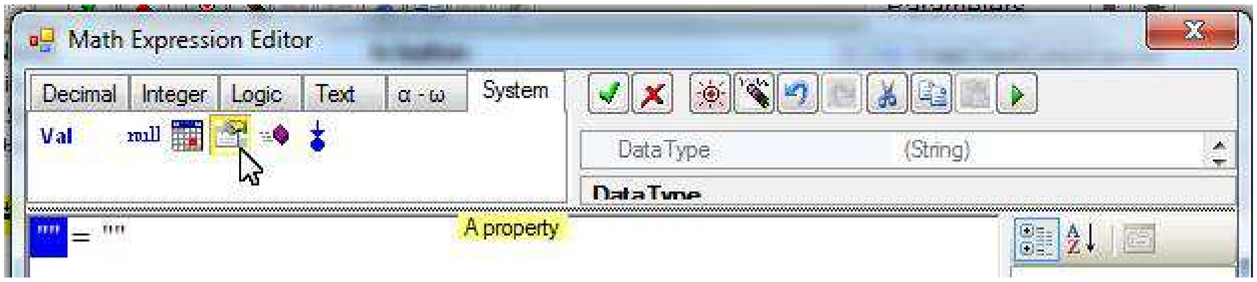


ComparisonStyle	OrdinalIgnoreCase
DataType	
IgnoreCase	
RunAt	

CurrentCulture
CurrentCultureIgnoreCase
InvariantCulture
InvariantCultureIgnoreCase
Ordinal
OrdinalIgnoreCase

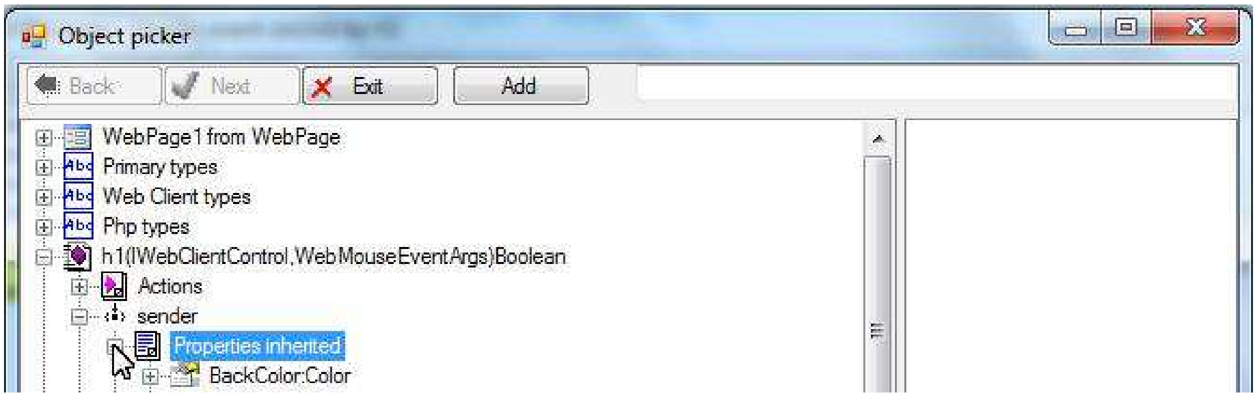
ComparisonStyle

In our case, we should use “OrdinalIgnoreCase”.

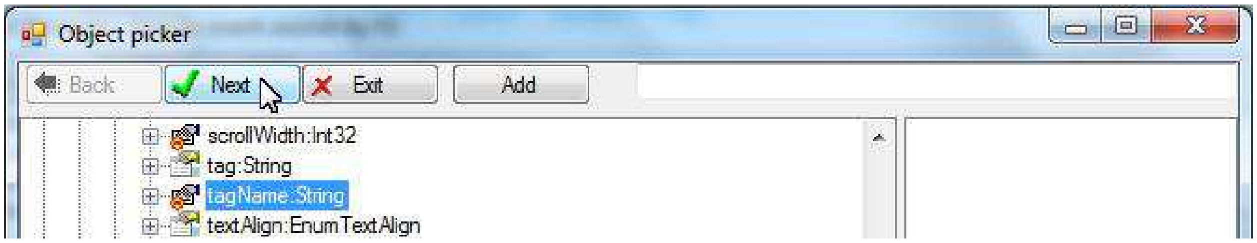


DataType	(String)

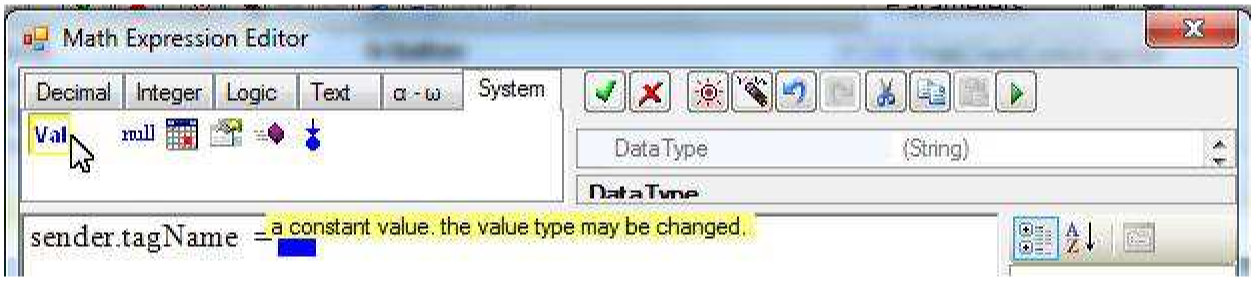
DataType



WebPage1 from WebPage
Primary types
Web Client types
Php types
h1(IWebClientControl,WebMouseEventArgs)Boolean
Actions
sender
Properties inherited
BackColor:Color

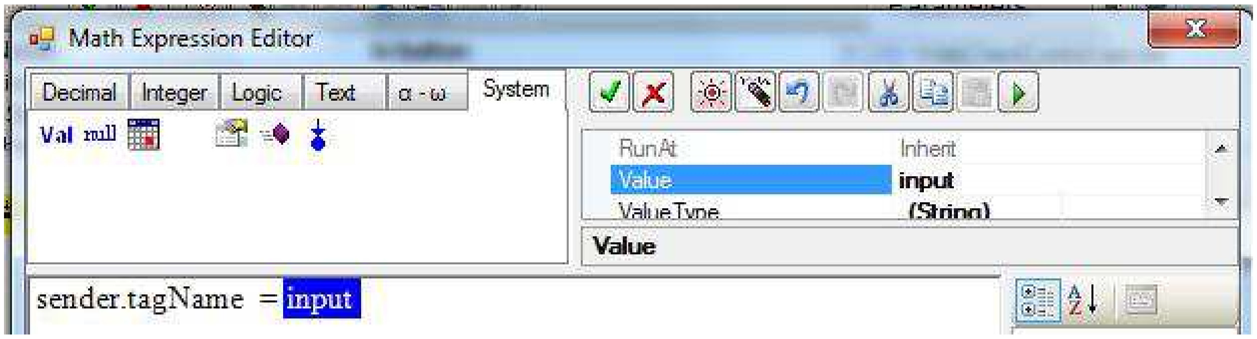


scrollWidth:Int32
tag:String
tagName:String
textAlign:EnumTextAlign



DataType (String)

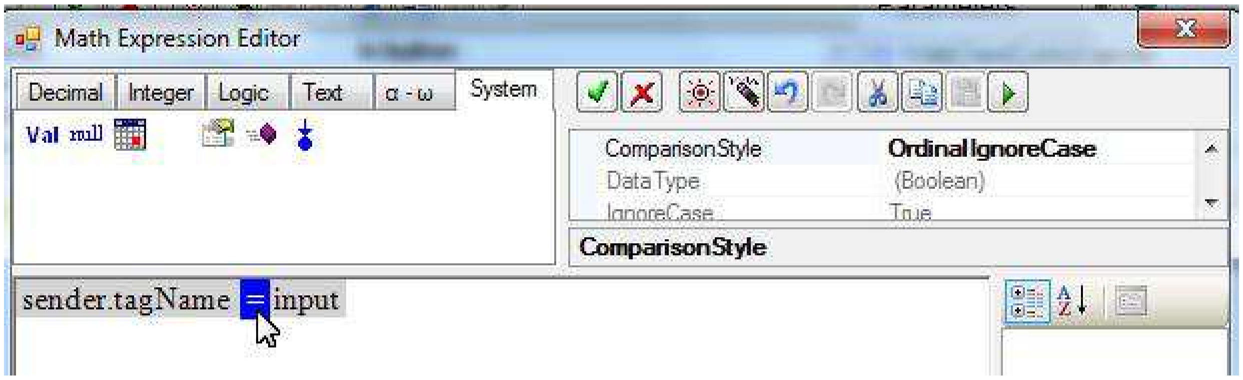
sender.tagName = a constant value. the value type may be changed.



RunAt	Inherit
Value	input
ValueType	(String)

Value

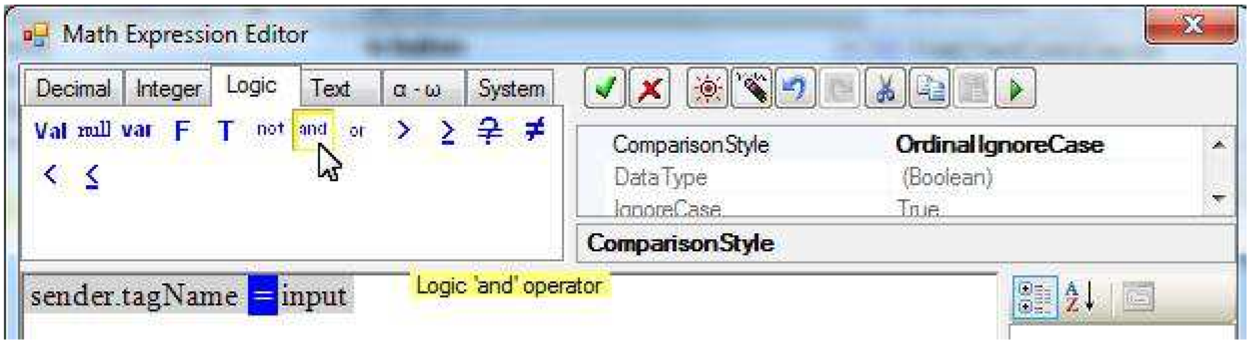
sender.tagName = input



ComparisonStyle	OrdinalIgnoreCase
DataType	(Boolean)
IgnoreCase	True

ComparisonStyle

sender.tagName = input

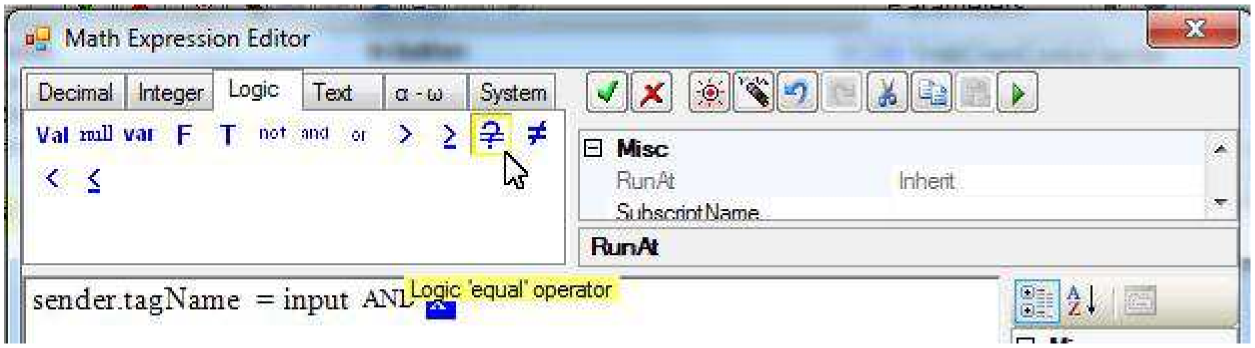


ComparisonStyle	OrdinalIgnoreCase
DataType	(Boolean)
IgnoreCase	True

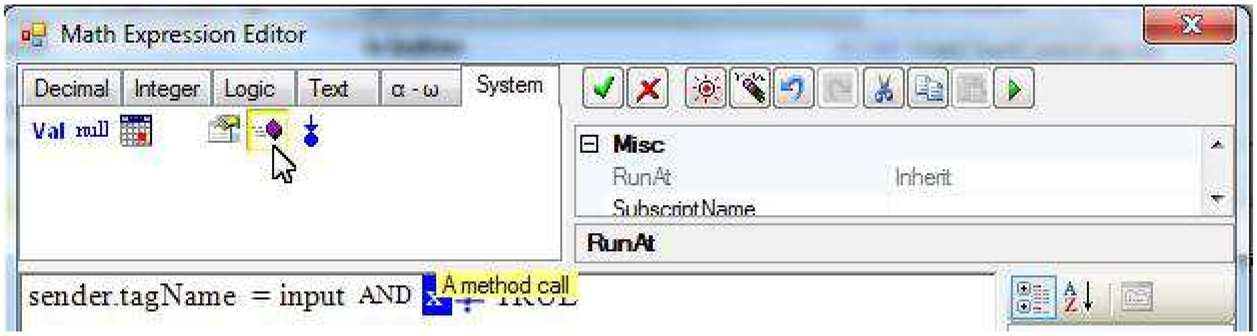
ComparisonStyle

sender.tagName = input Logic 'and' operator

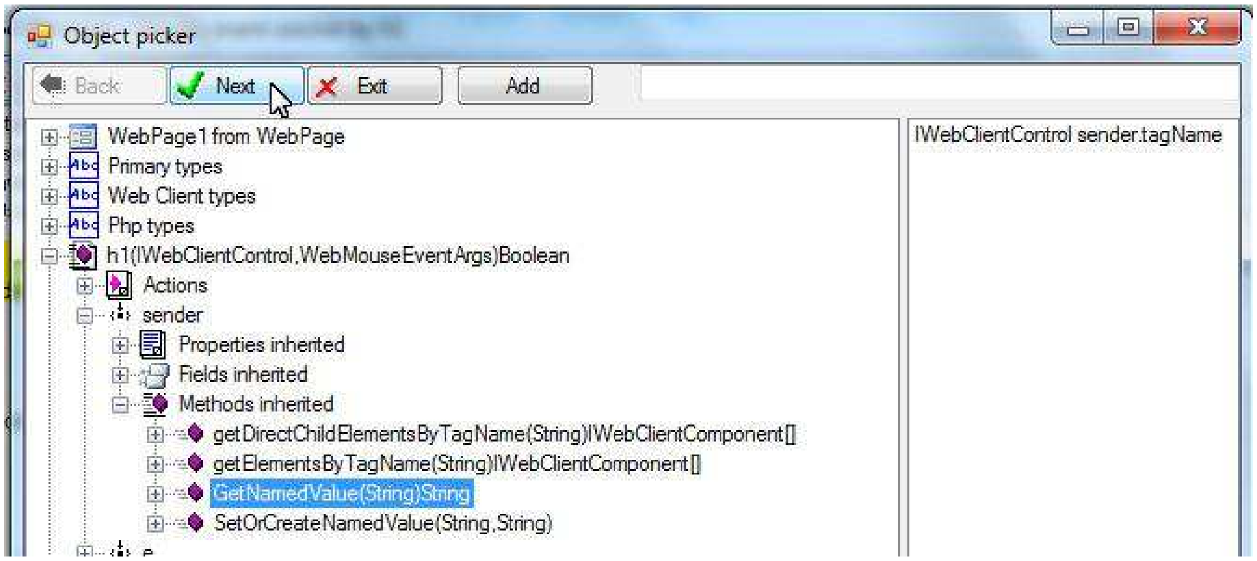
For checking sender.type = “button” we may use =?= from the Logic tab because this checking is case-sensitive:



sender.tagName = input AND x

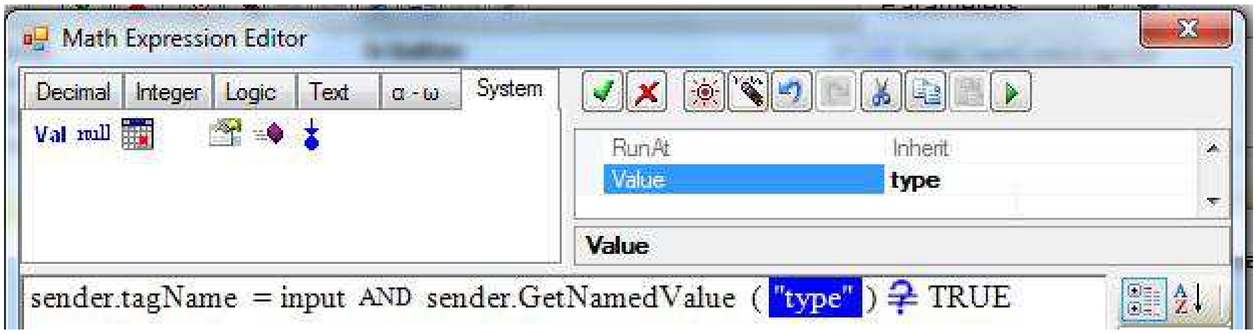


sender.tagName = input AND x ≟ TRUE



```
WebPage1 from WebPage
Primary types
Web Client types
Php types
h1(IWebClientControl,WebMouseEventArgs)Boolean
  Actions
  sender
    Properties inherited
    Fields inherited
    Methods inherited
      getDirectChildElementsByTagName(String)IWebClientComponent[]
      getElementsByTagName(String)IWebClientComponent[]
      GetNamedValue(String)String
      SetOrCreateNamedValue(String,String)
  e
```

