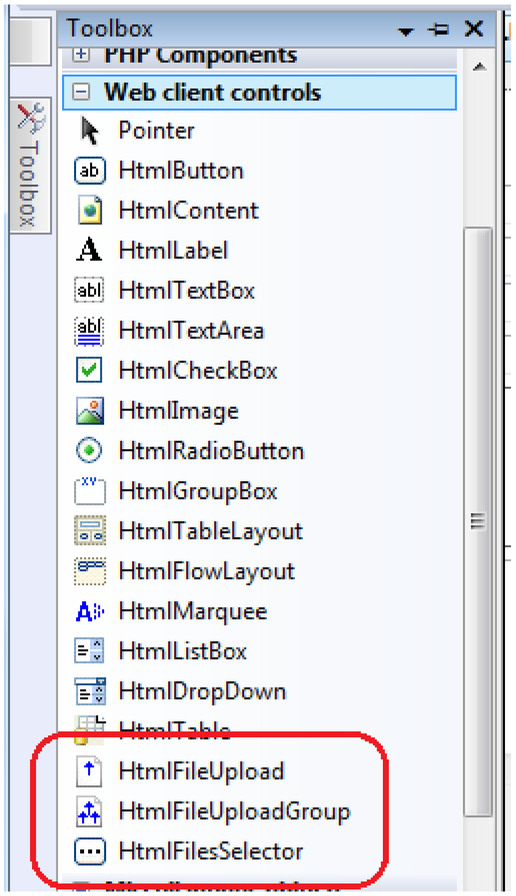
File Upload from Web Pages

Contents

Introduction

HtmlFileUpload can be used to upload a file from web page. HtmlFileUploadGroup may group many HtmlFileUpload controls to upload many files in one operation. HtmlFilesSelector can be used to select one or more files and upload the files as HtmlFileUploadGroup does, or it can be used by SendMail component so that the selected files are used as email attachments.

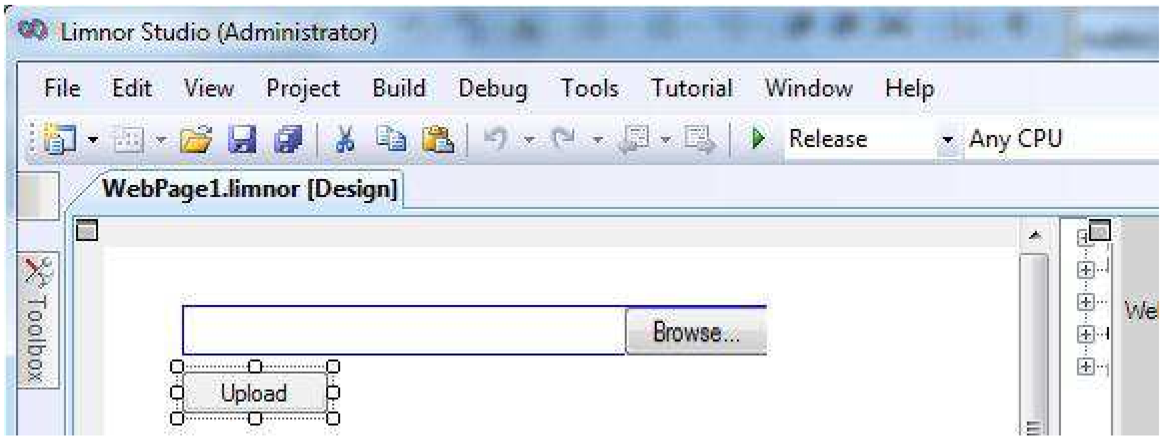


File-upload related procedures described in this document can be used for both PHP and ASPX web applications. The samples demonstrate the usages of HtmlFileUpload and HtmlFileUploadGroup. Web mail sample (http://www.limnor.com/support/WebEmail.pdf) shows usage pf HtmlFilesSelector.

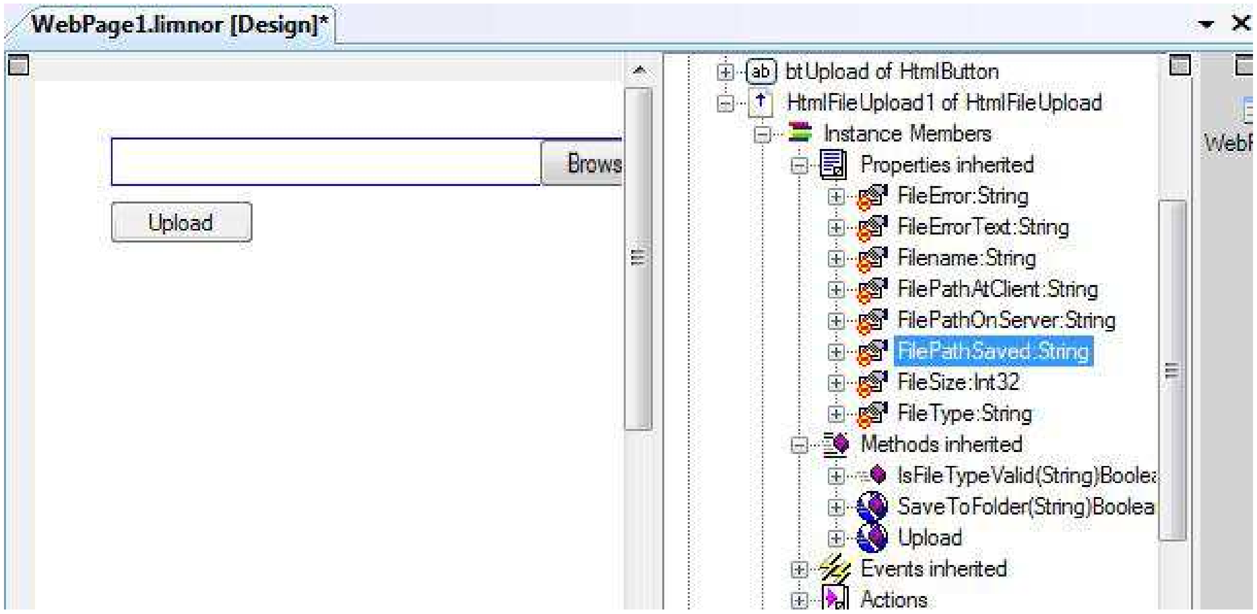
Upload Single File

Upload file to server

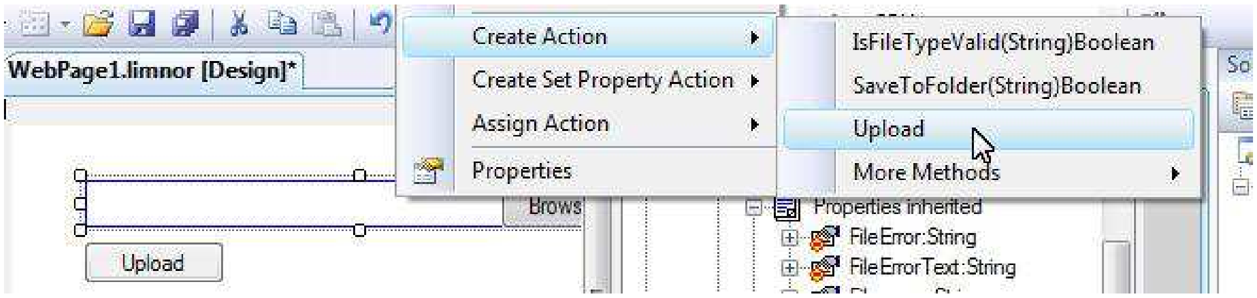
Drop an HtmlFileUpload to the web page. Add a button for initiate file uploading action:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window Help
Release
Any CPU
WebPage1.limnor [Design]
Toolbox
Browse...
Upload

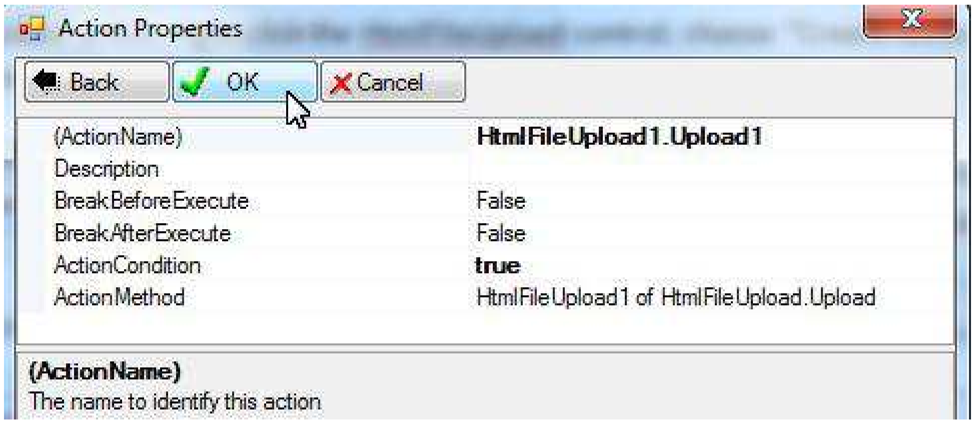
We may create an Upload action to upload the file to the web server. We may create a SaveToFolder action to save the uploaded file to a folder on the web server. FilePathSaved property is the file path on the web server after executing SaveToFolder.



To create an Upload action, right-click the HtmlFileUpload control; choose “Create Action”; choose “Upload” method:



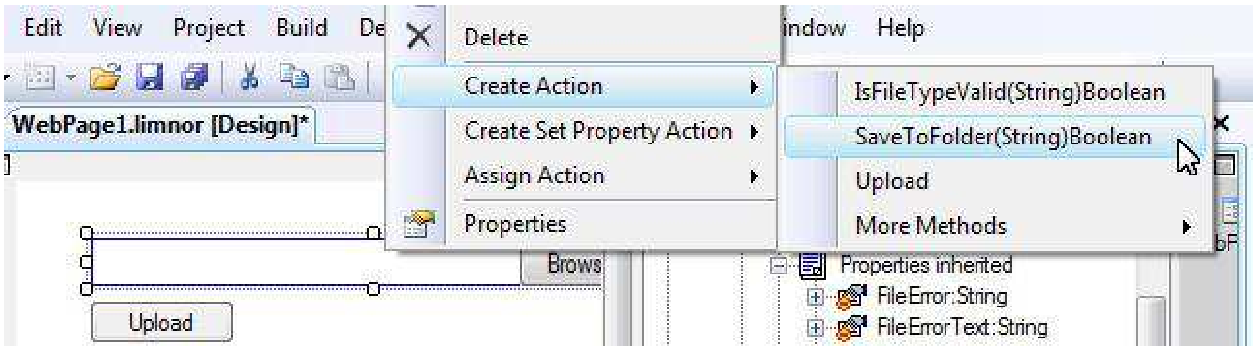
Click OK to finish creating this action:



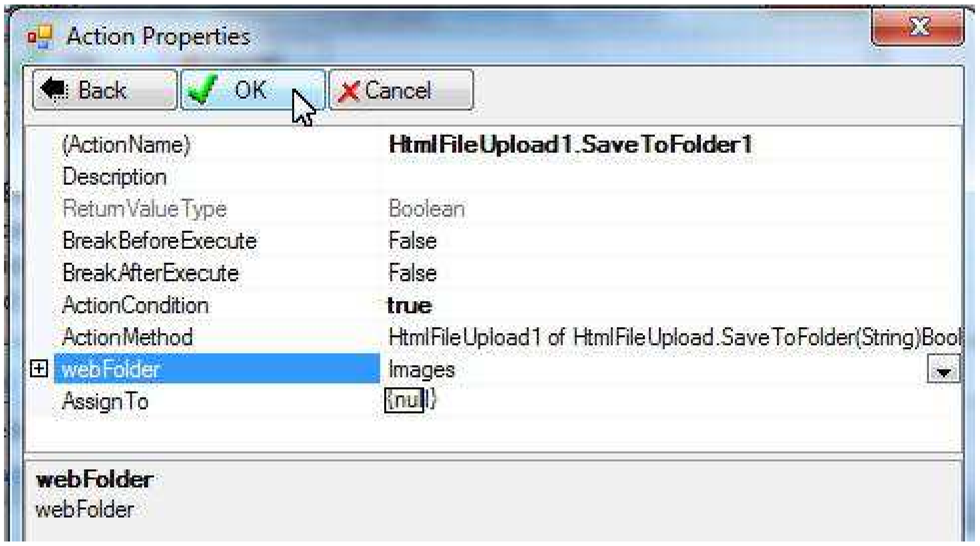
(ActionName)	HtmlFileUpload1.Upload1
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlFileUpload1 of HtmlFileUpload.Upload

(ActionName)
The name to identify this action

To create a SaveToFolder action; right-click the HtmlFileUpload control; choose “Create Action”; choose “SaveToFolder” method:



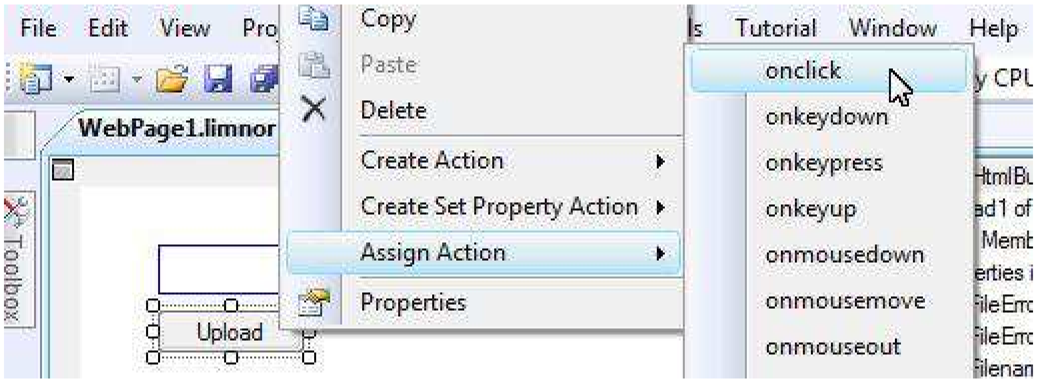
Specify a folder name on the web server. We used “Images” here. Click OK to finish creating this action:



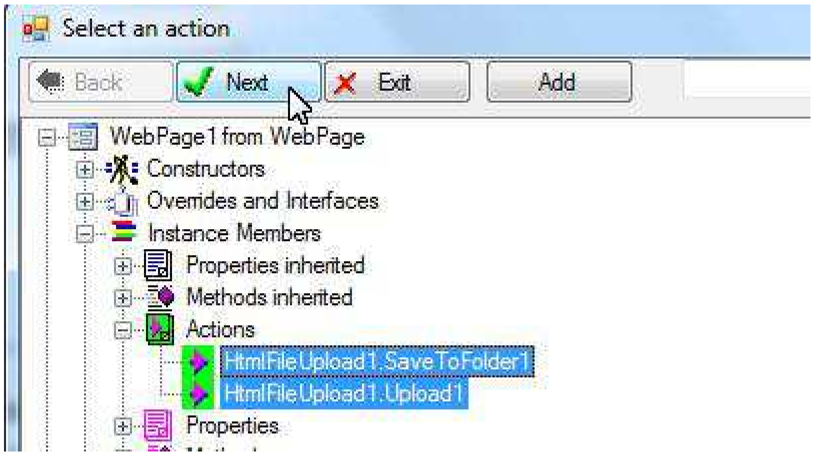
(ActionName)	HtmlFileUpload1.SaveToFolder1
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlFileUpload1 of HtmlFileUpload.SaveToFolder(String)Bool
webFolder	Images
AssignTo	{null}

webFolder
webFolder

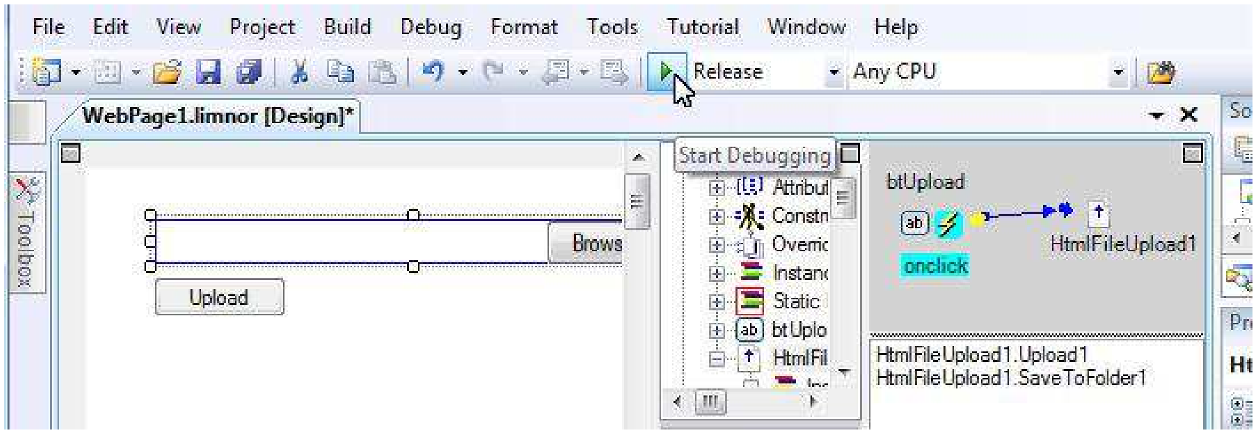
We want these two actions to be executed when the visitor clicks the button. Right-click the button; choose “Assign Action”; choose “onclick” event:



Choose the actions and click “Next”:

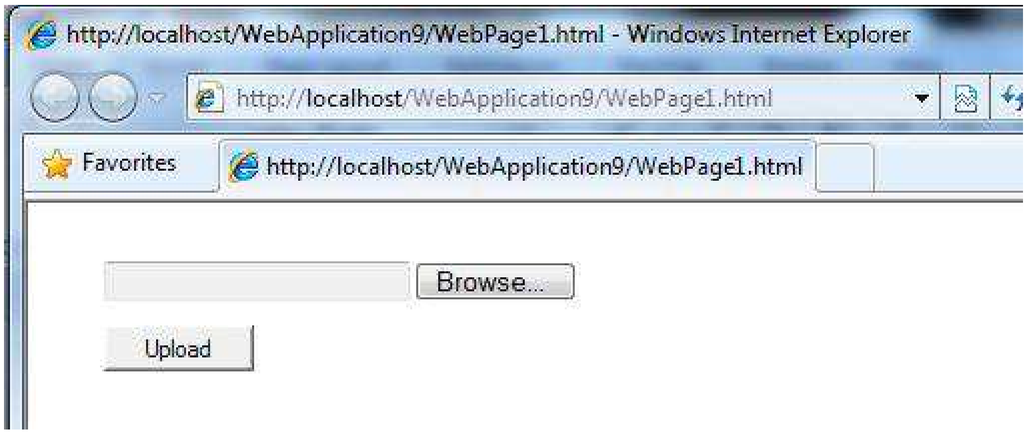


The actions are assigned to the button. Make sure that the Upload action is before the SaveToFolder action. We may test the web application now.

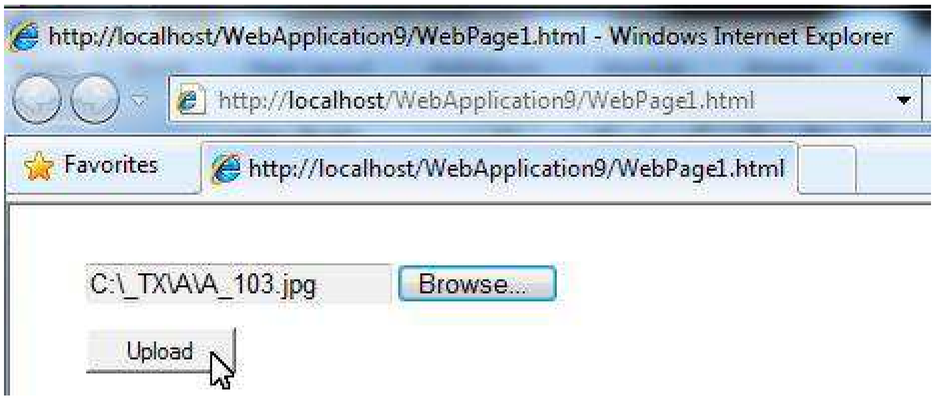


HtmlFileUpload1.Upload1
HtmlFileUpload1.SaveToFolder1

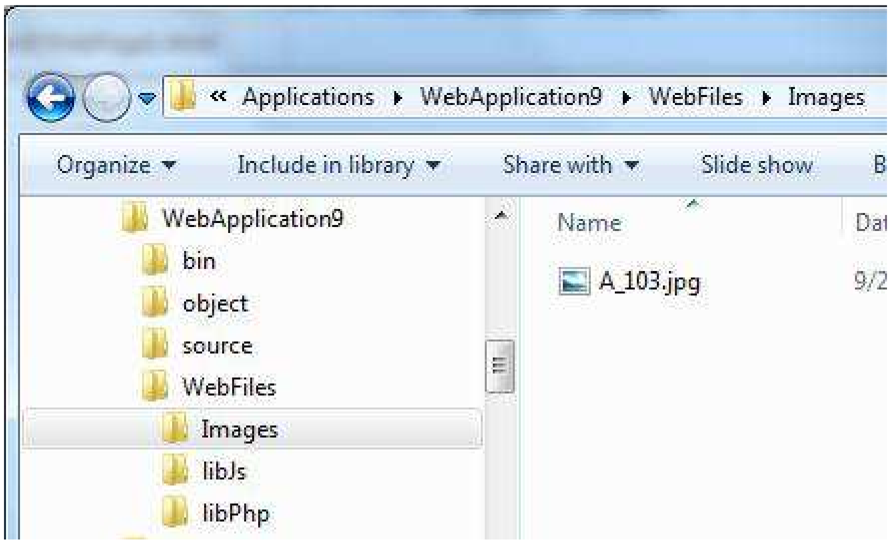
The web page appears:


http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
Browse...
Upload

Select a file and click the button.

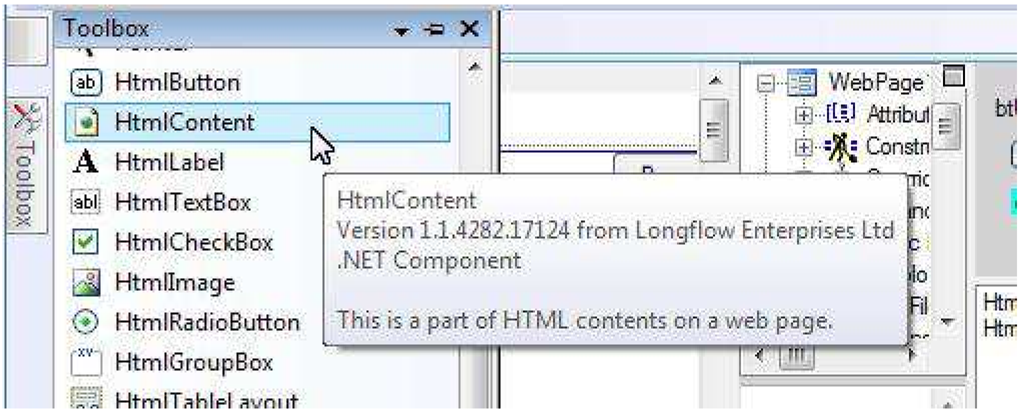

http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
C:_TX\A\A_103.jpg
Browse...
Upload

We can see that the file is uploaded to the Images folder:



Use uploaded file name at client

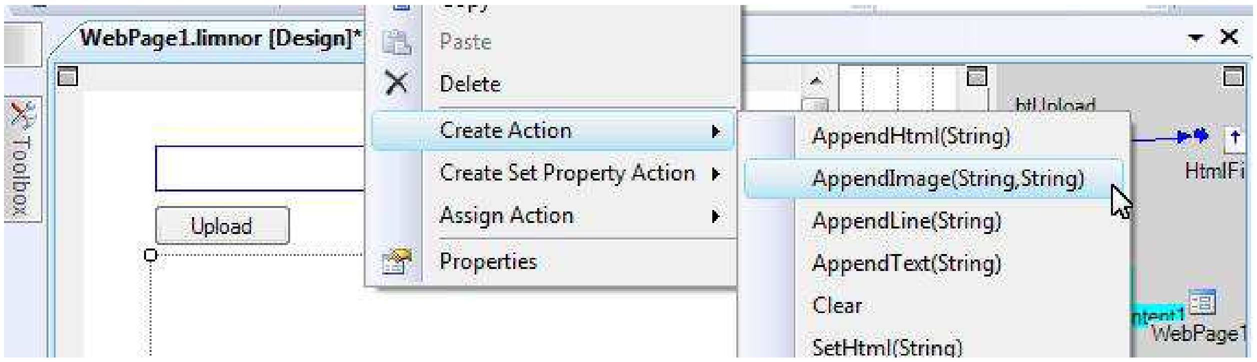
FilePathSaved property can be used at client after file upload. Let's use an HtmlContent control to demonstrate it. Note that such file upload related properties are only available during the same event handling processing immediately after the file uploading. For example, you cannot use one button to do file uploading and use another button to access the FilePathSaved property.



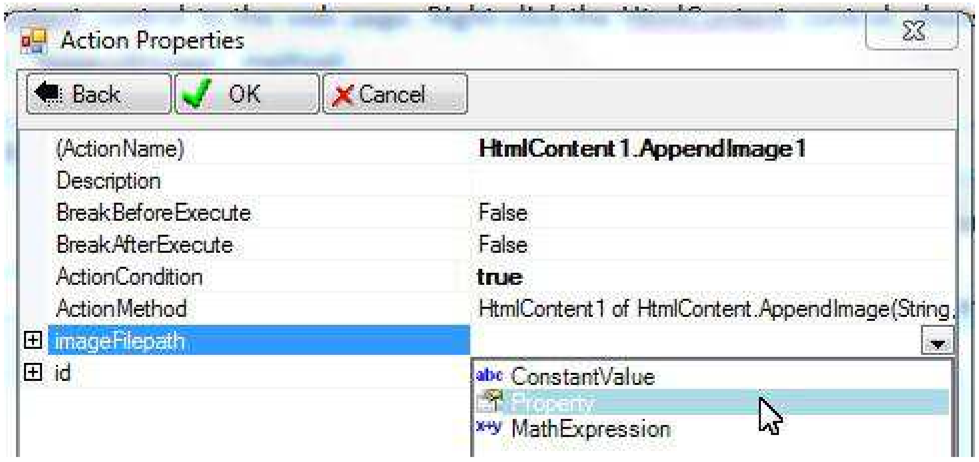
HtmlContent
Version 1.1.4282.17124 from Longflow Enterprises Ltd
.NET Component

This is a part of HTML contents on a web page.

Drop an HtmlContent control to the web page. Right-click the HtmlContent control; choose "Create Action"; choose "AppendImage" method:

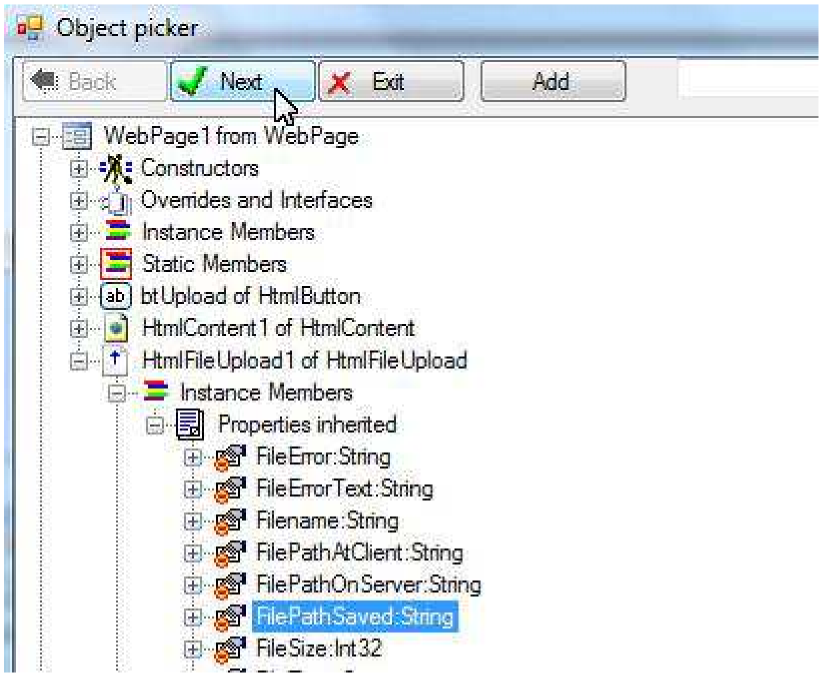


Select "Property" for the "imageFilepath" parameter:

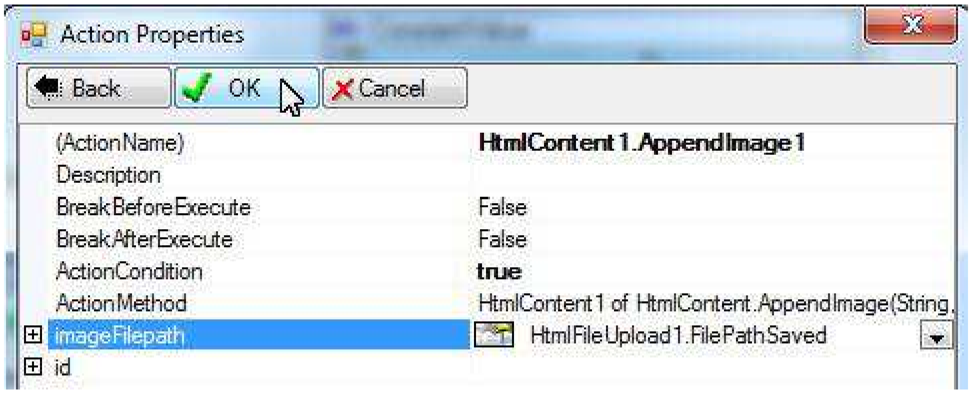


(ActionName)	HtmlContent1.AppendImage1
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlContent1 of HtmlContent.AppendImage(String,
imageFilepath	
id	

Select FilePathSaved property of the HtmlFileUpload control. Click "Next":

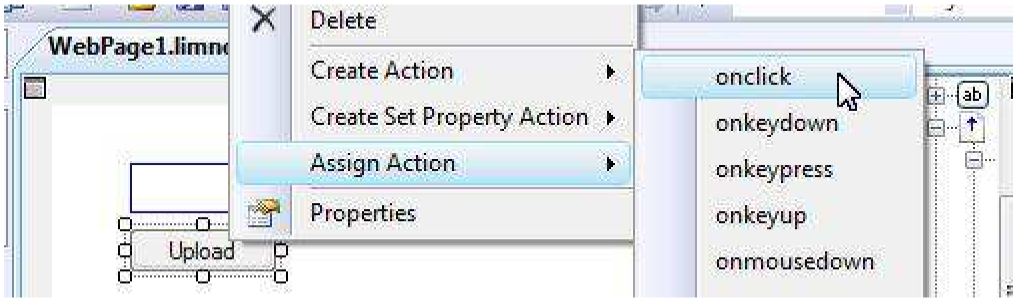


Click OK to finish creating this action:

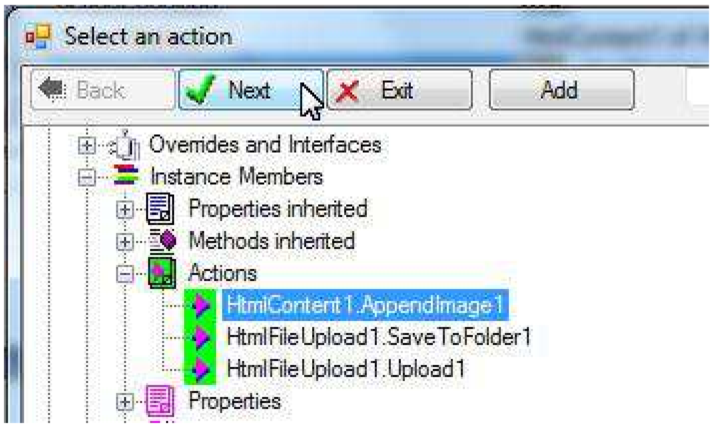


(ActionName)	**HtmlContent1.AppendImage1**
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	HtmlContent1 of HtmlContent.AppendImage(String,
imageFilepath	HtmlFileUpload1.FilePathSaved
id	

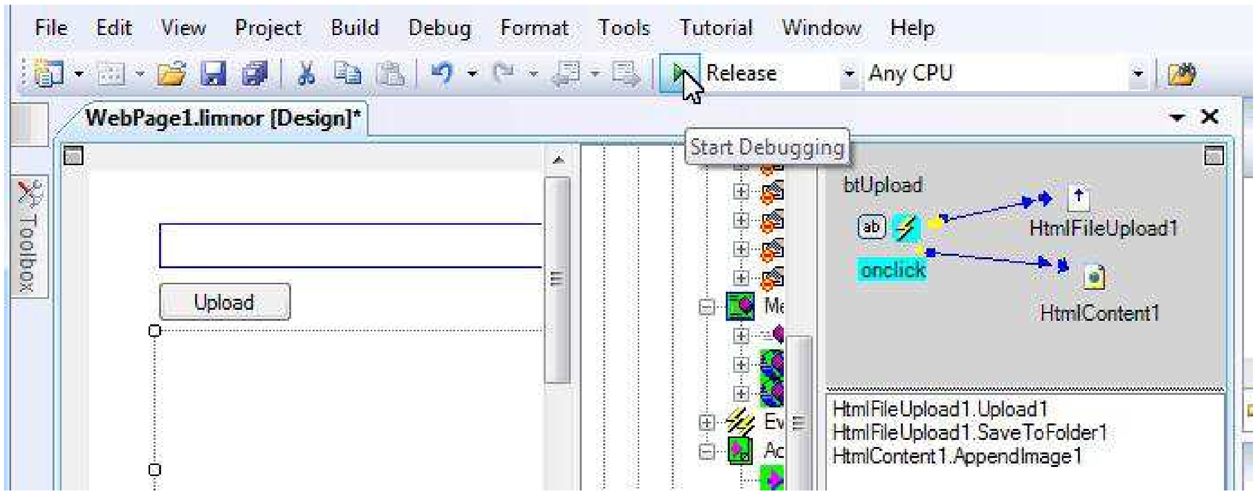
Assign this action to the button as well. Right-click the button; choose “Assign Action”; choose “onclick”:



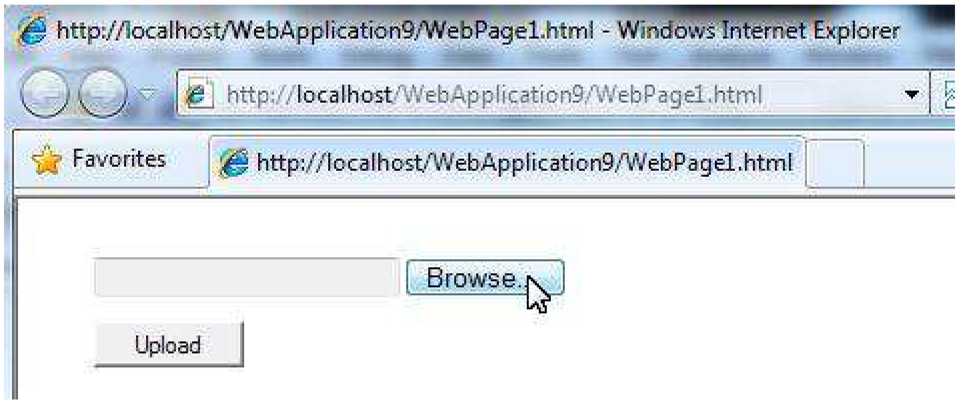
Choose the new action and click “Next”:



