Use Webcam in Web Pages

Contents

Introduction

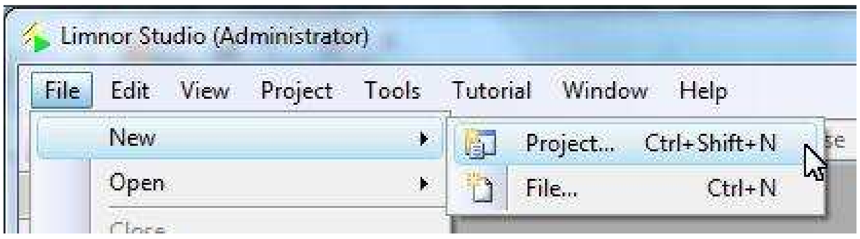
jpegcam (http://code.google.com/p/jpegcam/) is a JavaScript library for using webcam in web pages. This document presents an example of using this library. It uses a web page to display and capture live webcam image and uploads it to web server.

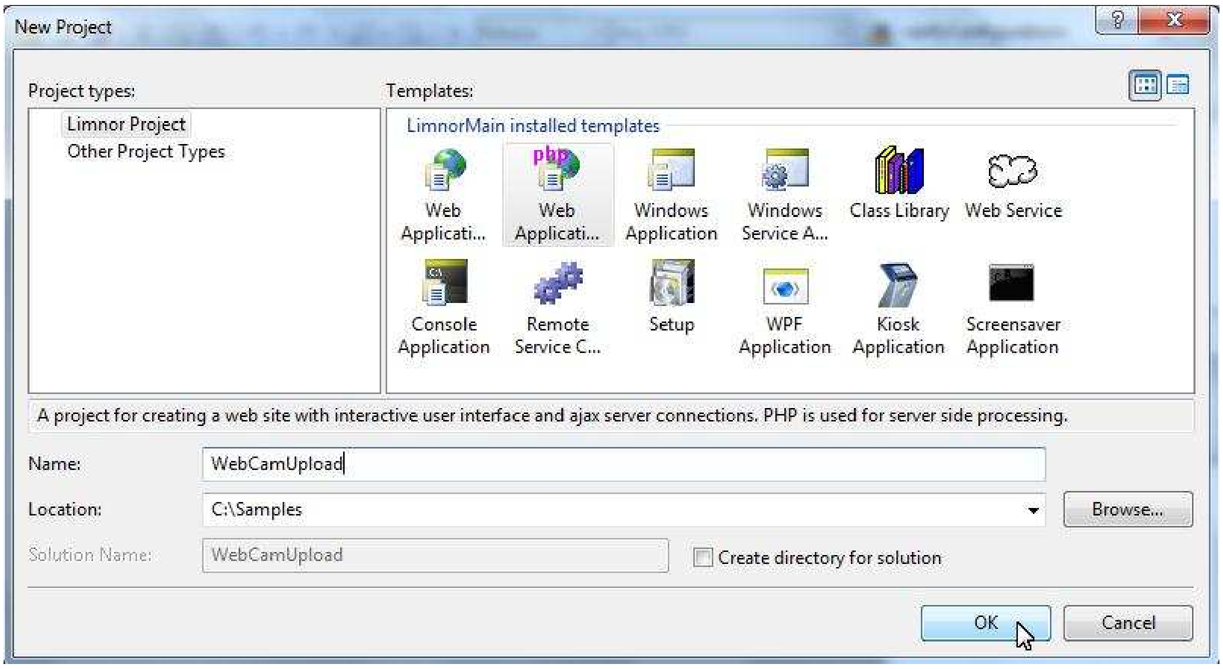
It demonstrates of using JavaScript libraries via a component named JavascriptFiles. The technologies used in each JavaScript library may be outdated in time but the ways of using JavaScript libraries in your web pages demonstrated in this sample will stay. You need to pay attention to see how a JavaScript library is merged into the rest of your web application, as demonstrated in this document.

Preparation of using jpegcam

Create a PHP web project

We create a PHP web project first because we need to explain where the files go.




New Project

Project types: Limnor Project, Other Project Types

Templates: LimnorMain installed templates

A project for creating a web site with interactive user interface and ajax server connections. PHP is used for server side processing.