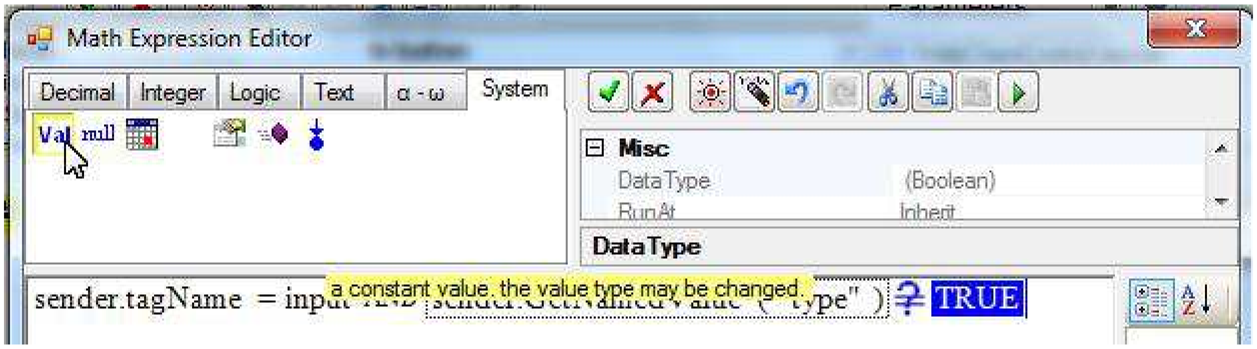
IWebClientControl sender.tagName



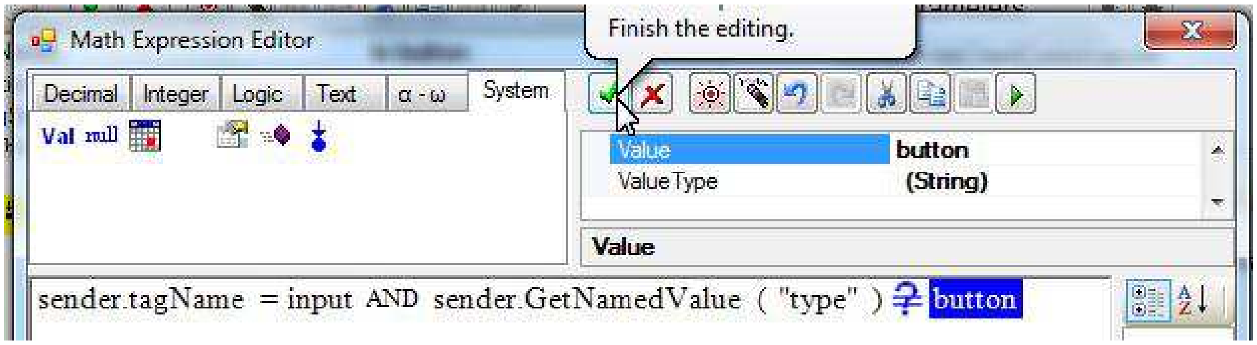
RunAt	Inherit
Value	type

Value

sender.tagName = input AND sender.GetNamedValue ("type") ≟ TRUE

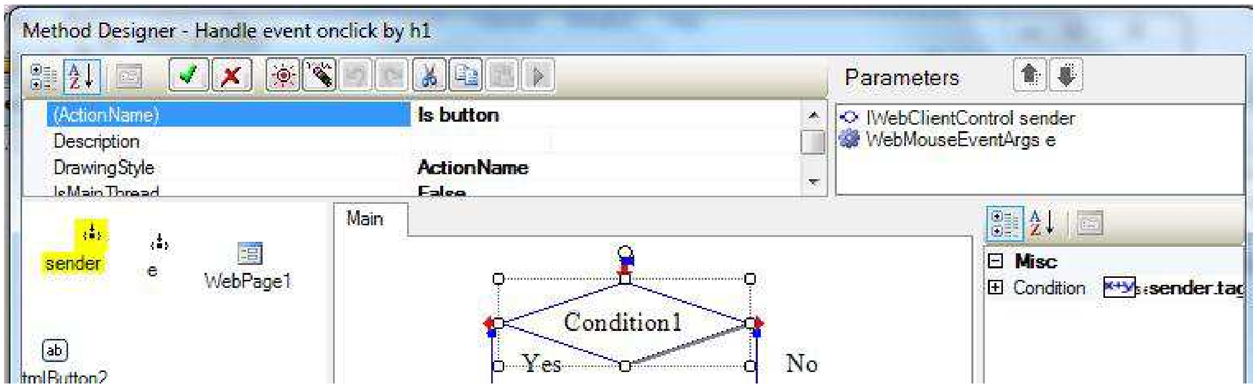


```
sender.tagName = input AND sender.GetNamedValue ( "type" ) ≠ TRUE
```

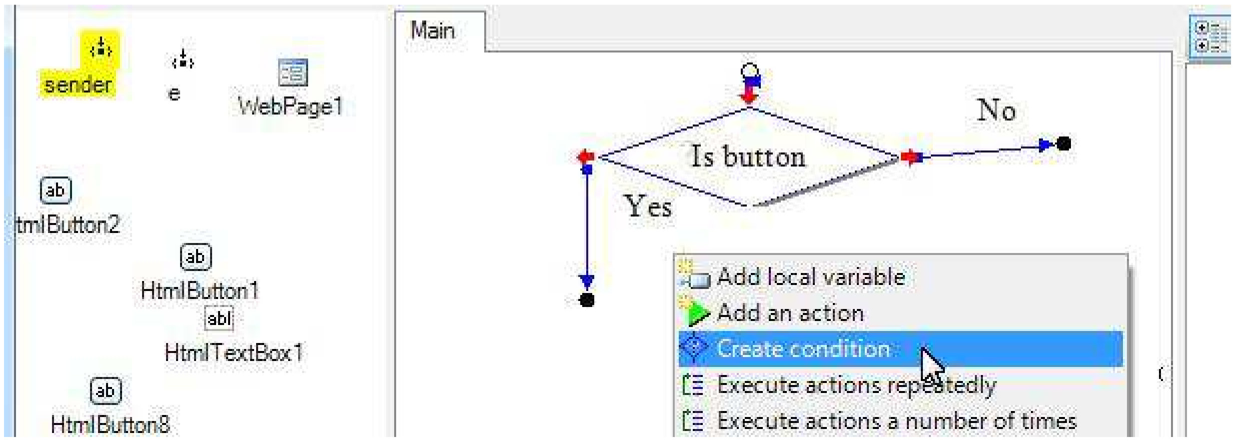


```
sender.tagName = input AND sender.GetNamedValue ( "type" ) ≠ button
```

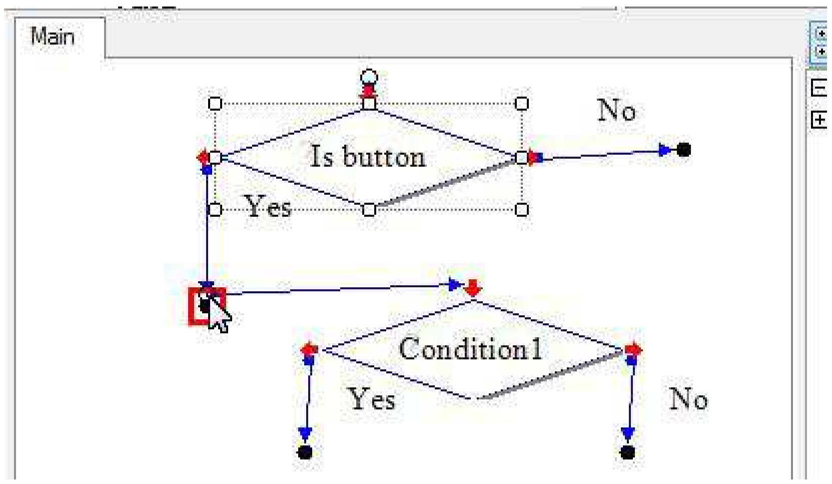
Change the condition name to “Is Button”:



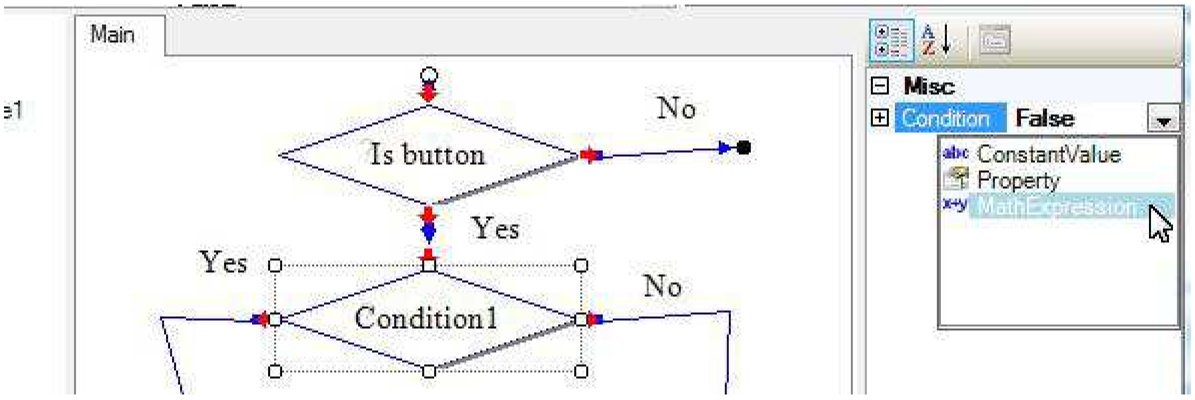
Add another condition action to check the background color of the sender:

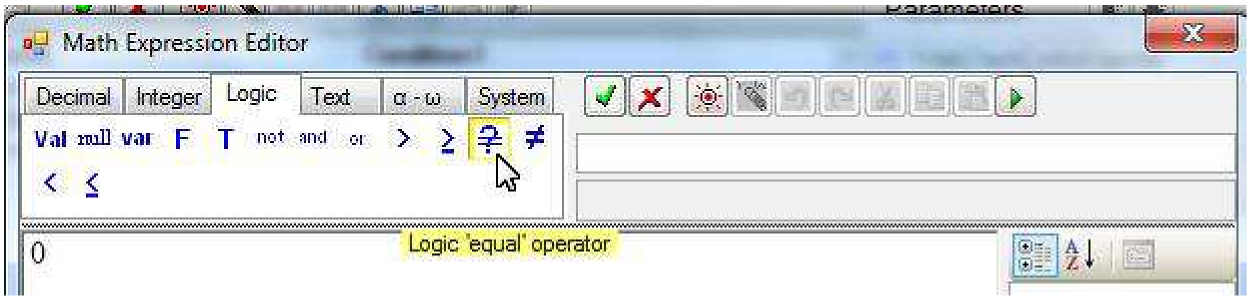


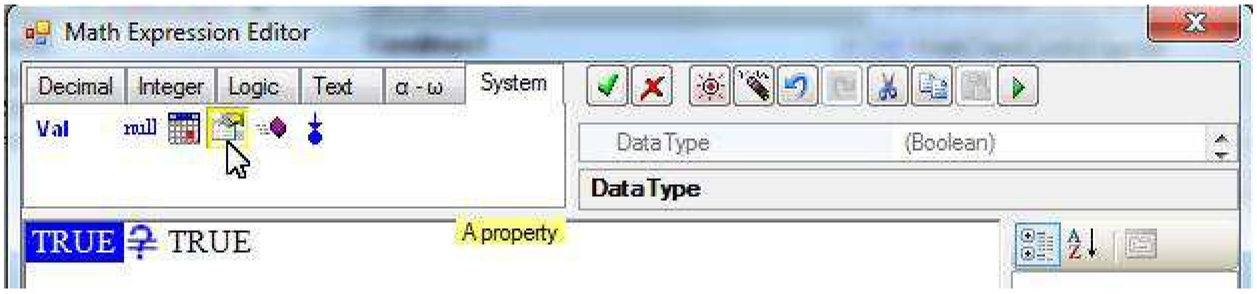
Link it to the Yes port of the last action:



Set Condition to check background color:

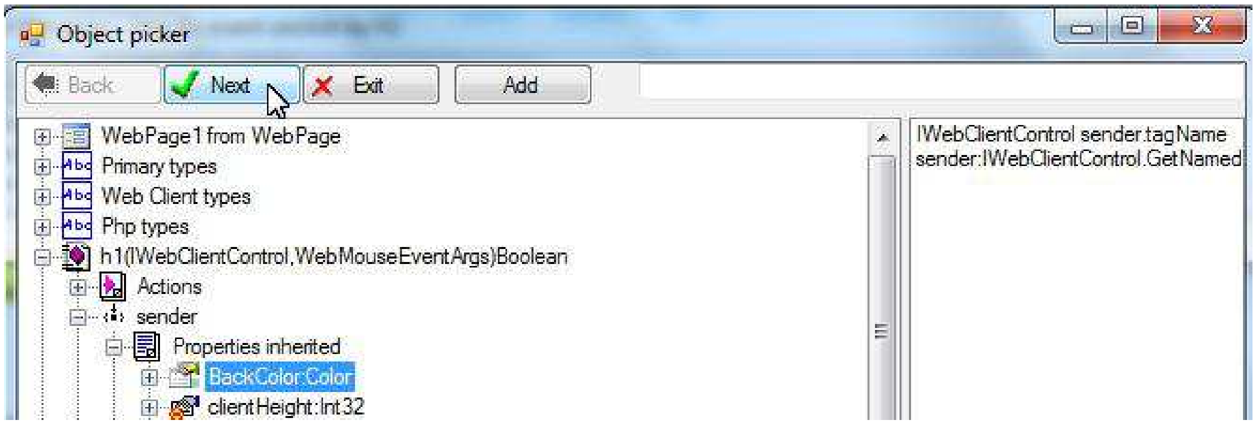






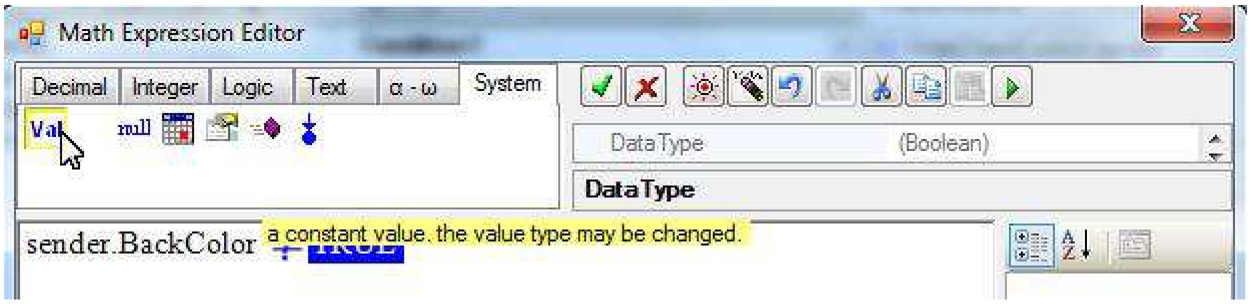
DataType (Boolean)

DataType

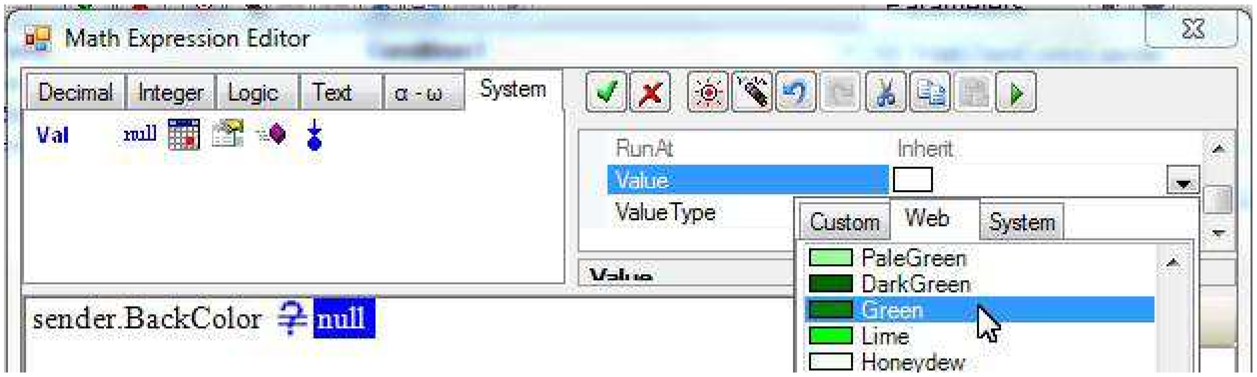


WebPage1 from WebPage
Primary types
Web Client types
Php types
h1(IWebClientControl,WebMouseEventArgs)Boolean
Actions
sender
Properties inherited
BackColor:Color
clientHeight:Int32