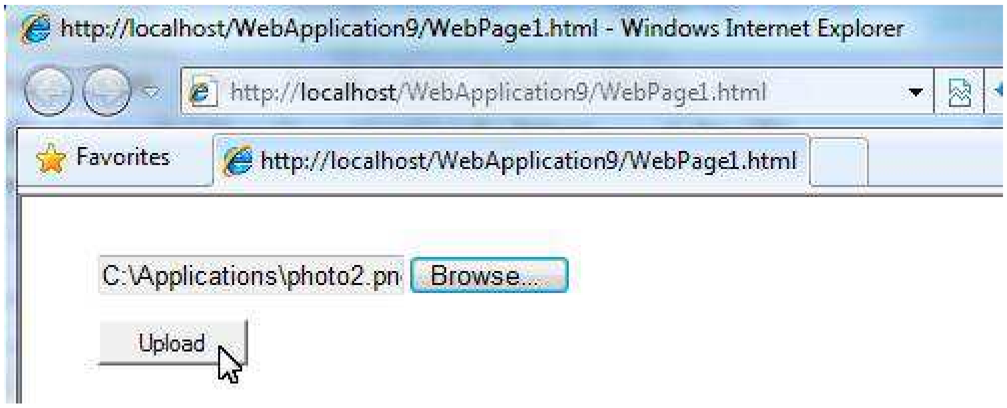
We may test the web application again:



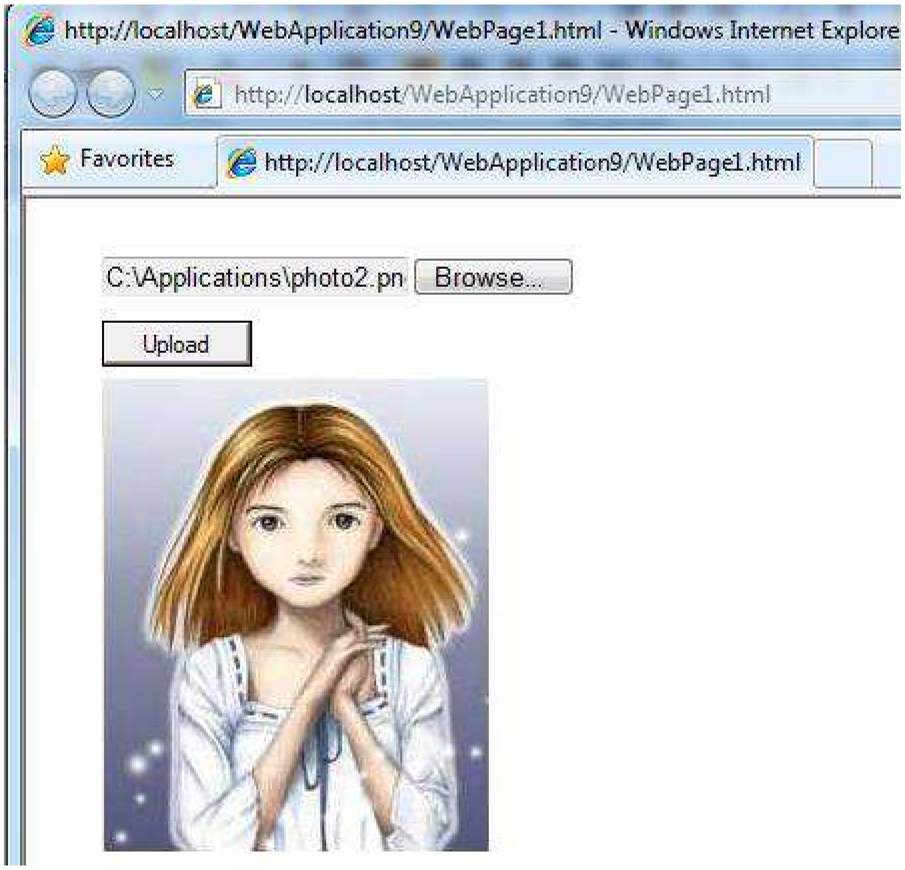
The web page appears:


http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
Browse...
Upload

Select a file and click the Upload button.


http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
C:\Applications\photo2.pn
Browse...
Upload

The image file is uploaded. The image also appears on the web page:

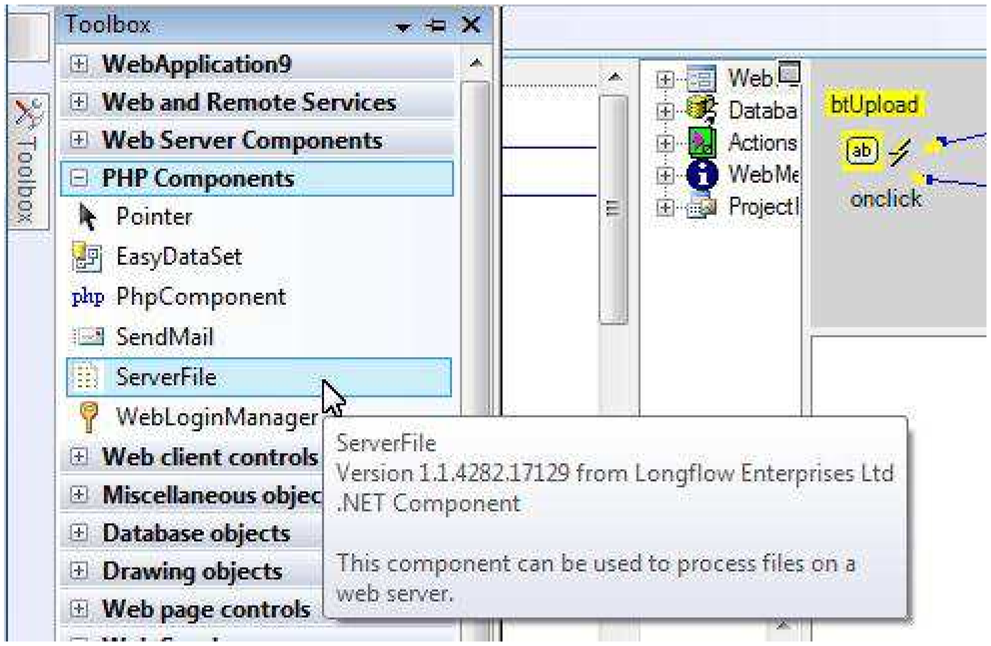

http://localhost/WebApplication9/WebPage1.html - Windows Internet Explore
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
C:\Applications\photo2.pn
Browse...
Upload

Use uploaded file name at server

FilePathSaved property can be used at server after file upload. Even though HtmlFileUpload component is used in the same way for PHP and Aspx, other server processing is usually different for PHP and Aspx. Suppose we want to write file names of uploaded files to a log file. For PHP we may use ServerFile component to do it. For Aspx, we may use StreamWriter class to do it. Below we show both ways.

Use PHP to do logging

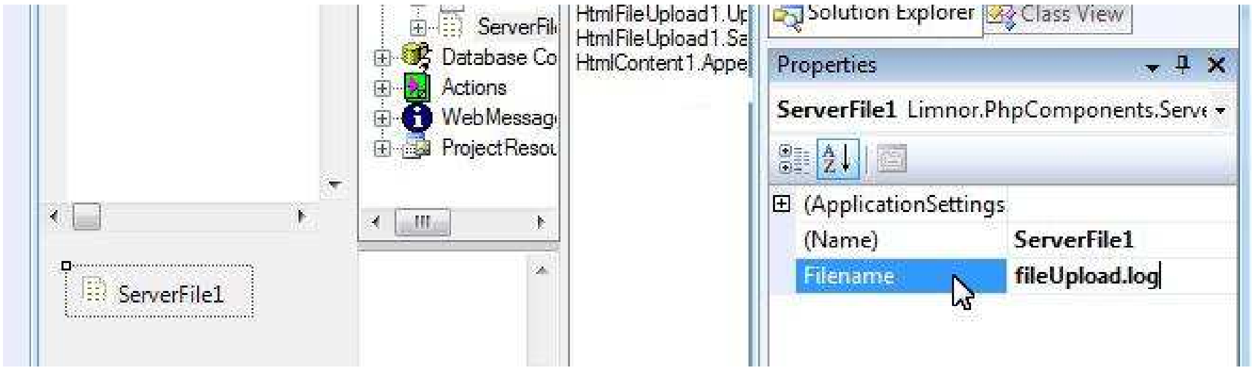
Let's use a **ServerFile** component to demonstrate it on a PHP web project. We'll show the same demonstration on an ASPX project later. Note that such file upload related properties are only available during the same event handling processing immediately after the file uploading. For example, you cannot use one button to do file uploading and use another button to access the FilePathSaved property.



ServerFile
Version 1.1.4282.17129 from Longflow Enterprises Ltd
.NET Component

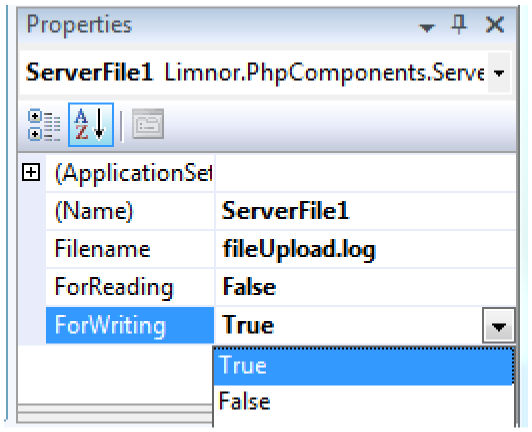
This component can be used to process files on a web server.

Drop a ServerFile component to the web page. Specify a file name on the web server:



(ApplicationSettings	
(Name)	ServerFile1
Filename	fileUpload.log

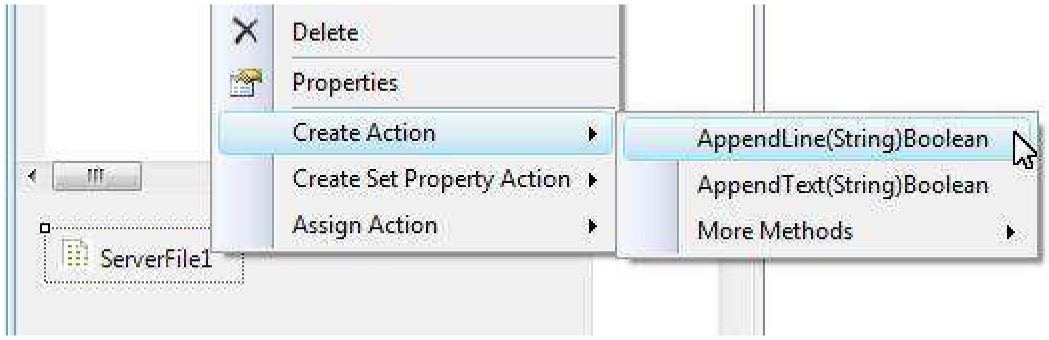
Set its ForWriting to True:


Properties

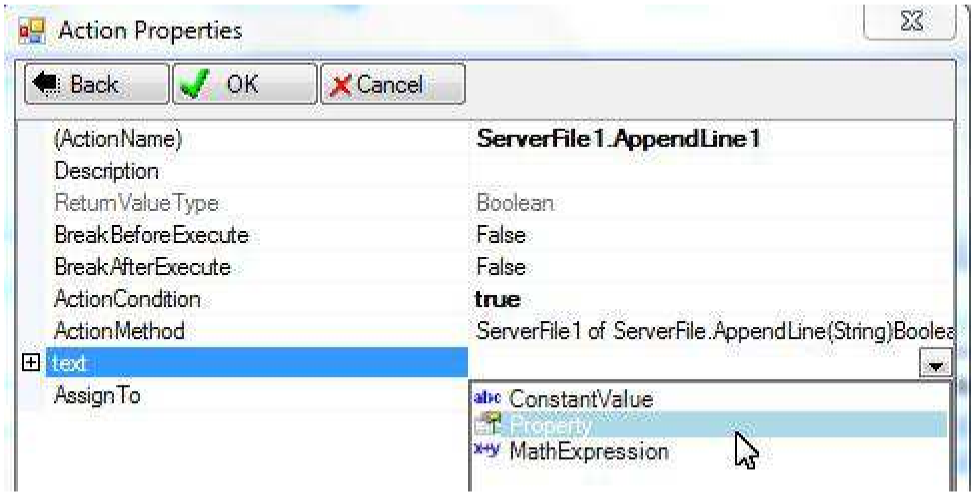
ServerFile1 Limnor.PhpComponents.Serve

(ApplicationSet	
(Name)	ServerFile1
Filename	fileUpload.log
ForReading	False
ForWriting	True

Create an AppendLine action:

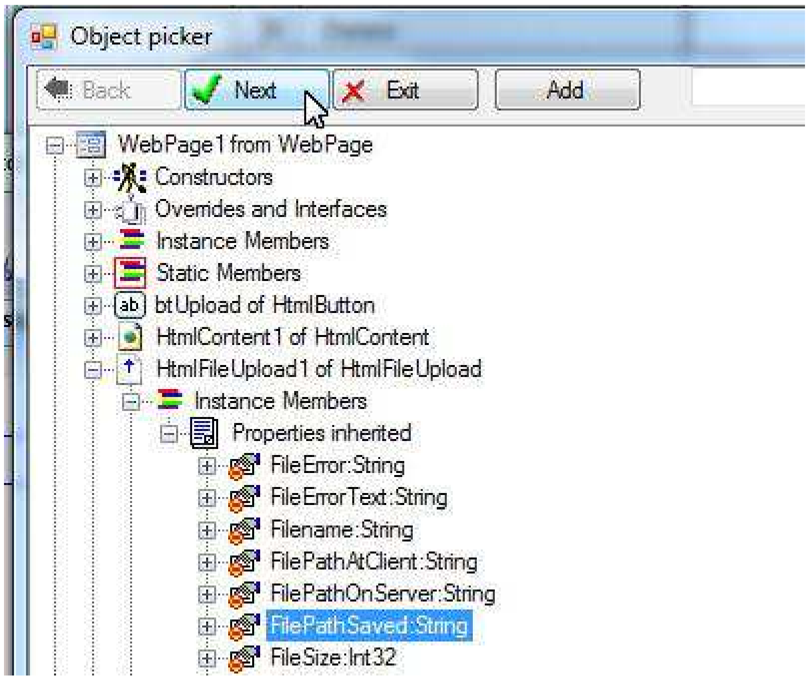

Delete
Properties
Create Action
Create Set Property Action
Assign Action
AppendLine(String)Boolean
AppendText(String)Boolean
More Methods
ServerFile1

Select “Property” for the “text” parameter of the action:

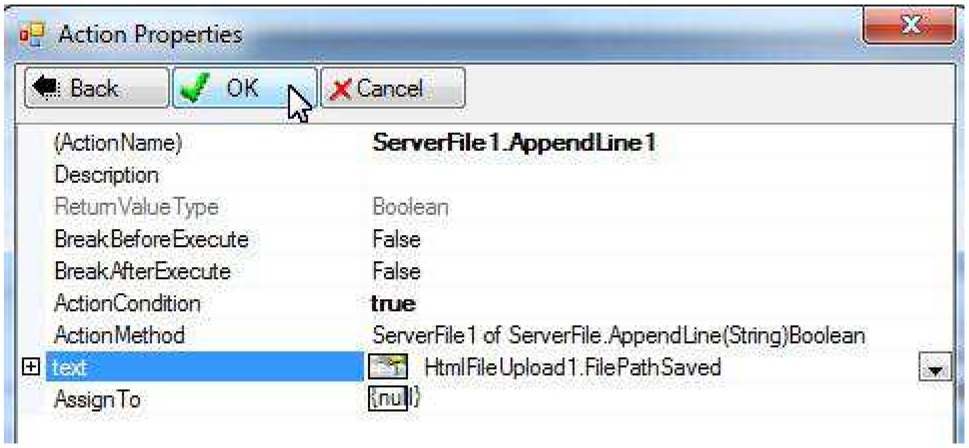

Action Properties

(ActionName)	ServerFile1.AppendLine1
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ServerFile1 of ServerFile.AppendLine(String)Boolea
text	
AssignTo	

Select the FilePathSaved property of the HtmlFileUpload control:

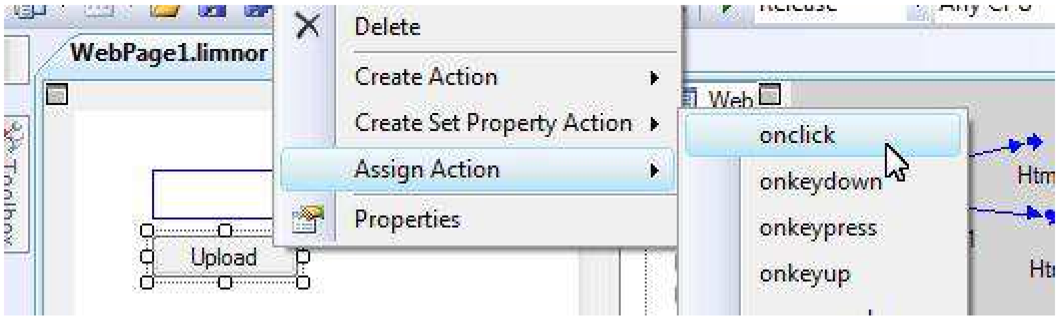


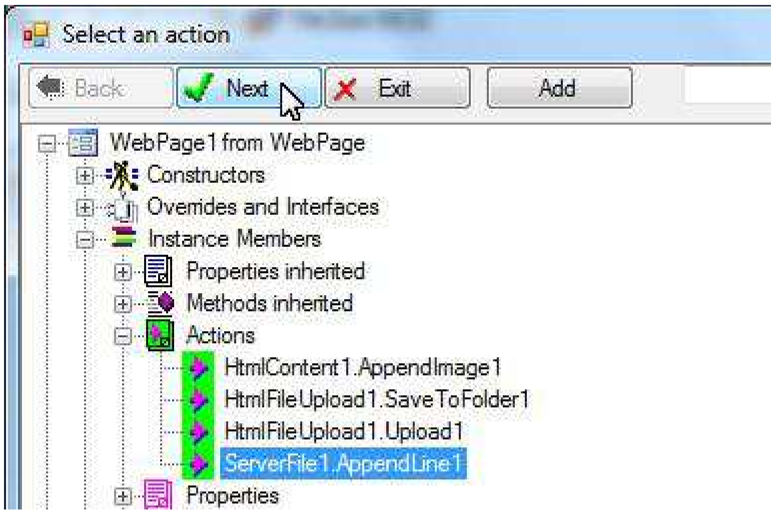
Click OK to finish creating this action:



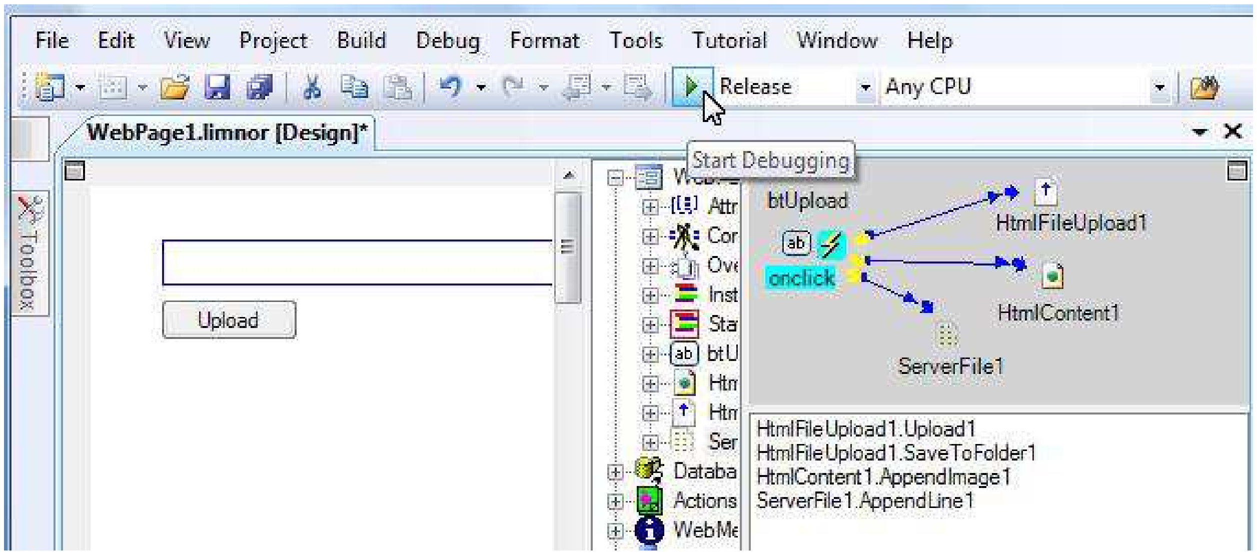
(ActionName)	ServerFile1.AppendLine1
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ServerFile1 of ServerFile.AppendLine(String)Boolean
text	HtmlFileUpload1.FilePathSaved
AssignTo	{null}

Also assign this action to the Upload button:

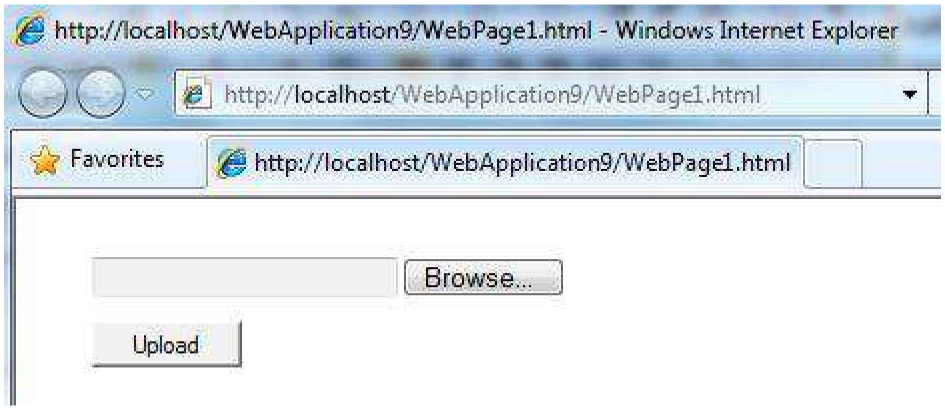




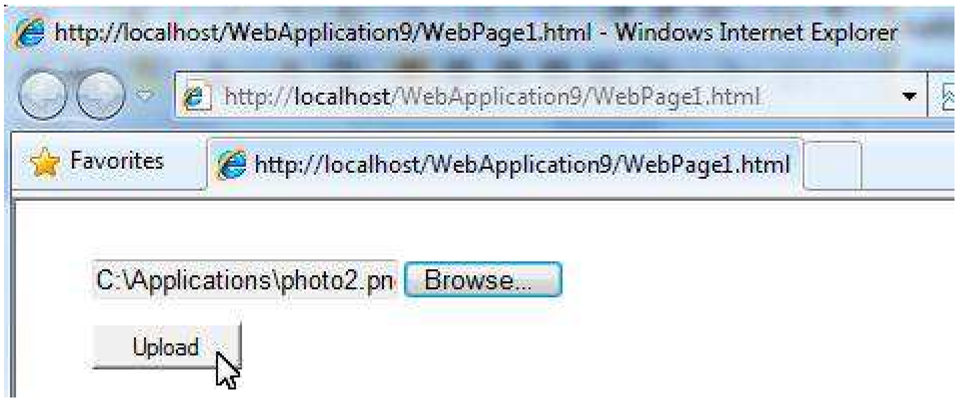
We may test the web application now:



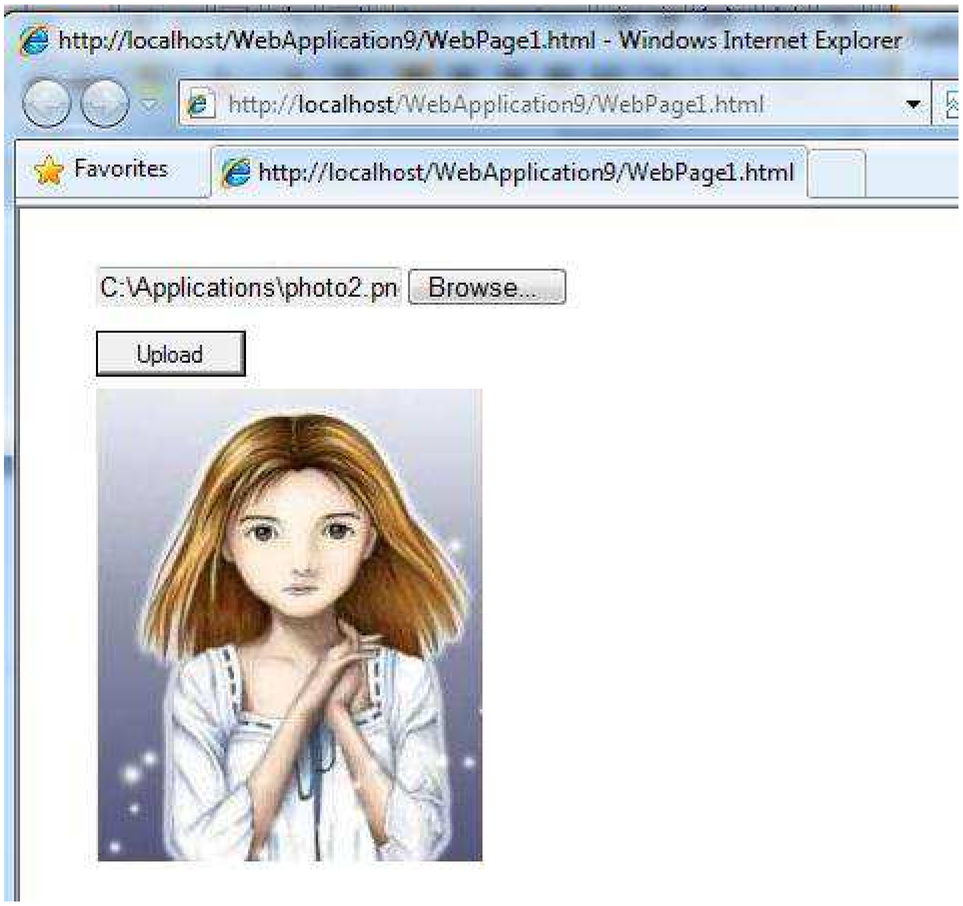
The web page appears:


http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
Browse...
Upload

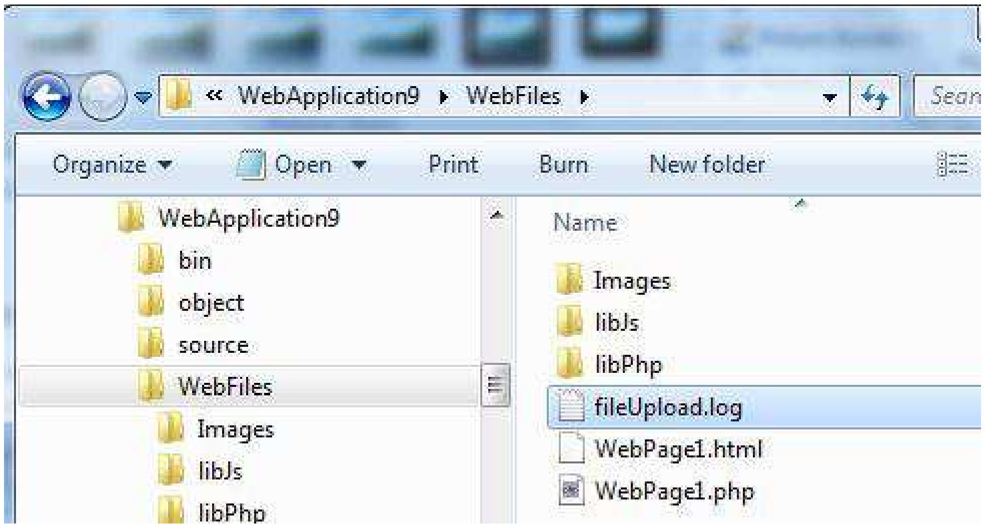
Select an image file and click Upload:


http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
C:\Applications\photo2.pn
Browse...
Upload

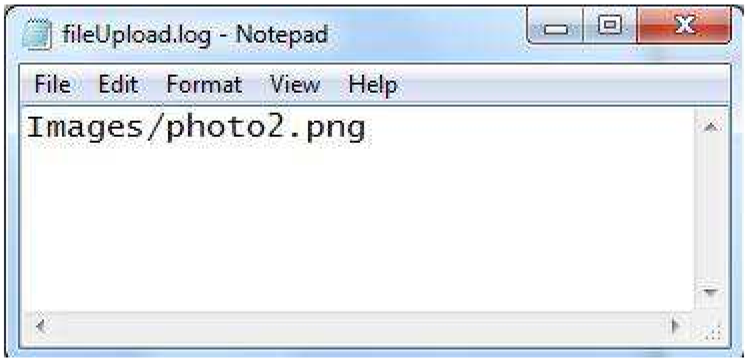
The file is uploaded. The image appears on the web page:


http://localhost/WebApplication9/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication9/WebPage1.html
Favorites
http://localhost/WebApplication9/WebPage1.html
C:\Applications\photo2.pn
Browse...
Upload

We also see a file named fileUpload.log appears at the web server:



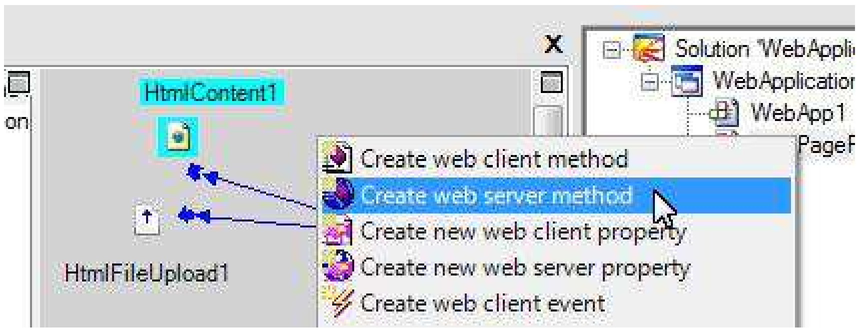
Open the file; we can see that the file name for the uploaded file is in the file:



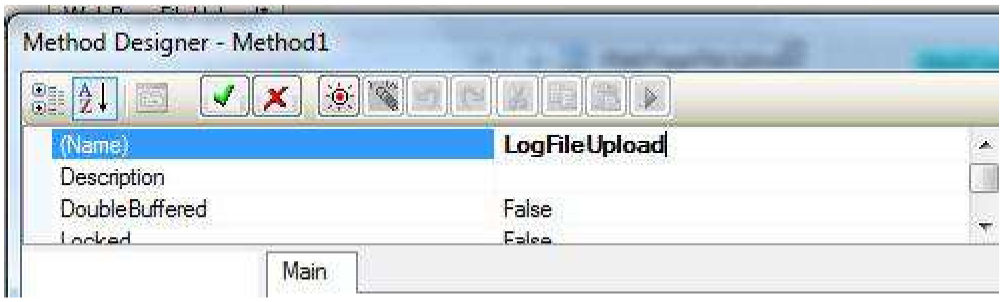
```
Images/photo2.png
```

Use ASPX to do logging

For an ASPX web application, we may use most classes in .Net Framework for server side processing. For this demonstration, let's use StreamWriter class to write file on server. Let's create a server side method to do it:

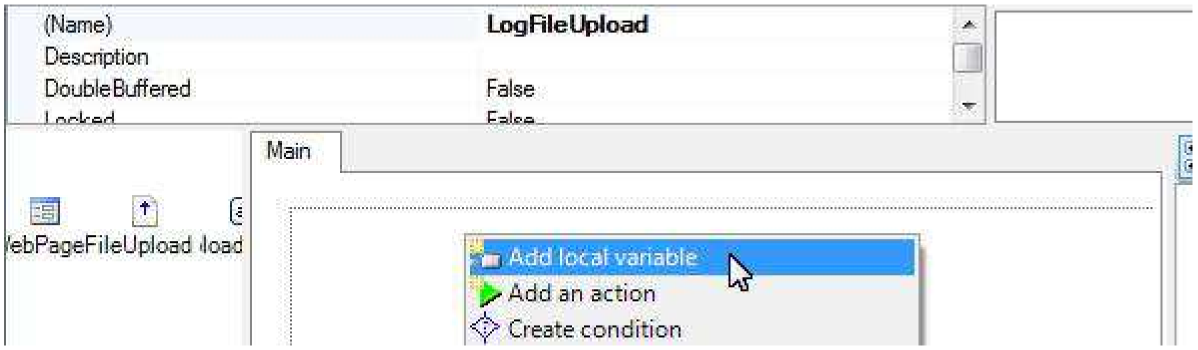


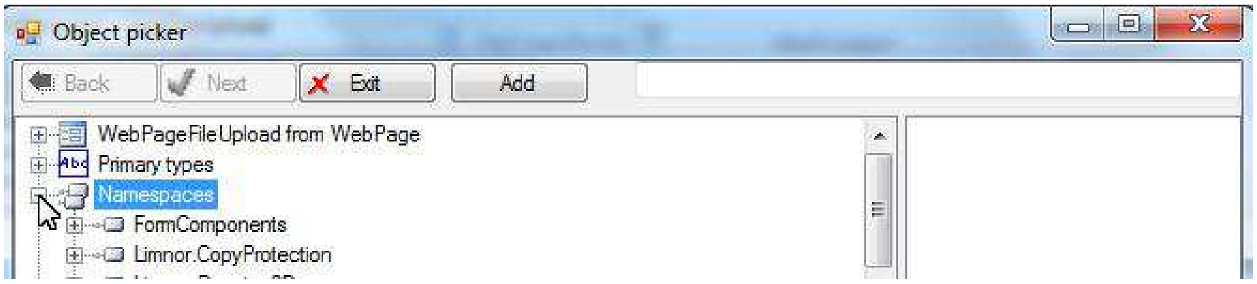
Rename it to "LogFileUpload":

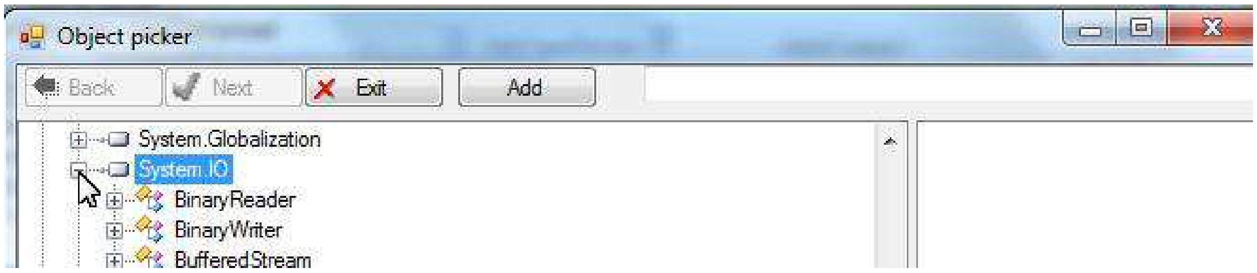


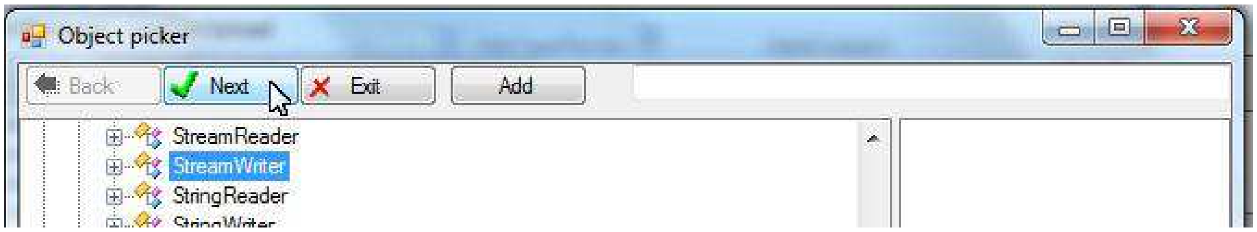
(Name)	LogFileUpload
Description	
DoubleBuffered	False
Locked	False