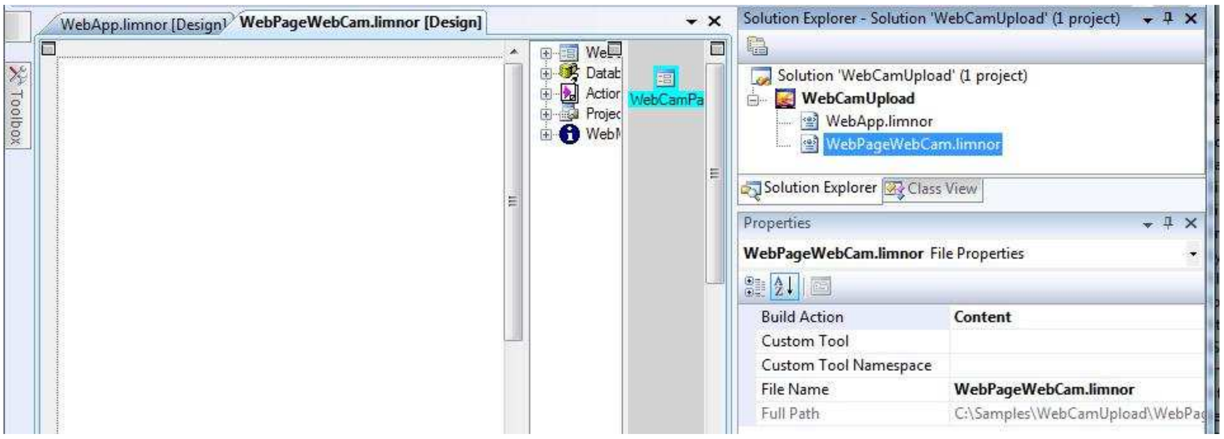
Name: WebCamUpload

Location: C:\Samples

Solution Name: WebCamUpload

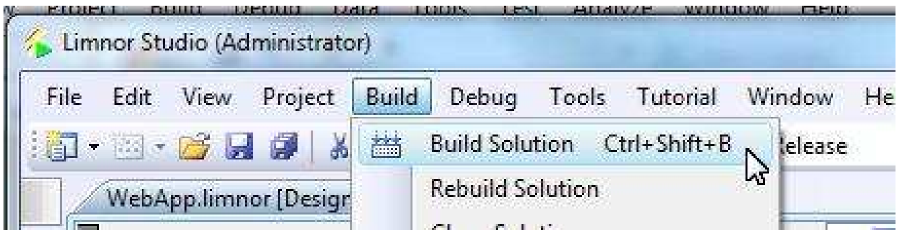
[ ] Create directory for solution

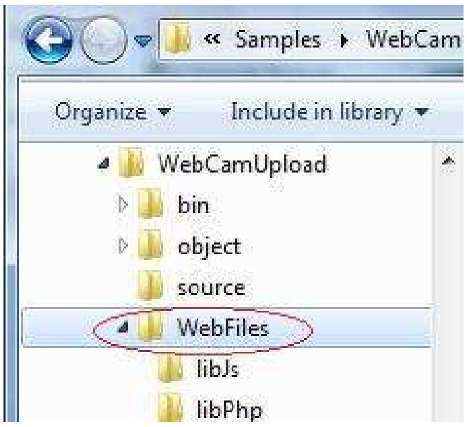
Rename the web page file name and web page name:


WebApp.limnor [Design]
WebPageWebCam.limnor [Design]
Toolbox
WebCamPa
Solution Explorer - Solution 'WebCamUpload' (1 project)
Solution 'WebCamUpload' (1 project)
WebCamUpload
WebApp.limnor
WebPageWebCam.limnor
Solution Explorer
Class View
Properties
WebPageWebCam.limnor File Properties

Build Action	Content
Custom Tool	
Custom Tool Namespace	
File Name	WebPageWebCam.limnor
Full Path	C:\Samples\WebCamUpload\WebPag

Compile the project so that a WebFiles folder will be generated under the project folder:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window He
Build Solution Ctrl+Shift+B
Rebuild Solution
WebApp.limnor [Design