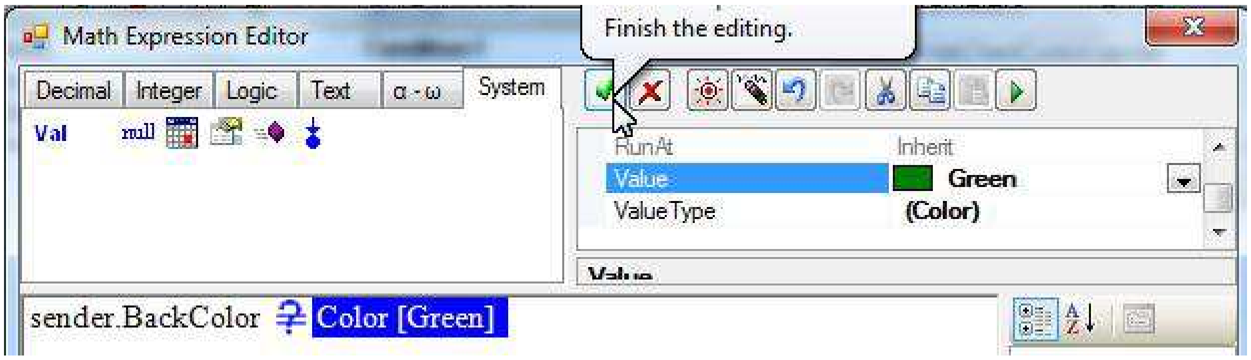
IWebClientControl sender.tagName
sender:IWebClientControl.GetNamed



sender.BackColor

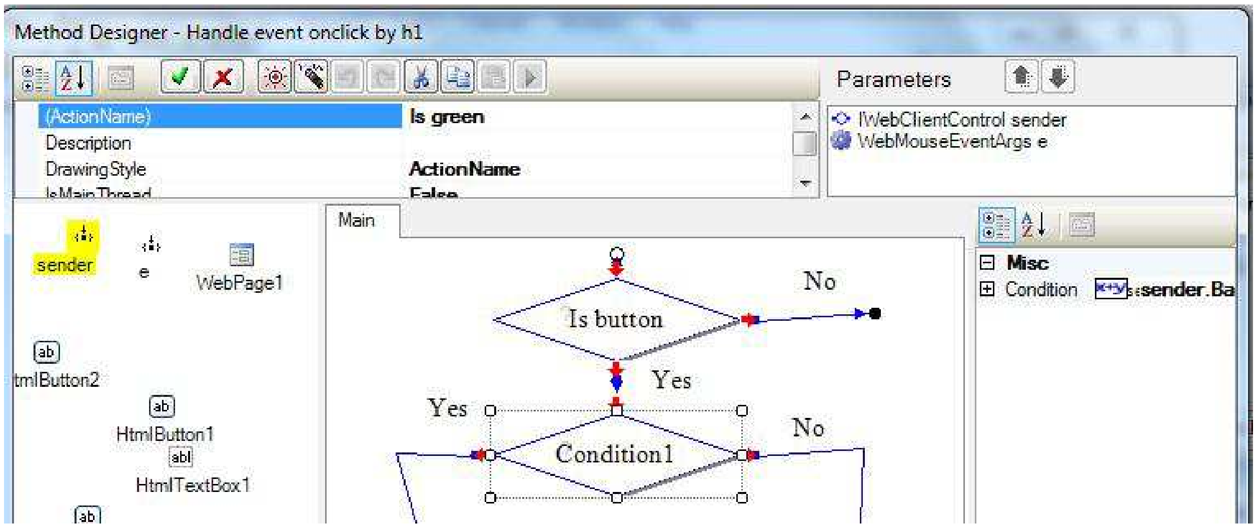


sender.BackColor null



sender.BackColor Color [Green]

Change the condition name to “Is green”:

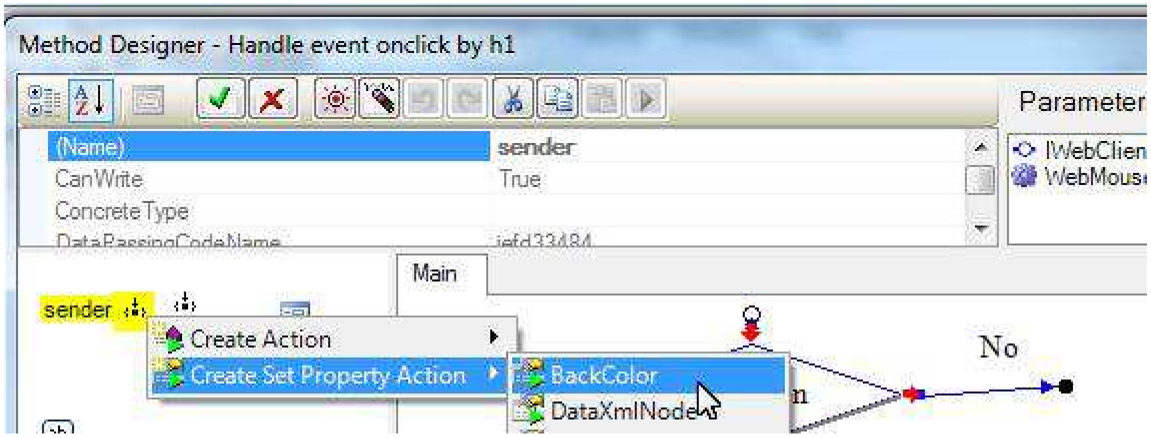


```
(ActionName)     Is green
Description
DrawingStyle     ActionName
IsMainThread     False
```

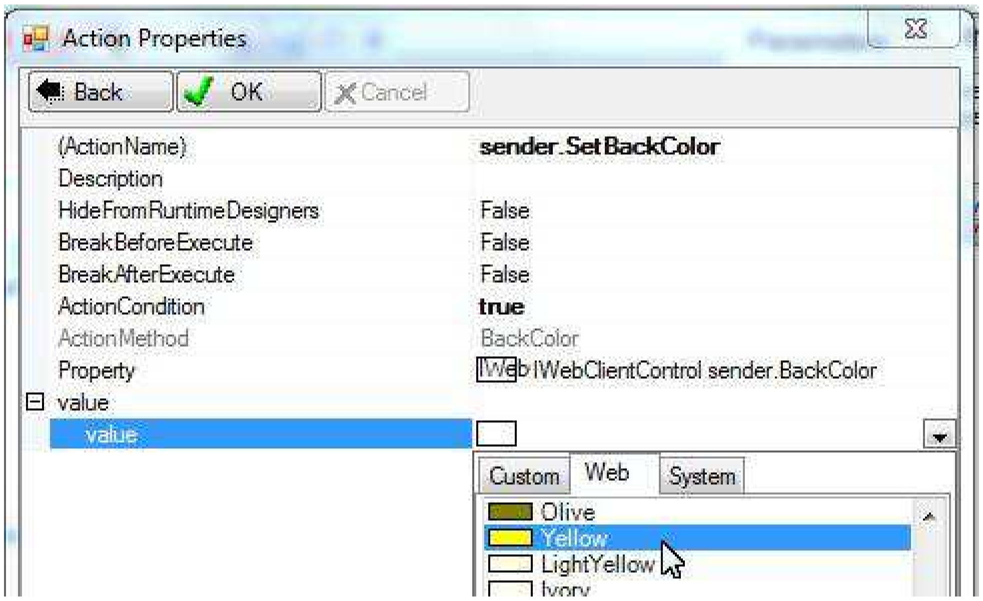
```
IWebClientControl sender
WebMouseEventArgs e
```

```
Condition   :sender.Ba
```

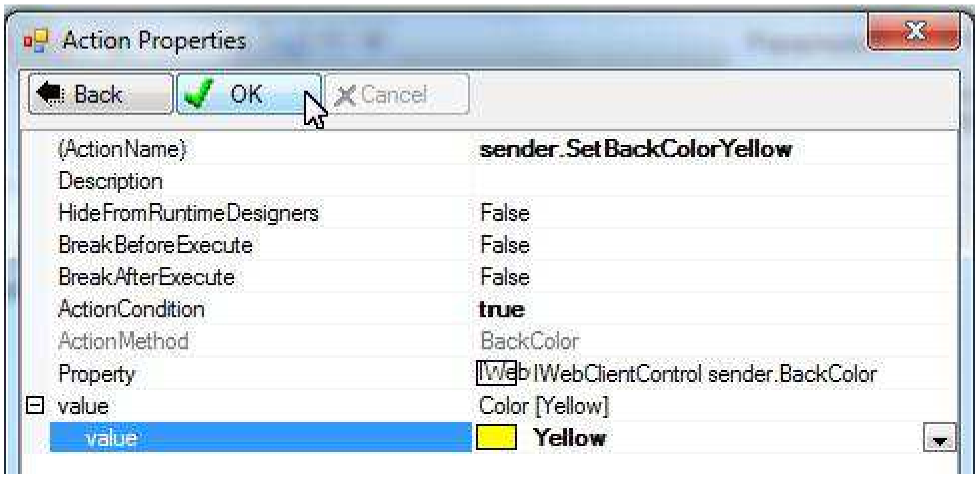
Create an action to set the background color of the sender to yellow:


Method Designer - Handle event onclick by h1
Parameter
IWebClien
WebMous
Main
sender
Create Action
Create Set Property Action
BackColor
DataXmlNode
No

```
(Name)               sender
CanWrite             True
ConcreteType
DataPassingCodeName  iefd33484
```

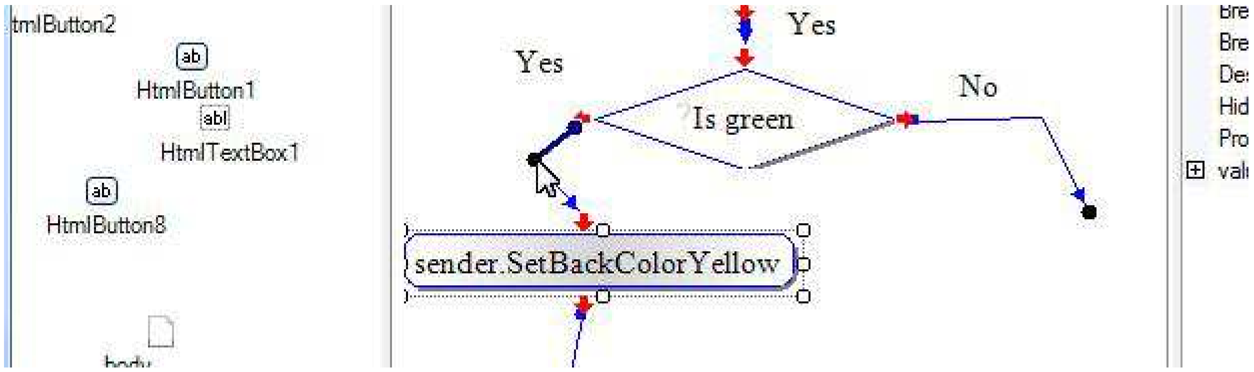


```
(ActionName)               sender.SetBackColor
Description
HideFromRuntimeDesigners   False
BreakBeforeExecute         False
BreakAfterExecute          False
ActionCondition            true
ActionMethod               BackColor
Property                   IWeb IWebClientControl sender.BackColor
⊟ value
    value
```

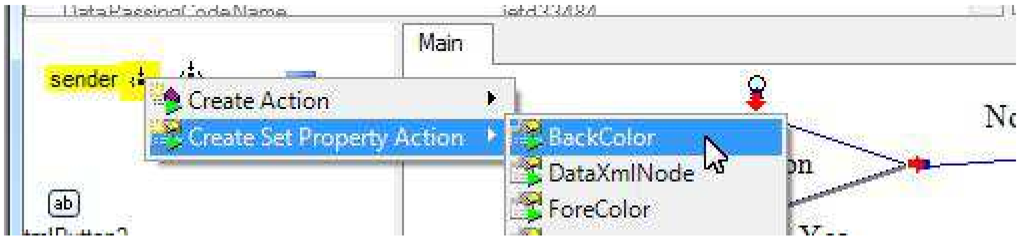


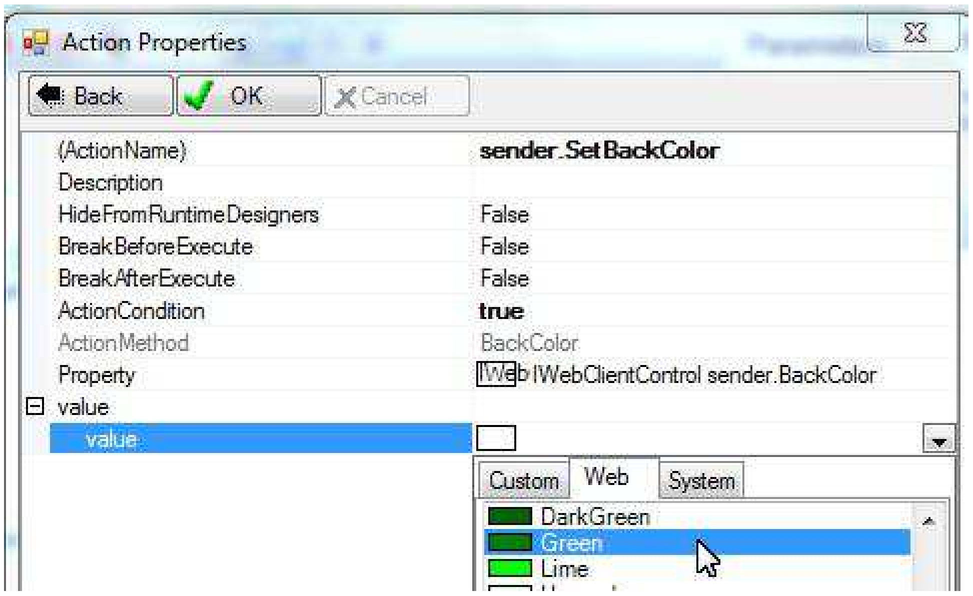
(ActionName)	sender.SetBackColorYellow
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	BackColor
Property	IWebClientControl sender.BackColor
value	Color [Yellow]
value	🟨 Yellow

Link the action to the Yes port:

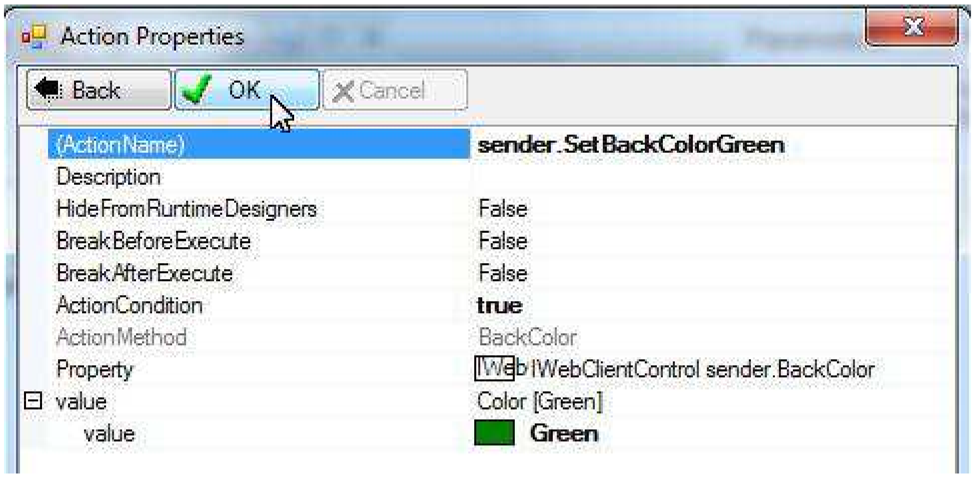


Create another action to set background color of the sender to green:



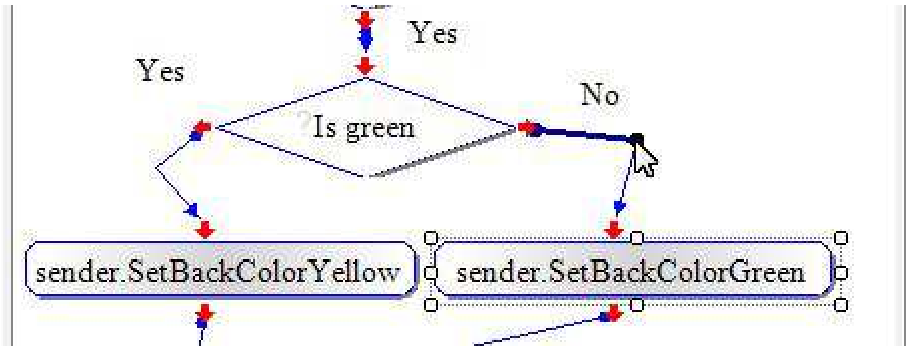


(ActionName)	sender.SetBackColor
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	BackColor
Property	IWeb IWebClientControl sender.BackColor
⊟ value	
value	

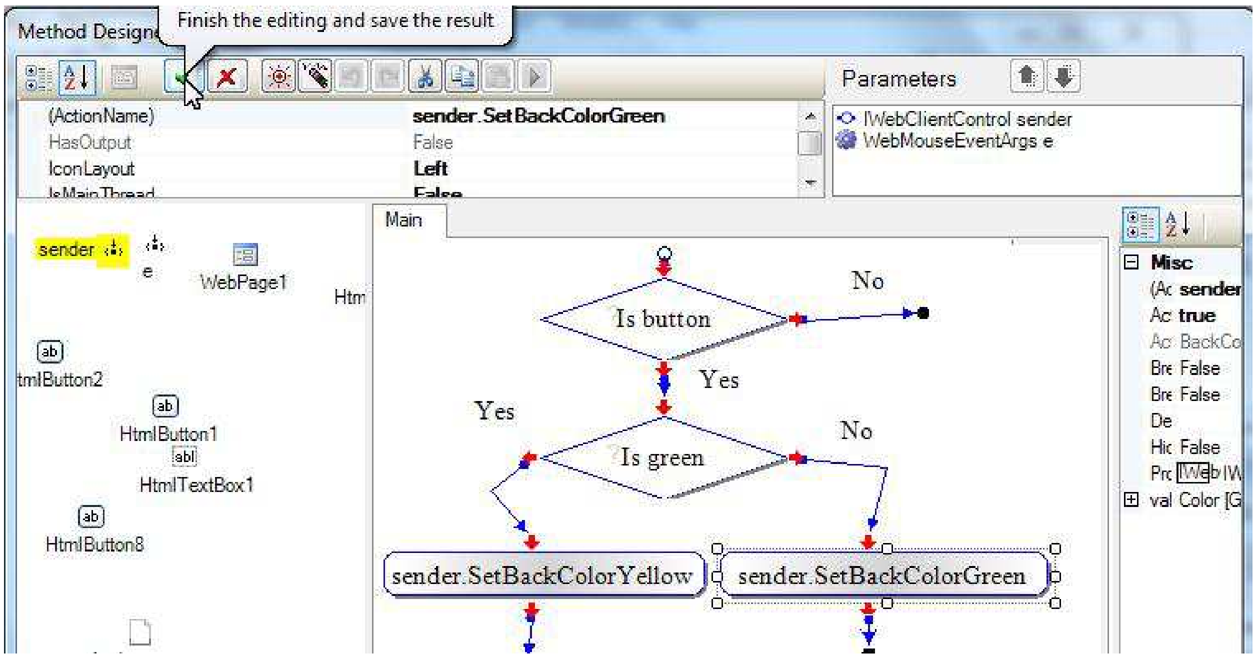


(ActionName)	sender.SetBackColorGreen
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	BackColor
Property	IWeb IWebClientControl sender.BackColor
⊟ value	Color [Green]
value	🟩 Green