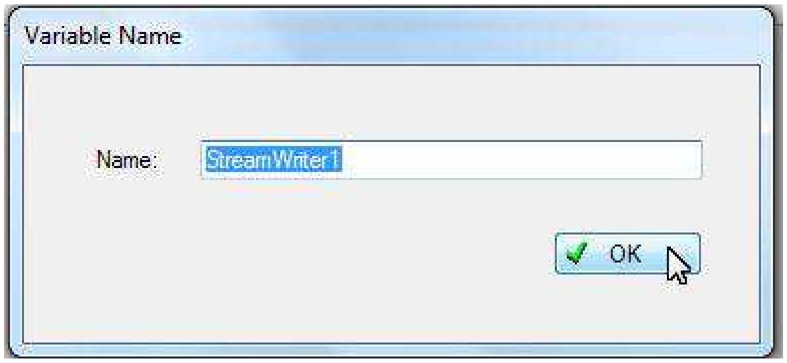
Create a StreamWriter to do the job:





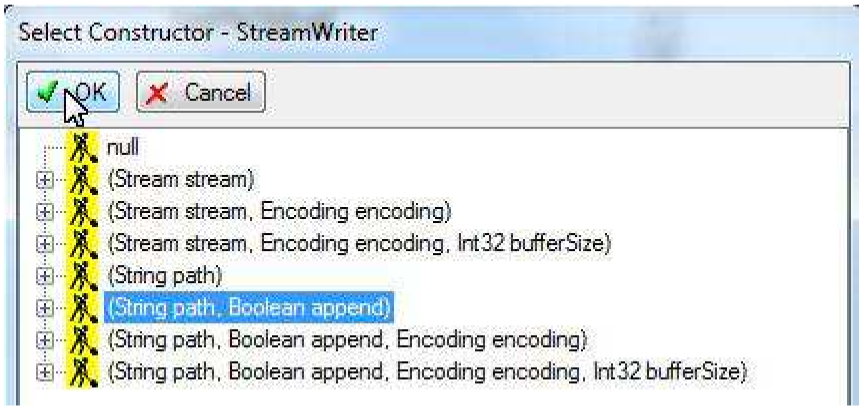






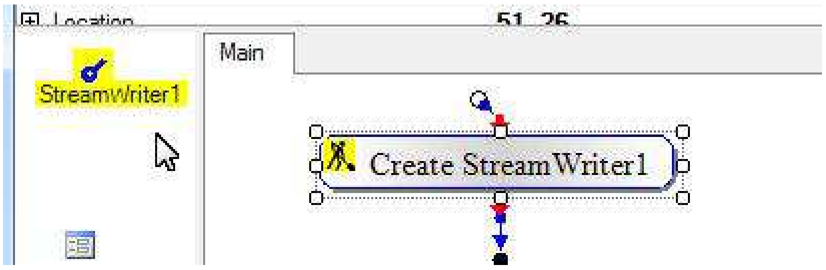
StreamWriter1

Select a constructor which accepts a file path and a flag indicating that we want to append the file:

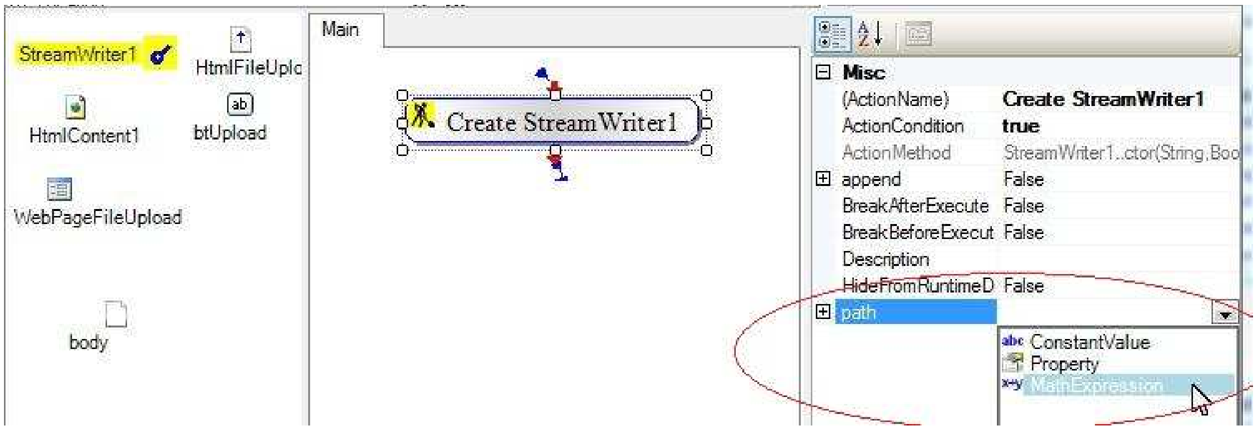


null
(Stream stream)
(Stream stream, Encoding encoding)
(Stream stream, Encoding encoding, Int32 bufferSize)
(String path)
(String path, Boolean append)
(String path, Boolean append, Encoding encoding)
(String path, Boolean append, Encoding encoding, Int32 bufferSize)

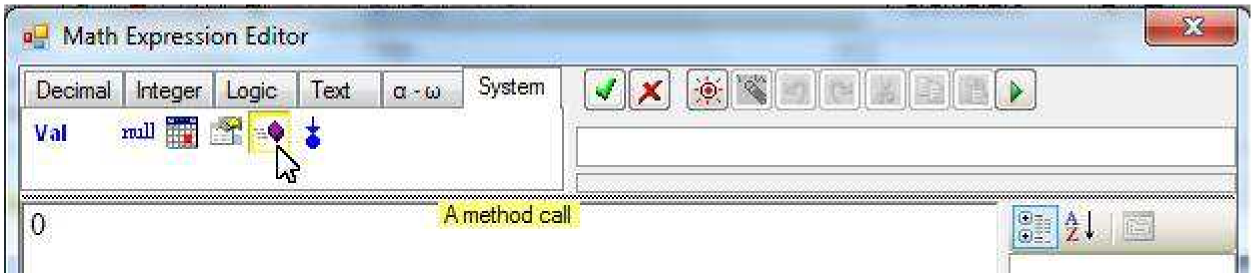
A variable of StreamWriter is created and an action appears for creating the variable:

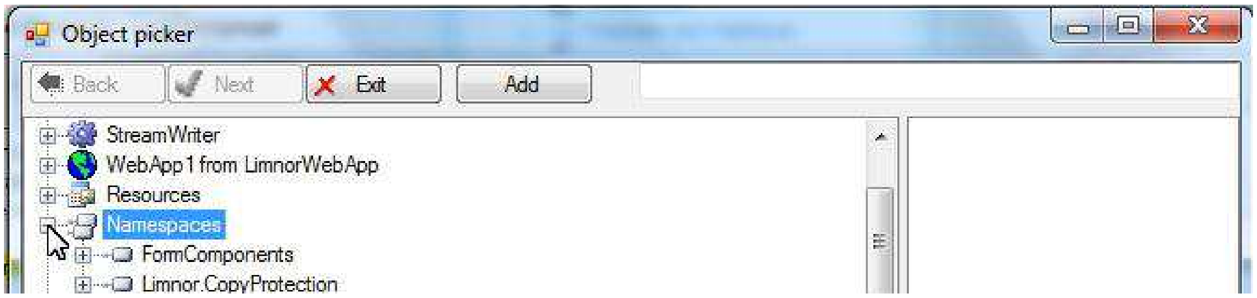


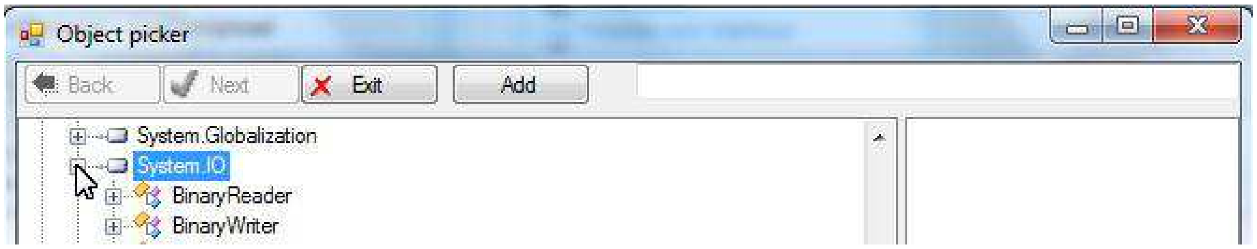
Suppose our log file name is “FileUpload.log”. Where should we save the file? For this sample, we simply save it on the same folder as the web application, which is represented by property `AspxPhysicalFolder` of the web page. We may use Path class’ Combine method to form the full path of the log file.

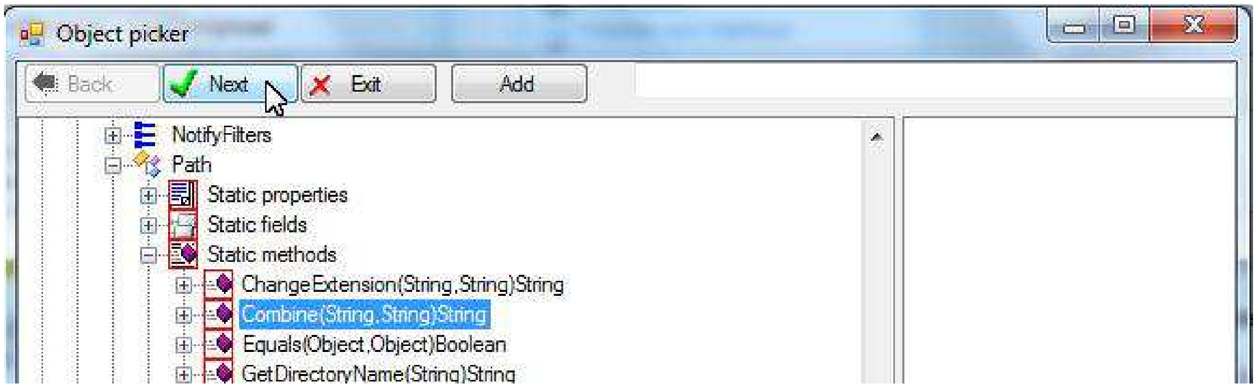


(ActionName)	Create StreamWriter1
ActionCondition	true
ActionMethod	StreamWriter1..ctor(String,Boo
append	False
BreakAfterExecute	False
BreakBeforeExecut	False
Description	
HideFromRuntimeD	False
path	



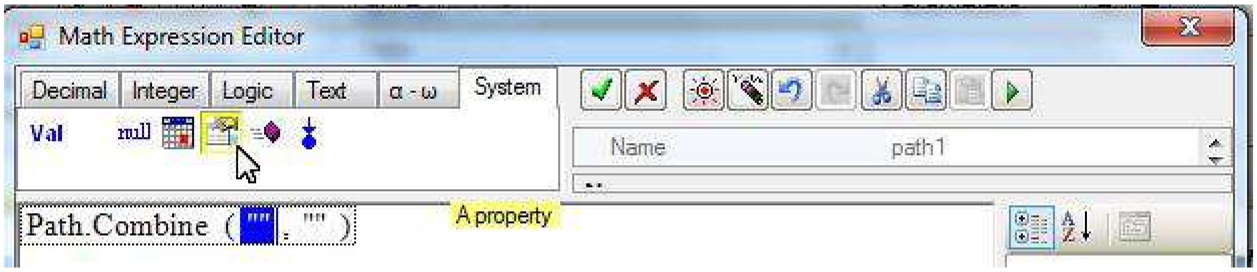






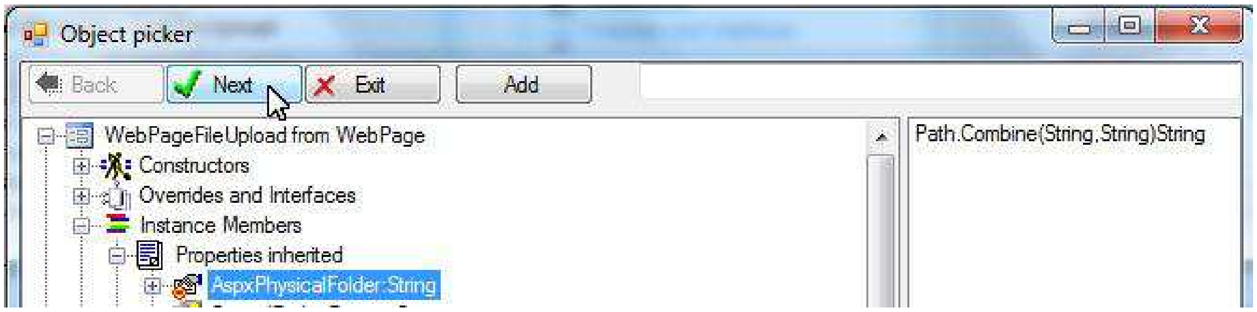
NotifyFilters
Path
Static properties
Static fields
Static methods
ChangeExtension(String,String)String
Combine(String,String)String
Equals(Object,Object)Boolean
GetDirectoryName(String)String

The first parameter of Combine is the folder. Let's use AspxPhysicalFolder:



Name path1

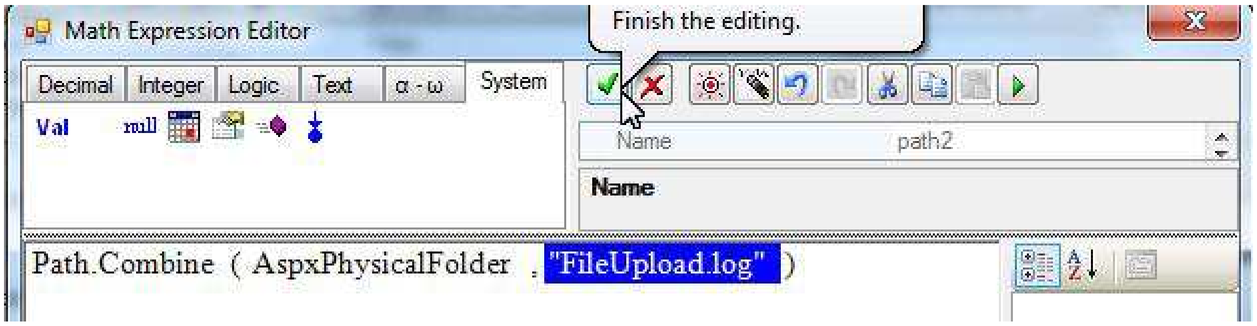
Path.Combine ("" , "")



WebPageFileUpload from WebPage
Constructors
Overrides and Interfaces
Instance Members
Properties inherited
AspxPhysicalFolder:String

Path.Combine(String,String)String

The second parameter of Combine is file name:

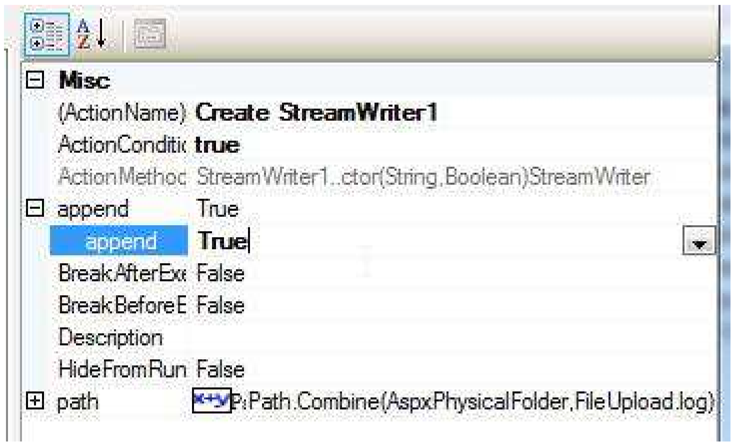


Name path2

Name

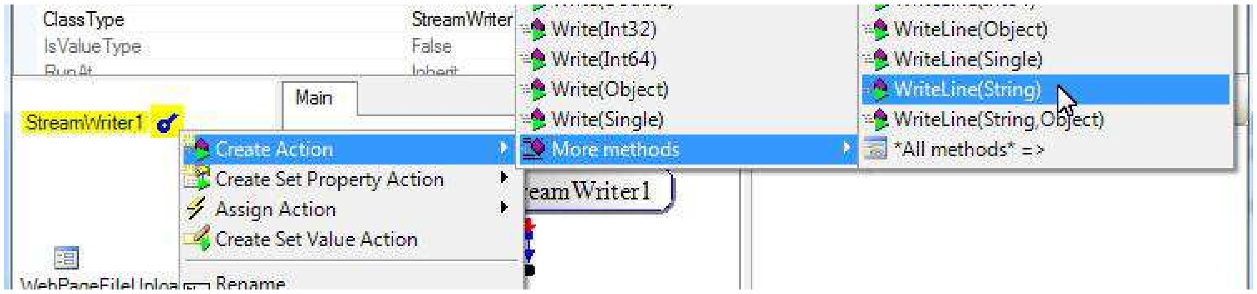
Path.Combine (AspxPhysicalFolder , "FileUpload.log")

Set "append" to True:

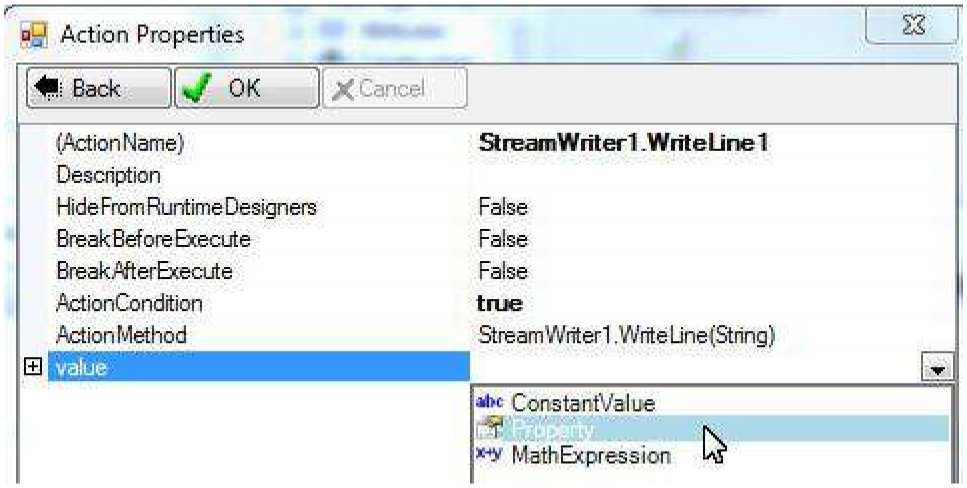


Misc	
(ActionName)	**Create StreamWriter1**
ActionConditic	**true**
ActionMethoc	StreamWriter1..ctor(String,Boolean)StreamWriter
append	True
append	**True**
BreakAfterExe	False
BreakBeforeE	False
Description	
HideFromRun	False
path	P:Path.Combine(AspxPhysicalFolder,FileUpload.log)

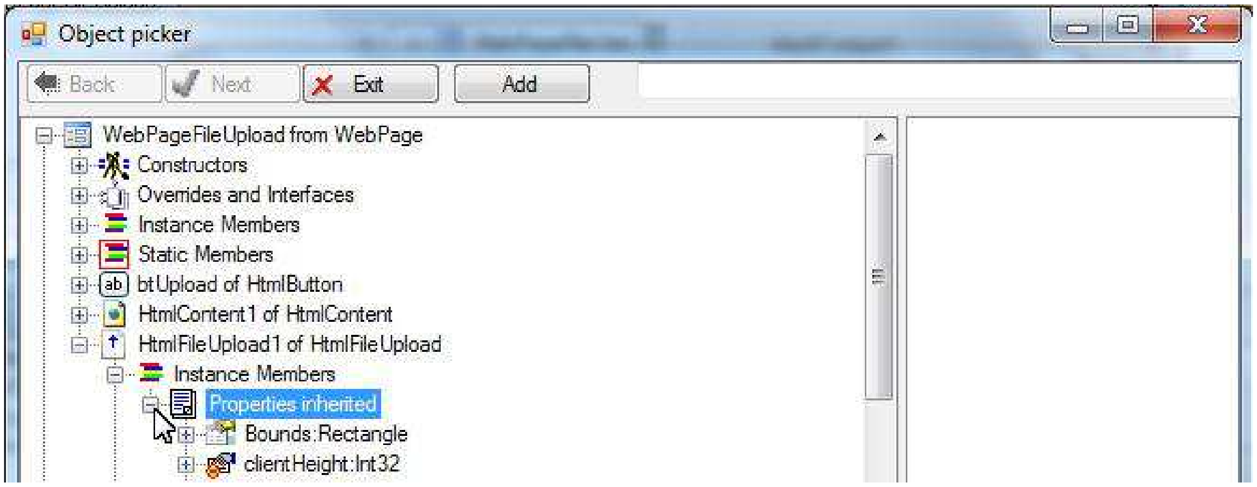
Create a WriteLine action:

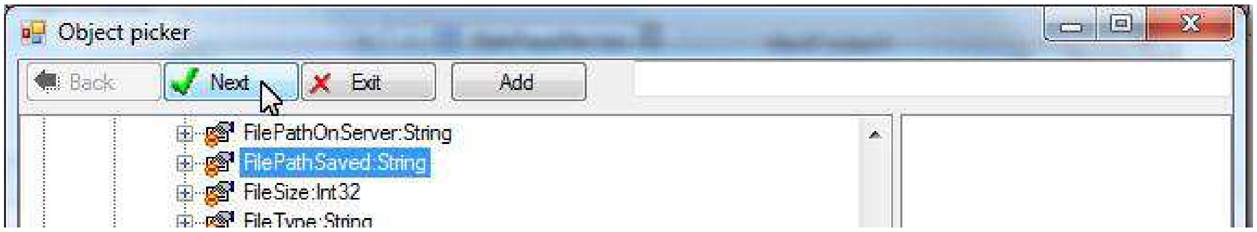


The value to write is the **FilePathSaved** property of the HtmlFileUpload component:

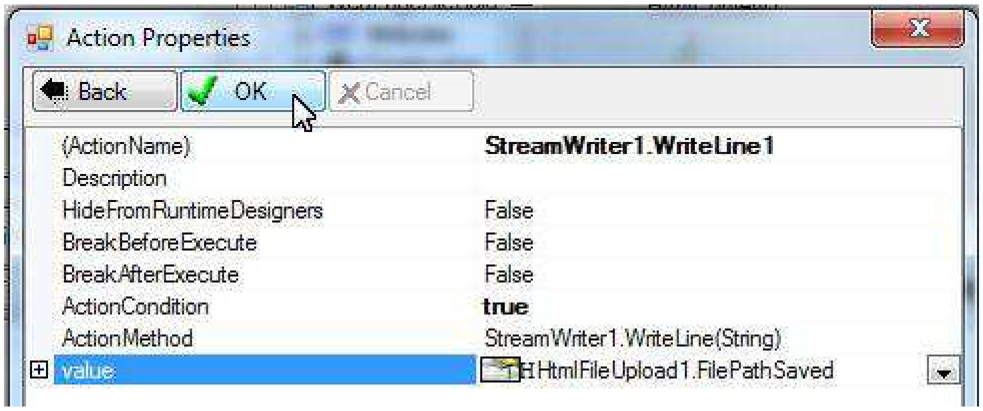


(ActionName)	**StreamWriter1.WriteLine1**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	StreamWriter1.WriteLine(String)
value	



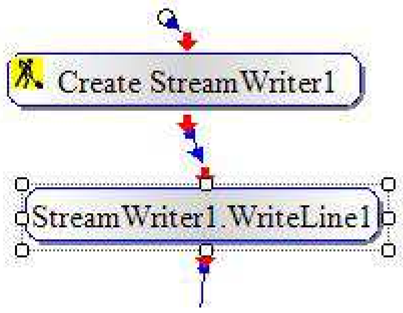


Click OK to finish creating the action:

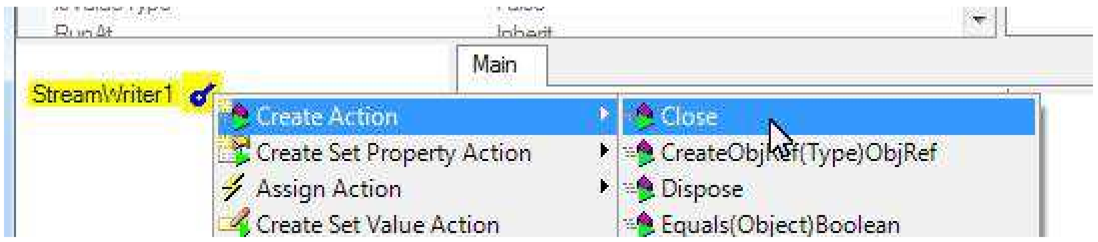


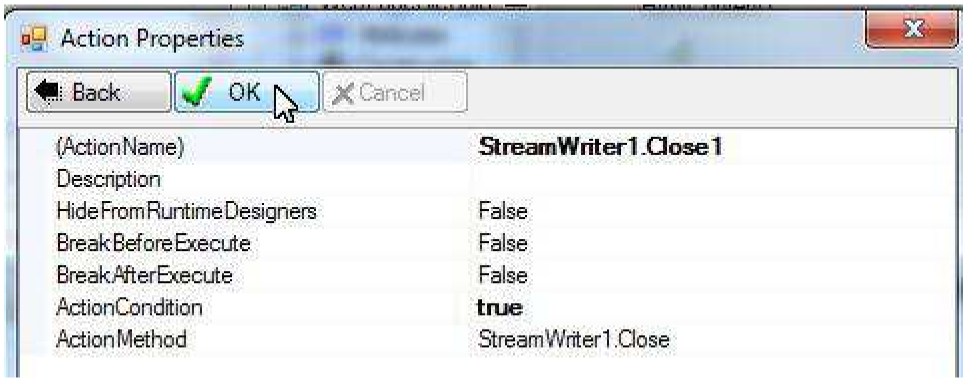
(ActionName)	**StreamWriter1.WriteLine1**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	StreamWriter1.WriteLine(String)
value	HtmlFileUpload1.FilePathSaved

This action shows the use of FilePathSaved property on the server side. Link the action to the last action:



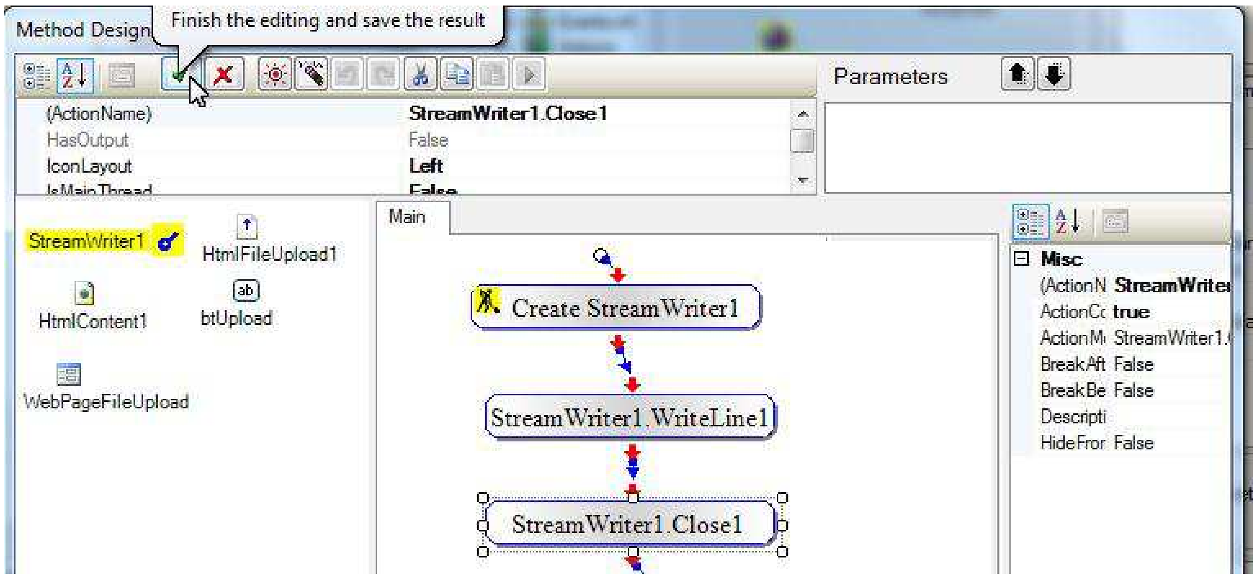
Create an action to close StreamWriter:



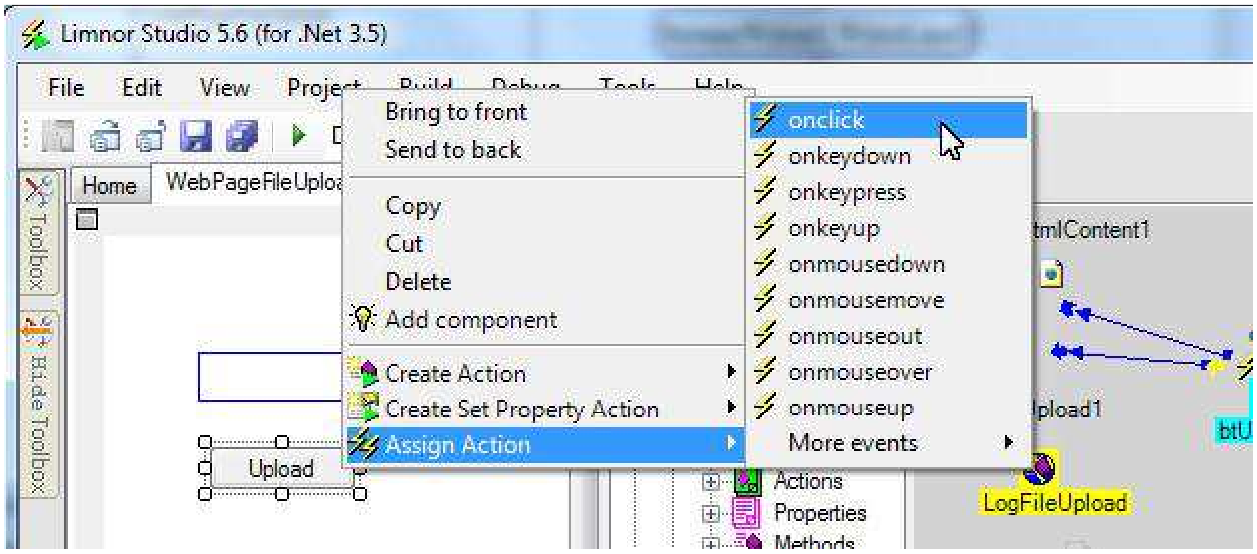


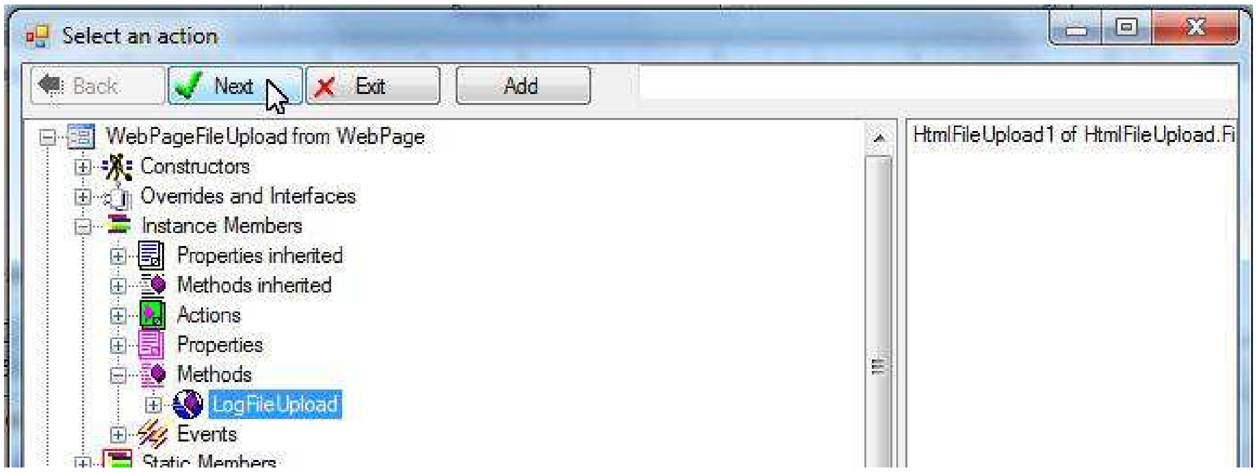
(ActionName)	StreamWriter1.Close1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	StreamWriter1.Close

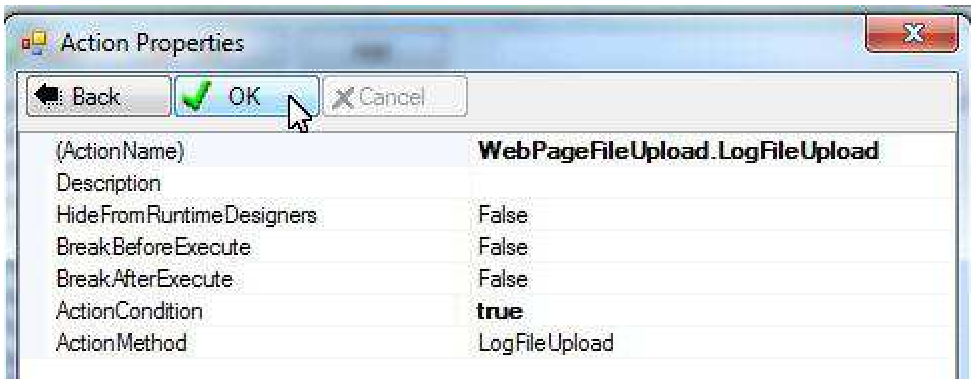
Link the action. We are done creating this server side method:



Let's execute the above method when the upload button is clicked:

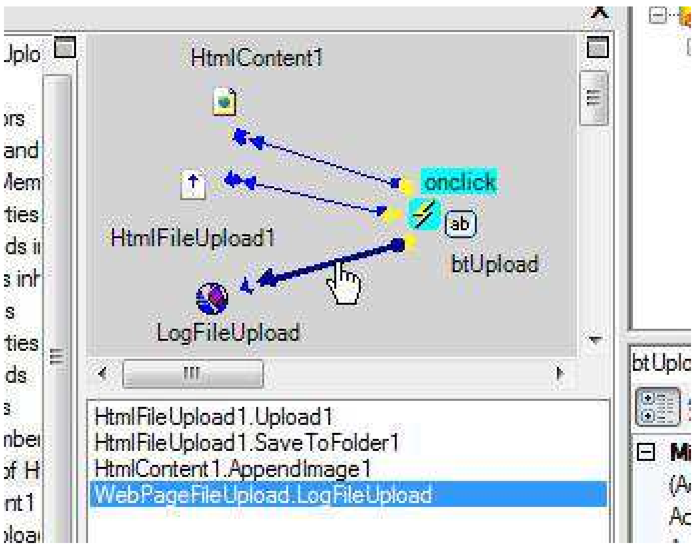






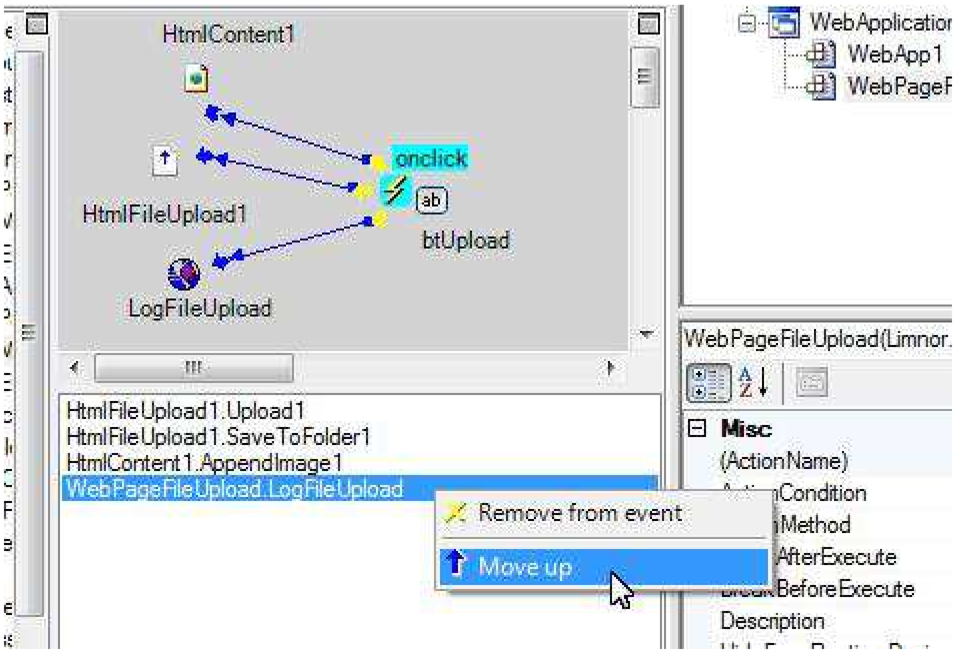
(ActionName)	**WebPageFileUpload.LogFileUpload**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	LogFileUpload

The action is created and assigned to the button:



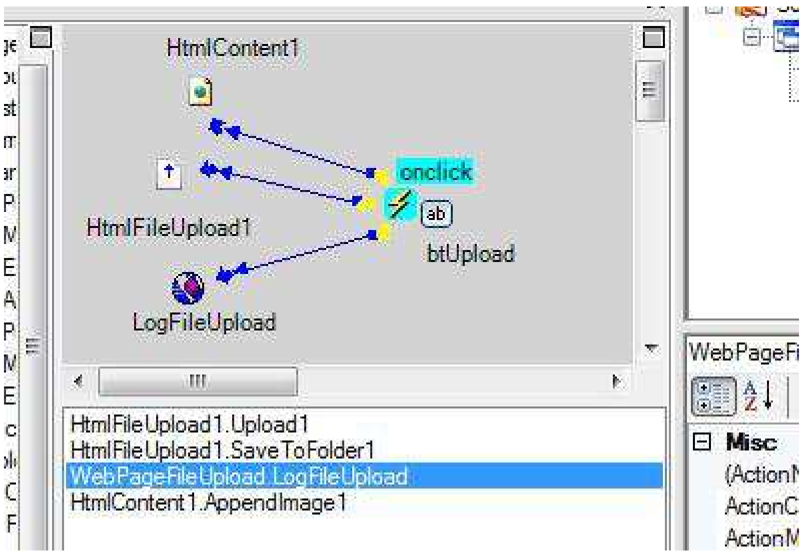
HtmlFileUpload1.Upload1
HtmlFileUpload1.SaveToFolder1
HtmlContent1.AppendImage1
WebPageFileUpload.LogFileUpload

The above arrangement works but there is room for performance improvement. SaveToFolder1 is an action on web server; AppendImage1 is an action on web client. On executing a web client action, all data on web server are lost because all web servers are stateless. Following AppendImage1 is another action on web server: LogFileUpload which uses server data FilePathSaved. Because FilePathSaved loses its value due to executing AppendImage1, the web page has to use extra network traffic between web client and web server to keep this value from losing. Such network traffic can be avoided if we move LogFileUpload up before AppendImage1:



HtmlFileUpload1.Upload1
HtmlFileUpload1.SaveToFolder1
HtmlContent1.AppendImage1
WebPageFileUpload.LogFileUpload

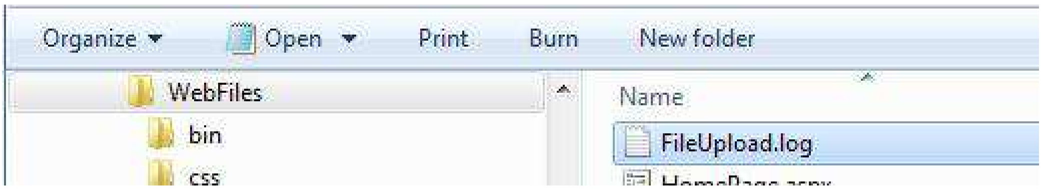
WebPageFileUpload(Limnor.
Misc
(ActionName)
ActionCondition
Method
AfterExecute
BeforeExecute
Description



HtmlFileUpload1.Upload1
HtmlFileUpload1.SaveToFolder1
WebPageFileUpload.LogFileUpload
HtmlContent1.AppendImage1

The above comments and performance improvement apply to previous PHP sample as well.