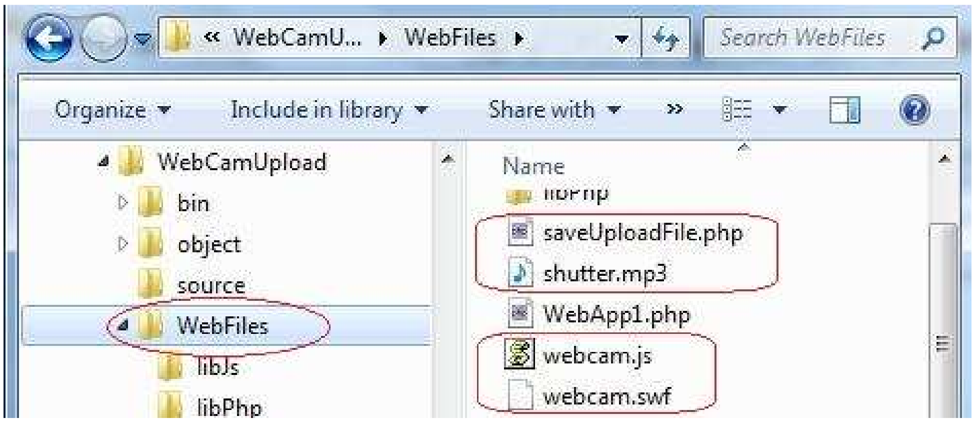


Add JavaScript Library

Add library files

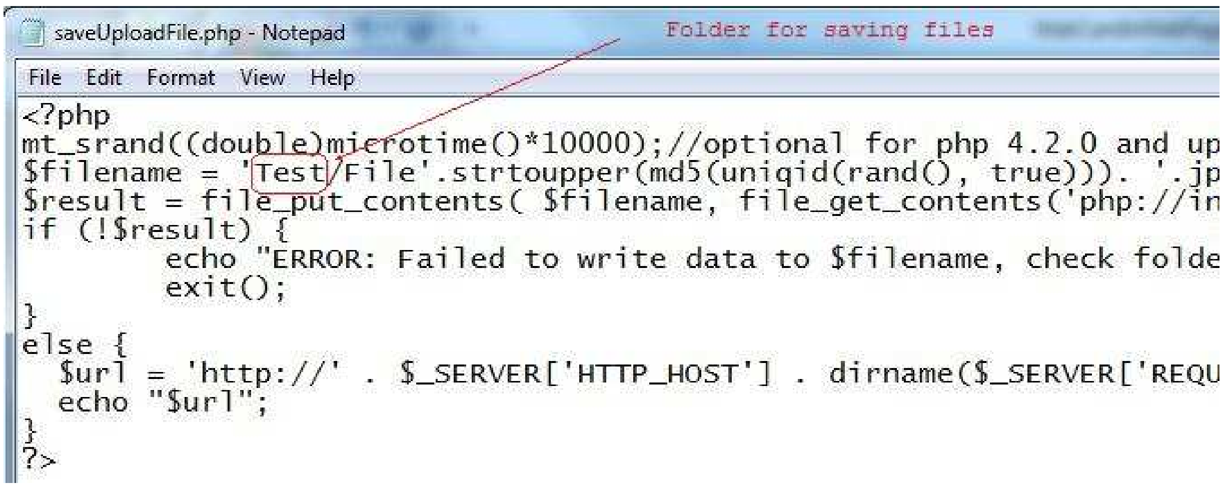
The files needed by jpegcam library can be downloaded from
http://www.limnor.com/studio/jpegWebCam.zip

Unzip the files to WebFiles folder:



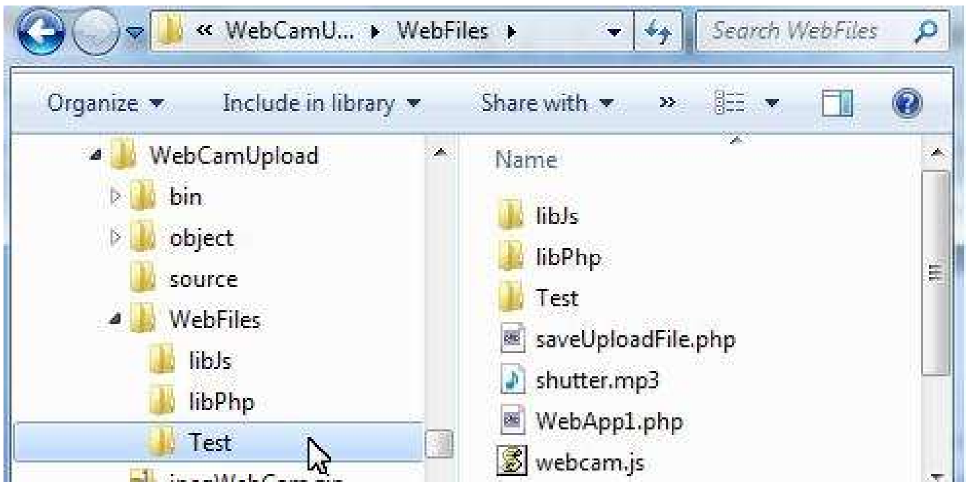
Prepare folder for saving uploaded file

The folder name for saving uploaded files is specified in a PHP file saveUploadFile.php. You may use the Notepad to open it and modified it:



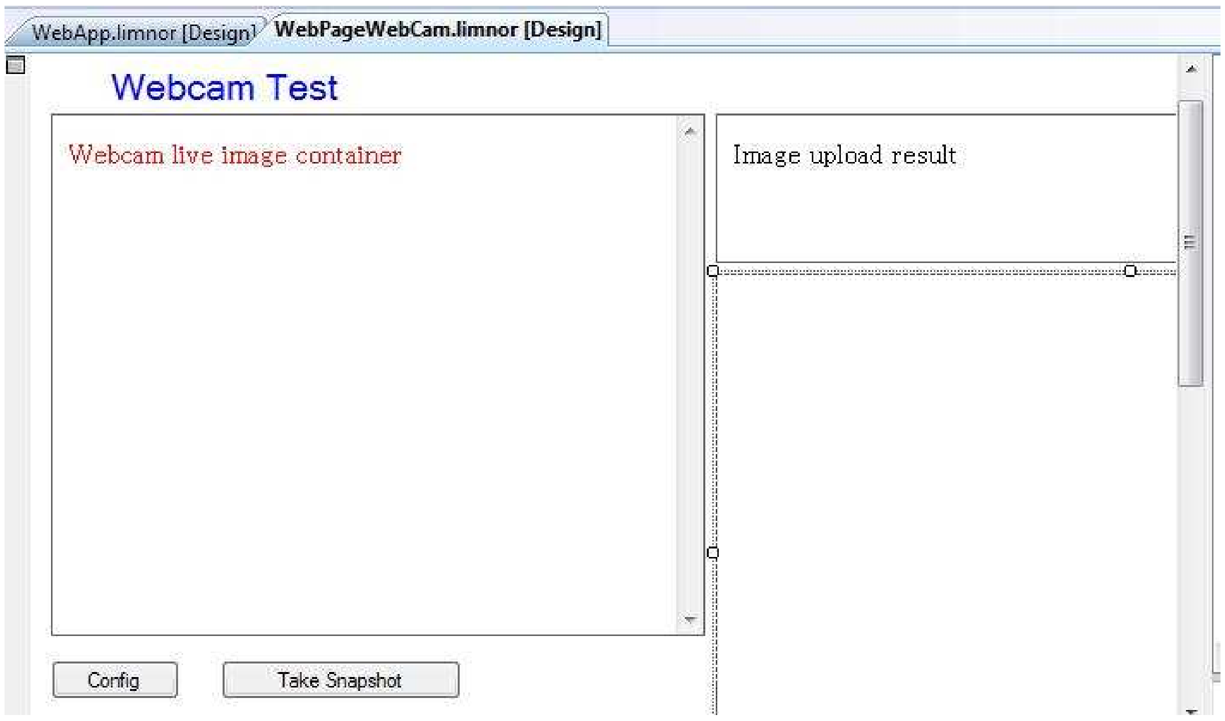
```
<?php
mt_srand((double)microtime()*10000);//optional for php 4.2.0 and up
$filename = 'Test/File'.strtoupper(md5(uniqid(rand(), true))). '.jp
$result = file_put_contents( $filename, file_get_contents('php://in
if (!$result) {
        echo "ERROR: Failed to write data to $filename, check folde
        exit();
}
else {
  $url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['REQU
  echo "$url";
}
?>
```