Link it to the No port:

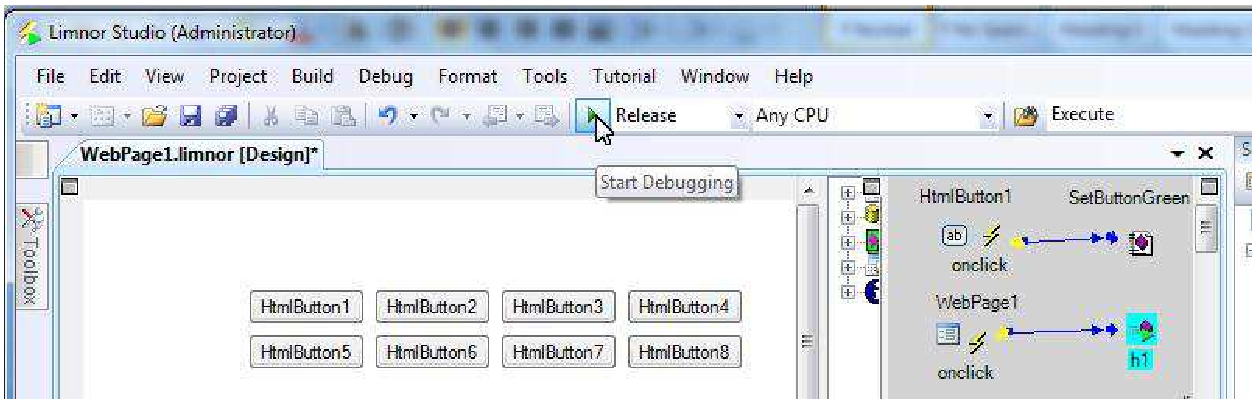


This is our event handler to handle click event from all buttons:

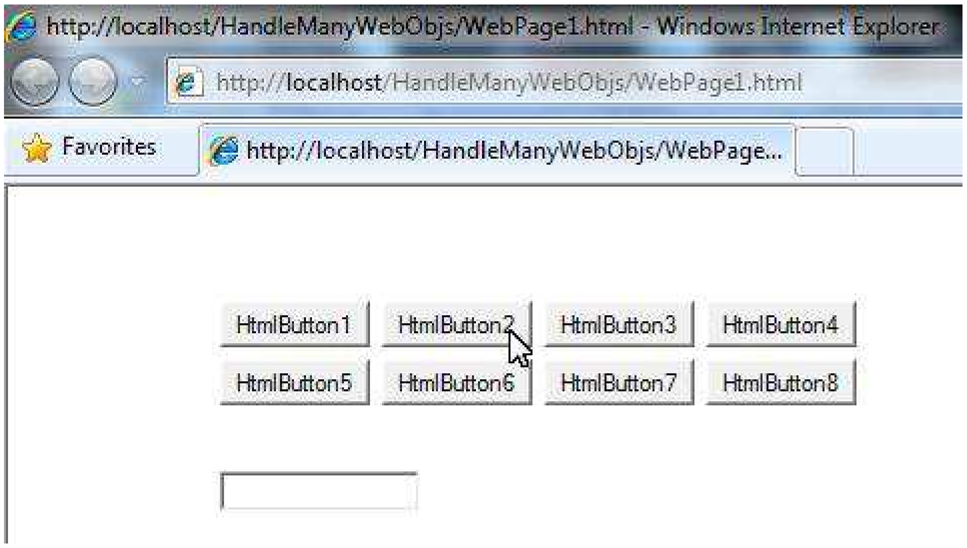


(ActionName)	sender.SetBackColorGreen
HasOutput	False
IconLayout	Left
IsMainThread	False

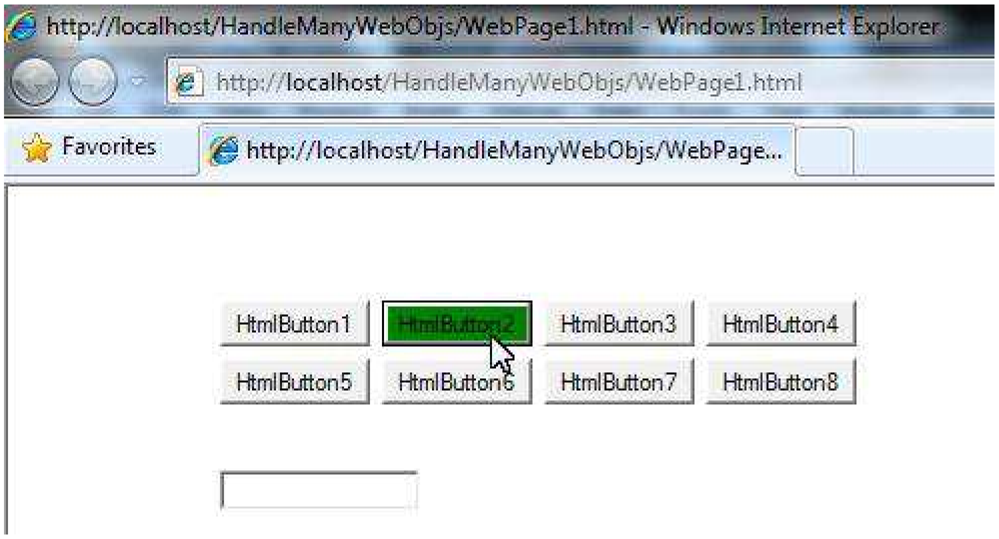
Test



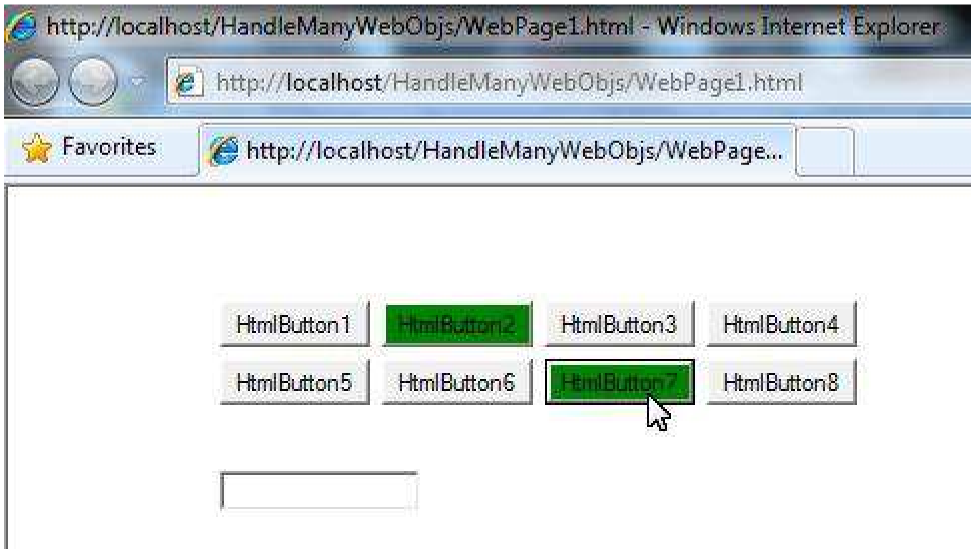
The page appears. Click a button:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

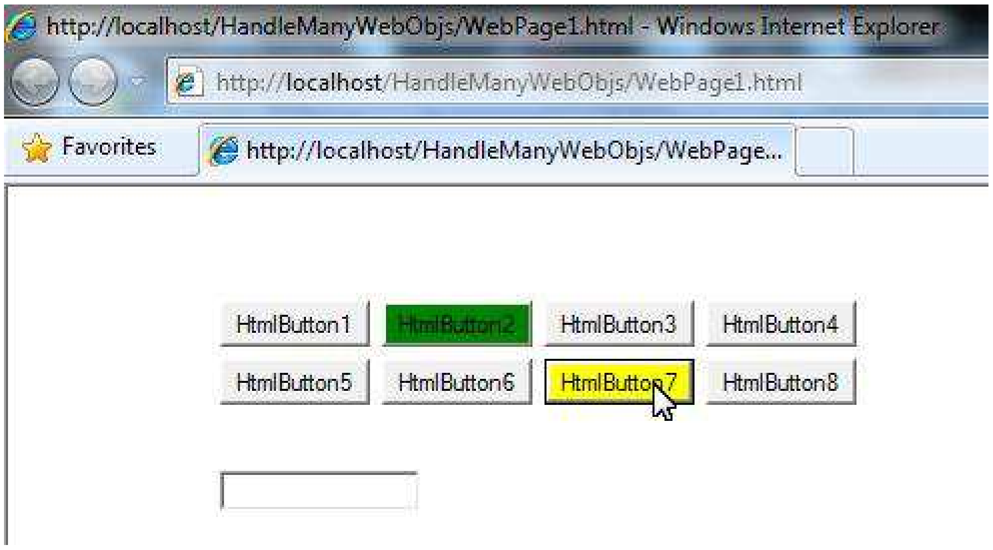
The background color of the button becomes green:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

Click another button, see, HtmlButton7, the button’s background color also changes to green:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

Click HtmlButton7 again, its background color changes to yellow:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

Use tag property

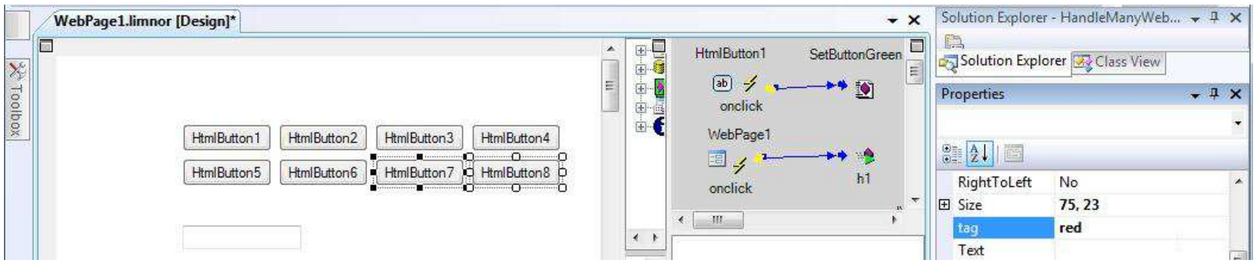
The above sample shows that one event handler can be used to handle a large number of elements. It uses a Condition action to identify buttons.

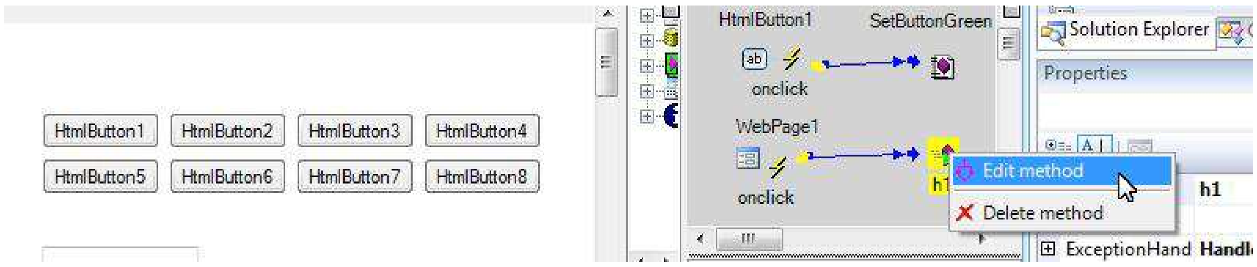
Suppose we want to handle the event based on button characteristics then we may use more condition actions.

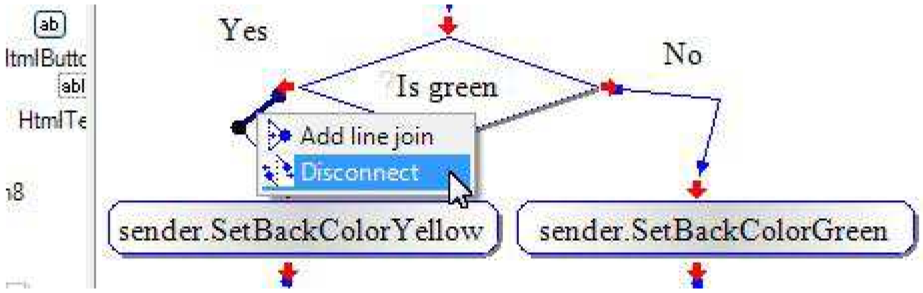
Every Html element has an “id” property. If the “id” is not empty then it will be unique. So we may use “id” to uniquely identify an element.

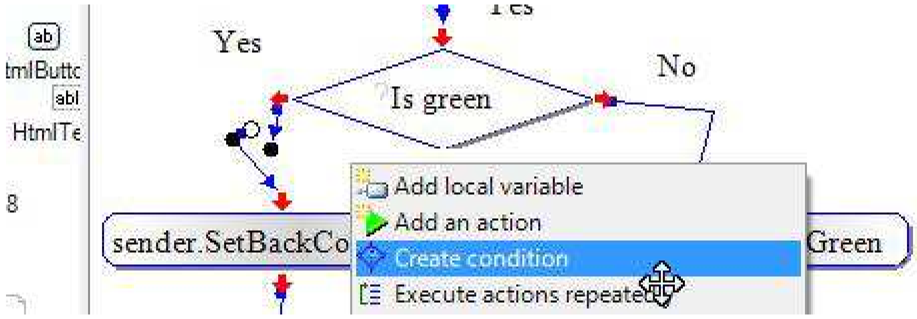
Every element also has a “tag” property, which is a string. There is not restriction on what text you may assign to it. You may assign value to it both at runtime and design time.

For example, let's set tag to "red" for button 7 and button 8. Modify the handler so that if the tag is "red" then instead of setting background color to yellow we set it to red.

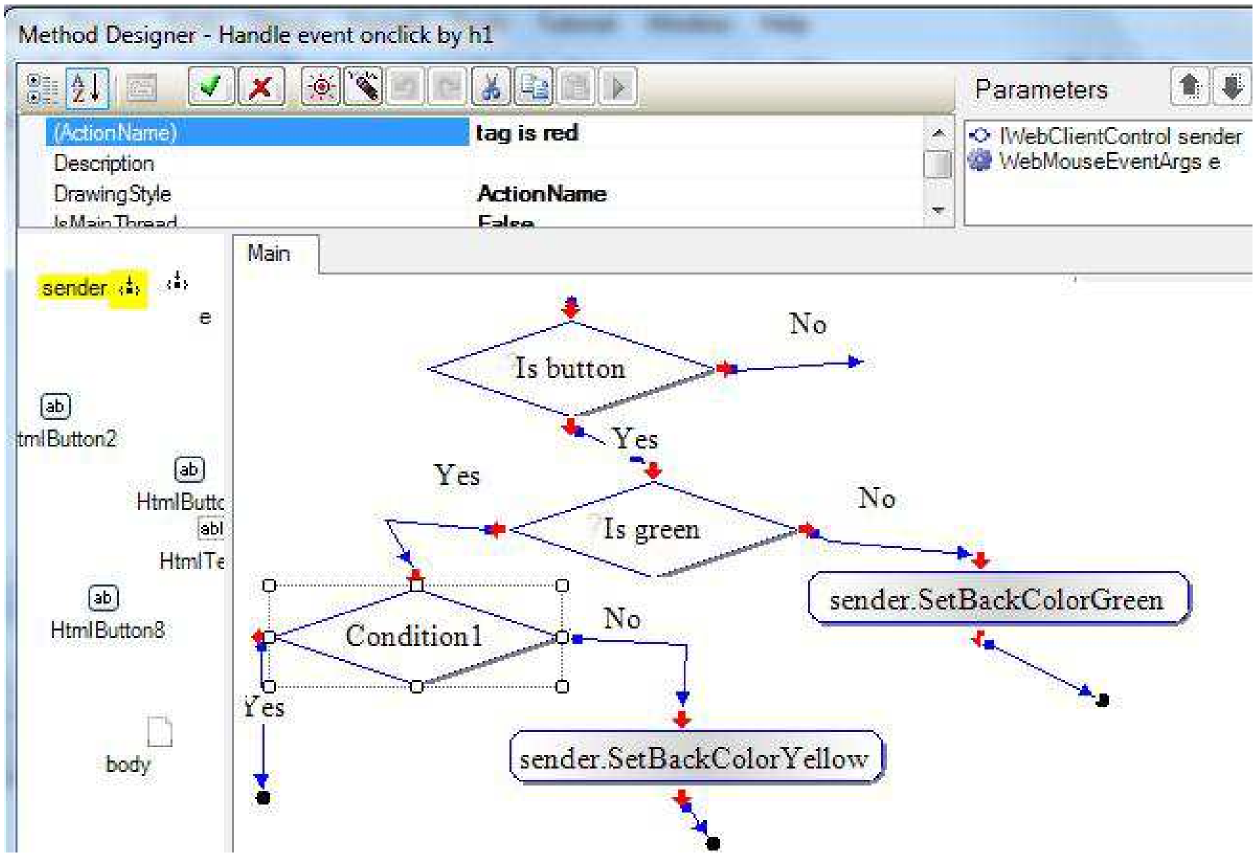








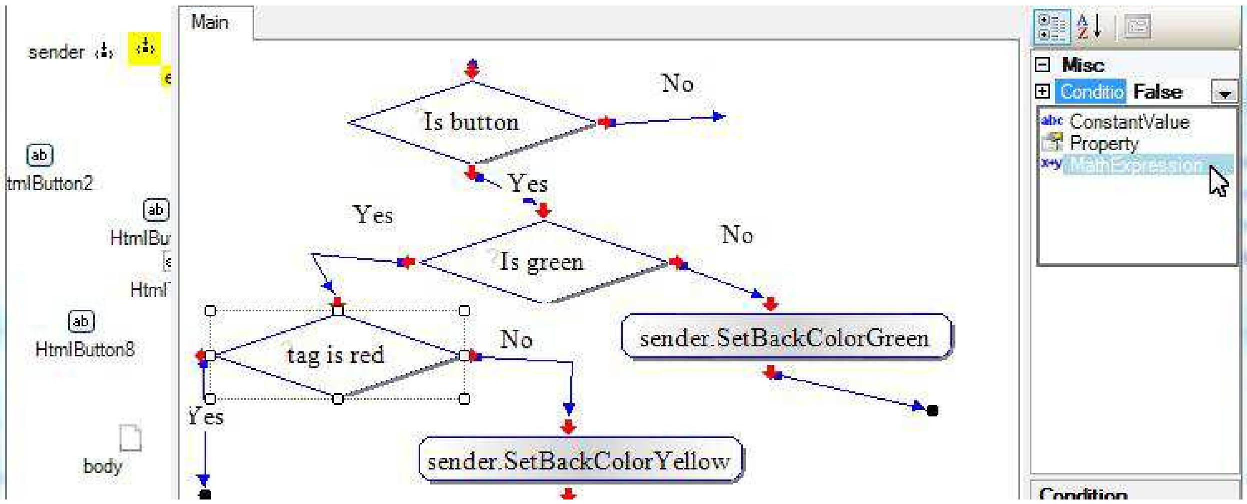
Insert a Condition to check tag = "red":