Run the web application and upload a file, we will see a log file created:

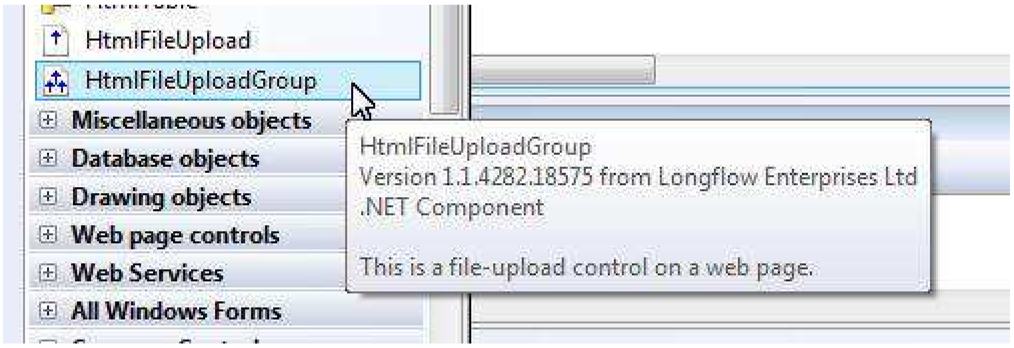


Upload Multiple Files

User Interface

Let’s create a new PHP web application to demonstrate uploading multiple files.

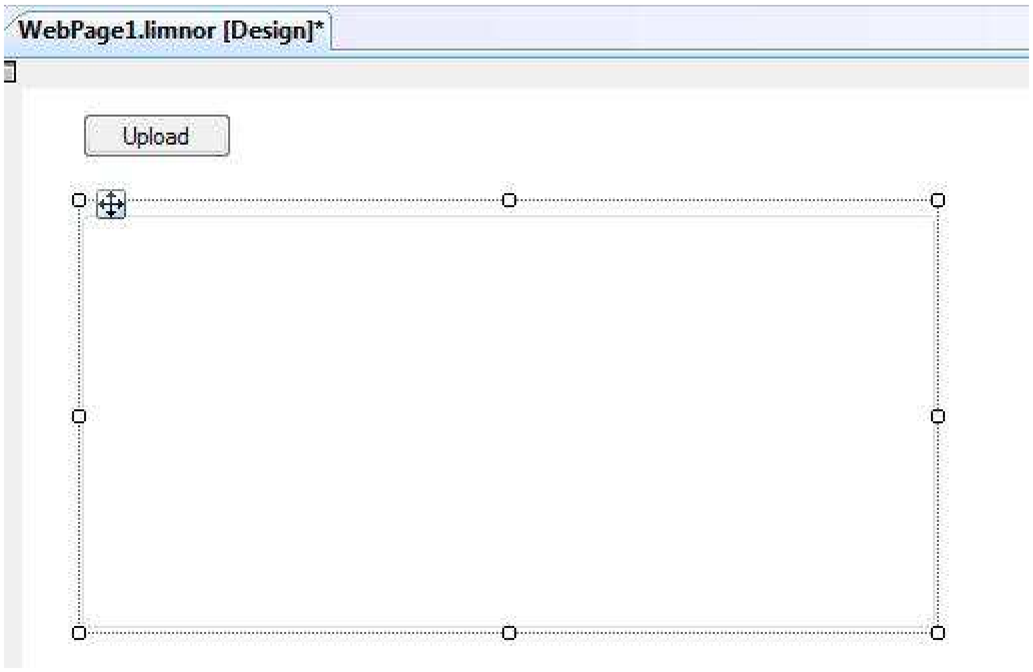
Drop an HtmlFileUploadGroup to the web page:



HtmlFileUploadGroup
Version 1.1.4282.18575 from Longflow Enterprises Ltd
.NET Component

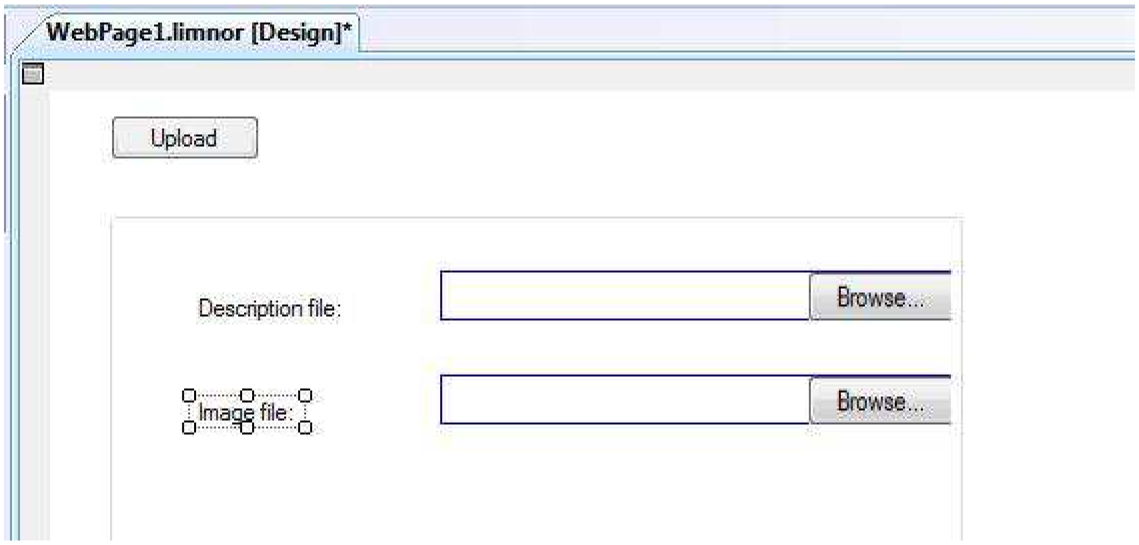
This is a file-upload control on a web page.

Drop a button to the page for initiating file upload action.



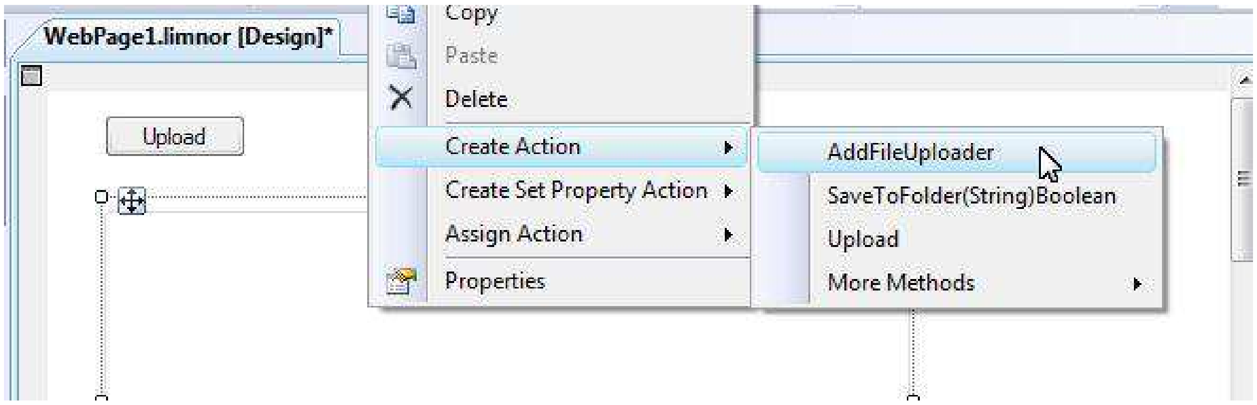
Add file upload controls at design time

Drop as many HtmlFileUpload control into the HtmlFileUploadGroup as needed. Other controls may also be added to the HtmlFileUploadGroup to form user interface:

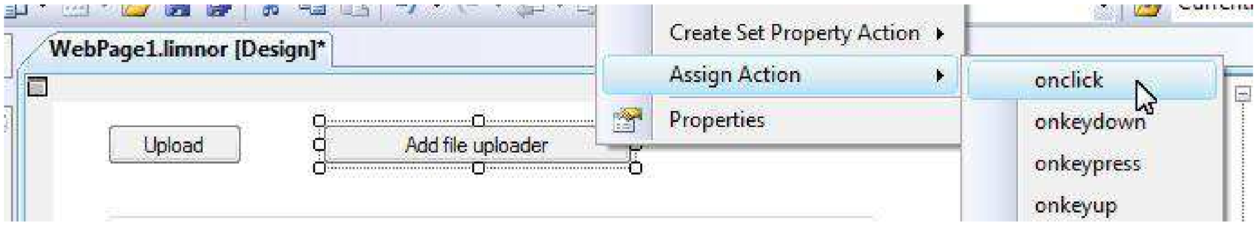


Add file upload controls at runtime

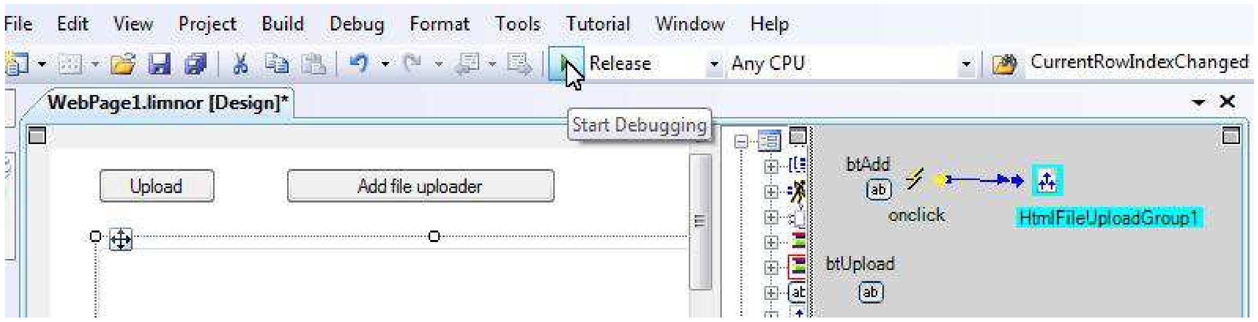
Another approach is to let the web visitor add file upload controls. Let’s remove all controls from HtmlFileUploadGroup. Create an “AddFileUploader” action to it:



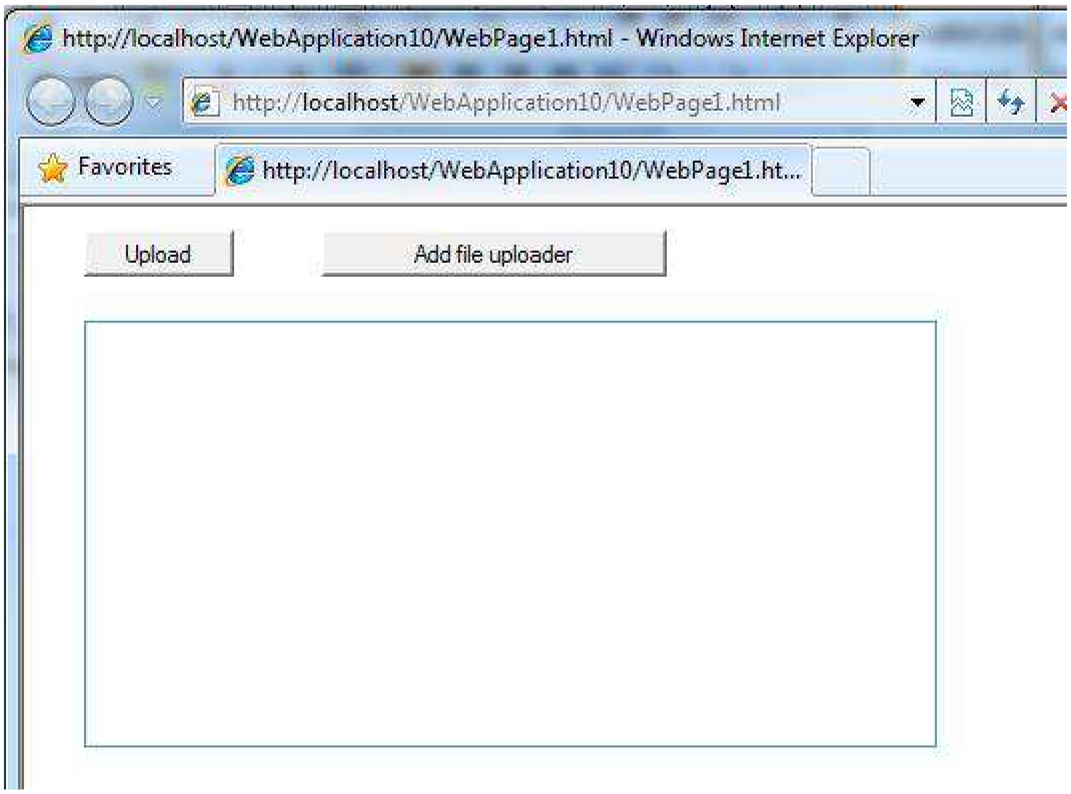
Assign the action to a button:



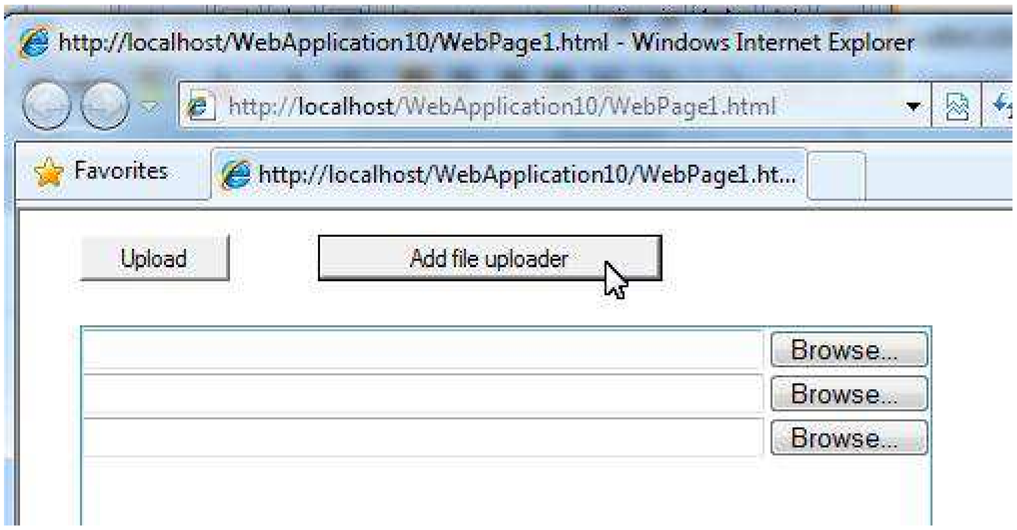
Let's test it:



The web page appears:

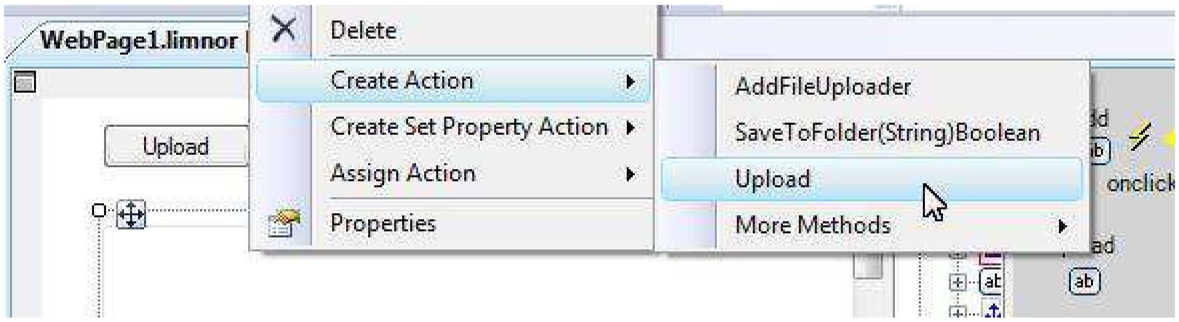

http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication10/WebPage1.html
Favorites
http://localhost/WebApplication10/WebPage1.ht...
Upload
Add file uploader

Click “Add file uploader” several times. Each time a new file upload control is added:

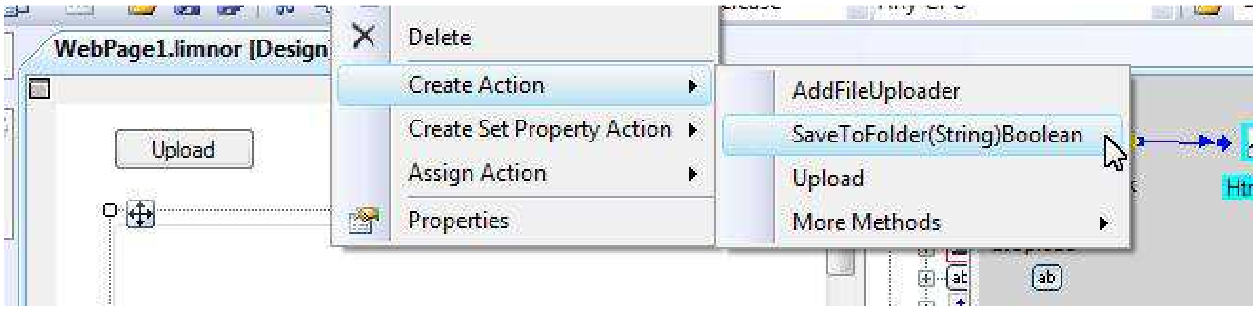

http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication10/WebPage1.html
Favorites
http://localhost/WebApplication10/WebPage1.ht...
Upload
Add file uploader
Browse...
Browse...
Browse...

Upload files to web server

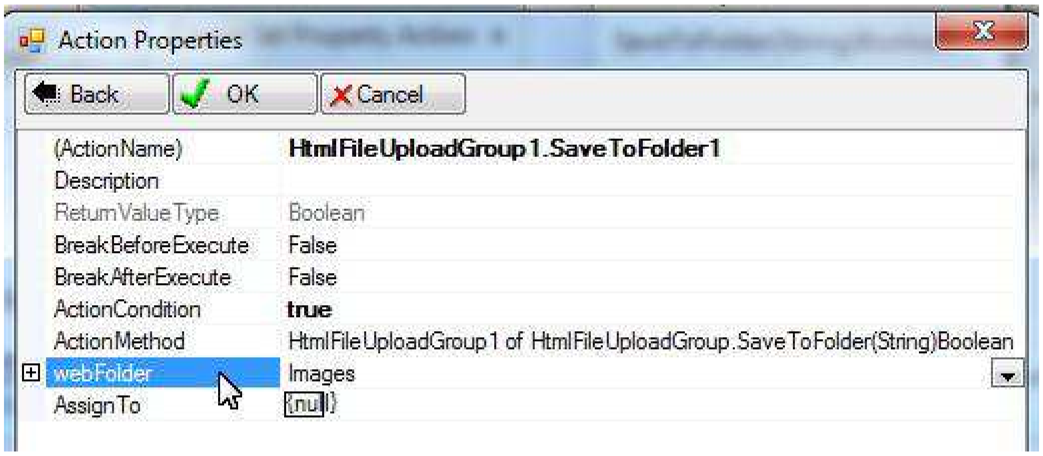
To upload all the files in the group, create an Upload action:



To save all the files to a folder at the web server, create a SaveToFolder action:

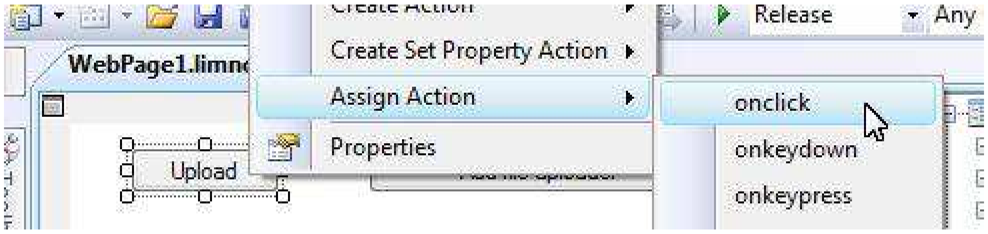


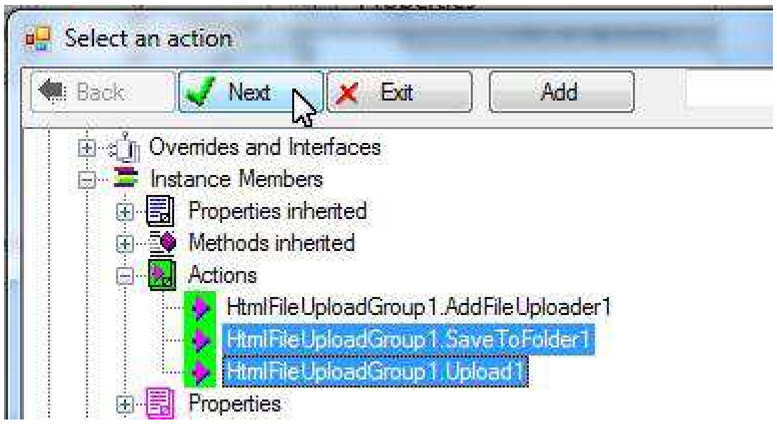
Give a folder name. We use “Images” here.



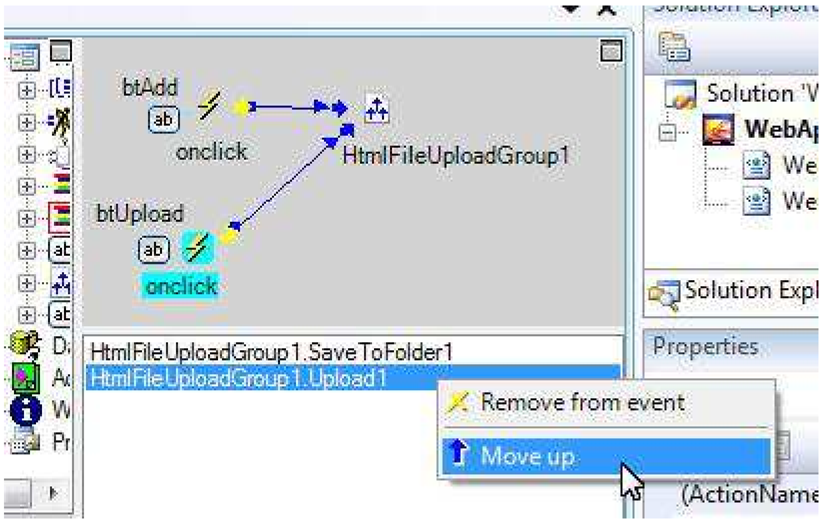
(ActionName)	HtmlFileUploadGroup1.SaveToFolder1
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlFileUploadGroup1 of HtmlFileUploadGroup.SaveToFolder(String)Boolean
webFolder	Images
AssignTo	{null}

Assign the actions to the Upload button:

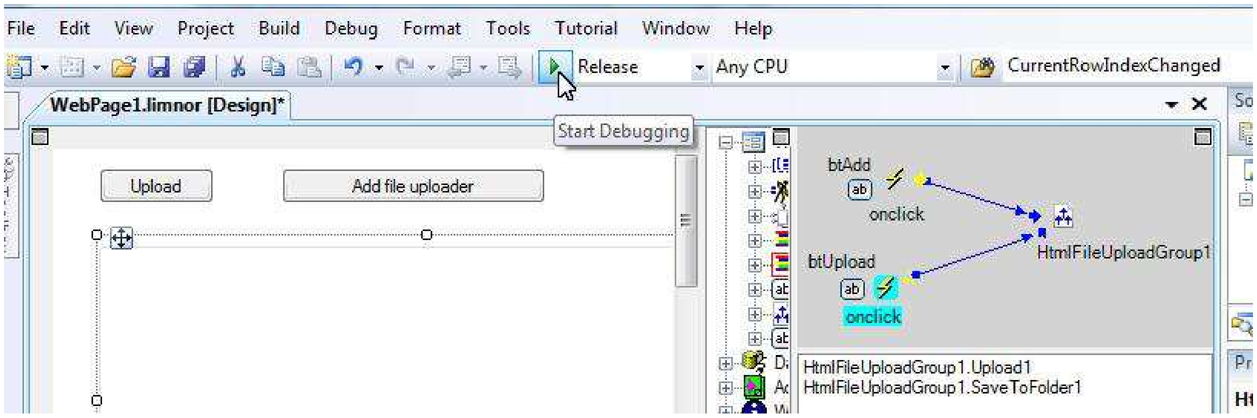




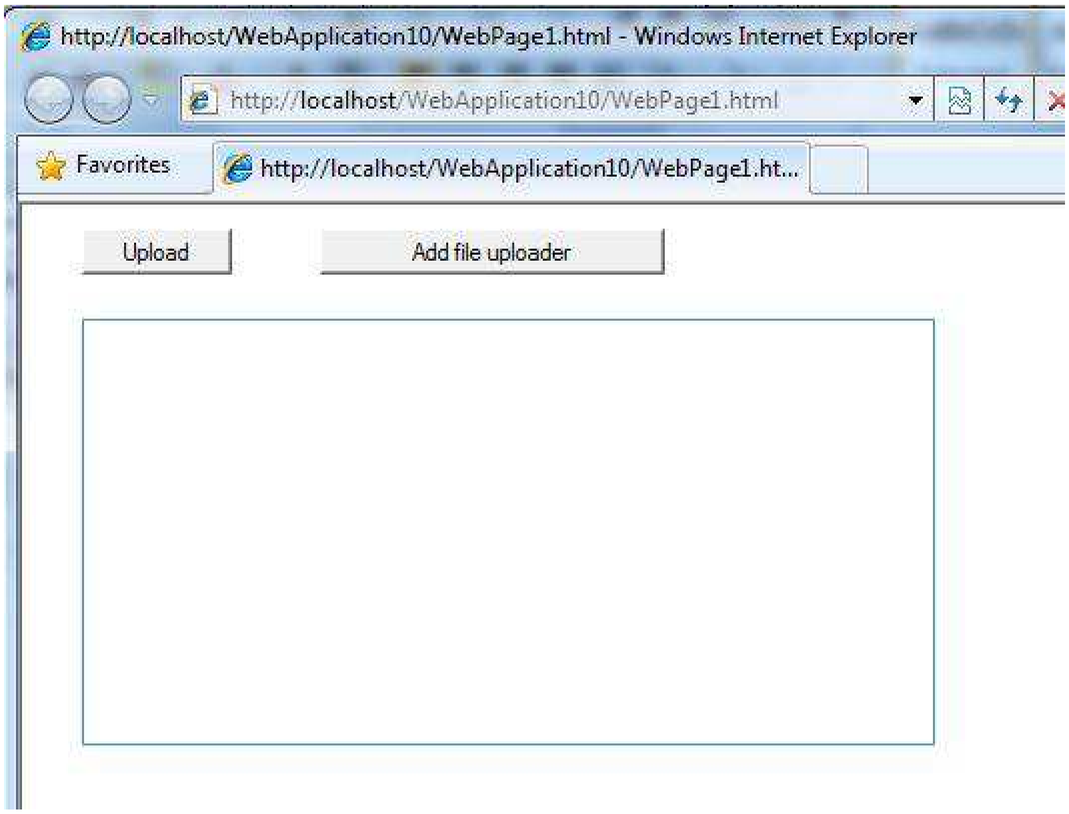
Move Upload action before SaveToFolder action:



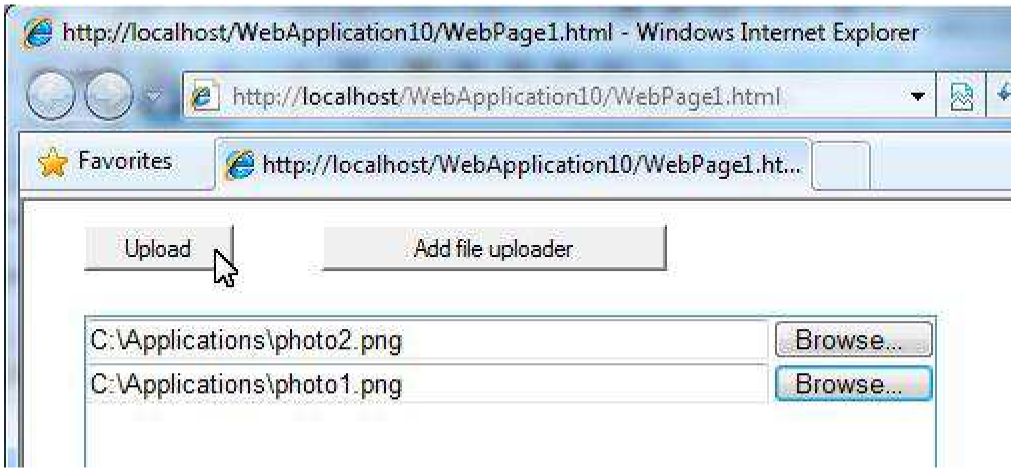
We may test it now:



The web page appears:

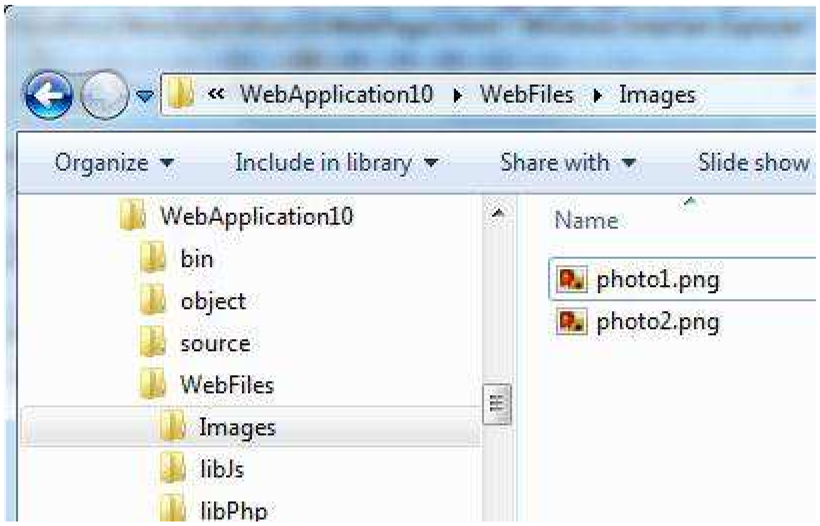

http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
Favorites
Upload
Add file uploader

Add some file upload controls and select files. Click Upload button.

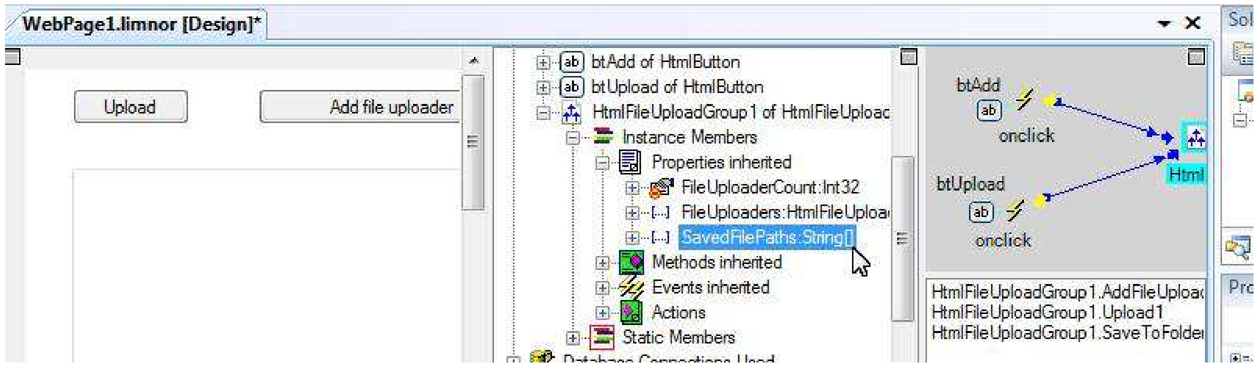

http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
Favorites
Upload
Add file uploader
Browse...
Browse...