You need to create the folder under WebFiles folder:



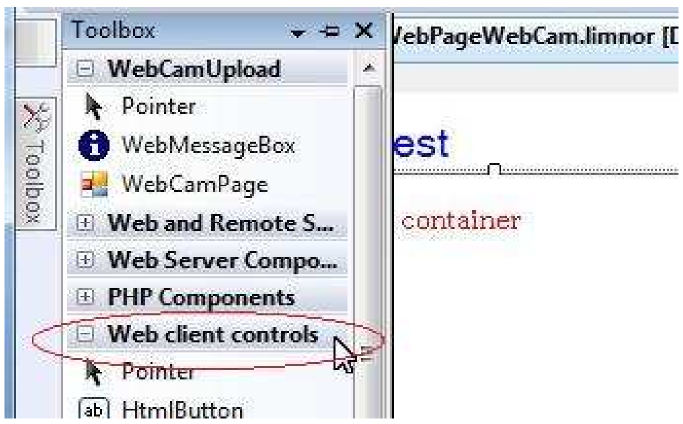
UI for Working with Webcam

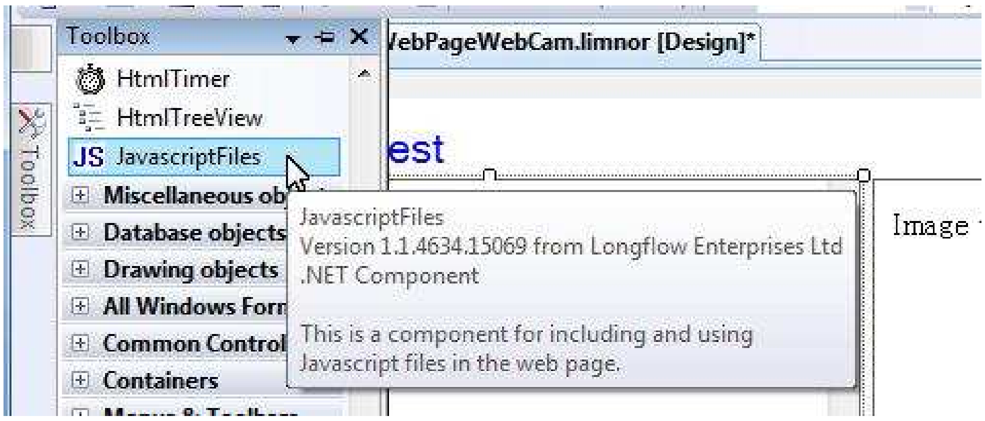
We use a button to configure webcam and a button to take snapshot and upload it to web server. We use one HtmlContent to show live webcam image. We use another HtmlContent to show operation messages. We use an HtmlImage to display the last image uploaded to the web server.



Add JavascriptFiles Component

JavascriptFiles component is used to interface with JavaScript libraries.

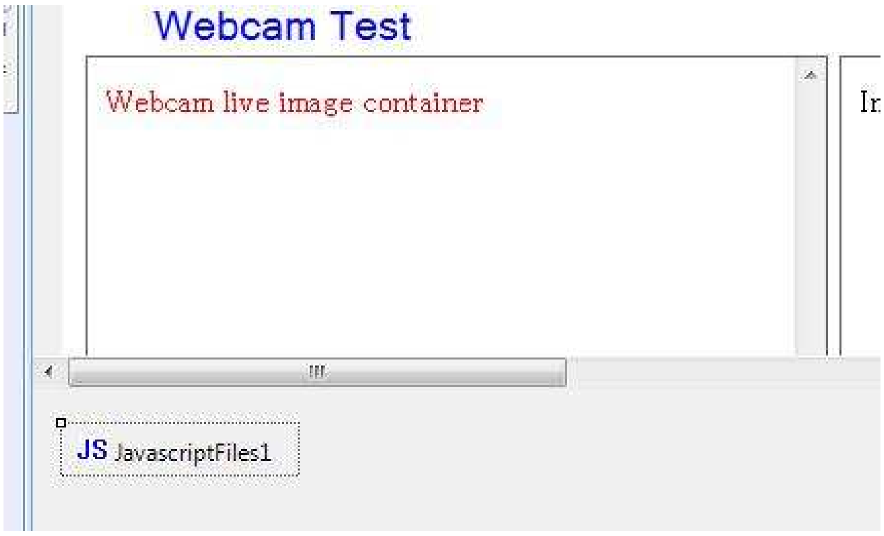




JavascriptFiles
Version 1.1.4634.15069 from Longflow Enterprises Ltd
.NET Component