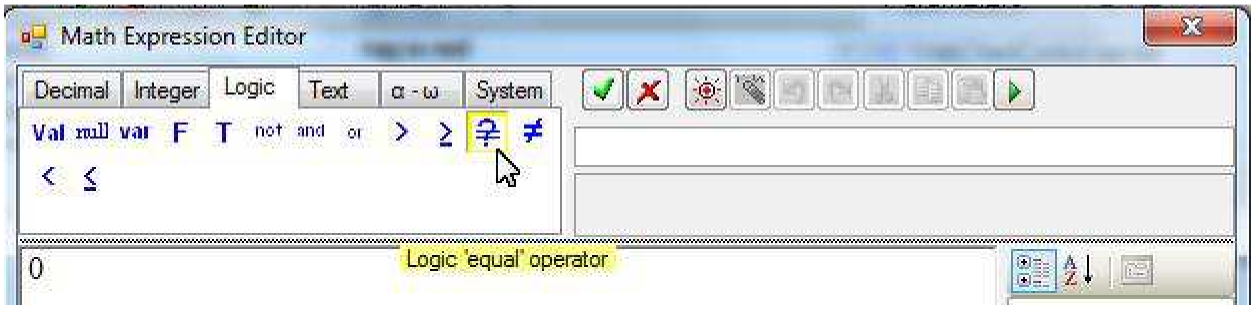

Method Designer - Handle event onclick by h1

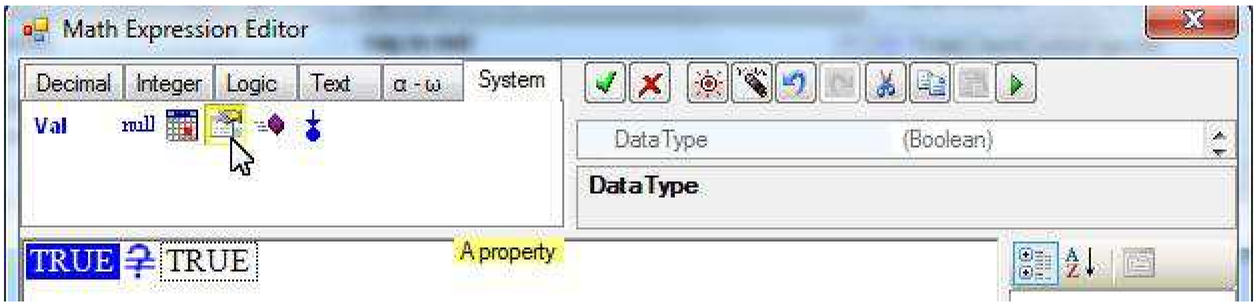
(ActionName)	tag is red
Description	
DrawingStyle	ActionName
IsMainThread	False

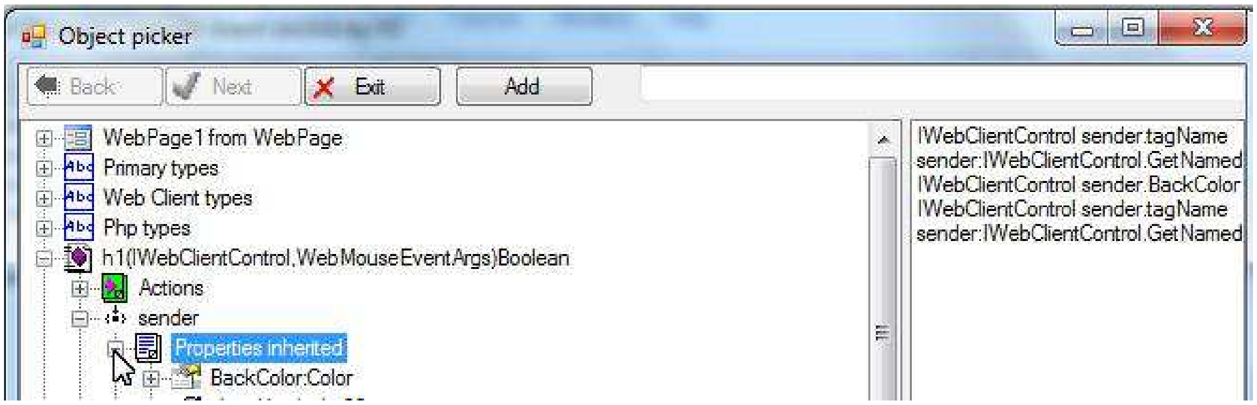
Parameters

- IWebClientControl sender
- WebMouseEventArgs e

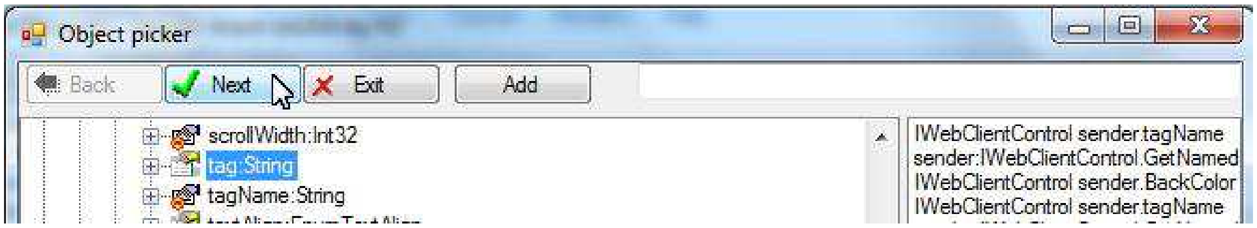



Math Expression Editor

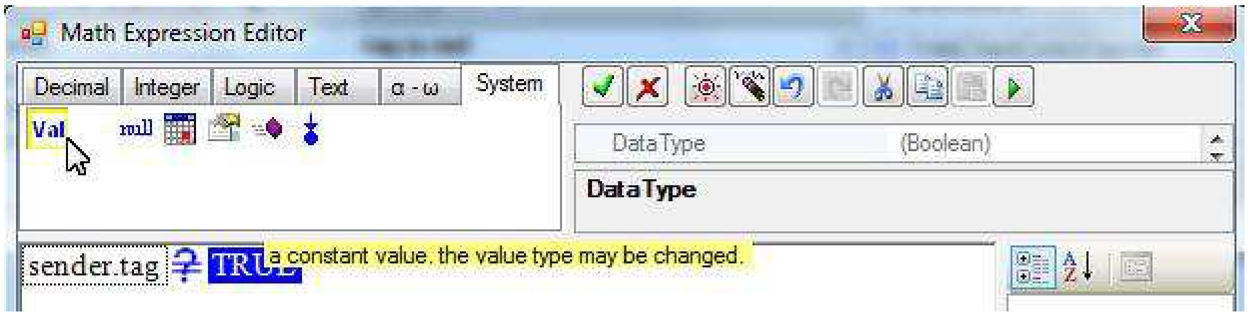


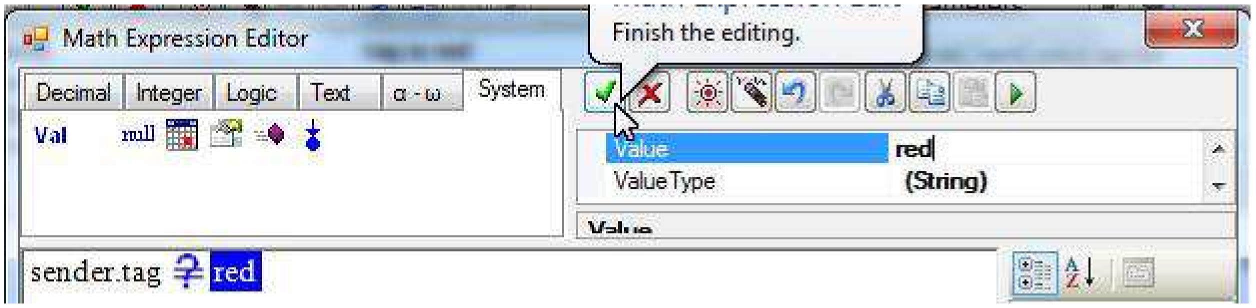


IWebClientControl sender.tagName
sender:IWebClientControl.GetNamed
IWebClientControl sender.BackColor
IWebClientControl sender.tagName
sender:IWebClientControl.GetNamed

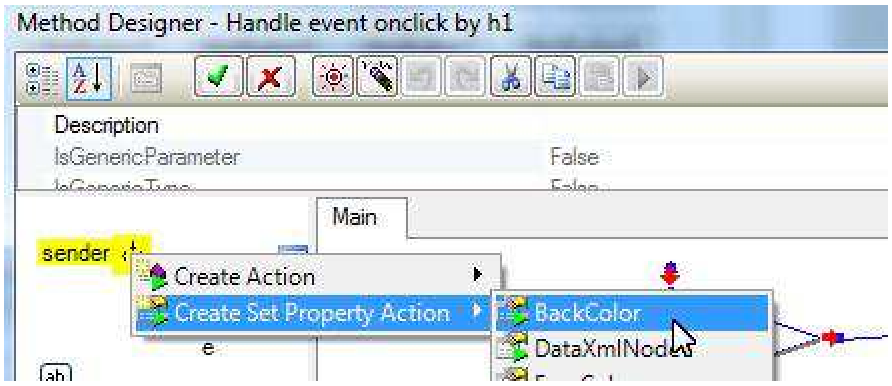


IWebClientControl sender.tagName
sender:IWebClientControl.GetNamed
IWebClientControl sender.BackColor
IWebClientControl sender.tagName

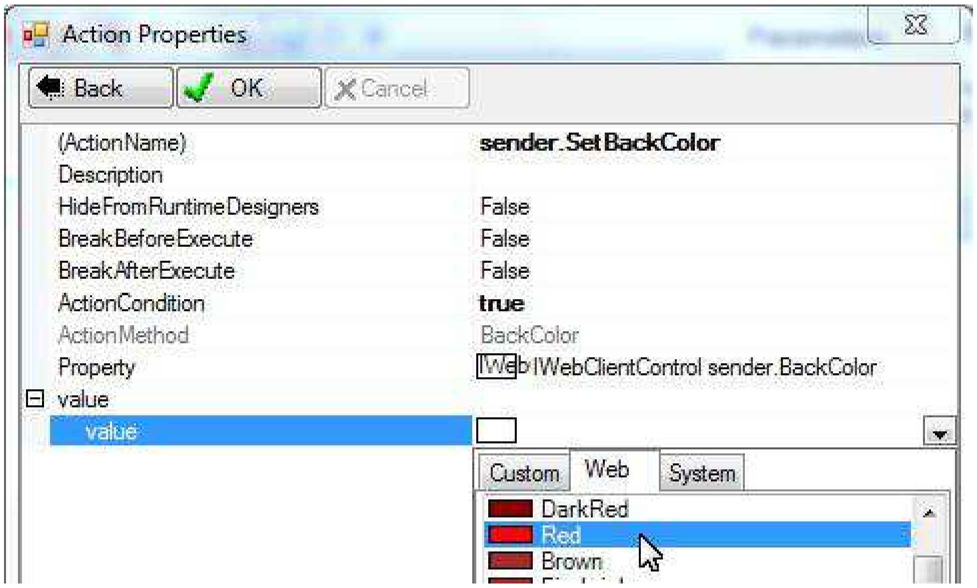




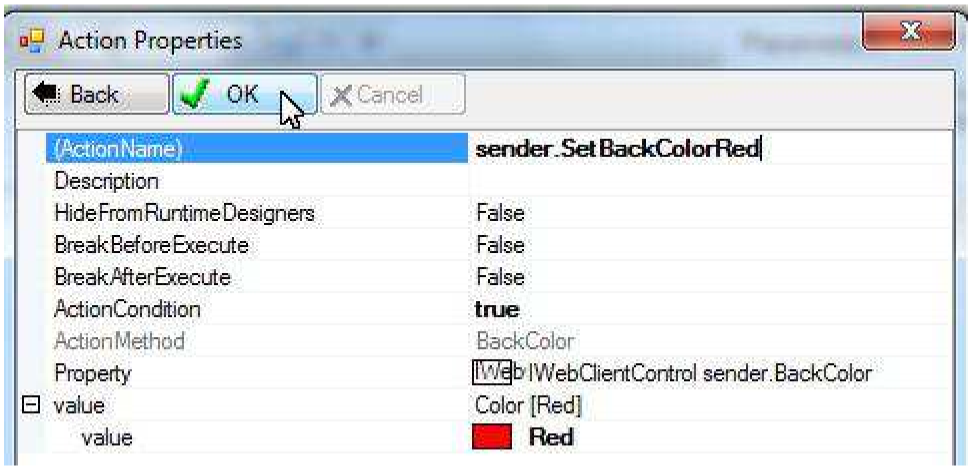
Create another action to set the background color of the sender to red:


Method Designer - Handle event onclick by h1

Description	
IsGenericParameter	False

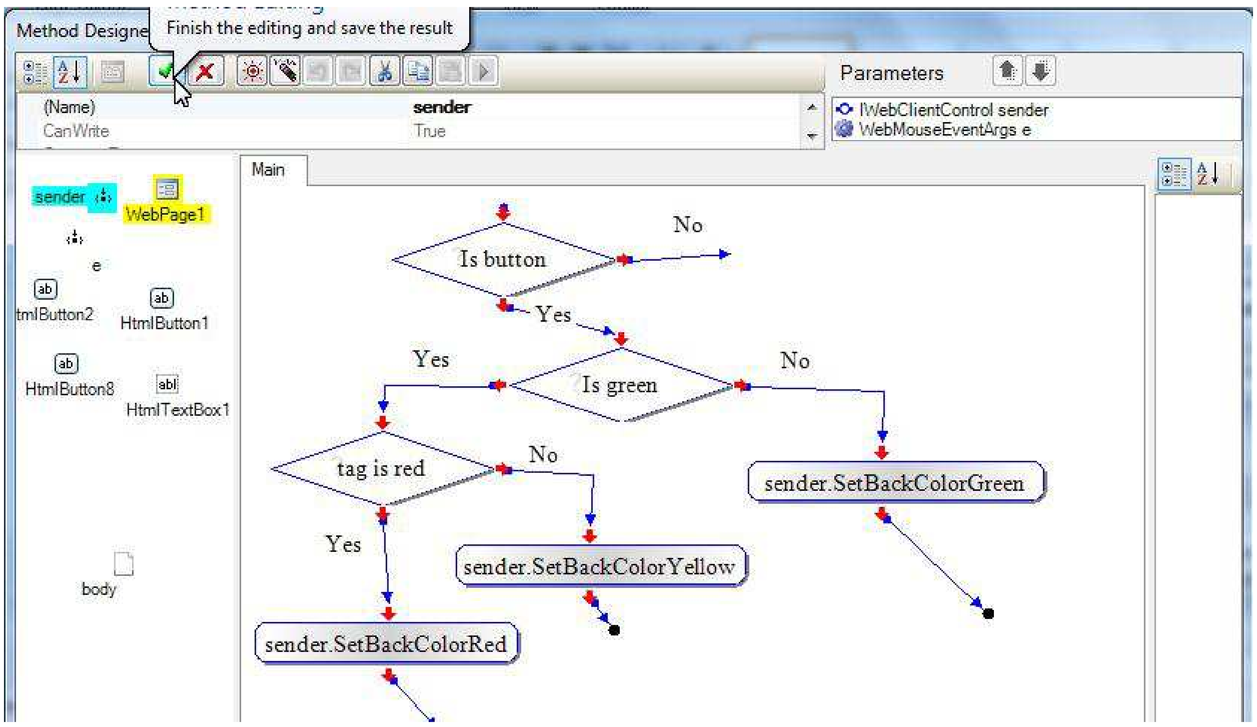

Action Properties

(ActionName)	**sender.SetBackColor**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	BackColor
Property	IWebClientControl sender.BackColor
⊟ value	
value	

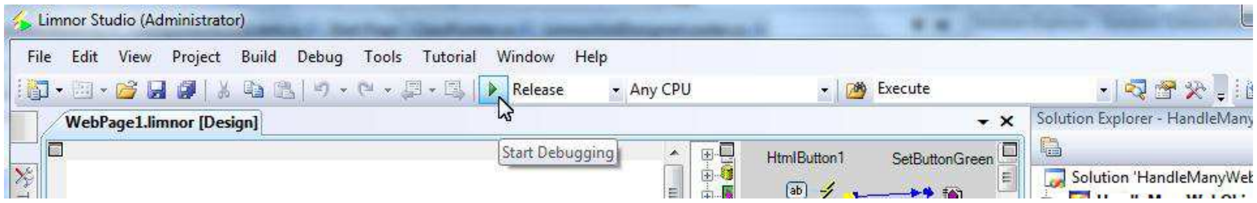

Action Properties

(ActionName)	**sender.SetBackColorRed**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	BackColor
Property	IWebClientControl sender.BackColor
⊟ value	Color [Red]
value	🟥 **Red**