C:\Applications\photo2.png
C:\Applications\photo1.png

The files will be uploaded to the web server:

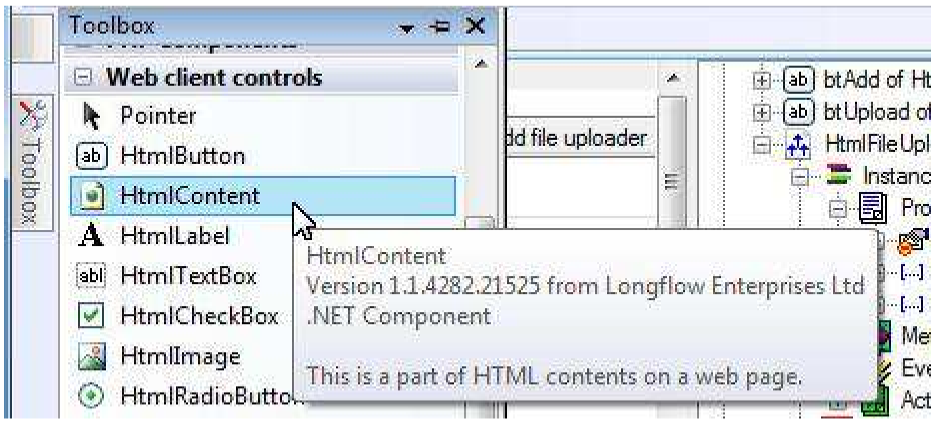


After file saving, the property SavedFilePaths of the HtmlFileUploadGroup can be used both at client and server.

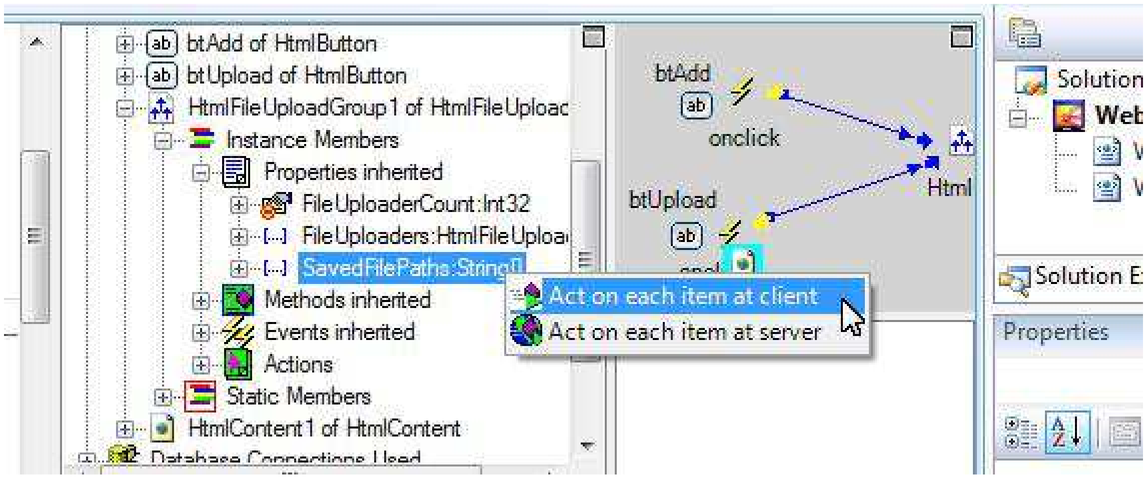


Use uploaded file names at client

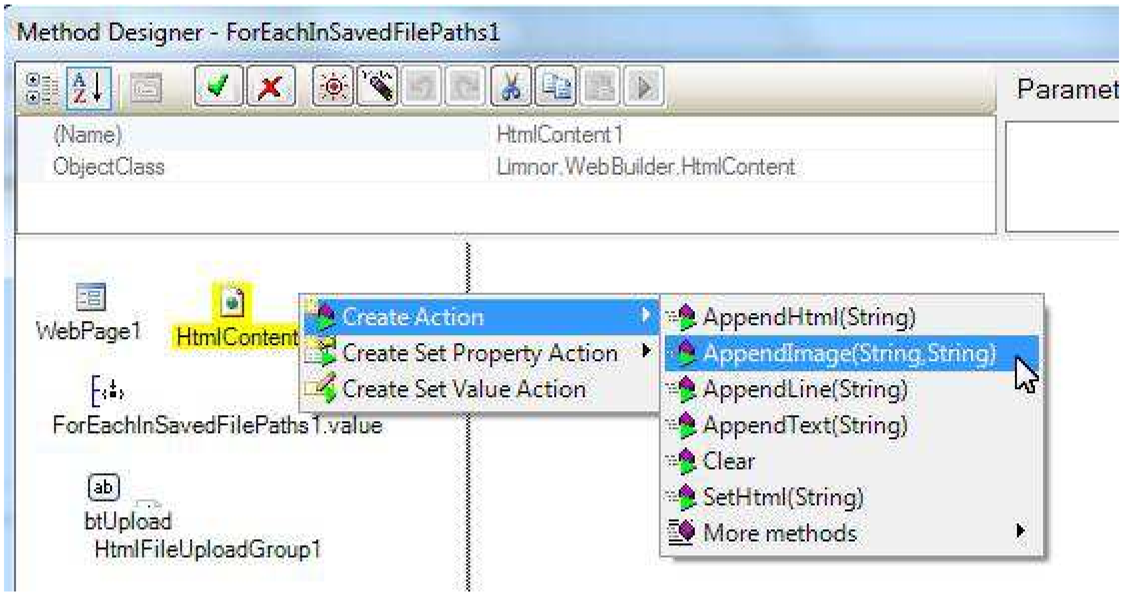
Drop an HtmlContent to the web page to show the usage of SavedFilePaths of the HtmlFileUploadGroup.



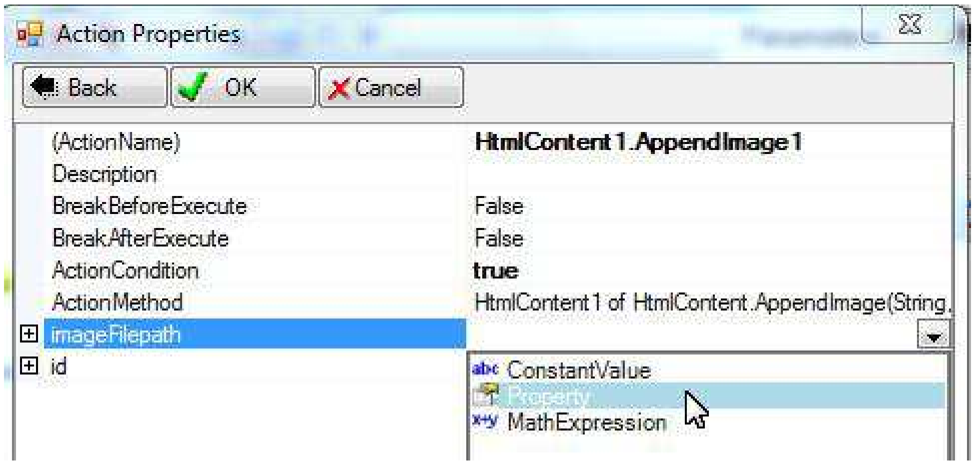
Right-click SavedFilePaths; choose “Act on each item at client”



Right-click the HtmlContent; choose “Create Action”; choose “AppendImage”:

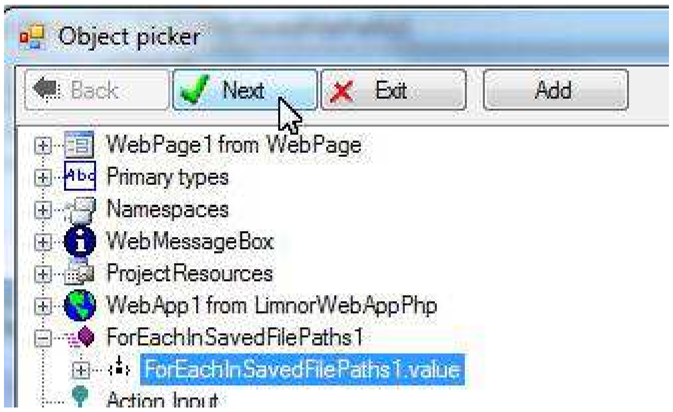


Select “Property” for the “imageFilepath”:

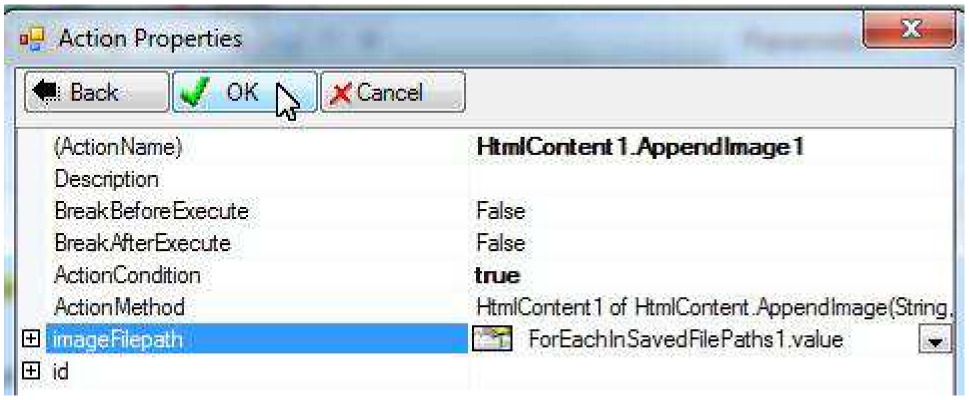


(ActionName)	HtmlContent1.AppendImage1
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlContent1 of HtmlContent.AppendImage(String,
imageFilepath	
id	

Select "ForeachInSavedFilePaths1.value":

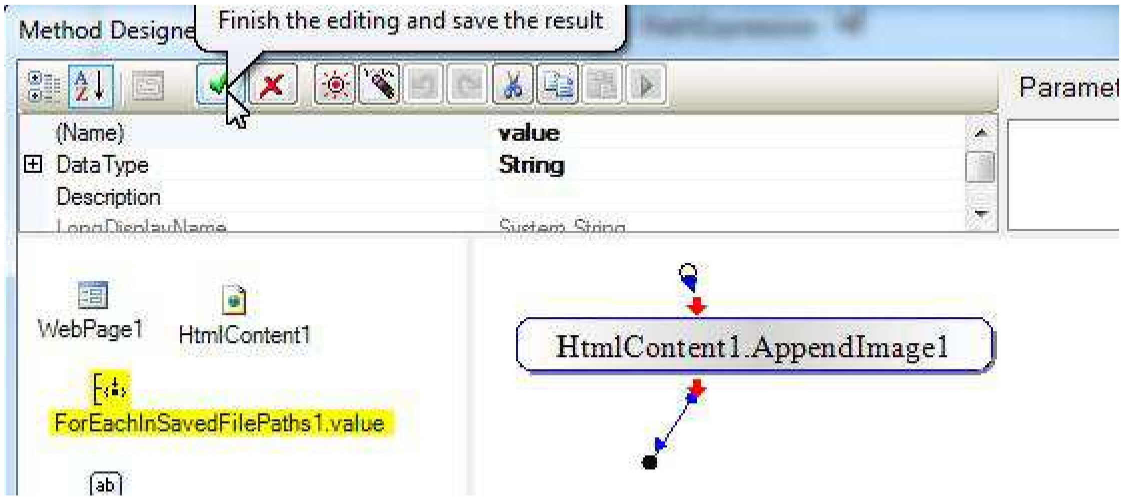


Click OK to finish creating this action:



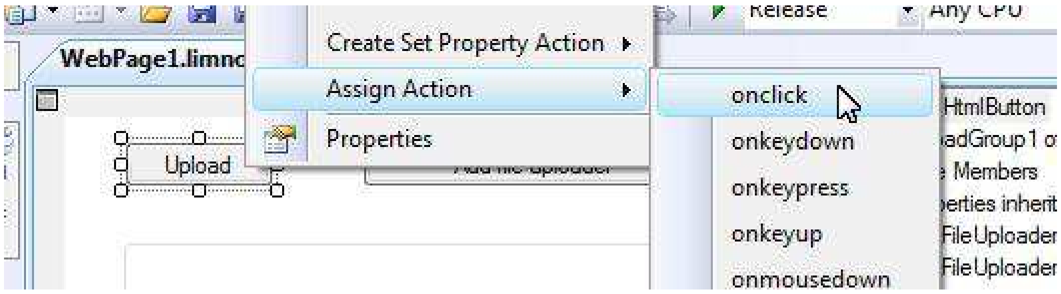
(ActionName)	HtmlContent1.AppendImage1
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlContent1 of HtmlContent.AppendImage(String,
imageFilepath	ForEachInSavedFilePaths1.value
id	

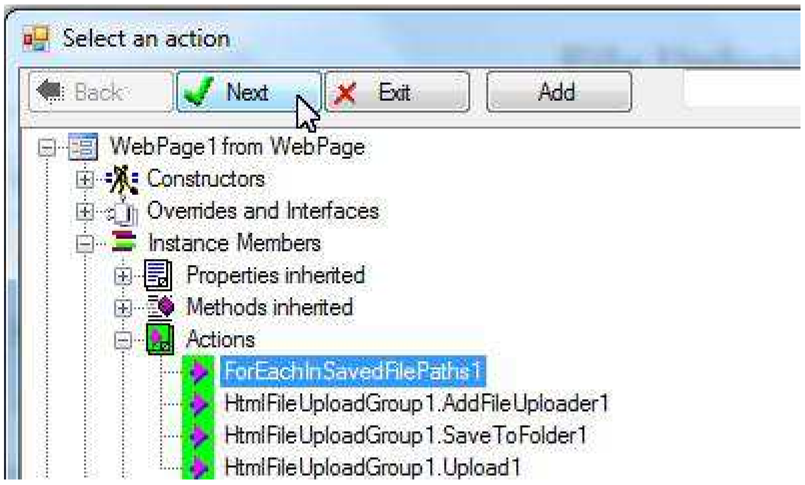
The action appears. For this sample, this is the only action we want to act on each file path:


Method Designer
Finish the editing and save the result
Paramet
(Name) value
Data Type String
Description
WebPage1
HtmlContent1
ForEachInSavedFilePaths1.value
HtmlContent1.AppendImage1

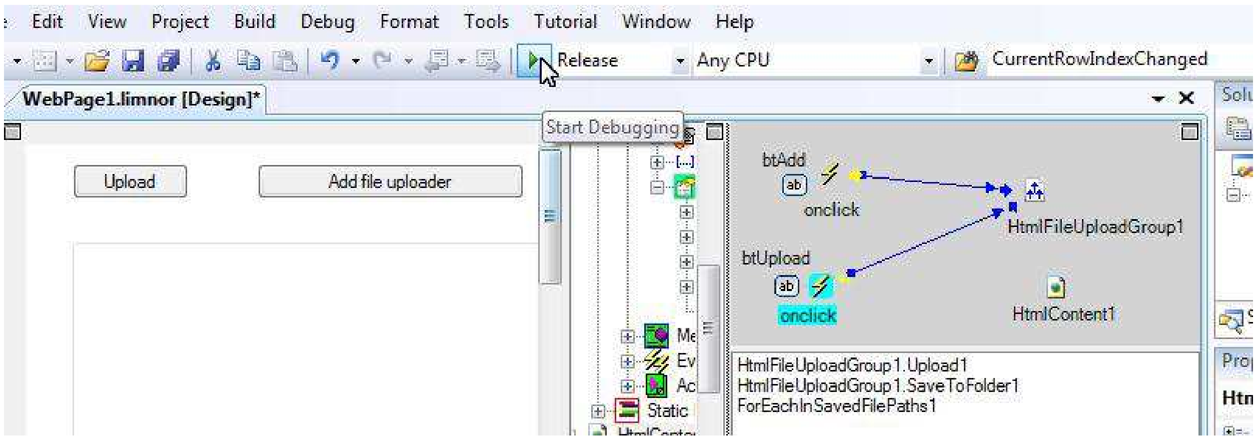
We created an action named ForeachInSavedFilepaths1, which contains another action named HtmlContent1.AppendImage1.

We may assign ForeachInSavedFilepaths1 to the Upload button:



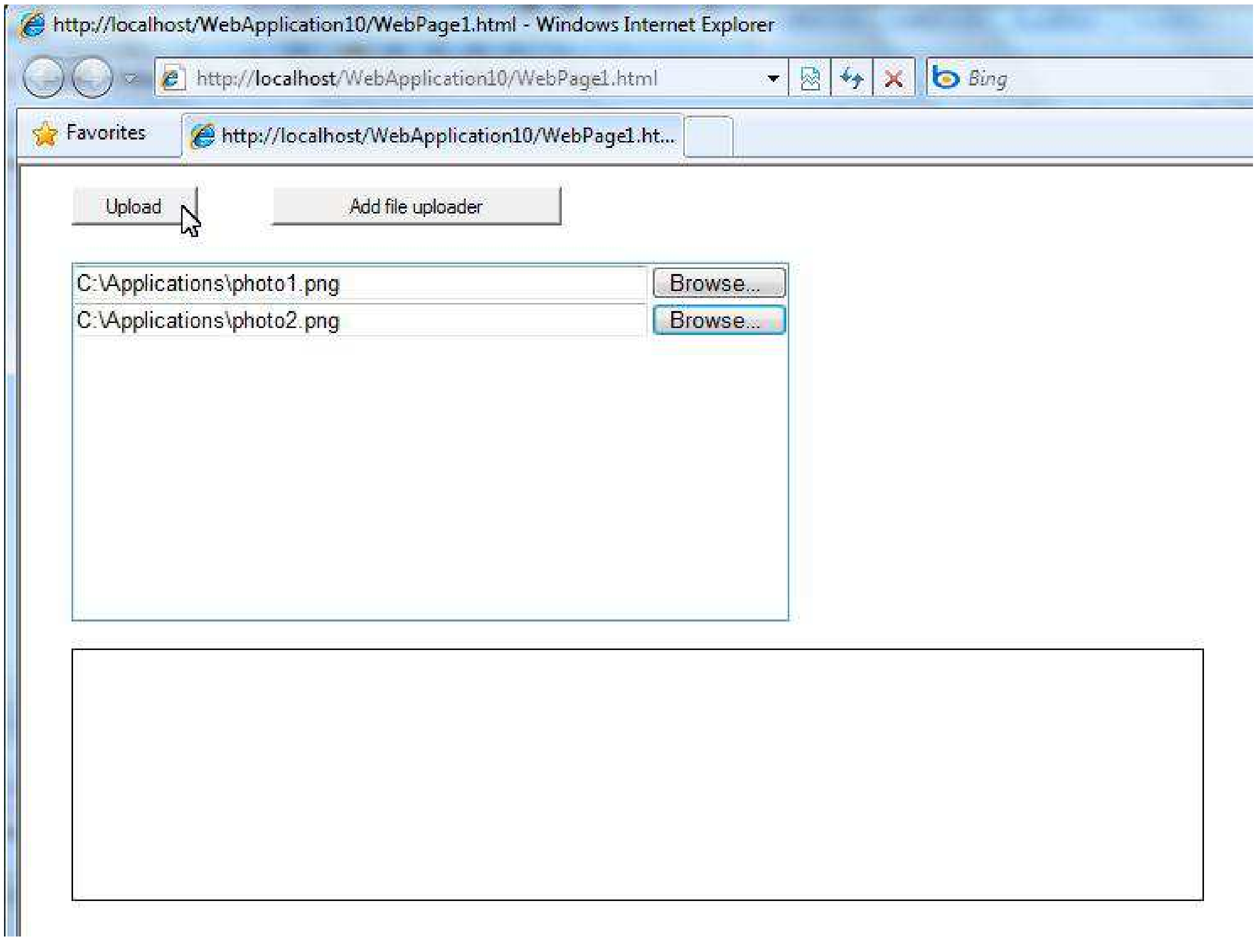


We may test it now:



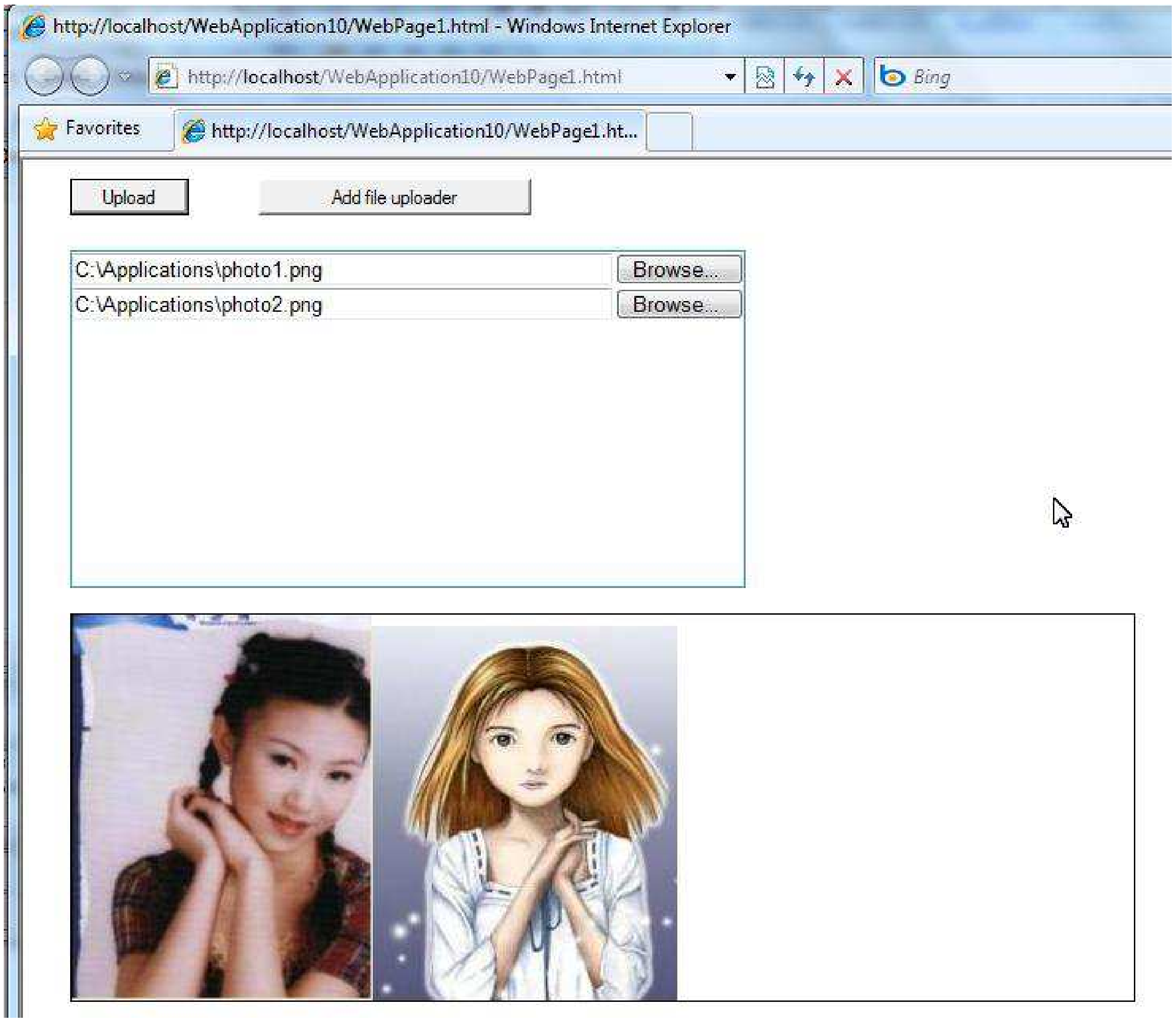
HtmlFileUploadGroup1.Upload1
HtmlFileUploadGroup1.SaveToFolder1
ForEachInSavedFilePaths1

The web page appears. Select two image files and click Upload:


http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication10/WebPage1.html
Bing
Favorites
http://localhost/WebApplication10/WebPage1.ht...
Upload
Add file uploader
Browse...
Browse...

C:\Applications\photo1.png
C:\Applications\photo2.png

The two image files are uploaded and displayed on the web page:


http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication10/WebPage1.html
Bing
Favorites
http://localhost/WebApplication10/WebPage1.ht...
Upload
Add file uploader
Browse...
Browse...

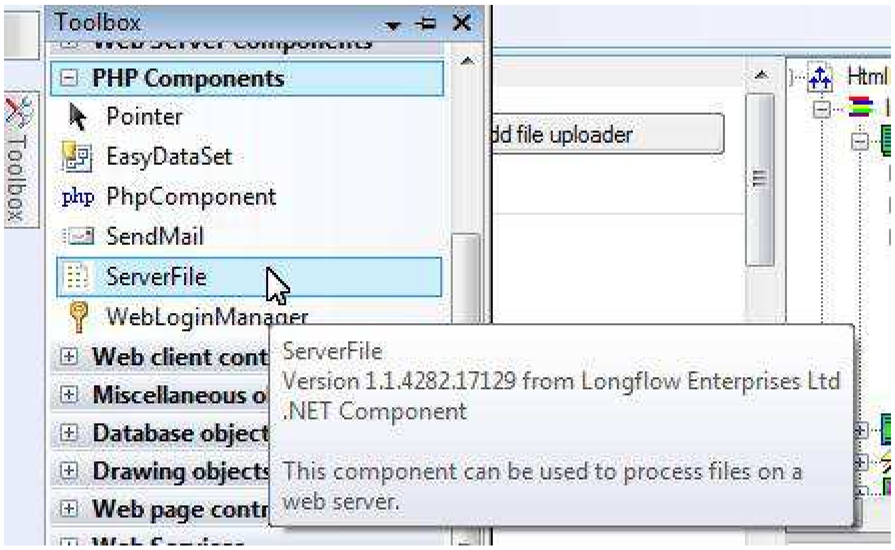
C:\Applications\photo1.png
C:\Applications\photo2.png

Use uploaded file names at server

SavedFilePaths property can be used at server after file upload. Even though HtmlFileUploadGroup component is used in the same way for PHP and Aspx, other server processing is usually different for PHP and Aspx. Suppose we want to write file names of uploaded files to a log file. For PHP we may use ServerFile component to do it. For Aspx, we may use StreamWriter class to do it. Below we show both ways.

Use PHP to do logging

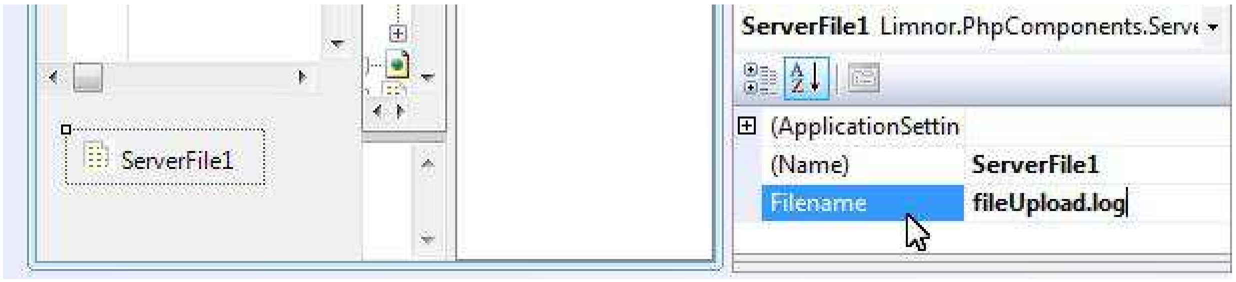
Drop a ServerFile component to the web page for logging the file names uploaded:



ServerFile
Version 1.1.4282.17129 from Longflow Enterprises Ltd
.NET Component

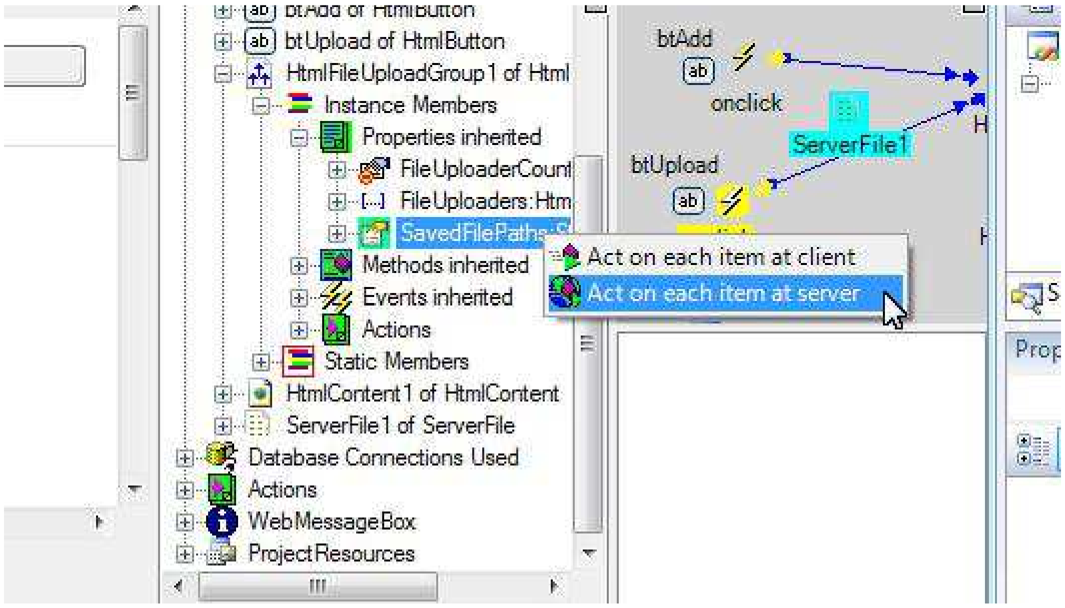
This component can be used to process files on a web server.

Set file name:

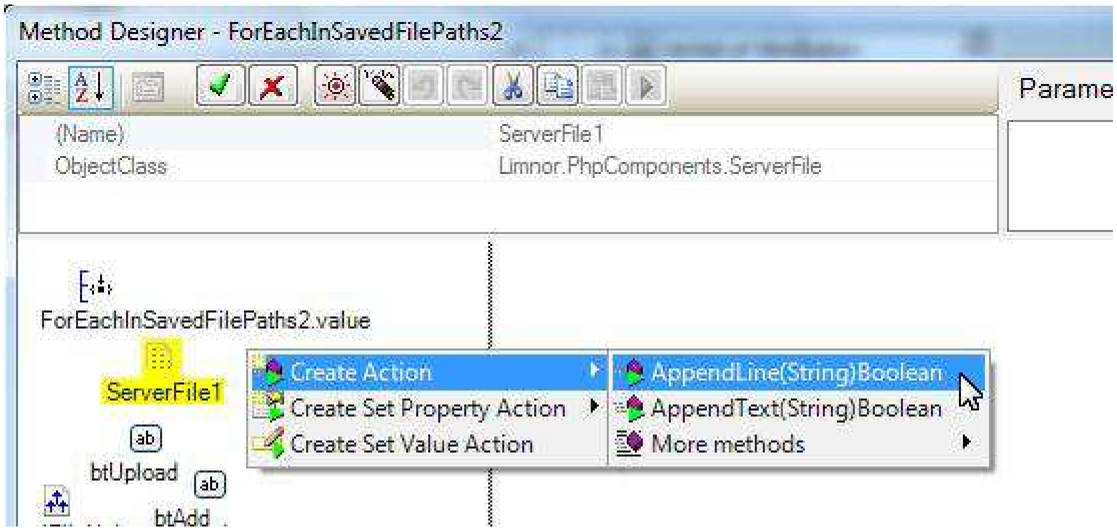


(Name)	ServerFile1
Filename	fileUpload.log

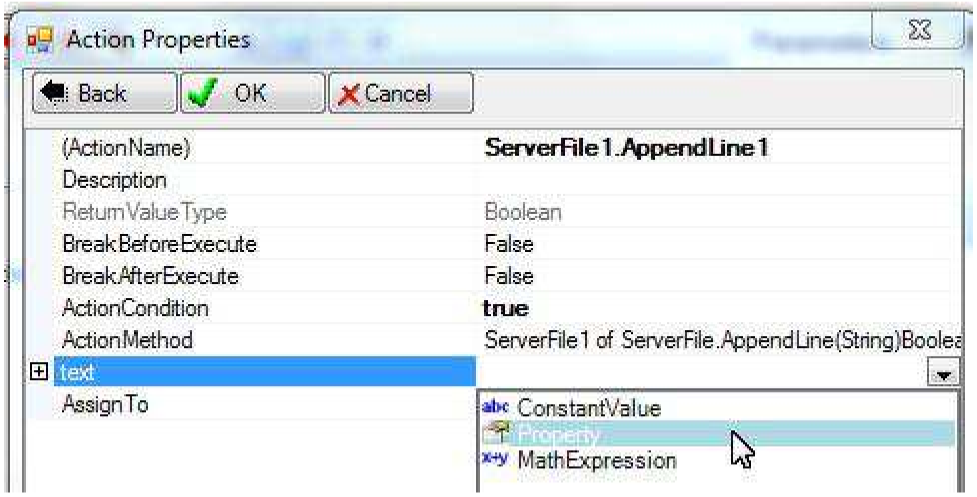
Right-click SavedFilePaths; choose “Act on each item at server”:



Right-click ServerFile1; choose “Create Action”; choose “AppendLine”:

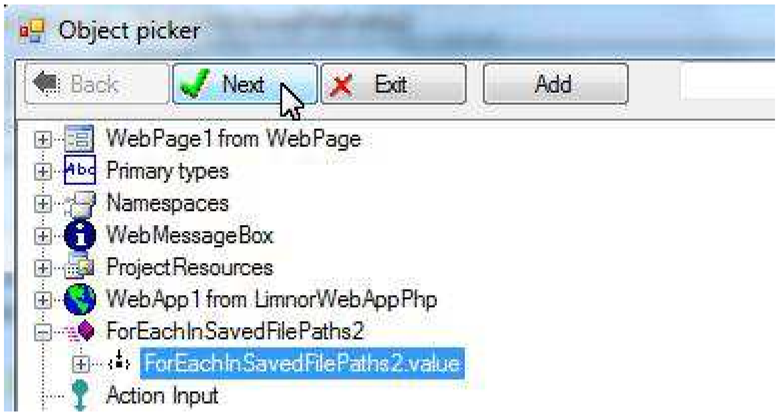

Method Designer - ForEachInSavedFilePaths2
Parame
(Name) ServerFile1
ObjectClass Limnor.PhpComponents.ServerFile
ForEachInSavedFilePaths2.value
ServerFile1
btUpload
btAdd
Create Action
Create Set Property Action
Create Set Value Action
AppendLine(String)Boolean
AppendText(String)Boolean
More methods

Select "Property" for the "text":

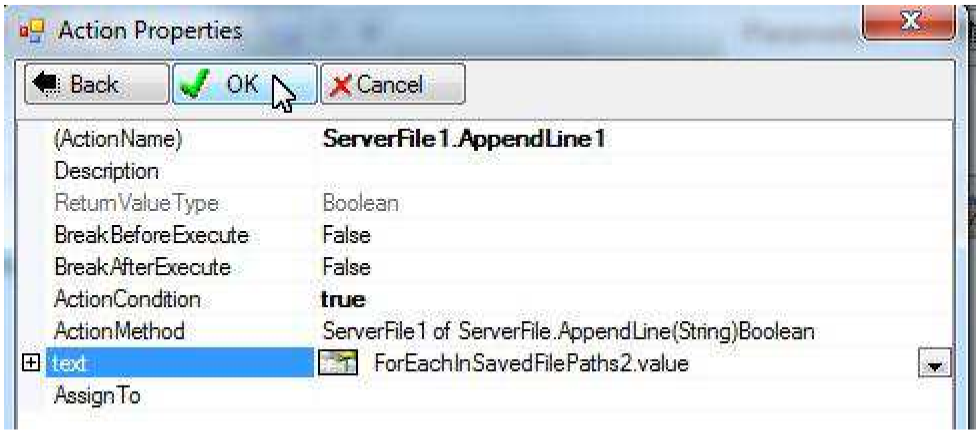


(ActionName)	ServerFile1.AppendLine1
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ServerFile1 of ServerFile.AppendLine(String)Boolea
text	
AssignTo	

Select "ForeachInSavedFilePaths2.value":

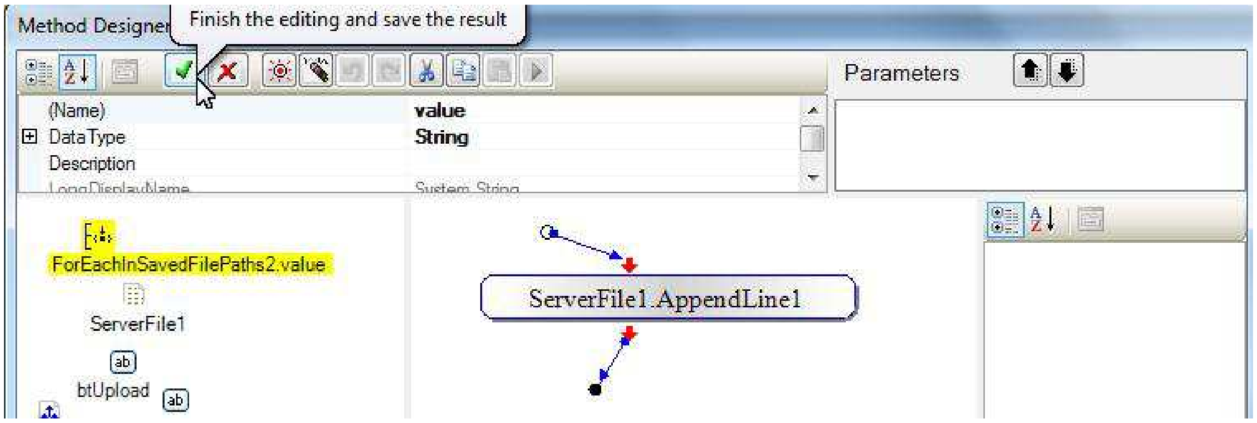


Click OK to finish creating this action:



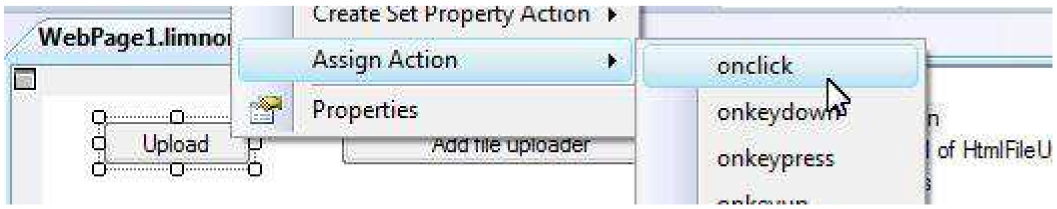
(ActionName)	ServerFile1.AppendLine1
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ServerFile1 of ServerFile.AppendLine(String)Boolean
text	ForEachInSavedFilePaths2.value
AssignTo	

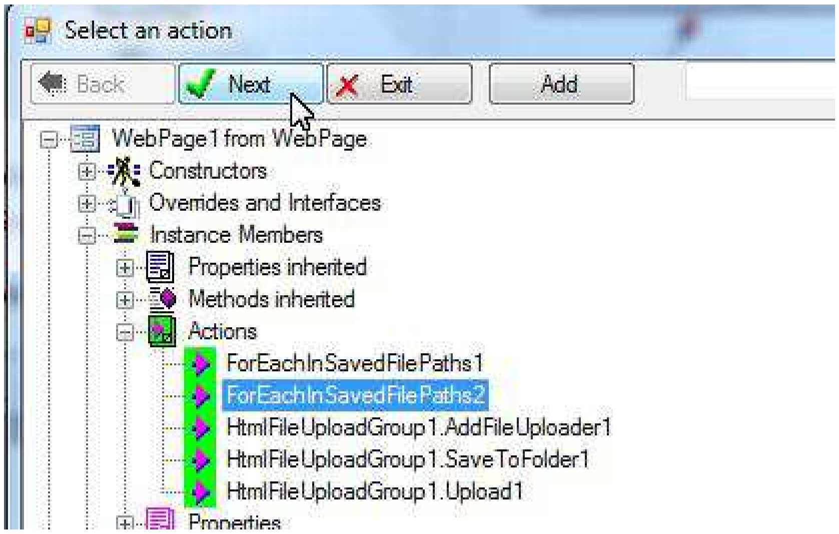
The action appears. For this sample, this is the only action we want to act on each file path:



We created an action named ForeachInSavedFilepaths2, which contains another action named ServerFile1.AppendLine1.