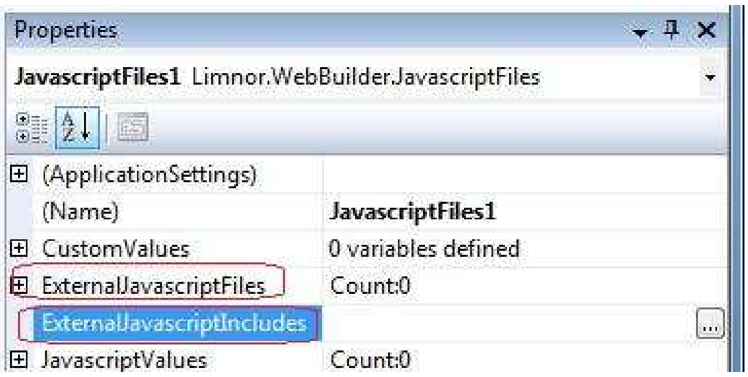
This is a component for including and using Javascript files in the web page.

Drop a JavascriptFiles to the web page:



Include JavaScript Library to Web Page

JavaScript libraries usually are *.js files. Two properties of JavascriptFiles component can be used to include *.js files in web pages:

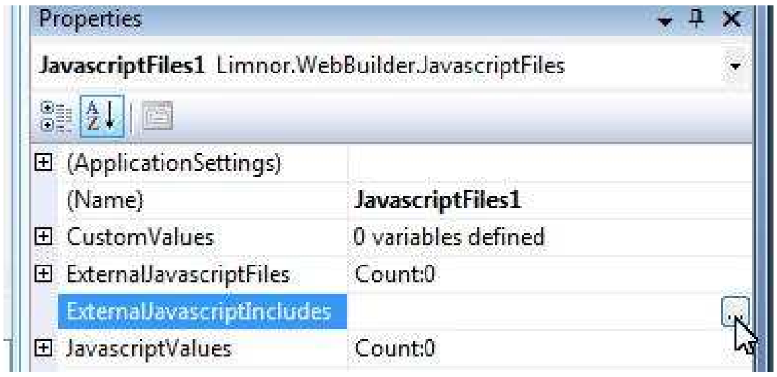


JavascriptFiles1 Limnor.WebBuilder.JavascriptFiles

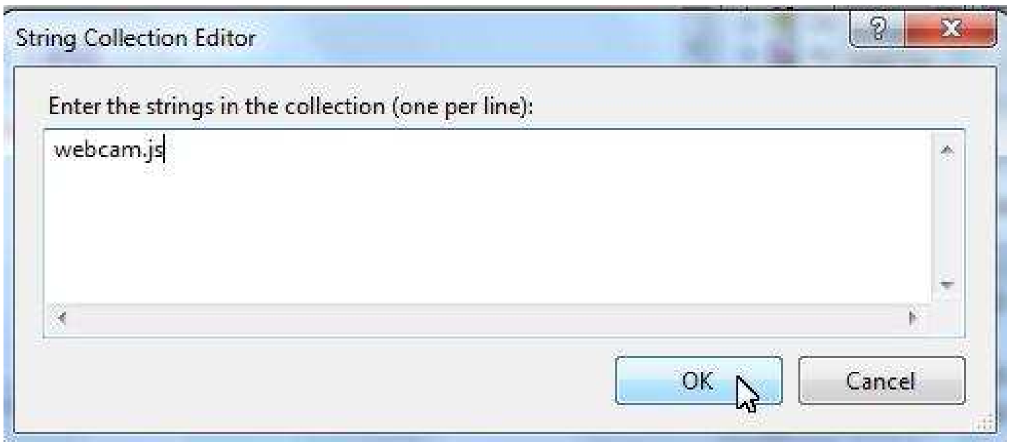
(ApplicationSettings)	
(Name)	JavascriptFiles1
CustomValues	0 variables defined
ExternalJavascriptFiles	Count:0
ExternalJavascriptIncludes	
JavascriptValues	Count:0