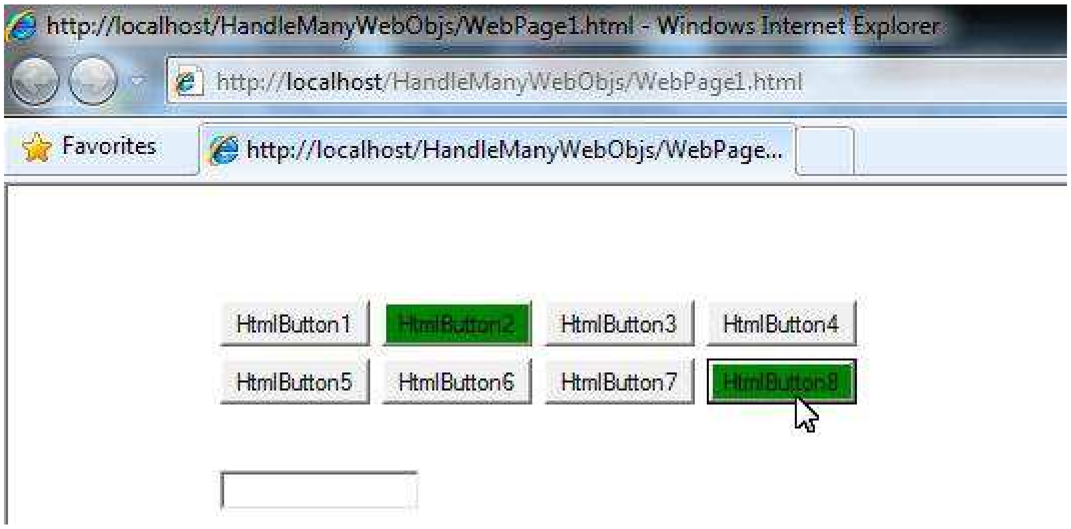
Link it to the Yes port of the condition. This is our new event handler:



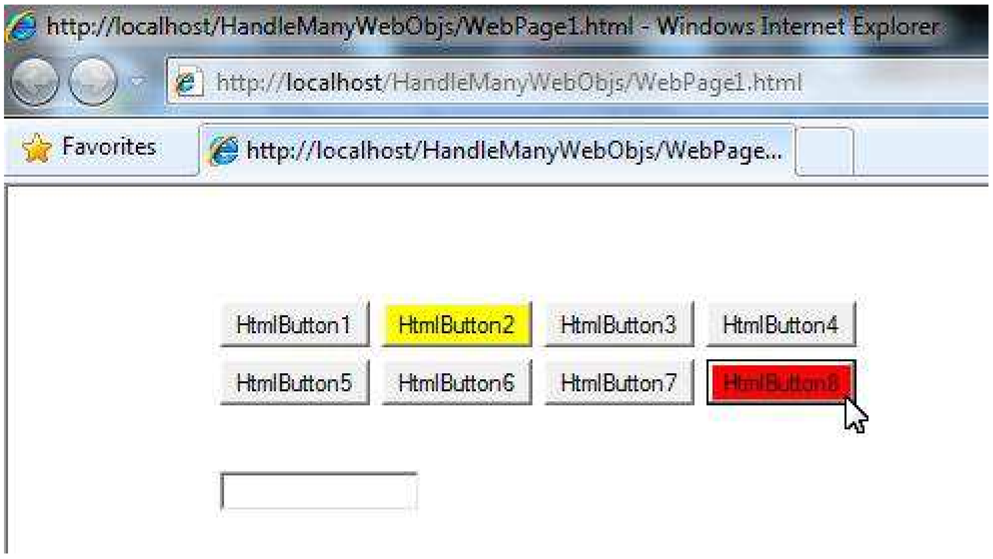
Run it again:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window Help
Release
Any CPU
Execute
WebPage1.limnor [Design]
Start Debugging
HtmlButton1
SetButtonGreen
Solution Explorer - HandleMany
Solution 'HandleManyWeb

Click button 2 and button 8, the buttons' background color change to green:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

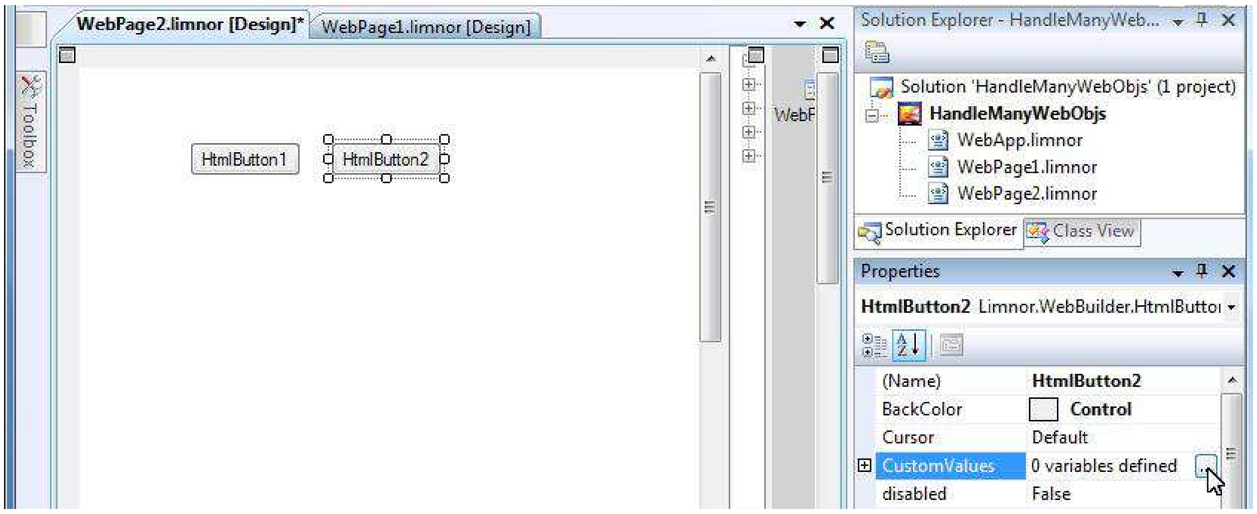
Click button 2 and button 8 again, button 2's background color becomes yellow and button 8's background color becomes red:


http://localhost/HandleManyWebObjs/WebPage1.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage1.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2
HtmlButton3
HtmlButton4
HtmlButton5
HtmlButton6
HtmlButton7
HtmlButton8

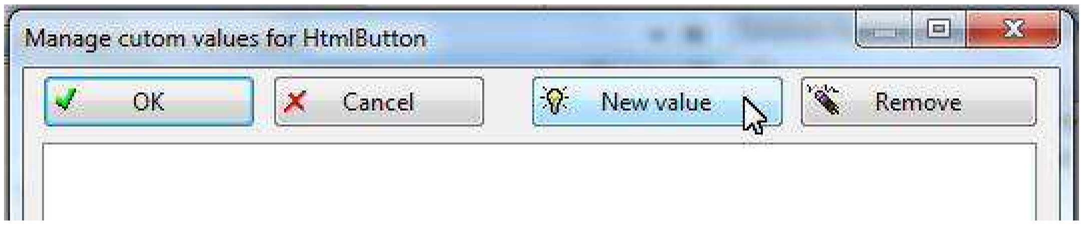
Use named values

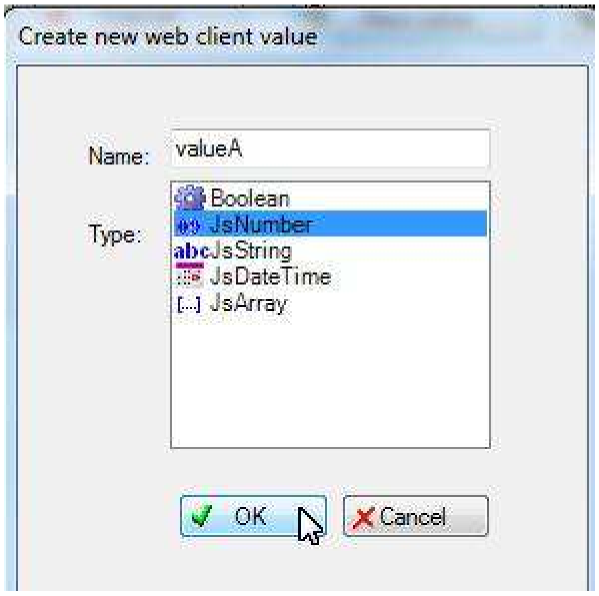
The tag property is a convenient way of associating a piece of information with an element. We may also associate more values to an element.

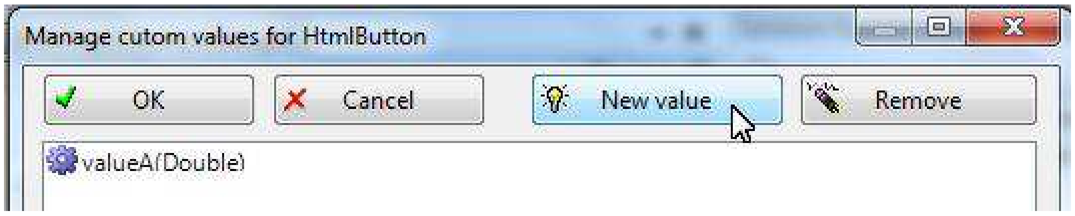
At design time, we may associate values to an element by setting its CustomValues property:

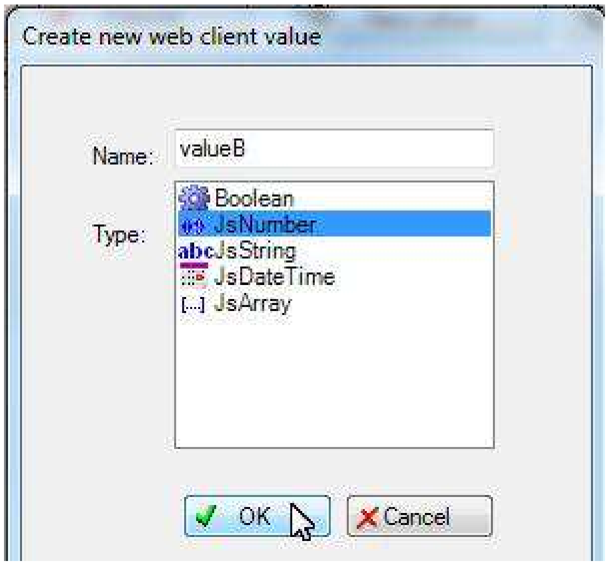


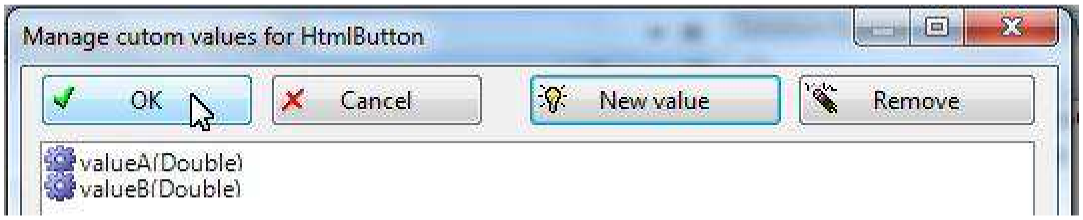
(Name)	HtmlButton2
BackColor	Control
Cursor	Default
CustomValues	0 variables defined
disabled	False



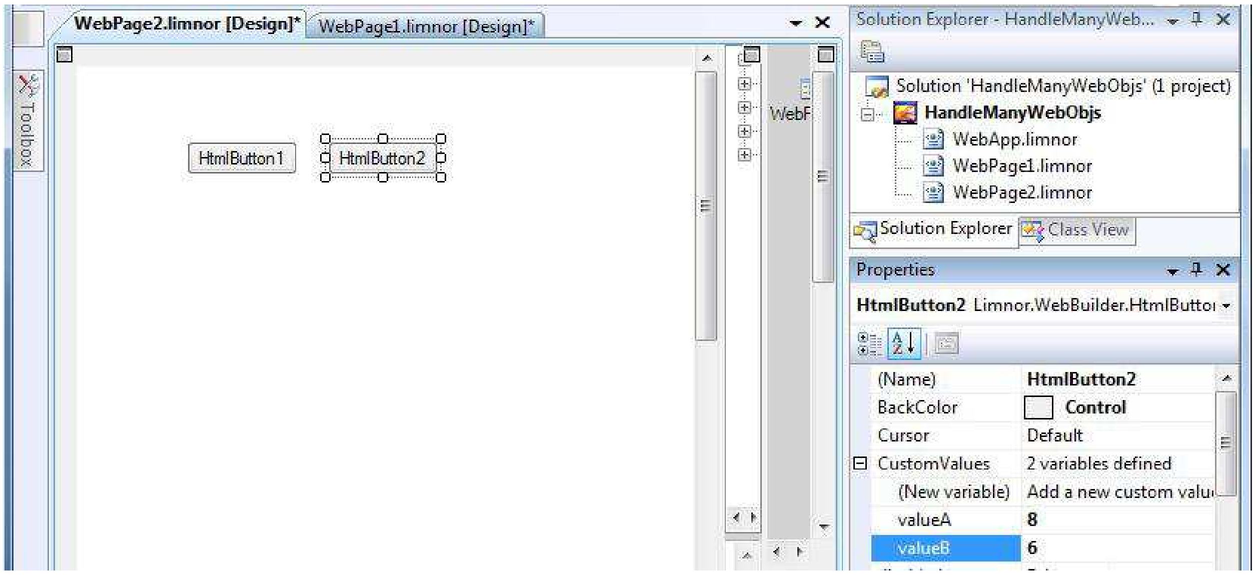








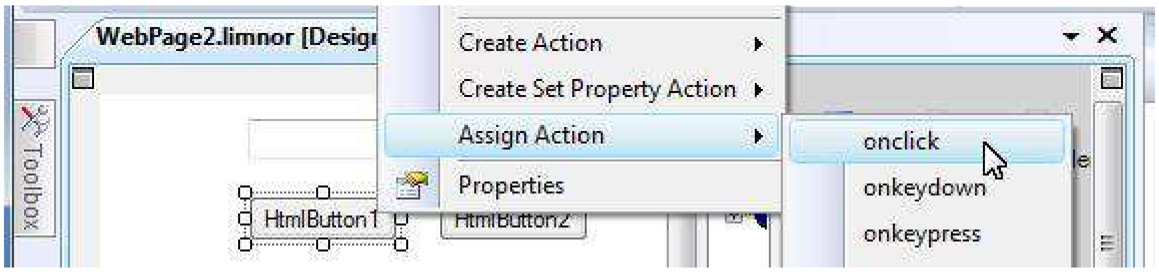
Values can be set at design time:

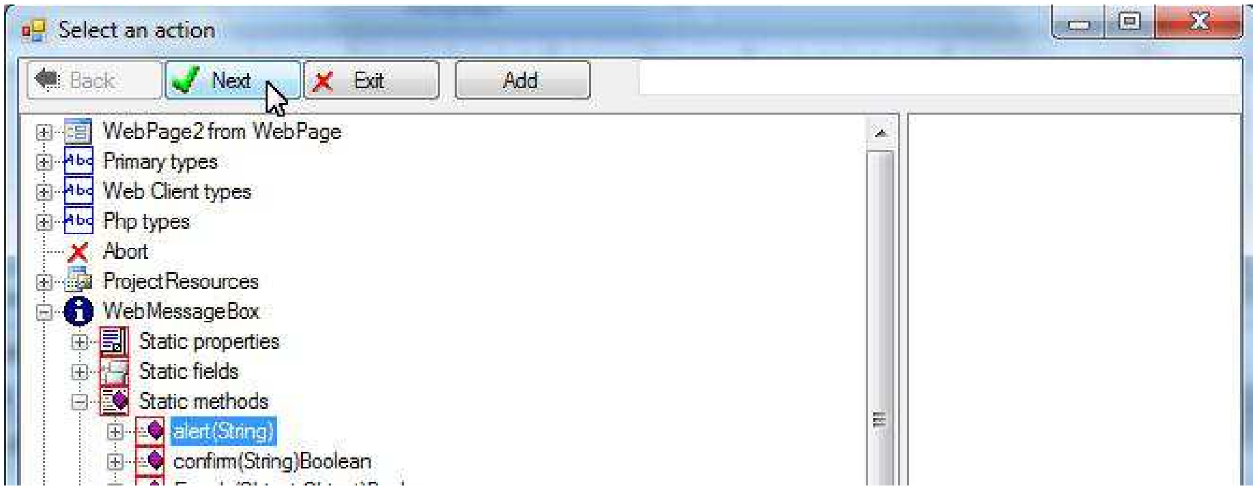

WebPage2.limnor [Design]*
WebPage1.limnor [Design]*
HtmlButton1
HtmlButton2
Solution Explorer - HandleManyWeb...
Solution 'HandleManyWebObjs' (1 project)
HandleManyWebObjs
WebApp.limnor
WebPage1.limnor
WebPage2.limnor
Solution Explorer
Class View
Properties
HtmlButton2 Limnor.WebBuilder.HtmlButto

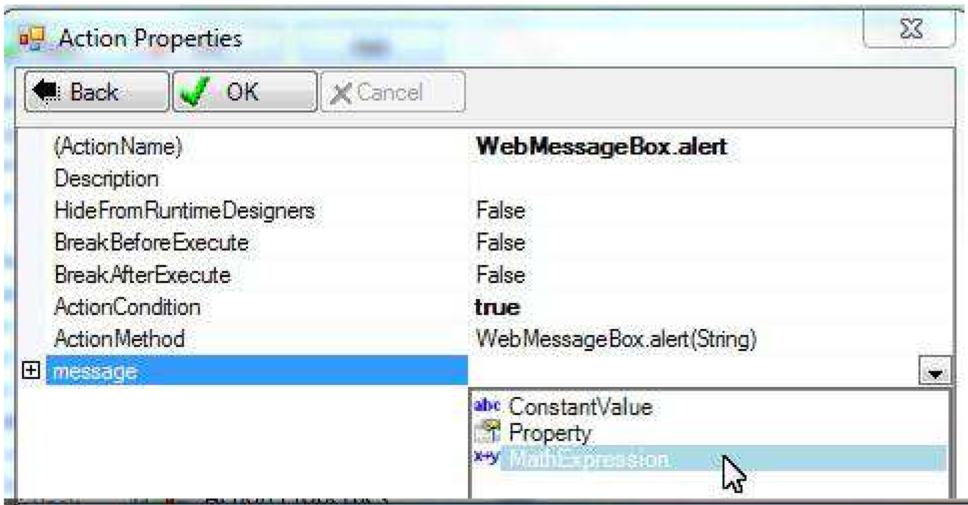
(Name)	HtmlButton2
BackColor	Control
Cursor	Default
CustomValues	2 variables defined
(New variable)	Add a new custom valu
valueA	8
valueB	6

At runtime, GetNamedValue can be used to get a value; SetOrCreateNamedValue can be used to associate a value to an element.