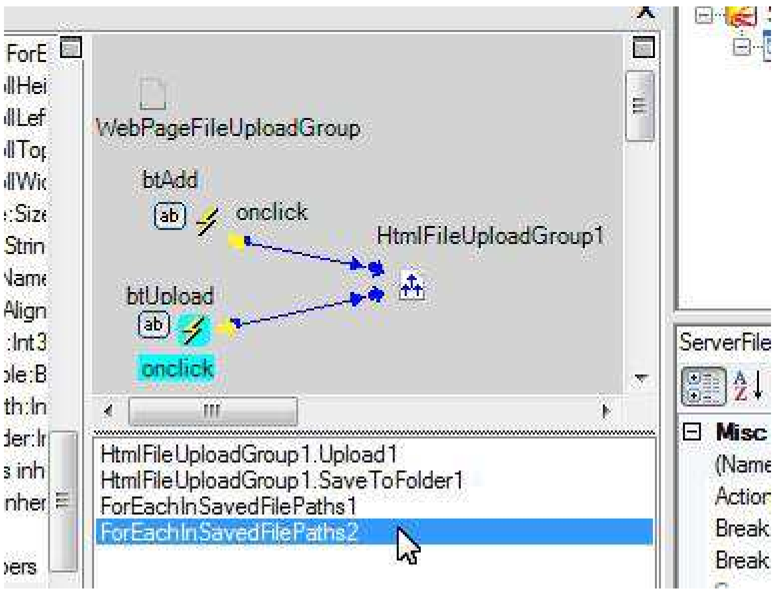
We may assign ForeachInSavedFilepaths2 to the Upload button:





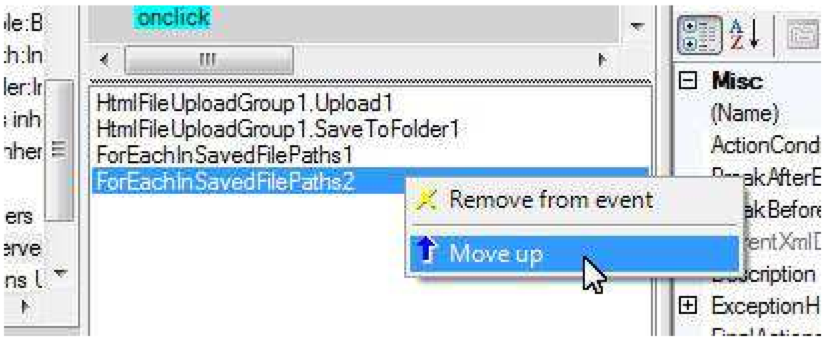
WebPage1 from WebPage
Constructors
Overrides and Interfaces
Instance Members
Properties inherited
Methods inherited
Actions
ForEachInSavedFilePaths1
ForEachInSavedFilePaths2
HtmlFileUploadGroup1.AddFileUploader1
HtmlFileUploadGroup1.SaveToFolder1
HtmlFileUploadGroup1.Upload1
Properties

The action appears at the end of the actions list:

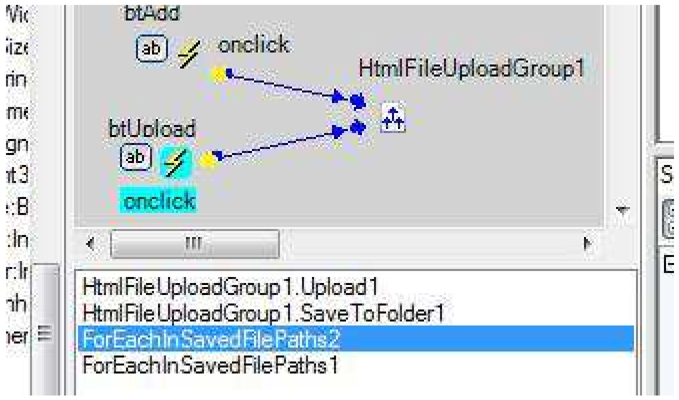


HtmlFileUploadGroup1.Upload1
HtmlFileUploadGroup1.SaveToFolder1
ForEachInSavedFilePaths1
ForEachInSavedFilePaths2

We may move this action up to make it appear before action ForEachInSavedFilePaths1:



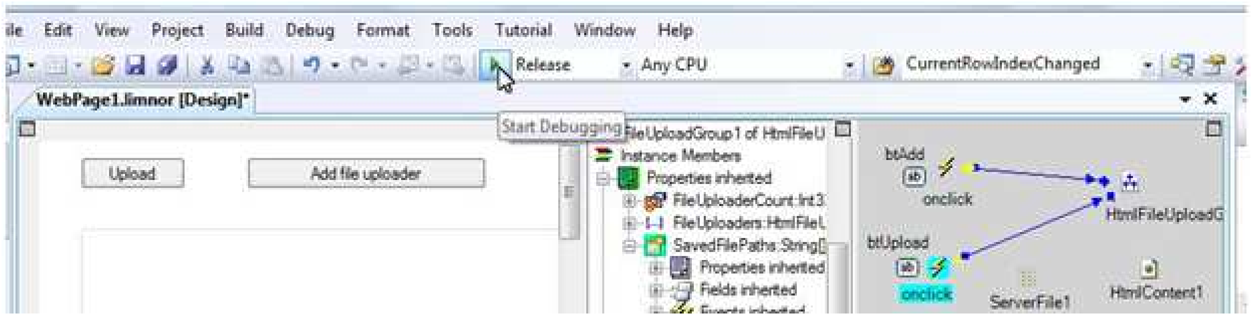
HtmlFileUploadGroup1.Upload1
HtmlFileUploadGroup1.SaveToFolder1
ForEachInSavedFilePaths1
ForEachInSavedFilePaths2



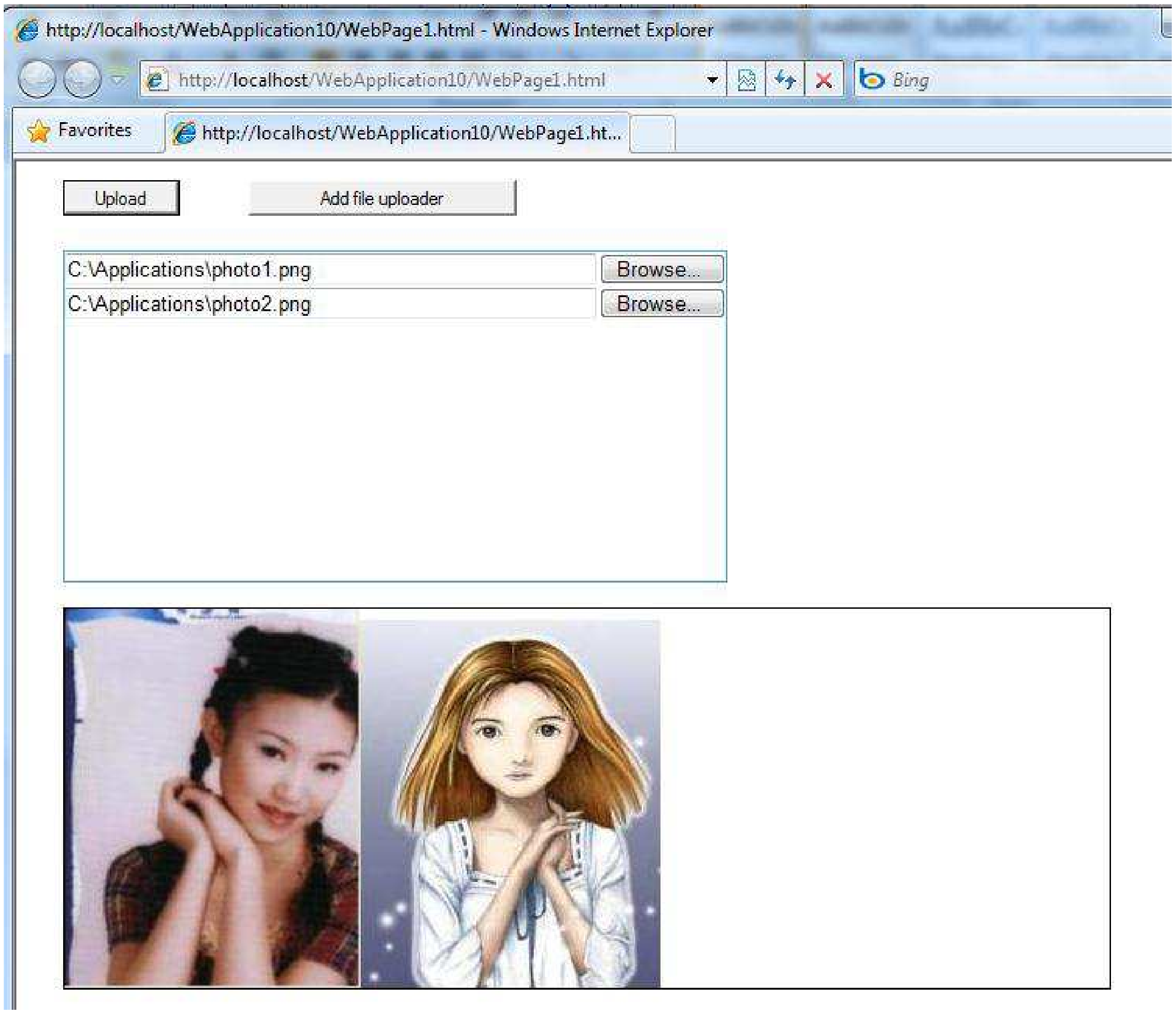
HtmlFileUploadGroup1.Upload1
HtmlFileUploadGroup1.SaveToFolder1
ForEachInSavedFilePaths2
ForEachInSavedFilePaths1

The above re-ordering of the action sequence helps improve web page performance in this situation. *ForEachInSavedFilePaths**2*** is a server side action which uses the values in a server side object, HtmlFileUploadGroup1. *ForEachInSavedFilePaths**1*** is a client side action. After executing a client side action, the server side object, *HtmlFileUploadGroup1*, loses all its values. For executing a server side action, *ForEachInSavedFilePaths**2***, the values have to be uploaded to the server. If *ForEachInSavedFilePaths**2*** executes before *ForEachInSavedFilePaths**1*** then it avoids the data uploading.

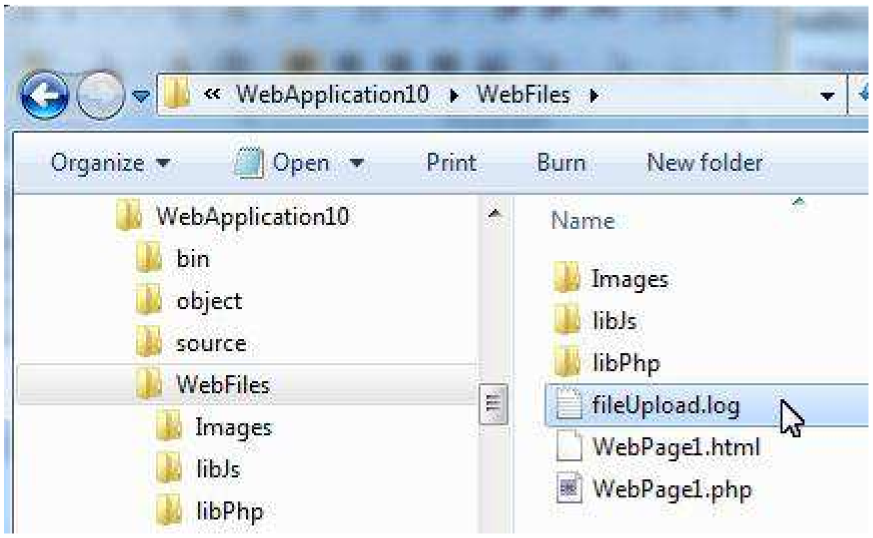
We may test it now:

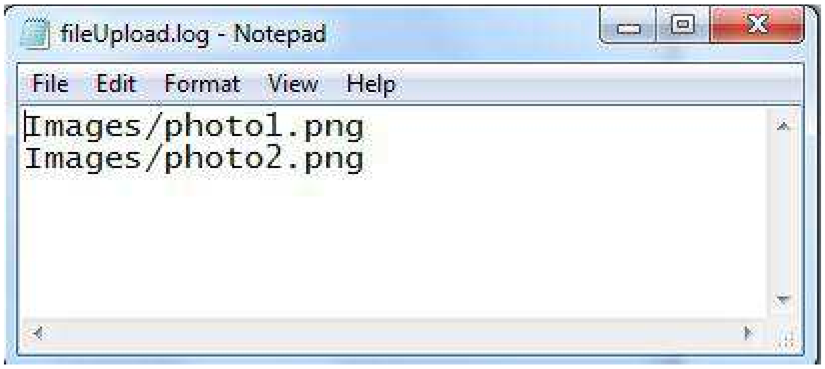


The web page appears. Add 2 image files. Click Upload. The files are uploaded. The images appear on the web page.


http://localhost/WebApplication10/WebPage1.html - Windows Internet Explorer
http://localhost/WebApplication10/WebPage1.html
Bing
Favorites
http://localhost/WebApplication10/WebPage1.ht...
Upload
Add file uploader
C:\Applications\photo1.png
Browse...
C:\Applications\photo2.png
Browse...

The log file on the web server is also uploaded:



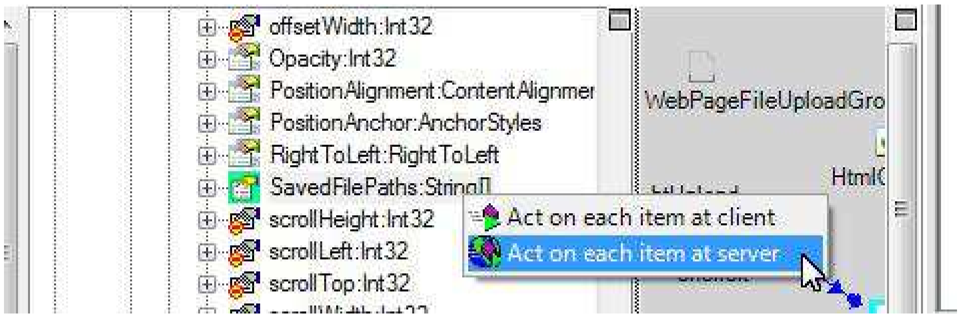


```
Images/photo1.png
Images/photo2.png
```

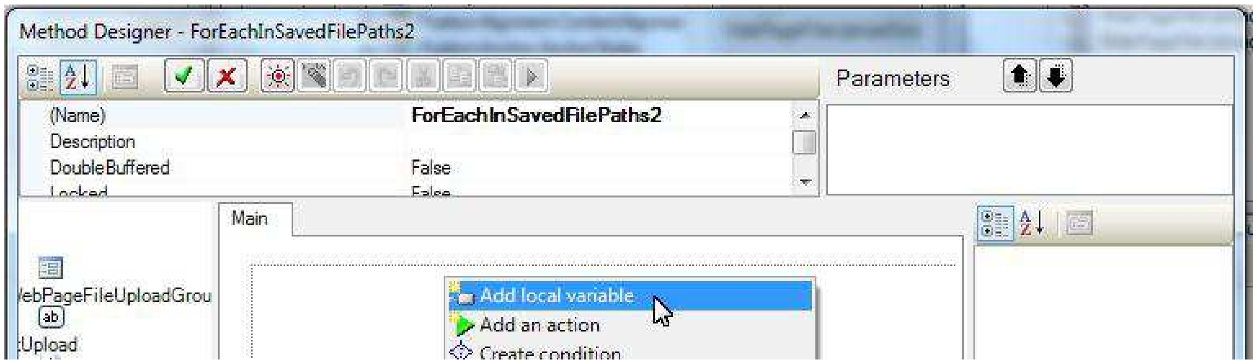
We can see that the names for uploaded files are logged in the file.

Use ASPX to do logging

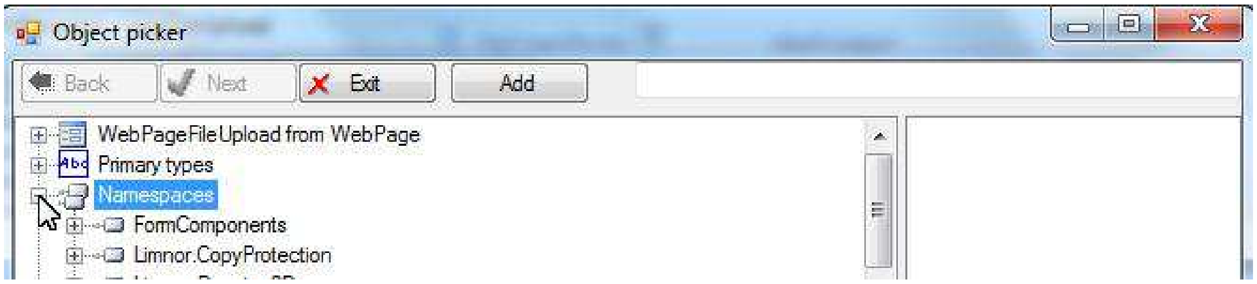
Right-click SavedFilePaths of HtmlFileUploadGroup1; choose “Act on each item at server”

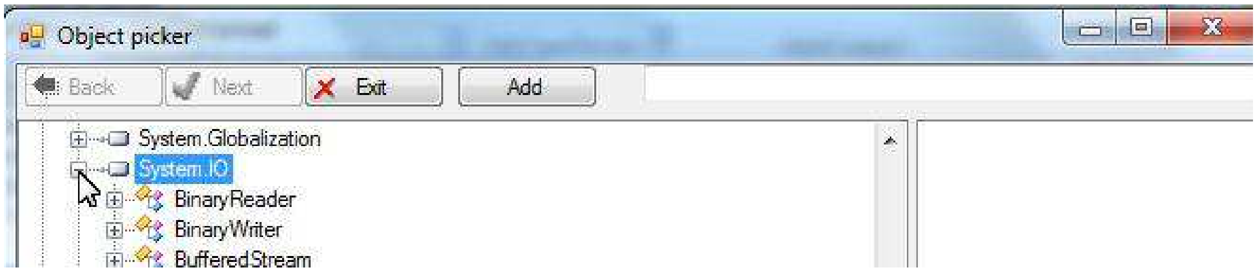


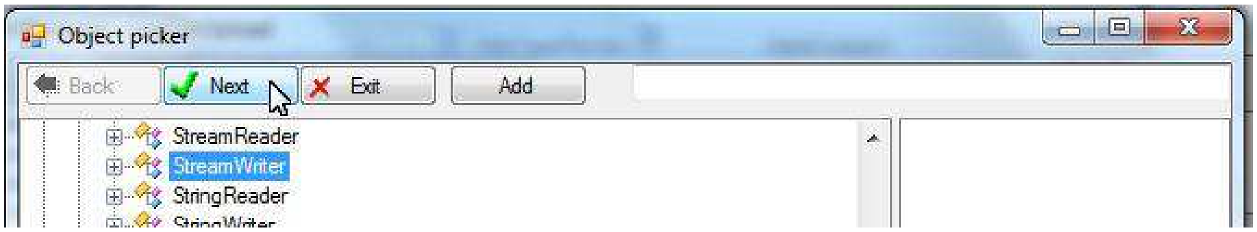
Create a StreamWriter to do the job:

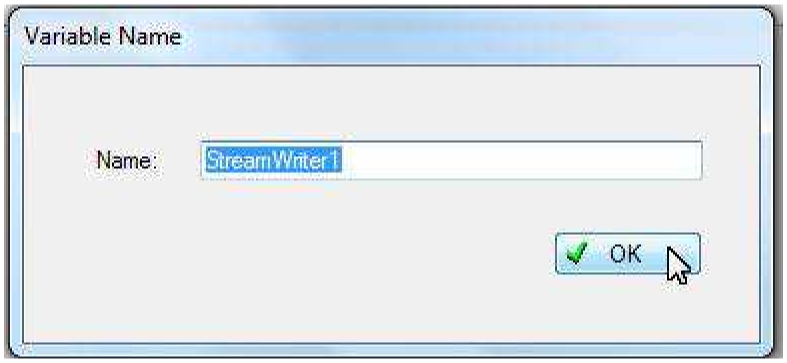


(Name)	ForEachInSavedFilePaths2
Description	
DoubleBuffered	False
Locked	False

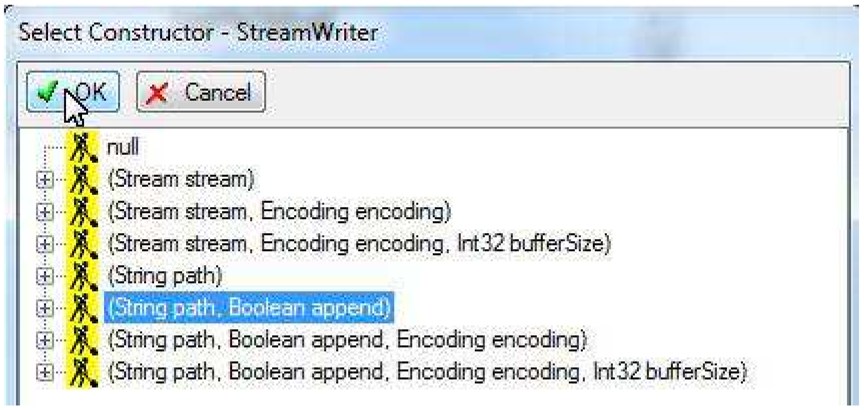




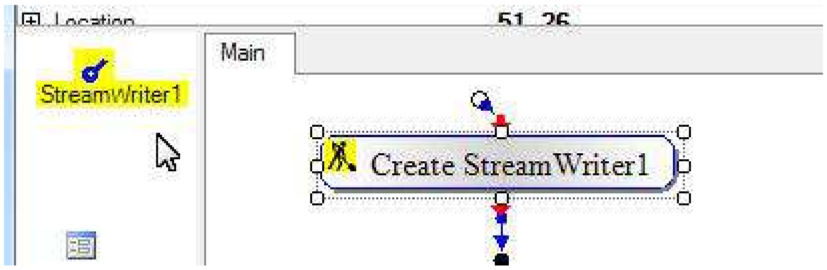




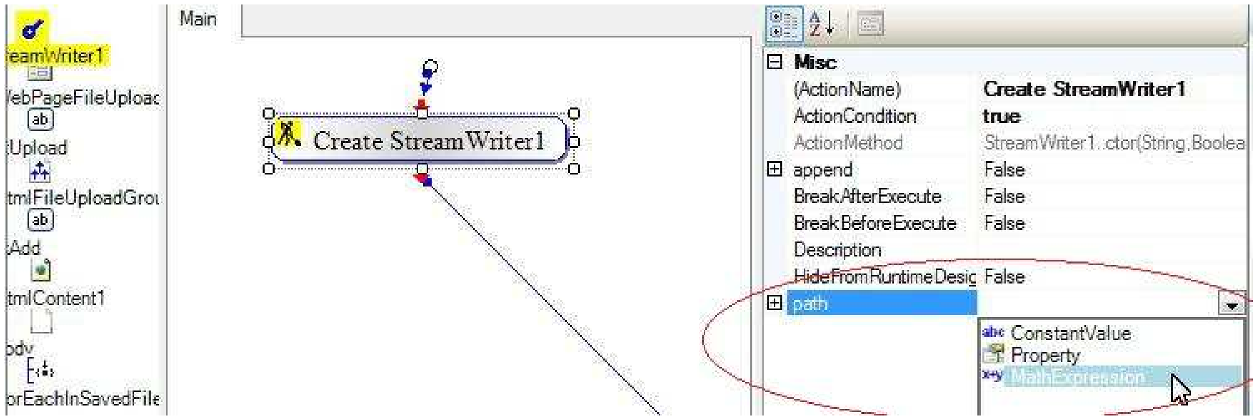
Select a constructor which accepts a file path and a flag indicating that we want to append the file:


Select Constructor - StreamWriter
OK
Cancel
null
(Stream stream)
(Stream stream, Encoding encoding)
(Stream stream, Encoding encoding, Int32 bufferSize)
(String path)
(String path, Boolean append)
(String path, Boolean append, Encoding encoding)
(String path, Boolean append, Encoding encoding, Int32 bufferSize)

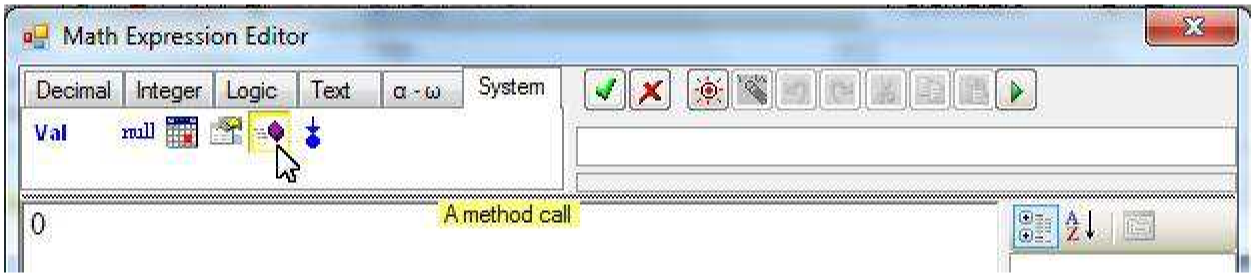
A variable of StreamWriter is created and an action appears for creating the variable:

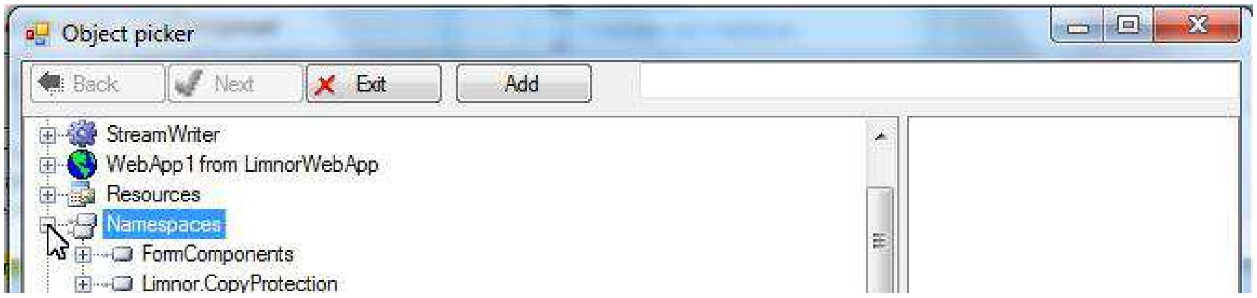


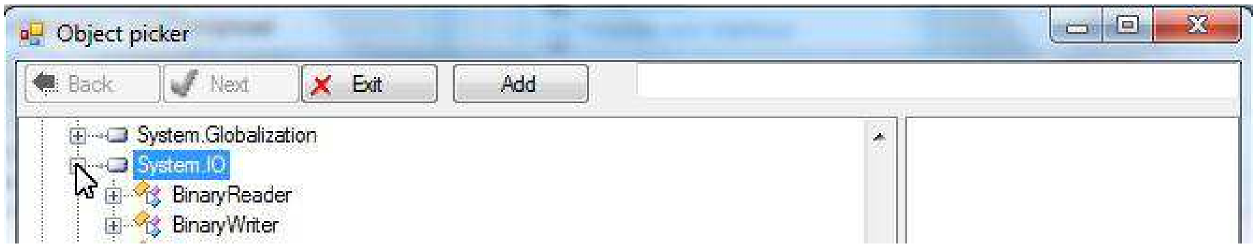
Suppose our log file name is “FileUpload.log”. Where should we save the file? For this sample, we simply save it on the same folder as the web application, which is represented by property `AspxPhysicalFolder` of the web page. We may use Path class’ Combine method to form the full path of the log file.