- **ExternalJavascriptFiles** – The files added to this property will be copied to libJs folder under WebFiles folder during compiling. Therefore the *.js files must exist in your hard disk. Any support files needed by the *.js files also need to be copied to folders relative to folder libJs.
- **ExternalJavascriptIncludes** – The files added to this property will be included in the web page as it is. No file copying is performed during compiling. It can be used to include files from other web servers. For example, it can be used to include Google Maps library.

Here we use ExternalJavascriptIncludes to include jpegcam file, webcam.js. It is not because webcam.js is in another web server. It is because we unzipped webcam.js and supporting files to WebFiles folder and we do not want the compiler to copy webcam.js to libJs folder.

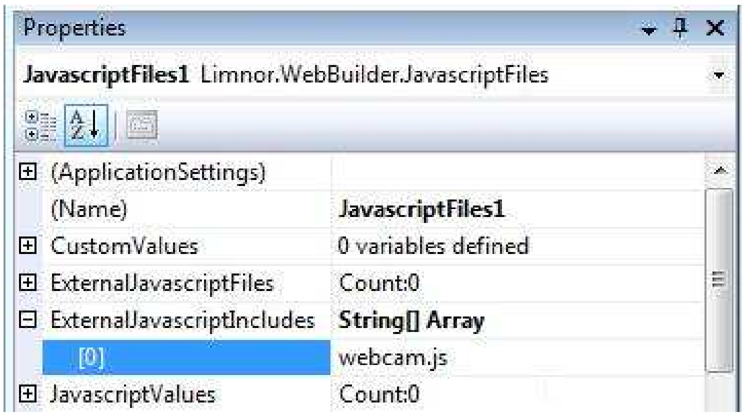


(ApplicationSettings)	
(Name)	JavascriptFiles1
CustomValues	0 variables defined
ExternalJavascriptFiles	Count:0
ExternalJavascriptIncludes	
JavascriptValues	Count:0



```
webcam.js
```

The js file is added to the property:



Properties	
JavascriptFiles1 Limnor.WebBuilder.JavascriptFiles	
(ApplicationSettings)	
(Name)	JavascriptFiles1
CustomValues	0 variables defined
ExternalJavascriptFiles	Count:0
ExternalJavascriptIncludes	String[] Array
[0]	webcam.js
JavascriptValues	Count:0

Use of jpegcam JavaScript Library

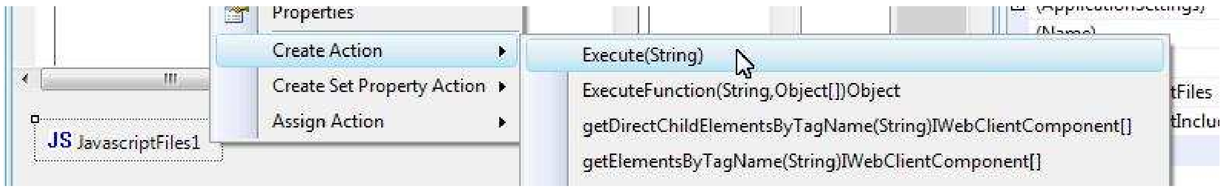
Sample web pages are the best way of learning the use of a JavaScript library. jpegcam contains some samples. We take its test.html to do this sample.