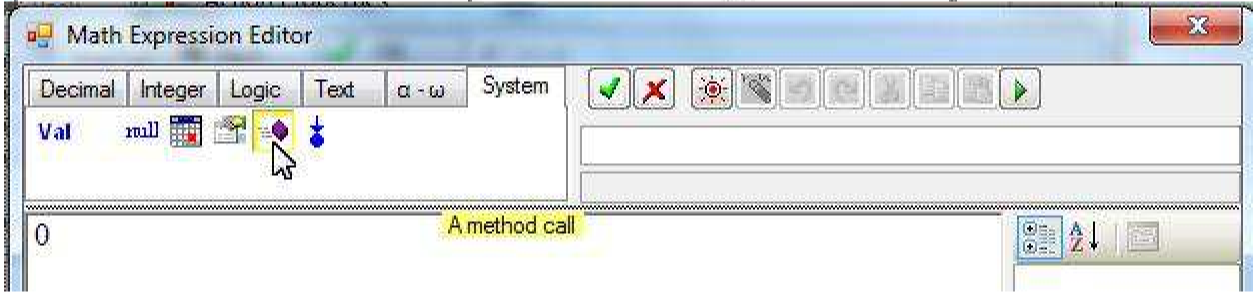
Let's use a message box to show value of valueA:



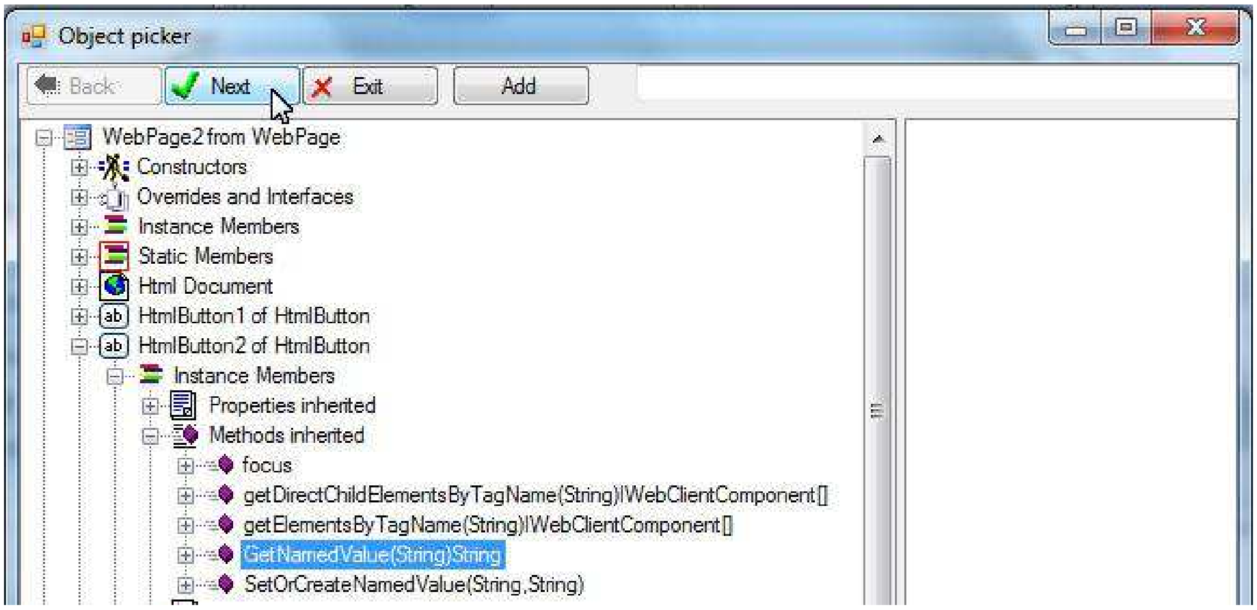




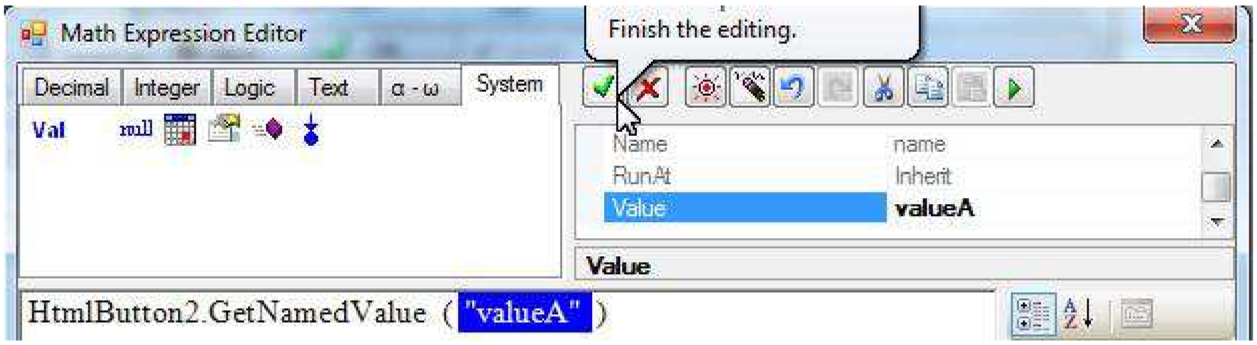
(ActionName)	**WebMessageBox.alert**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	WebMessageBox.alert(String)
message	



```
0
```

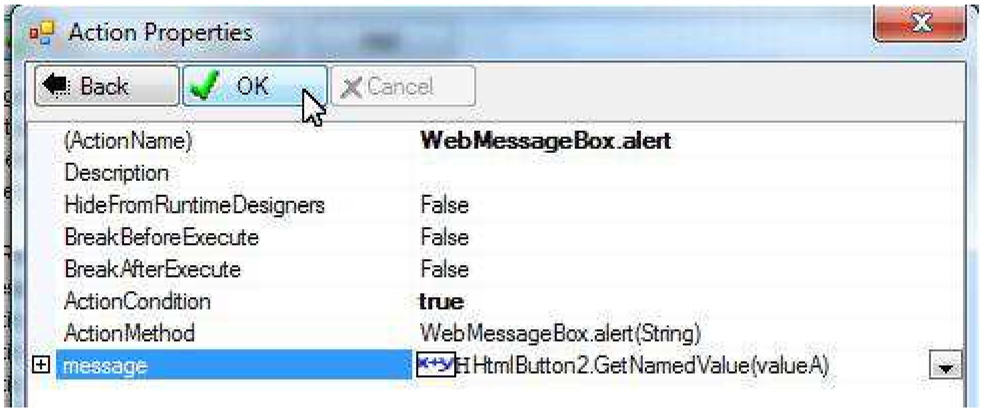


```
WebPage2 from WebPage
    Constructors
    Overrides and Interfaces
    Instance Members
    Static Members
    Html Document
    HtmlButton1 of HtmlButton
    HtmlButton2 of HtmlButton
        Instance Members
            Properties inherited
            Methods inherited
                focus
                getDirectChildElementsByTagName(String)IWebClientComponent[]
                getElementsByTagName(String)IWebClientComponent[]
                GetNamedValue(String)String
                SetOrCreateNamedValue(String,String)
```



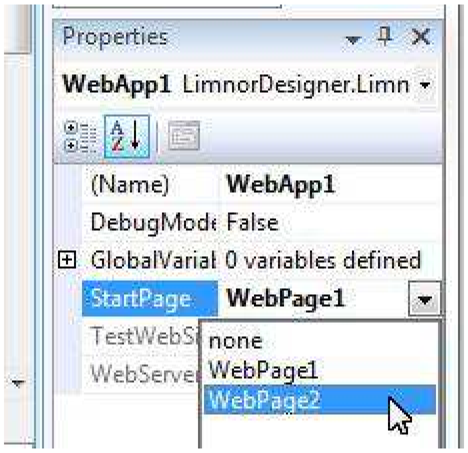
Name	name
RunAt	Inherit
Value	valueA

HtmlButton2.GetNamedValue ("valueA")

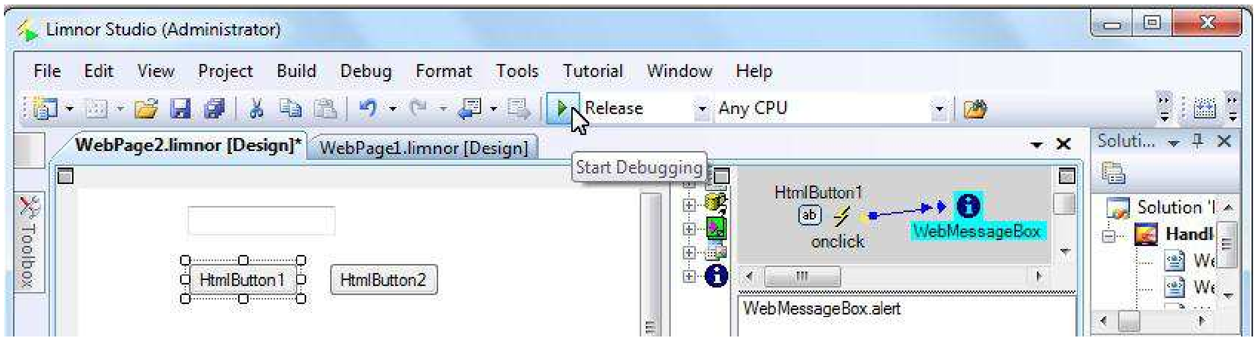


(ActionName)	WebMessageBox.alert
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	WebMessageBox.alert(String)
message	HtmlButton2.GetNamedValue(valueA)

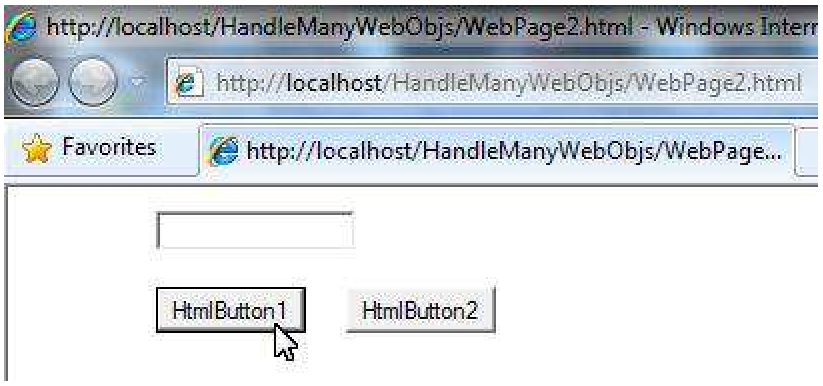
Let’s test it:



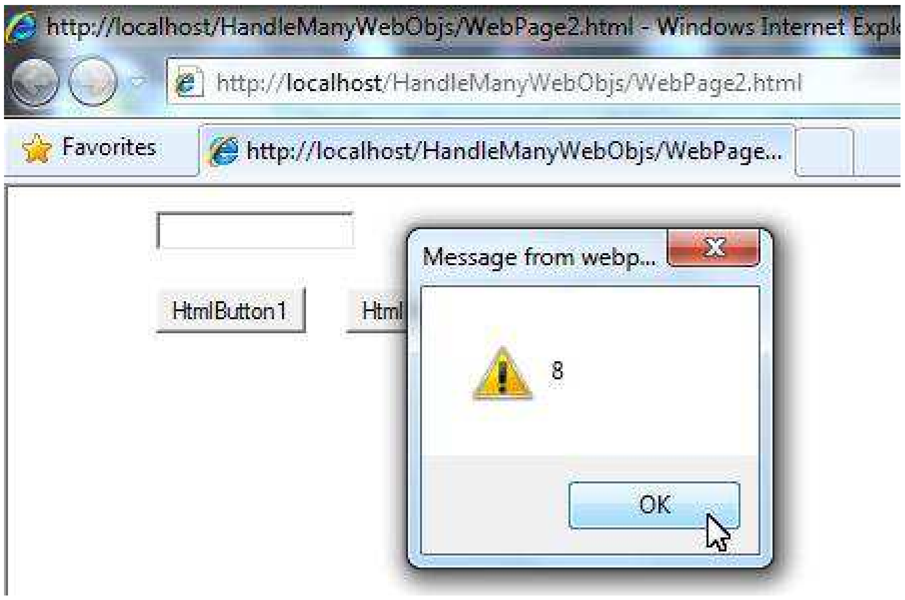
(Name)	WebApp1
DebugMode	False
GlobalVarial	0 variables defined
StartPage	WebPage1
TestWebS	
WebServer	



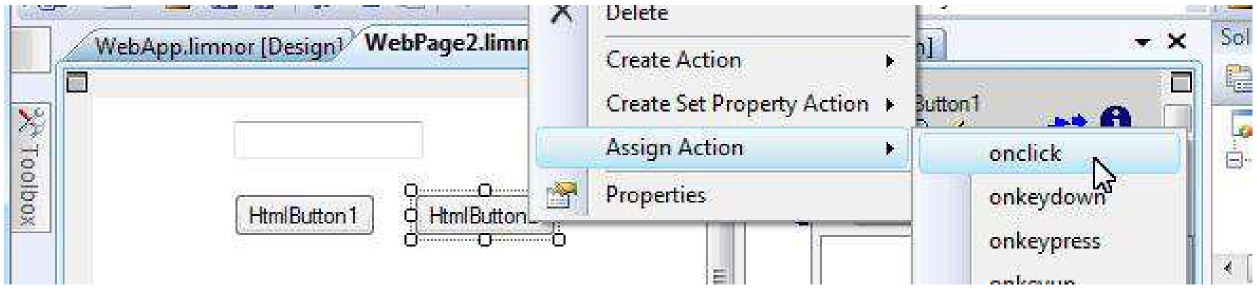
Click the button:

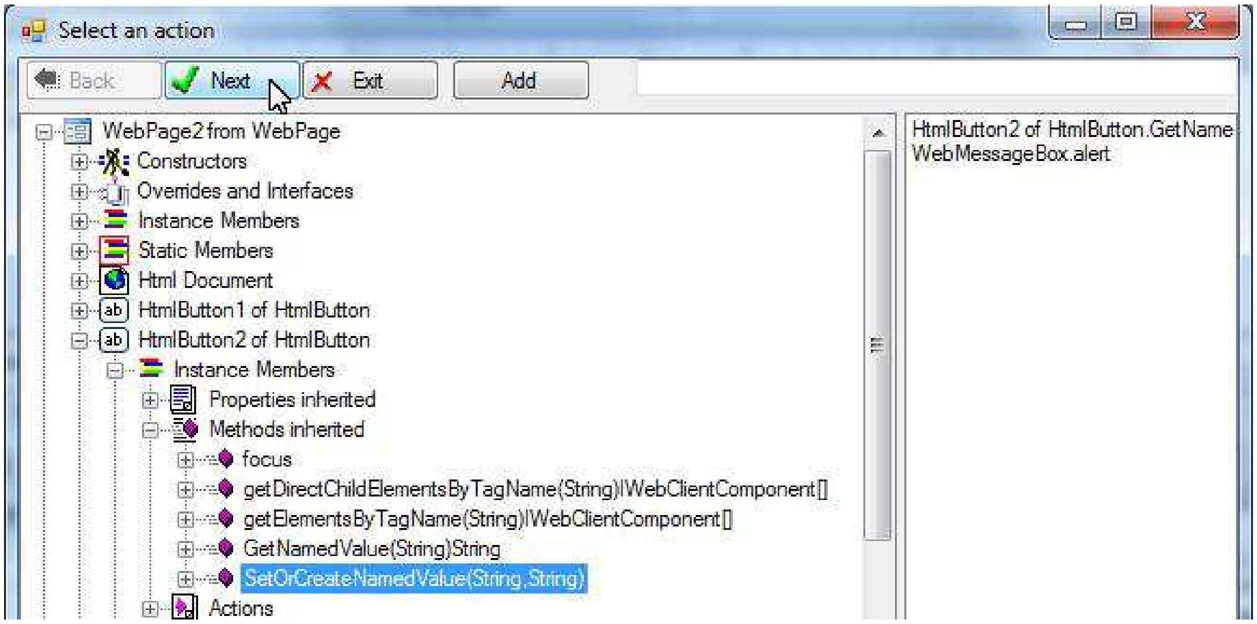

http://localhost/HandleManyWebObjs/WebPage2.html - Windows Intern
http://localhost/HandleManyWebObjs/WebPage2.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
HtmlButton2