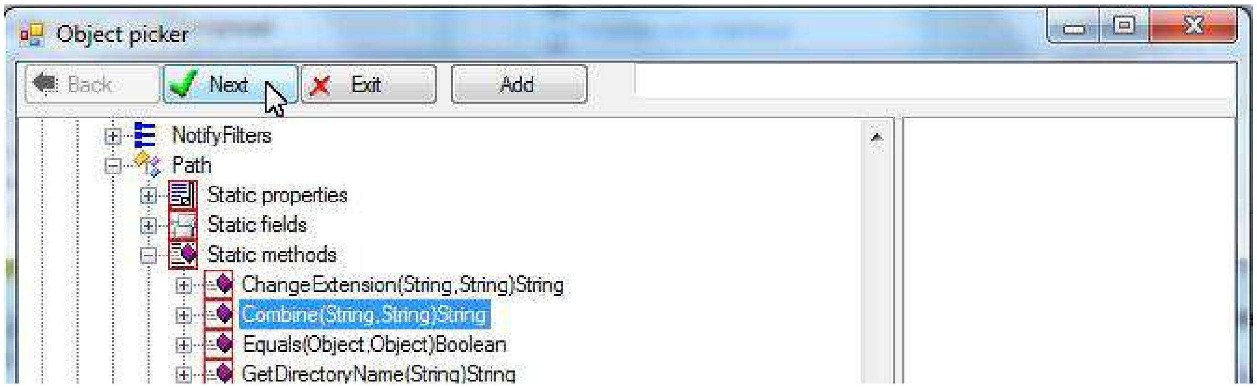


(ActionName)	Create StreamWriter1
ActionCondition	true
ActionMethod	StreamWriter1..ctor(String,Boolea
append	False
BreakAfterExecute	False
BreakBeforeExecute	False
Description	
HideFromRuntimeDesig	False
path	

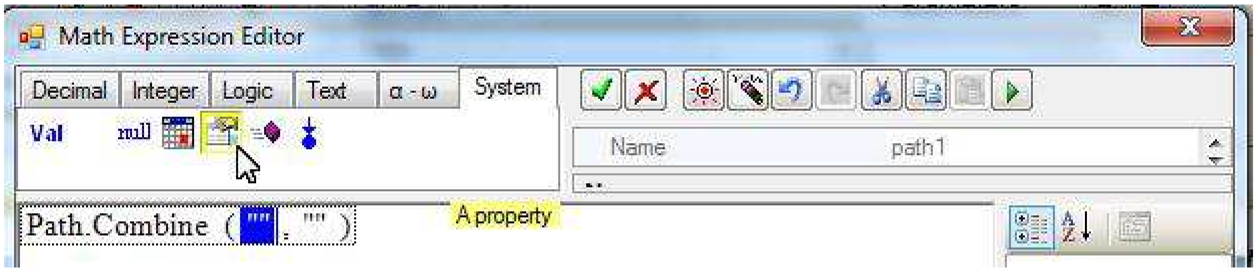




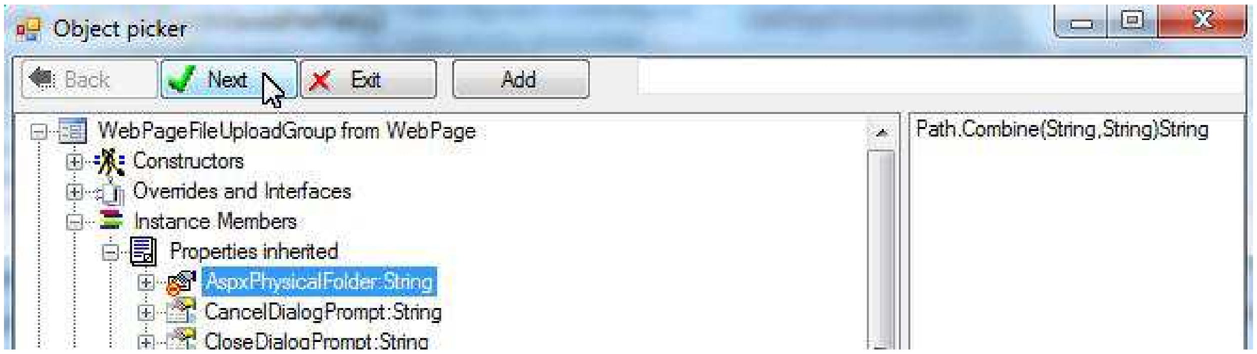



Object picker
Back
Next
Exit
Add
NotifyFilters
Path
Static properties
Static fields
Static methods
ChangeExtension(String,String)String
Combine(String,String)String
Equals(Object,Object)Boolean
GetDirectoryName(String)String

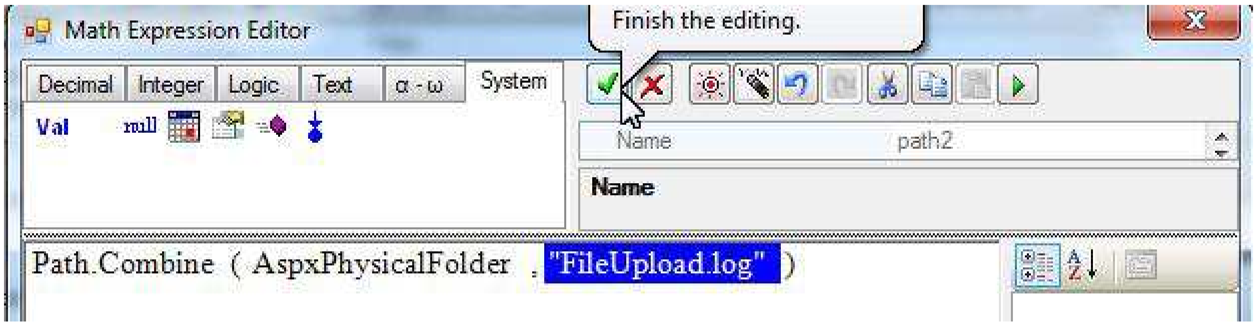
The first parameter of Combine is the folder. Let's use AspxPhysicalFolder:



```
Path.Combine ( "" , "" )
```

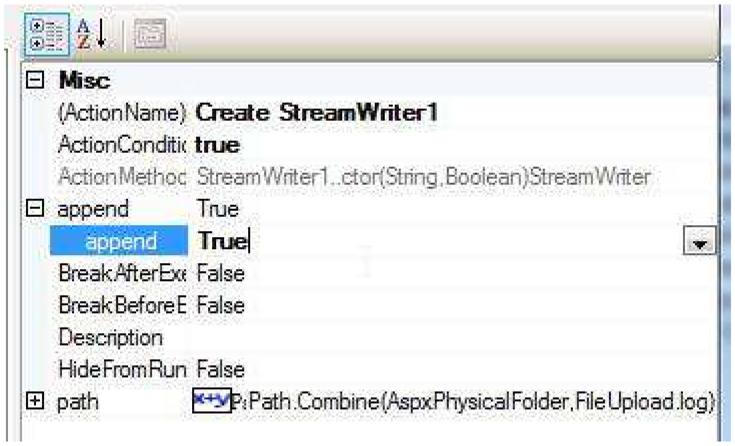


The second parameter of Combine is file name:



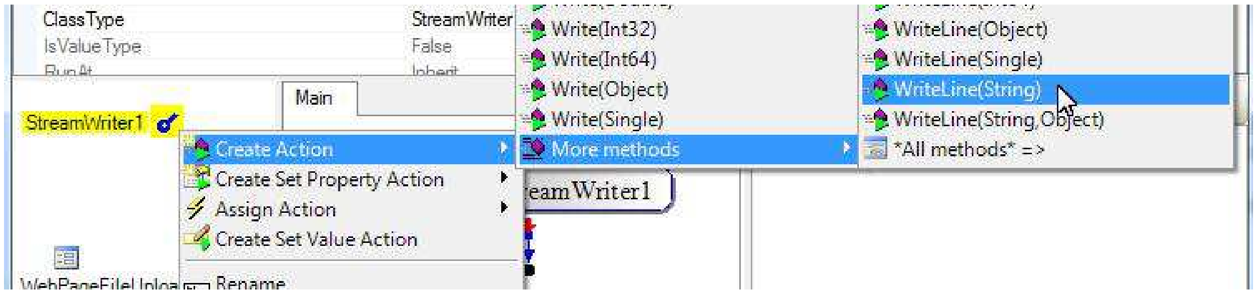
Path.Combine (AspxPhysicalFolder , "FileUpload.log")

Set “append” to True:

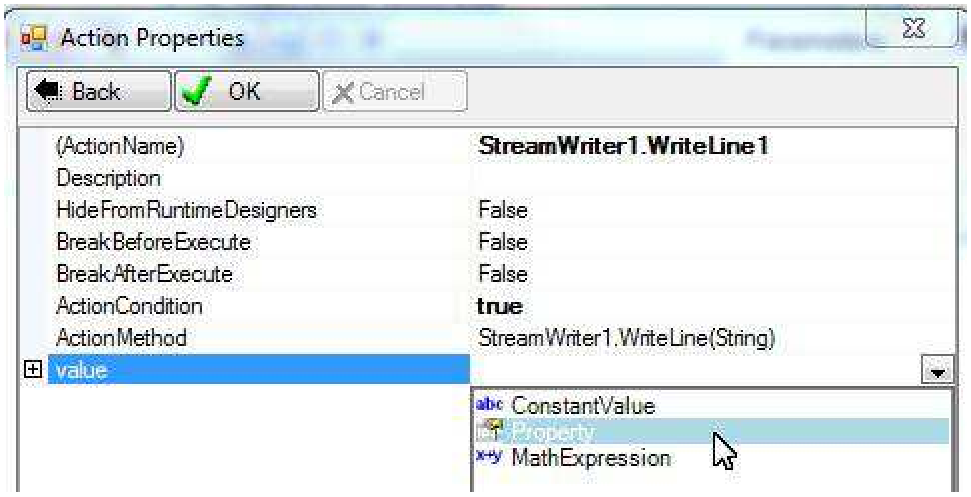


Misc	
(ActionName)	Create StreamWriter1
ActionConditic	true
ActionMethoc	StreamWriter1..ctor(String,Boolean)StreamWriter
append	True
append	True
BreakAfterExe	False
BreakBeforeE	False
Description	
HideFromRun	False
path	P:Path.Combine(AspxPhysicalFolder,FileUpload.log)

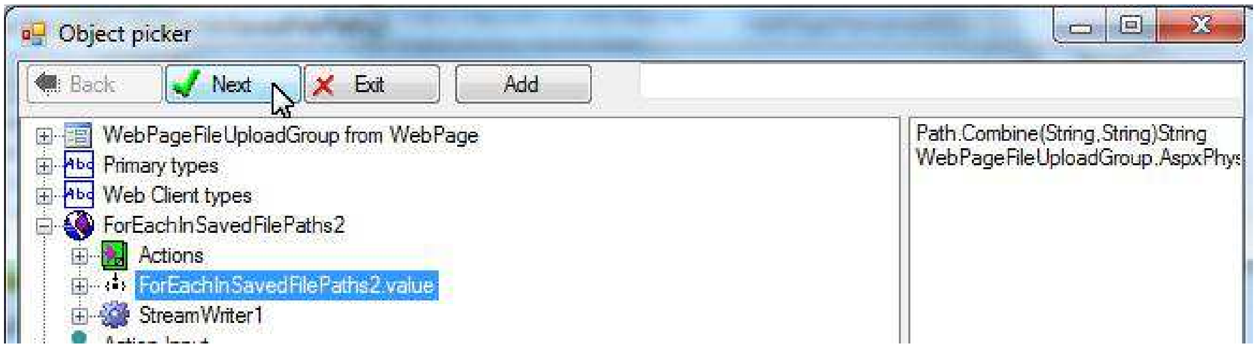
Create a WriteLine action:



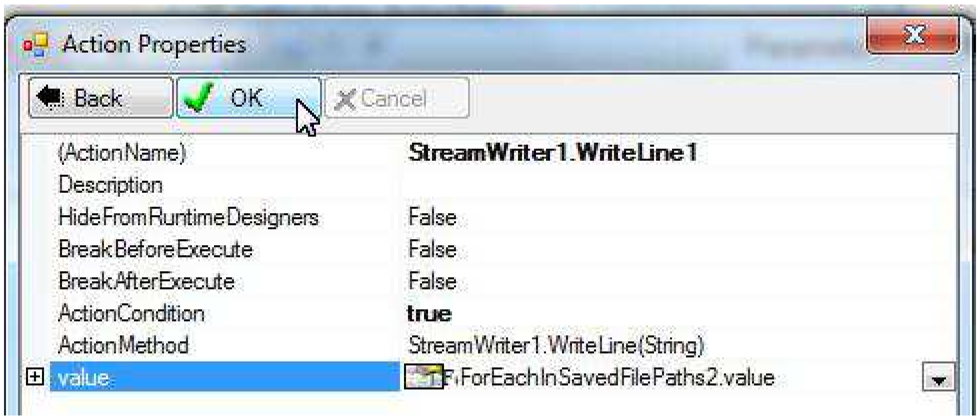
The value to write is the array item we are processing, which is a file name:



(ActionName)	**StreamWriter1.WriteLine1**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	StreamWriter1.WriteLine(String)
value	

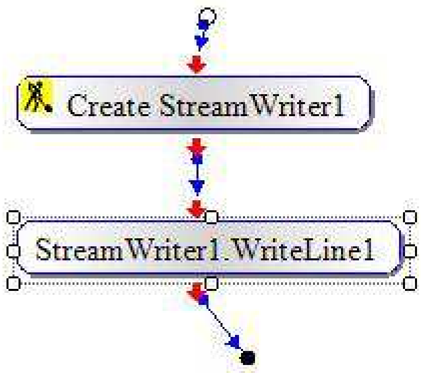

Object picker
Back
Next
Exit
Add
WebPageFileUploadGroup from WebPage
Primary types
Web Client types
ForEachInSavedFilePaths2
Actions
ForEachInSavedFilePaths2.value
StreamWriter1
Path.Combine(String,String)String
WebPageFileUploadGroup.AspxPhys

Click OK to finish creating the action:

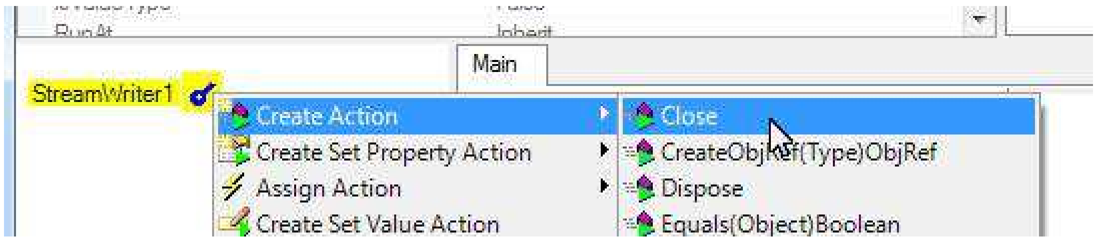


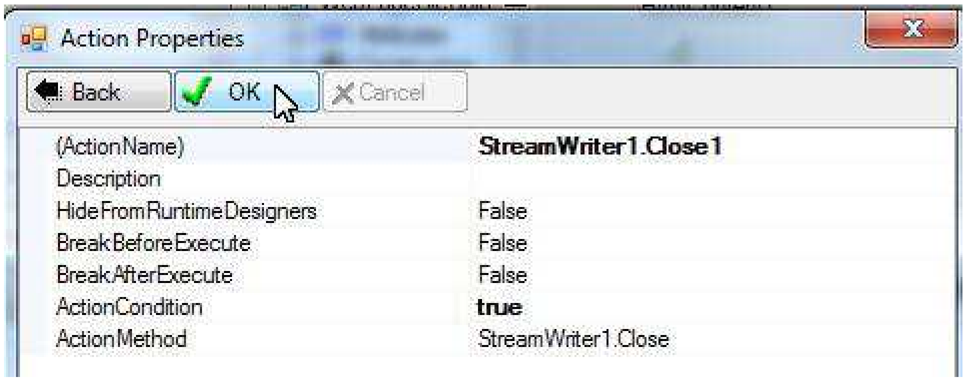
(ActionName)	**StreamWriter1.WriteLine1**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	StreamWriter1.WriteLine(String)
value	ForEachInSavedFilePaths2.value

This action shows the use of SavedFilePaths property on the server side. Link the action to the last action:



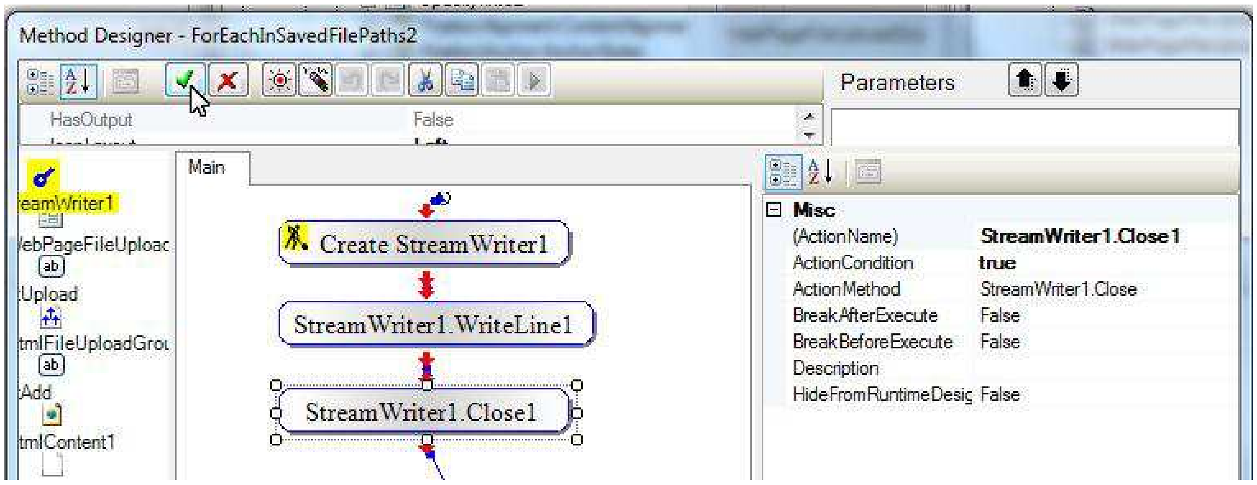
Create an action to close StreamWriter:





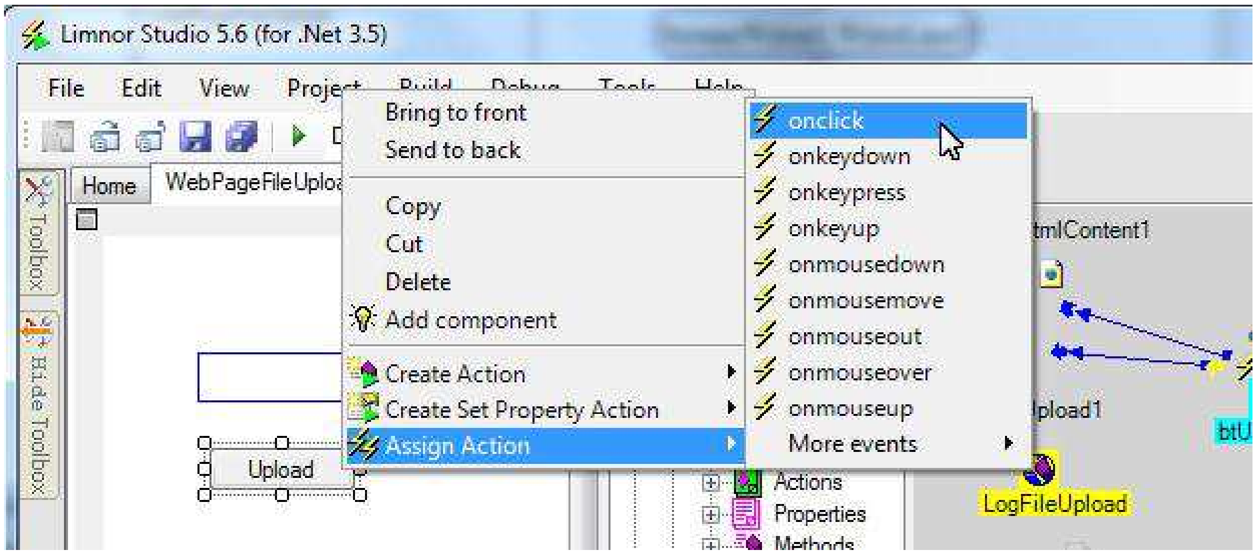
(ActionName)	StreamWriter1.Close1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	StreamWriter1.Close

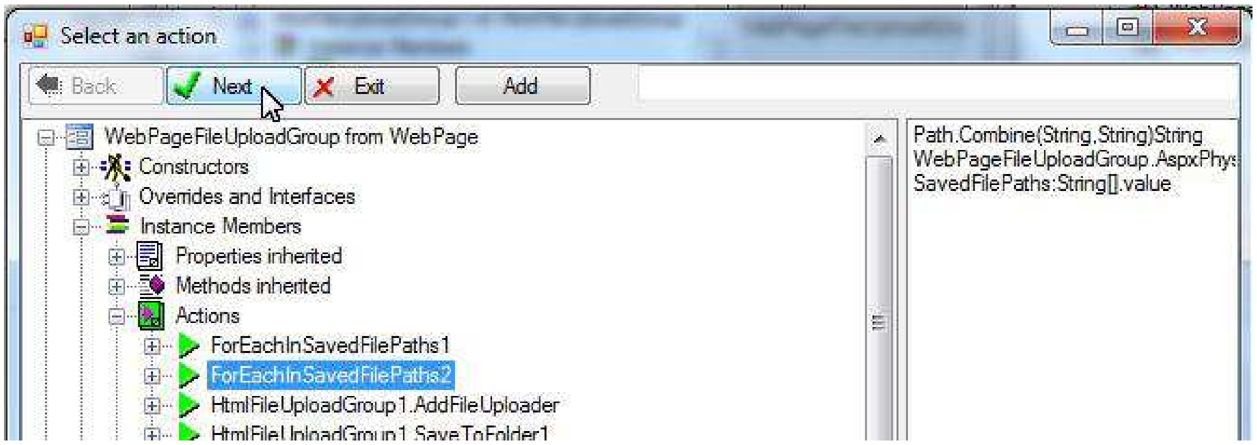
Link the action. We are done creating this server side method:



(ActionName)	StreamWriter1.Close1
ActionCondition	true
ActionMethod	StreamWriter1.Close
BreakAfterExecute	False
BreakBeforeExecute	False
Description	
HideFromRuntimeDesig	False

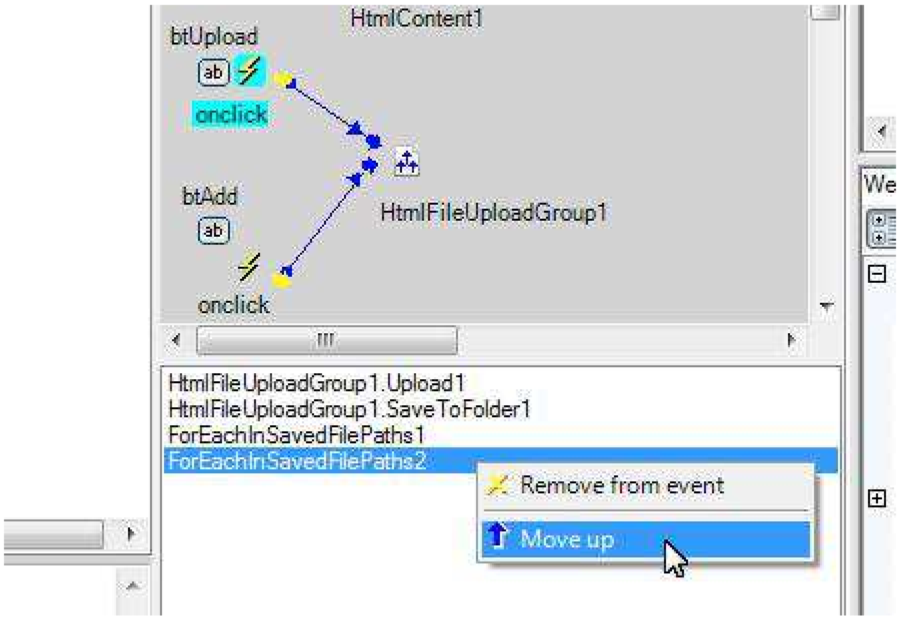
Let's execute the above method when the upload button is clicked:





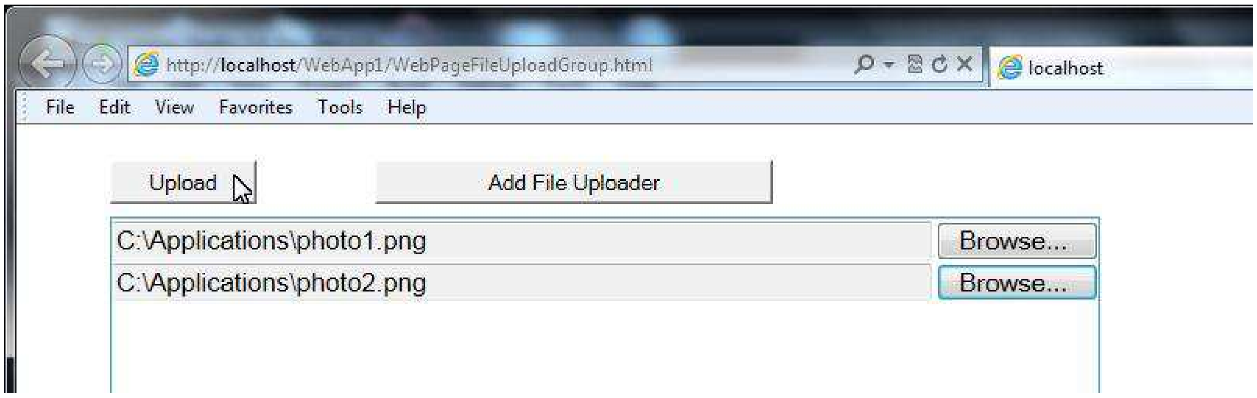
Path.Combine(String,String)String
WebPageFileUploadGroup.AspxPhys
SavedFilePaths:String[].value

The action is assigned to the button. Move it up to make all server side actions together to improve performance:

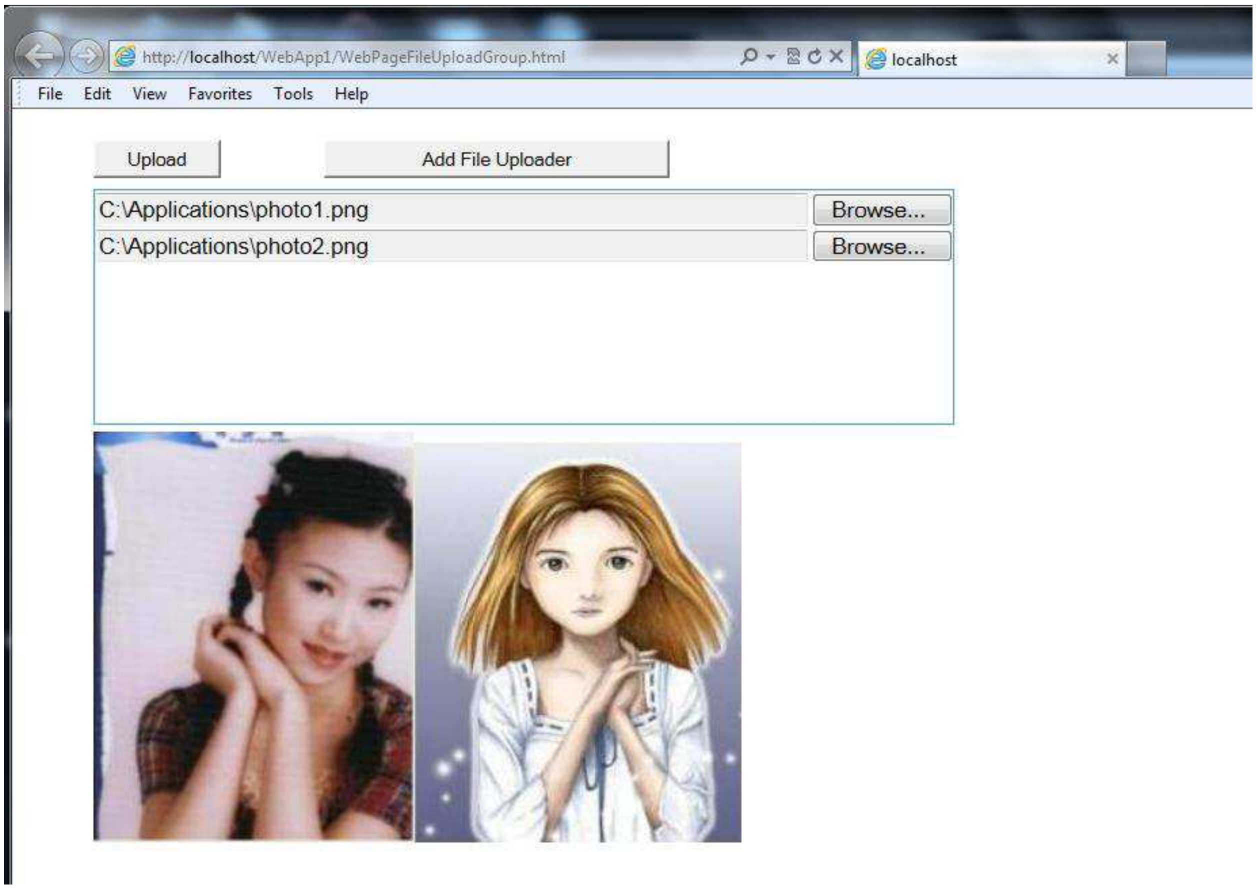


HtmlFileUploadGroup1.Upload1
HtmlFileUploadGroup1.SaveToFolder1
ForEachInSavedFilePaths1
ForEachInSavedFilePaths2

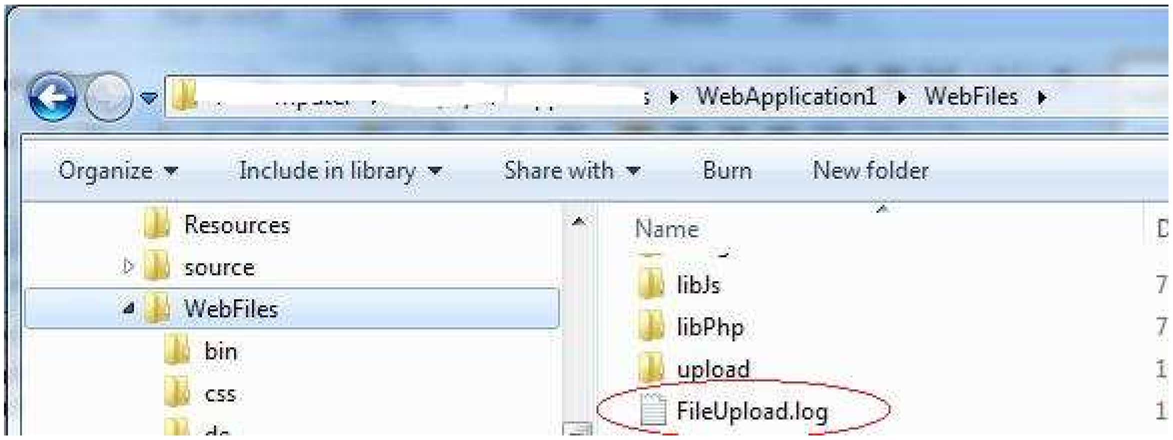
Compile and launch the Aspx web page. Upload two files:


http://localhost/WebApp1/WebPageFileUploadGroup.html
localhost
File Edit View Favorites Tools Help
Upload
Add File Uploader
Browse...
Browse...

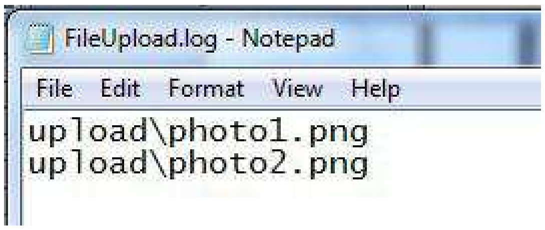
C:\Applications\photo1.png
C:\Applications\photo2.png


http://localhost/WebApp1/WebPageFileUploadGroup.html
localhost
File Edit View Favorites Tools Help
Upload
Add File Uploader
C:\Applications\photo1.png
Browse...
C:\Applications\photo2.png
Browse...

A log file is created:


WebApplication1 ▸ WebFiles ▸
Organize ▾ Include in library ▾ Share with ▾ Burn New folder
Resources
source
WebFiles
bin
css
Name
libJs
libPhp
upload
FileUpload.log

The log file records the file names:



```
upload\photo1.png
upload\photo2.png
```

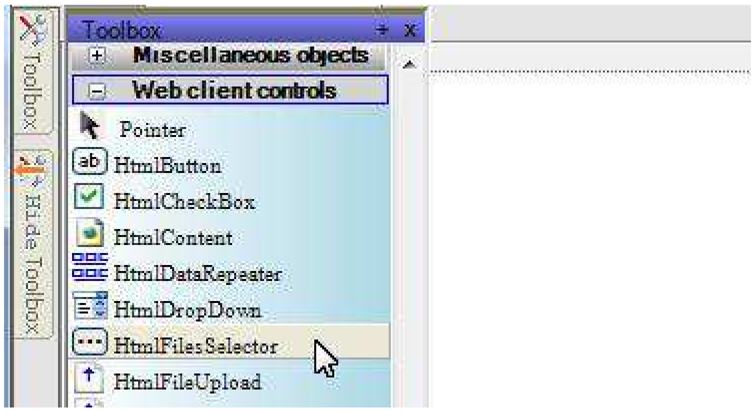
Files Selector

HtmlFileSelector appears as an image on the web page. You may change the image. The web visitors click the image to select files. It can be programmed in a similar way as programming using HtmlFileUploadGroup but displays file selections as icons and file names. It can also be used by PHP SendMail component for file attachments.

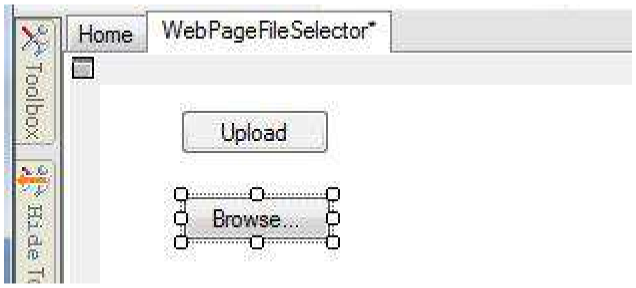
For examples of using HtmlFilesSelector in creating PHP Web Mail, see http://www.limnor.com/support/WebEmail.pdf

Web UI with File Selector

Add an HtmlFileSelector and a button to a web page:

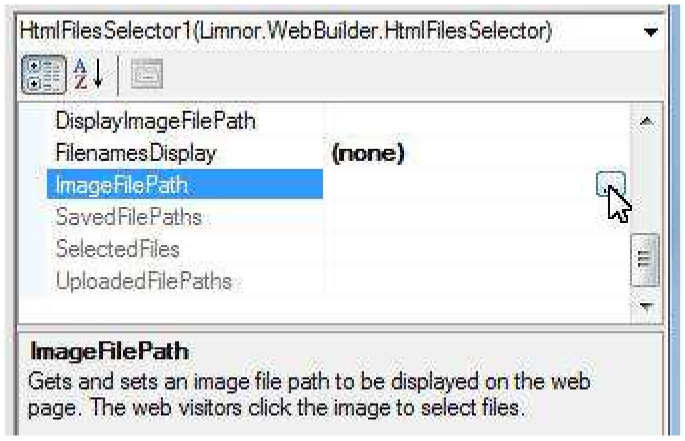


We can see that HtmlFileSelector appears as a button:



Change File Selector Appearance

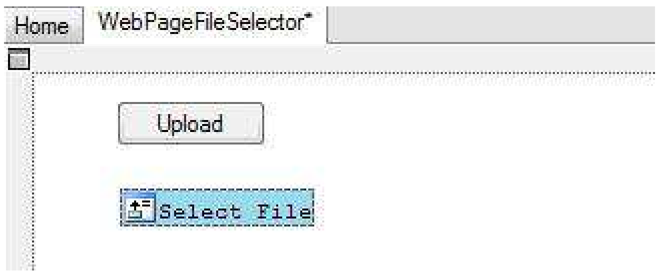
We may change its appearance by using a picture for its ImageFilePath property:


HtmlFilesSelector1(Limnor.WebBuilder.HtmlFilesSelector)

DisplayImageFilePath	
FilenamesDisplay	(none)
ImageFilePath	
SavedFilePaths	
SelectedFiles	
UploadedFilePaths	

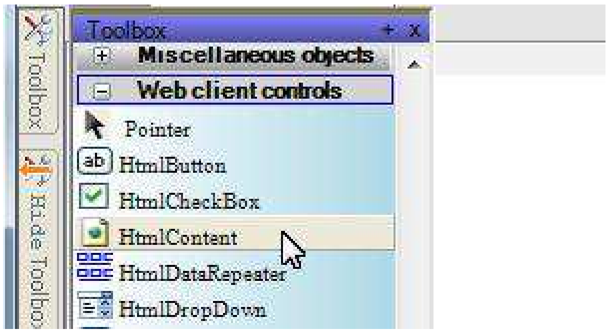
ImageFilePath
Gets and sets an image file path to be displayed on the web page. The web visitors click the image to select files.

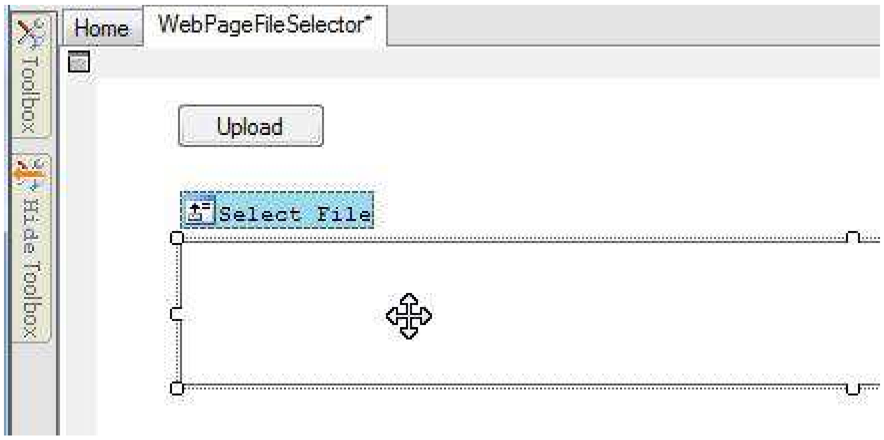
A file selection dialogue box appears for selecting an image file. Selected image appears in place of the HtmlFileSelector:



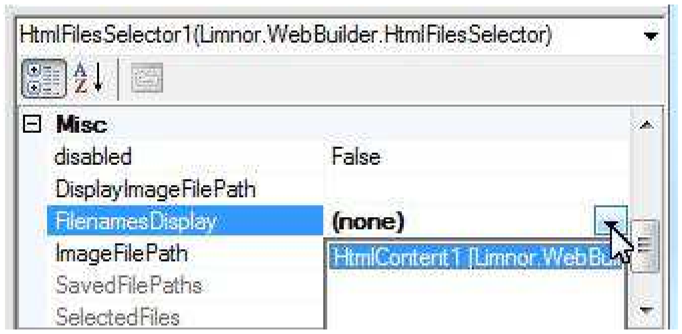
Display Selected File Names

HtmlFileSelector does not have an UI for displaying selected file names. We may use an HtmlContent component to display selected file names.



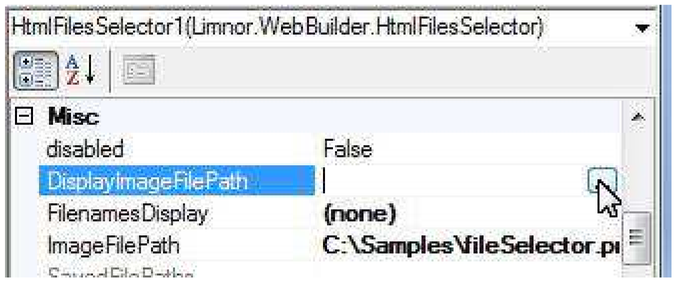


Set “FilenamesDisplay” property of HtmlFileSelector to the HtmlContent:

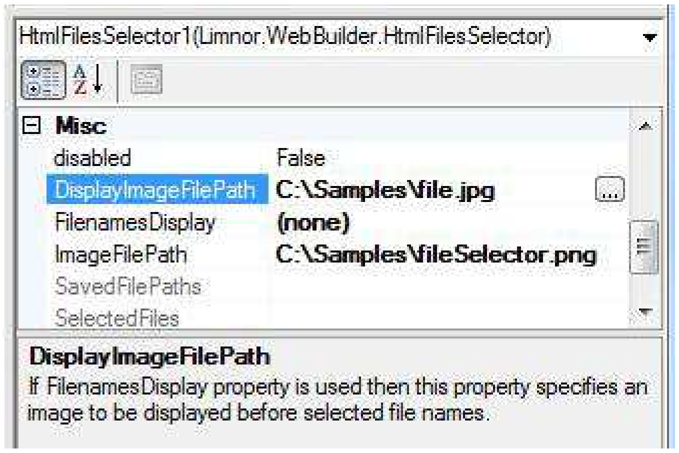

HtmlFilesSelector1(Limnor.WebBuilder.HtmlFilesSelector)

Misc	
disabled	False
DisplayImageFilePath	
FilenamesDisplay	**(none)**
ImageFilePath	HtmlContent1 [Limnor.WebBu
SavedFilePaths	
SelectedFiles	

All selected file names will be displayed in the HtmlContent. We may also display an image before each file name so as to separate file names:


HtmlFilesSelector1(Limnor.WebBuilder.HtmlFilesSelector)

Misc	
disabled	False
DisplayImageFilePath	
FilenamesDisplay	**(none)**
ImageFilePath	**C:\Samples\fileSelector.p**



HtmlFilesSelector1(Limnor.WebBuilder.HtmlFilesSelector)

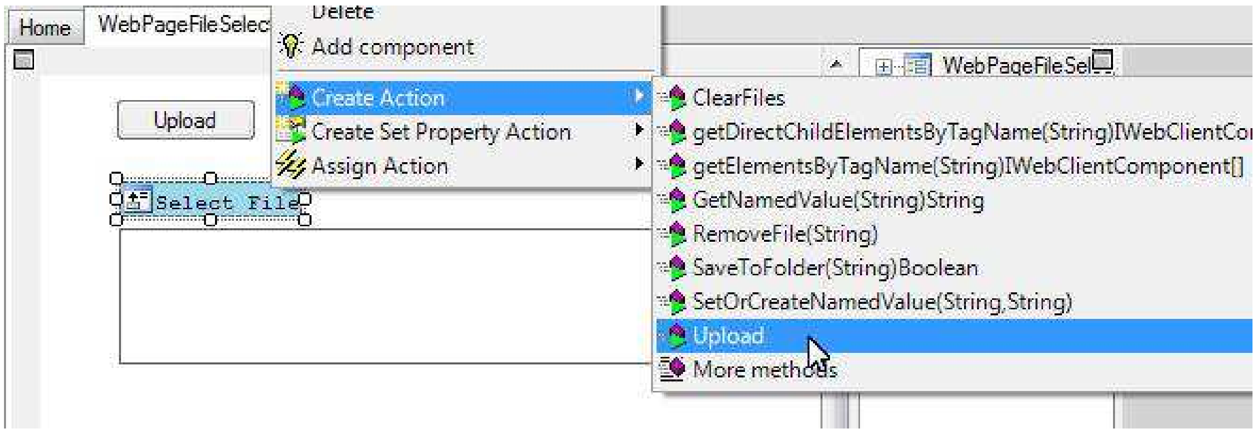
Misc	
disabled	False
DisplayImageFilePath	C:\Samples\file.jpg
FilenamesDisplay	(none)
ImageFilePath	C:\Samples\fileSelector.png
SavedFilePaths	
SelectedFiles	

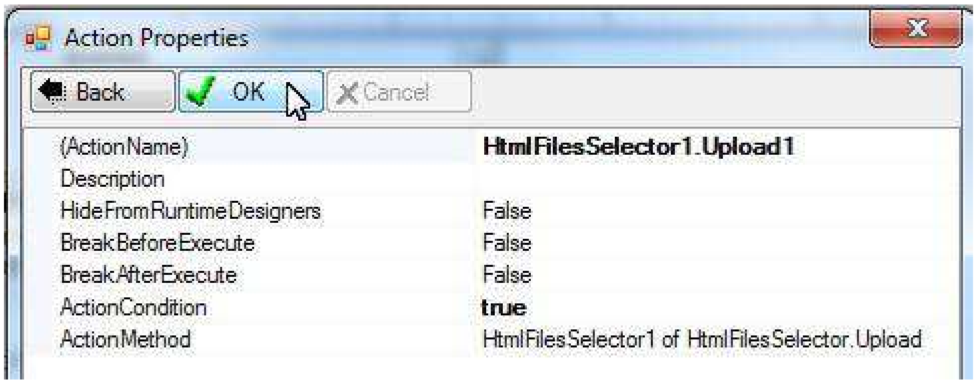
DisplayImageFilePath
If FilenamesDisplay property is used then this property specifies an image to be displayed before selected file names.

Actions

Create Upload Actions

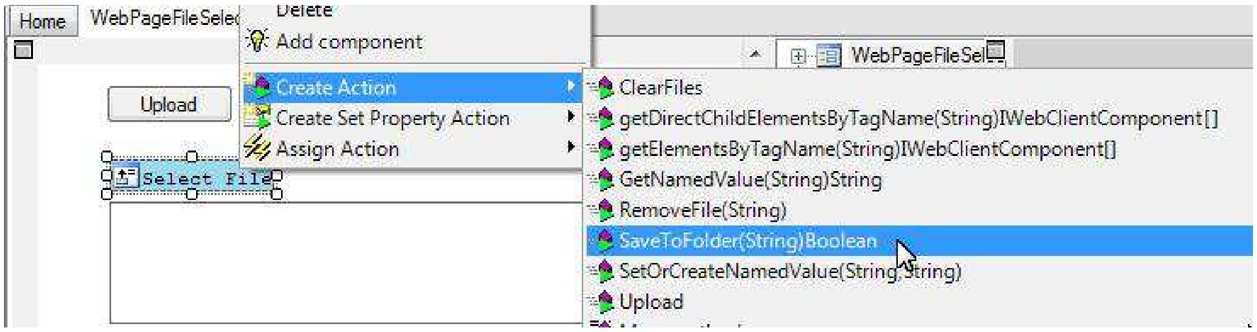
Create an Upload action to upload all selected files to web server:


Home
WebPageFileSelec
Delete
Add component
Create Action
Create Set Property Action
Assign Action
Upload
Select File
ClearFiles
getDirectChildElementsByTagName(String)IWebClientCo
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
RemoveFile(String)
SaveToFolder(String)Boolean
SetOrCreateNamedValue(String,String)
Upload
More methods
WebPageFileSel

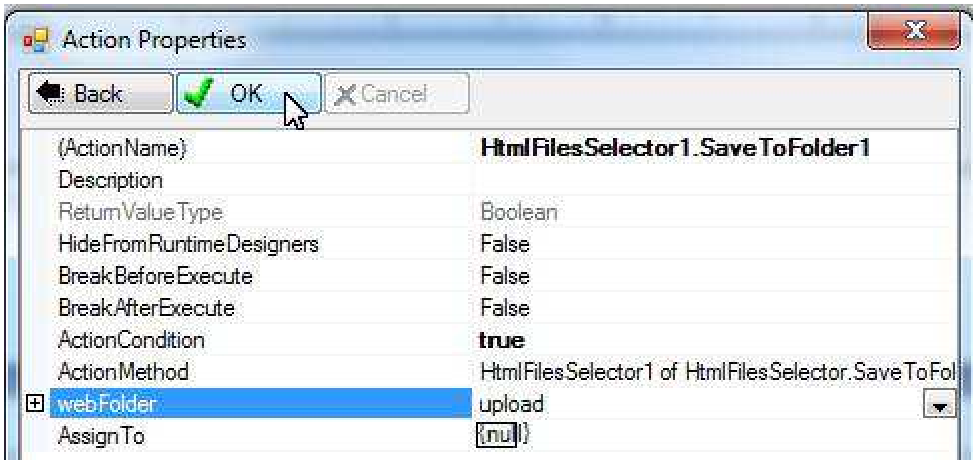


(ActionName)	HtmlFilesSelector1.Upload1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlFilesSelector1 of HtmlFilesSelector.Upload

Create a SaveToFolder action to save all uploaded files to a folder on web server:


Home
WebPageFileSelec
Upload
Select File
Delete
Add component
Create Action
Create Set Property Action
Assign Action
WebPageFileSel
ClearFiles
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
RemoveFile(String)
SaveToFolder(String)Boolean
SetOrCreateNamedValue(String,String)
Upload

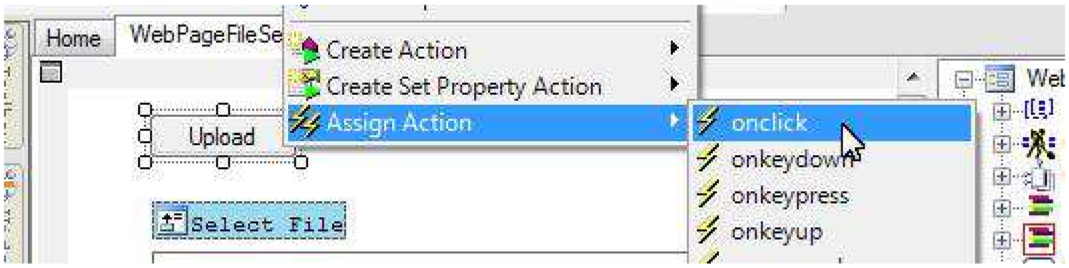
Give a folder for “webFolder” parameter and click OK:

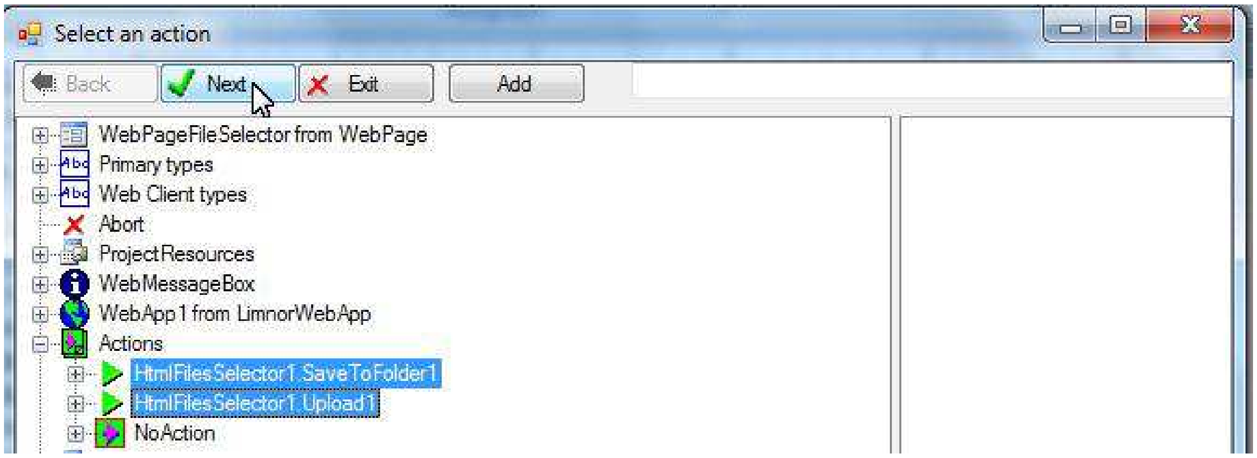


(ActionName)	HtmlFilesSelector1.SaveToFolder1
Description	
ReturnValueType	Boolean
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	HtmlFilesSelector1 of HtmlFilesSelector.SaveToFol
webFolder	upload
AssignTo	(null)

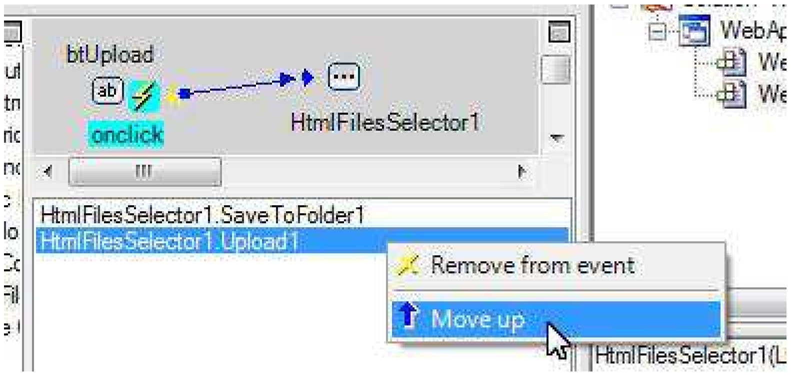
Assign actions

We may assign the above two actions to the Upload button:

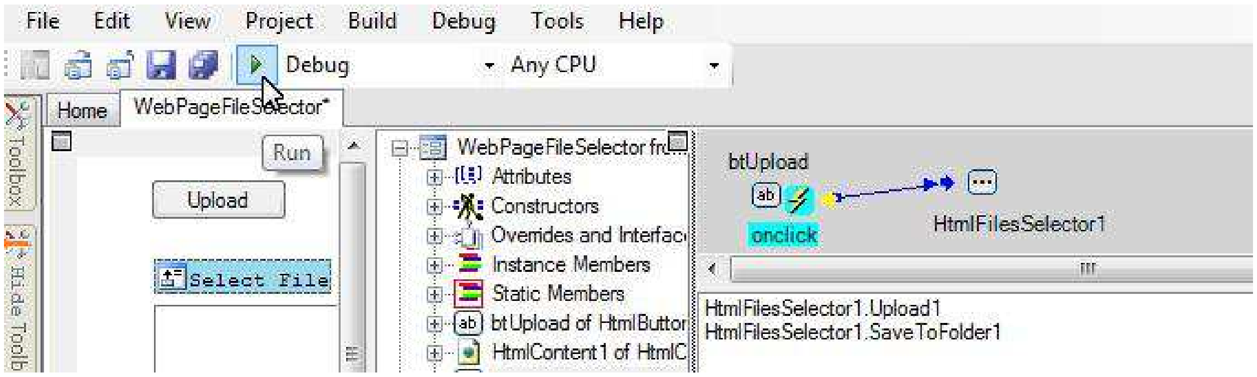



Select an action
Back
Next
Exit
Add
WebPageFileSelector from WebPage
Primary types
Web Client types
Abort
ProjectResources
WebMessageBox
WebApp1 from LimnorWebApp
Actions
HtmlFilesSelector1.SaveToFolder1
HtmlFilesSelector1.Upload1
NoAction

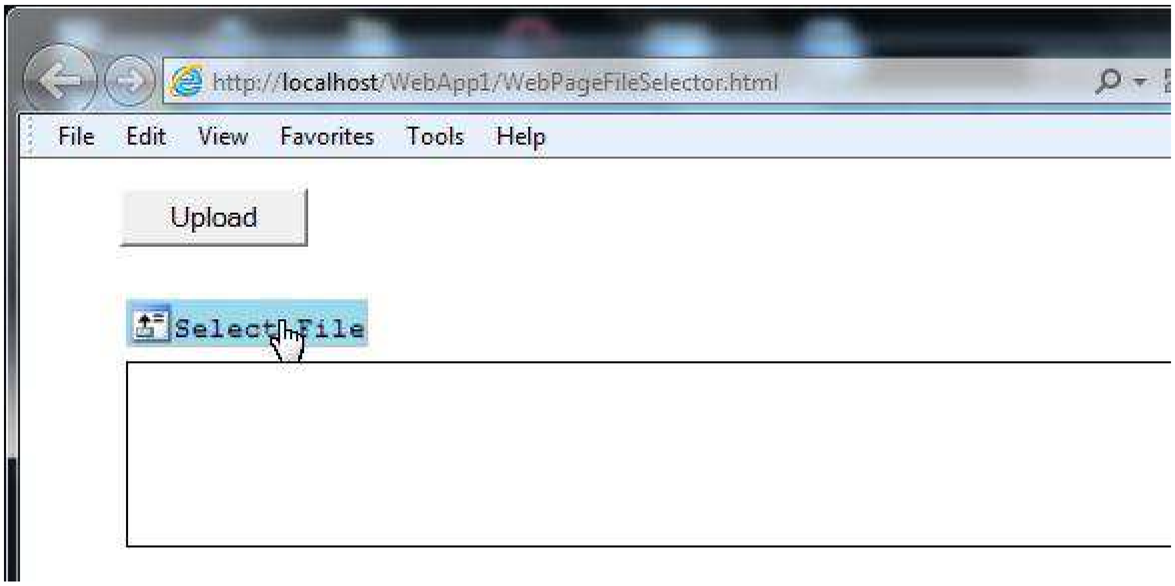
These actions are assigned to the button. Move Upload action to be the first one:



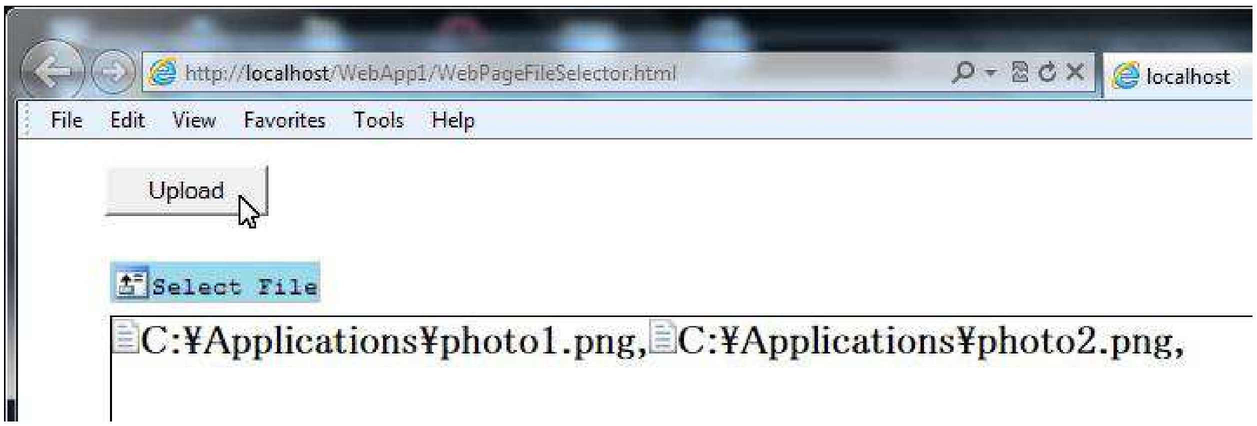
We may test our programming now:



The web page appears. Click the file selector to select a few files:

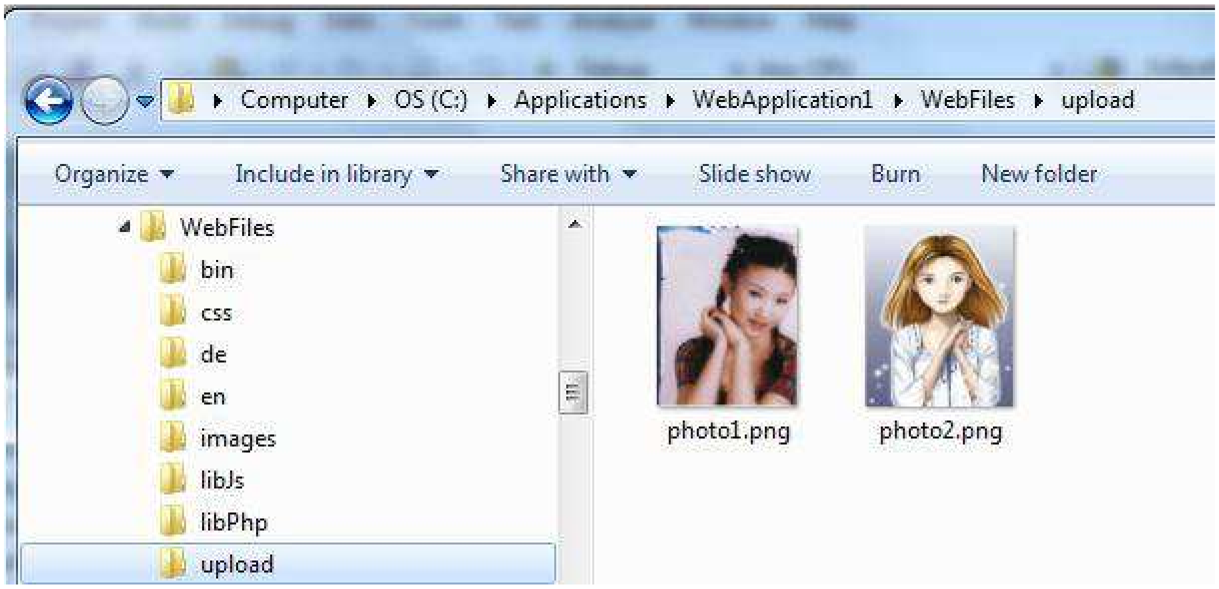

http://localhost/WebApp1/WebPageFileSelector.html
File Edit View Favorites Tools Help
Upload
Select File

Selected file names appear in the HtmlContent component. We may click Upload to upload selected files:


http://localhost/WebApp1/WebPageFileSelector.html
localhost
File Edit View Favorites Tools Help
Upload
Select File

C:¥Applications¥photo1.png,C:¥Applications¥photo2.png,

We can see that the selected files are uploaded to the web server:

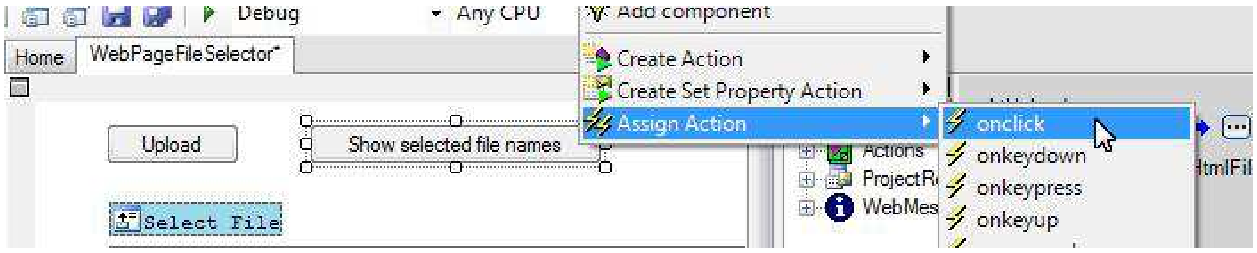


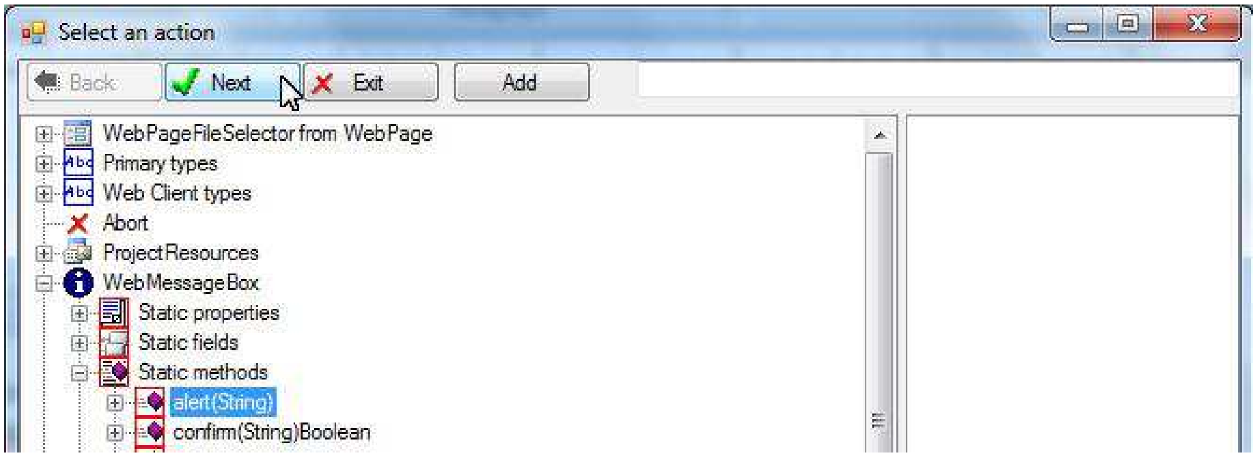
Information about uploaded files

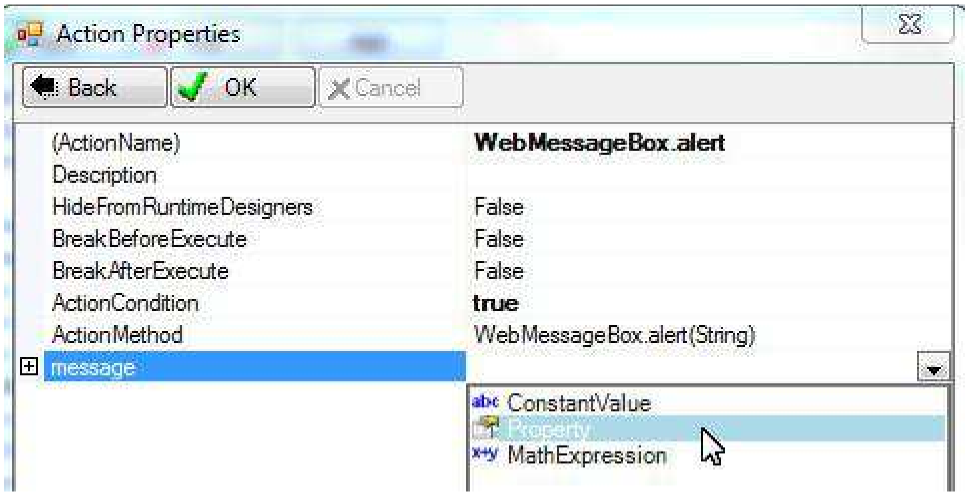
HtmlFileSelector has properties about uploaded files.

Like HtmlFileUploadGroup, it has a SavedFilePaths property, which is an array containing all file URL's for all uploaded files. You may use it as previous HtmlFileUploadGroup sample did.

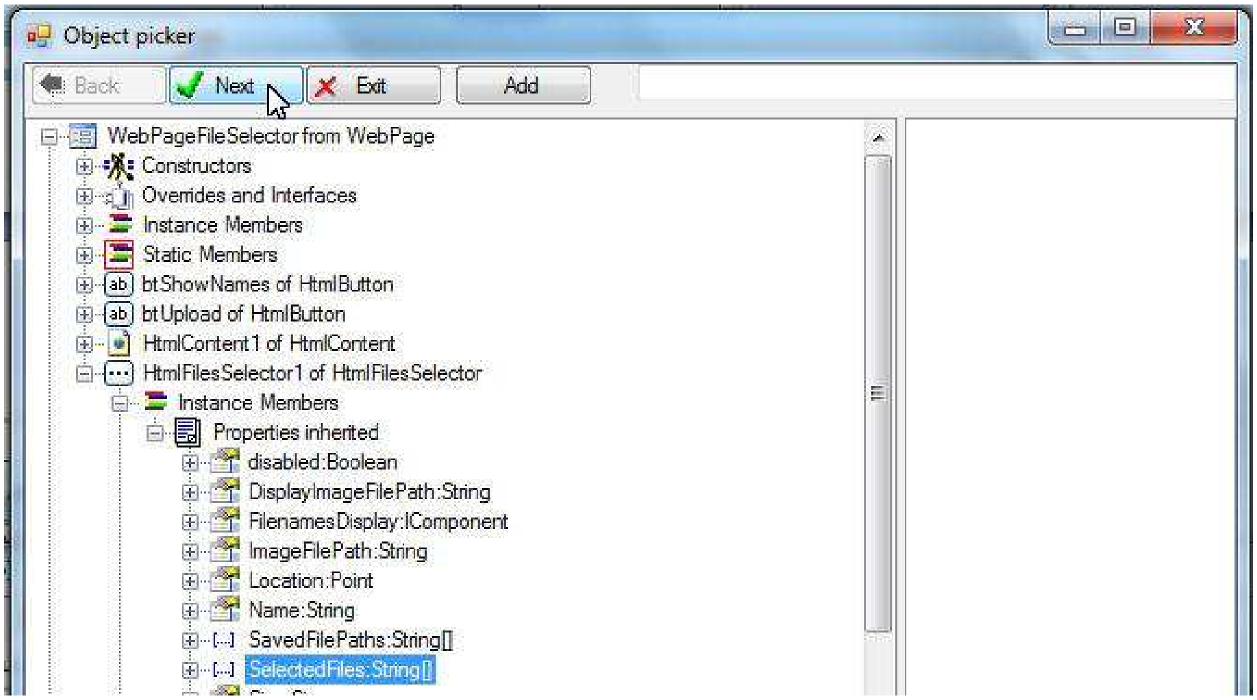
It has a SelectedFiles property, which is an array containing all file paths of selected files. We use a simple action to show a usage of it.



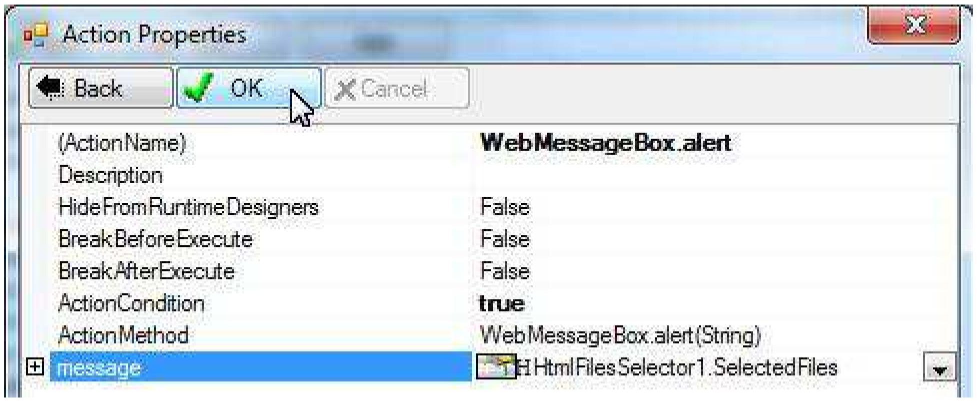




(ActionName) WebMessageBox.alert
Description
HideFromRuntimeDesigners False
BreakBeforeExecute False
BreakAfterExecute False
ActionCondition true
ActionMethod WebMessageBox.alert(String)
message

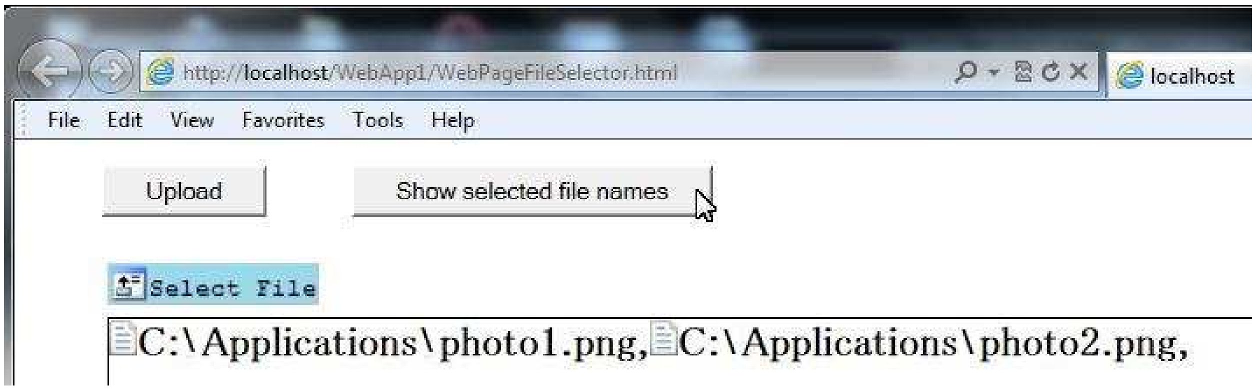


WebPageFileSelector from WebPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
btShowNames of HtmlButton
btUpload of HtmlButton
HtmlContent1 of HtmlContent
HtmlFilesSelector1 of HtmlFilesSelector
Instance Members
Properties inherited
disabled:Boolean
DisplayImageFilePath:String
FilenamesDisplay:IComponent
ImageFilePath:String
Location:Point
Name:String
SavedFilePaths:String[]
SelectedFiles:String[]



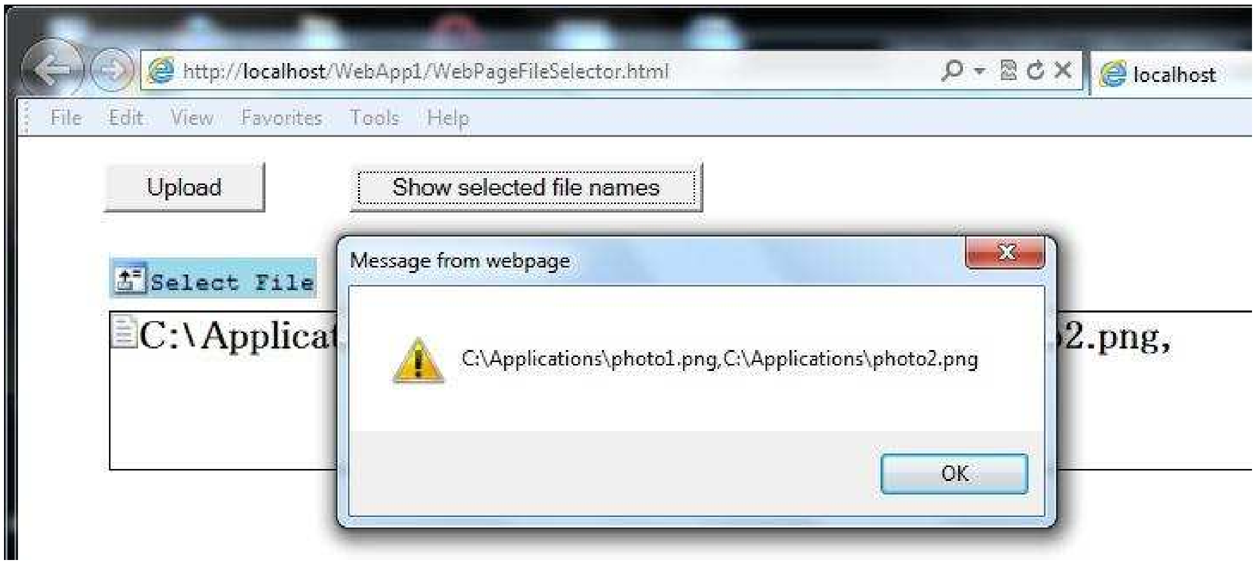
(ActionName) WebMessageBox.alert
Description
HideFromRuntimeDesigners False
BreakBeforeExecute False
BreakAfterExecute False
ActionCondition true
ActionMethod WebMessageBox.alert(String)
message HtmlFilesSelector1.SelectedFiles

Let’s test it. Launch the web application. Select a few files. Click the button:


http://localhost/WebApp1/WebPageFileSelector.html
localhost
File Edit View Favorites Tools Help
Upload
Show selected file names
Select File

C:\Applications\photo1.png, C:\Applications\photo2.png,

A message box appears, showing selected file names:


http://localhost/WebApp1/WebPageFileSelector.html
localhost
File Edit View Favorites Tools Help
Upload
Show selected file names
Select File
Message from webpage
OK

C:\Applications\photo1.png,C:\Applications\photo2.png

Feedbacks

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