A message box appears showing the value we set at design time:


http://localhost/HandleManyWebObjs/WebPage2.html - Windows Internet Expl
http://localhost/HandleManyWebObjs/WebPage2.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton1
Html
Message from webp...
8
OK

Let’s use SetOrCreateNamedValue to modify values at runtime:

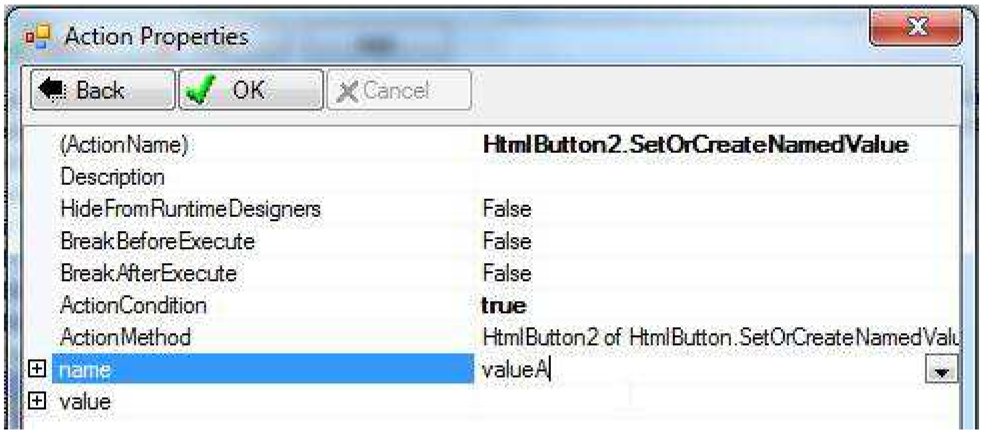




WebPage2 from WebPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
Html Document
HtmlButton1 of HtmlButton
HtmlButton2 of HtmlButton
Instance Members
Properties inherited
Methods inherited
focus
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
SetOrCreateNamedValue(String,String)
Actions

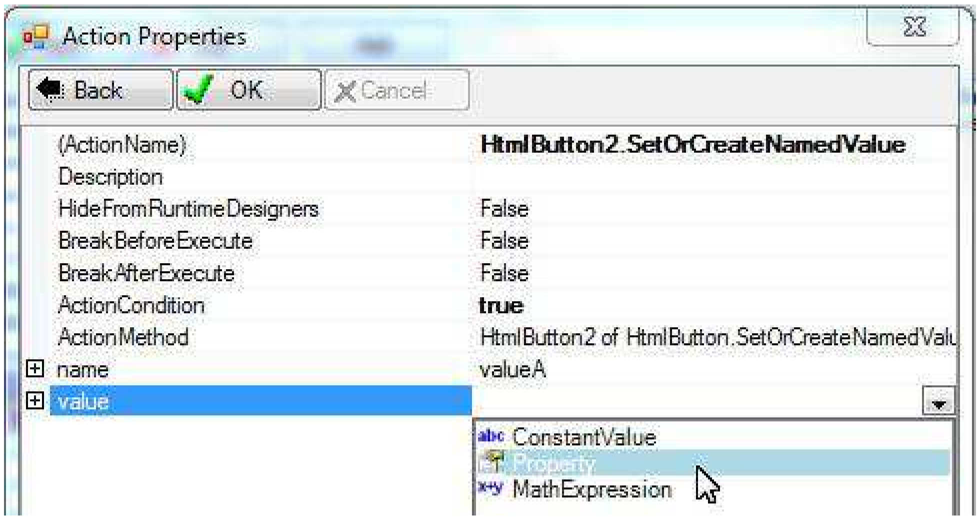
HtmlButton2 of HtmlButton.GetName
WebMessageBox.alert

Specify the “name” of the value:

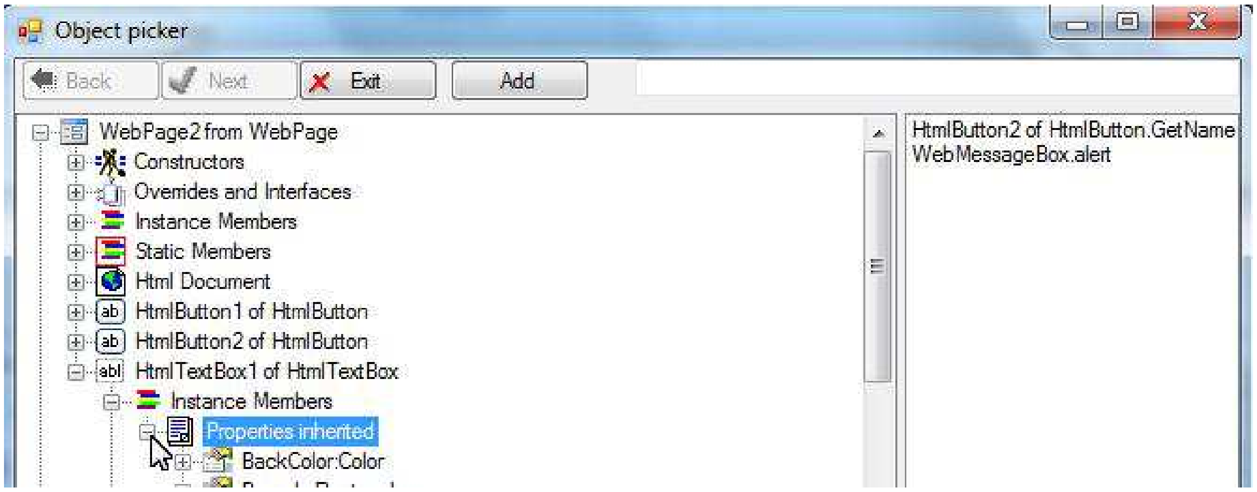


(ActionName)	**HtmlButton2.SetOrCreateNamedValue**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	HtmlButton2 of HtmlButton.SetOrCreateNamedValu
name	valueA
value	

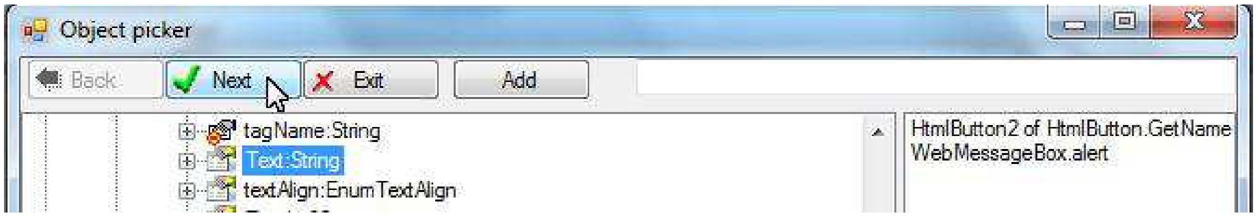
We use the Text property of a text box for the “value” of the action:



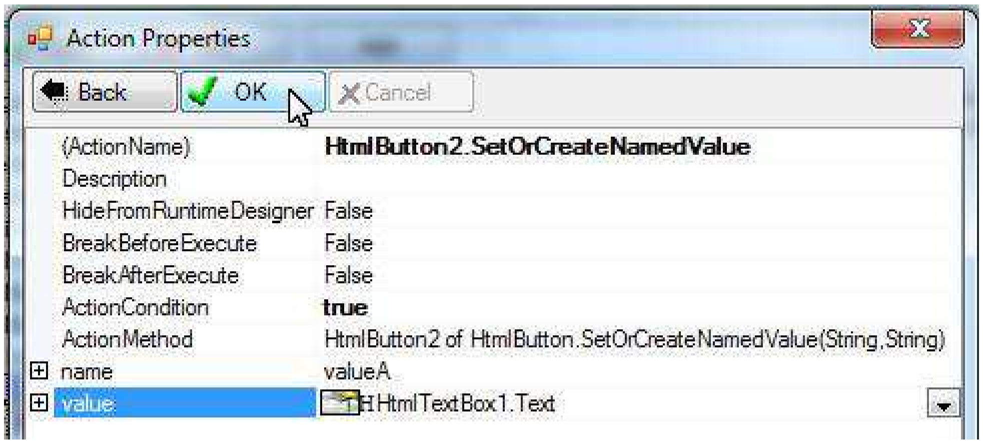
(ActionName) HtmlButton2.SetOrCreateNamedValue
Description
HideFromRuntimeDesigners False
BreakBeforeExecute False
BreakAfterExecute False
ActionCondition **true**
ActionMethod HtmlButton2 of HtmlButton.SetOrCreateNamedValu
name valueA
value



HtmlButton2 of HtmlButton.GetName
WebMessageBox.alert

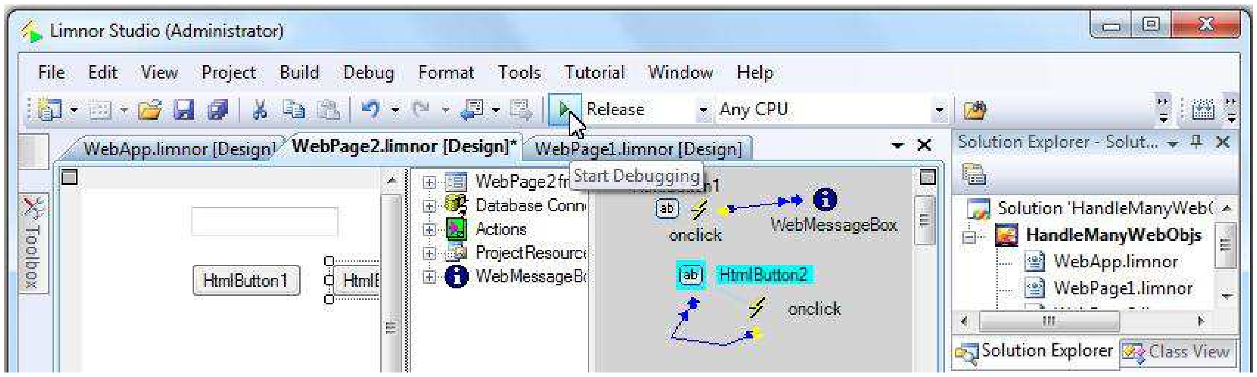


HtmlButton2 of HtmlButton.GetName
WebMessageBox.alert

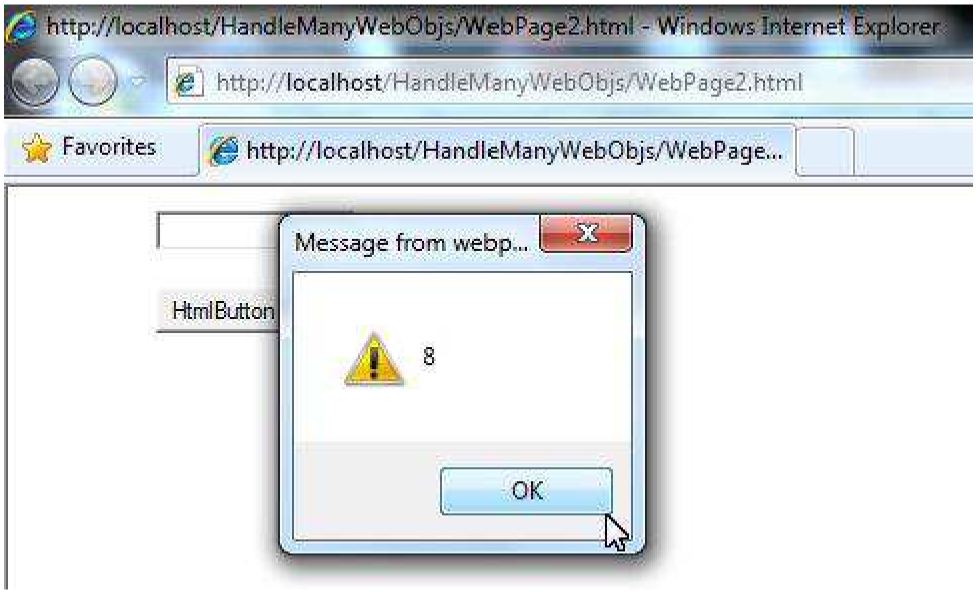


(ActionName)	**HtmlButton2.SetOrCreateNamedValue**
Description	
HideFromRuntimeDesigner	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	HtmlButton2 of HtmlButton.SetOrCreateNamedValue(String,String)
name	valueA
value	HtmlTextBox1.Text

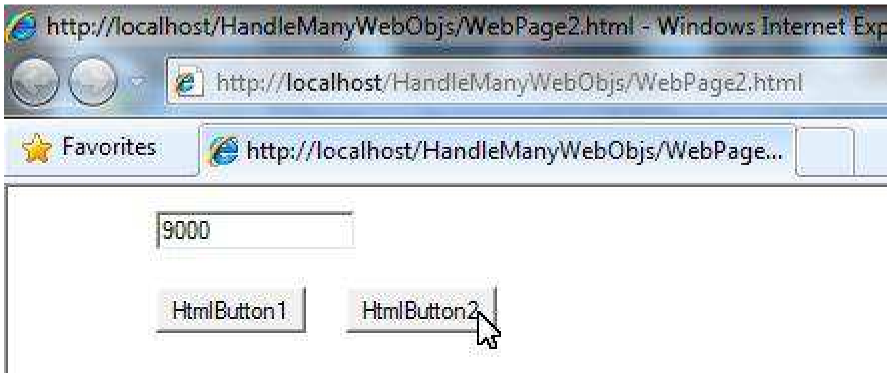
We may test it:



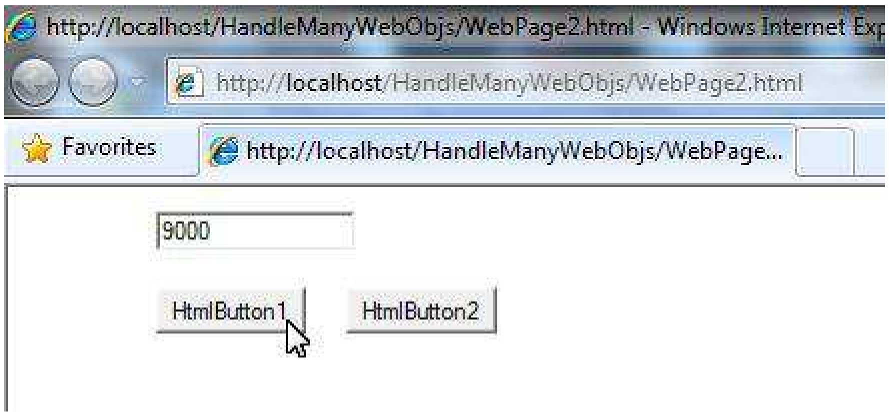
Click button 1, we still see the value set at design time:


http://localhost/HandleManyWebObjs/WebPage2.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage2.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
HtmlButton
Message from webp...
8
OK

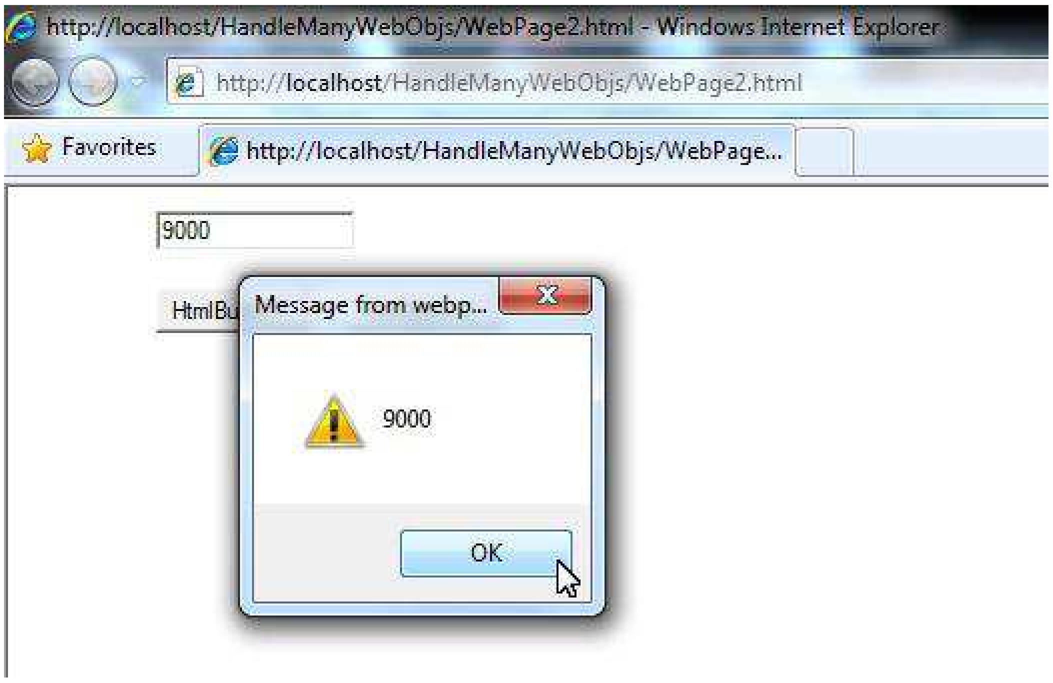
Now we type in some value in the text box and click button 2:


http://localhost/HandleManyWebObjs/WebPage2.html - Windows Internet Exp
http://localhost/HandleManyWebObjs/WebPage2.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
9000
HtmlButton1
HtmlButton2

According to our programming, the value should have been passed to the value named “valueA”. Let’s click button 1 to show it:


http://localhost/HandleManyWebObjs/WebPage2.html - Windows Internet Exp
http://localhost/HandleManyWebObjs/WebPage2.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
9000
HtmlButton1
HtmlButton2

As we expected it, the value is changed:


http://localhost/HandleManyWebObjs/WebPage2.html - Windows Internet Explorer
http://localhost/HandleManyWebObjs/WebPage2.html
Favorites
http://localhost/HandleManyWebObjs/WebPage...
9000
HtmlBu
Message from webp...
OK

```
9000
```

Feedbacks

Please send your feedbacks to support@limnor.com. Thanks!