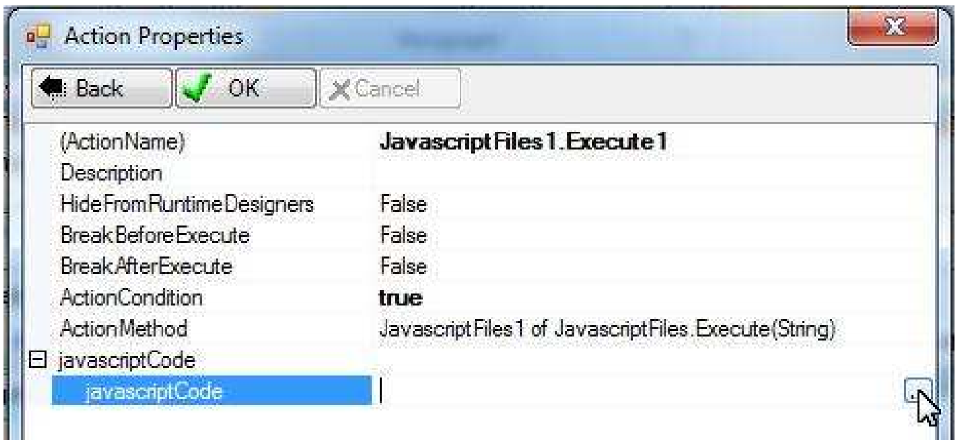
Setup webcam

In test.html, it uses following code to setup webcam:

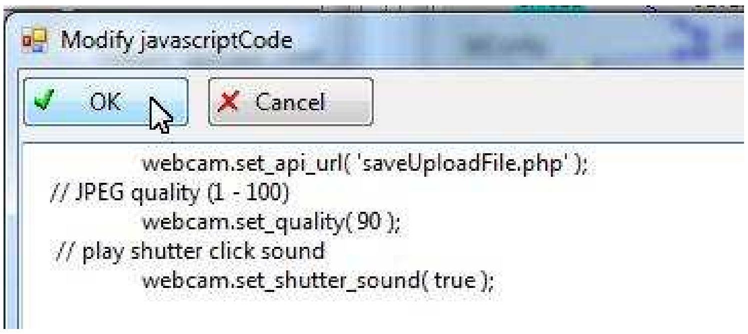
```
      <!-- Configure a few settings -->
      <script language="JavaScript">
            webcam.set_api_url( 'test.php' );
            webcam.set_quality( 90 ); // JPEG quality (1 - 100)
            webcam.set_shutter_sound( true ); // play shutter click sound
      </script>
```

We may create an action to execute the above code:


Properties
Create Action
Create Set Property Action
Assign Action
JS JavascriptFiles1
Execute(String)
ExecuteFunction(String,Object[])Object
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]



(ActionName)	JavascriptFiles1.Execute1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)
javascriptCode	
javascriptCode	



```
            webcam.set_api_url( 'saveUploadFile.php' );
 // JPEG quality (1 - 100)
            webcam.set_quality( 90 );
  // play shutter click sound
            webcam.set_shutter_sound( true );
```

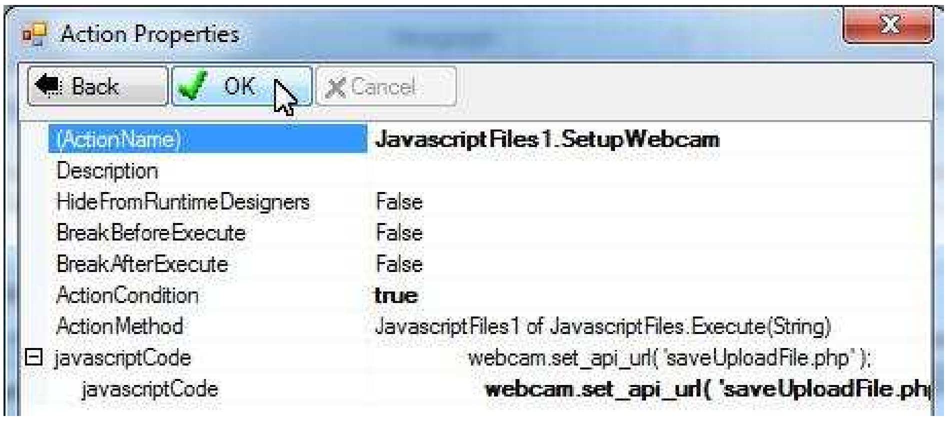
The code we use is as following:

```
                  webcam.set_api_url( 'saveUploadFile.php' );
 // JPEG quality (1 - 100)
         webcam.set_quality( 90 );
  // play shutter click sound
         webcam.set_shutter_sound( true );
```

Note that we changed test.php to saveUploadFile.php.

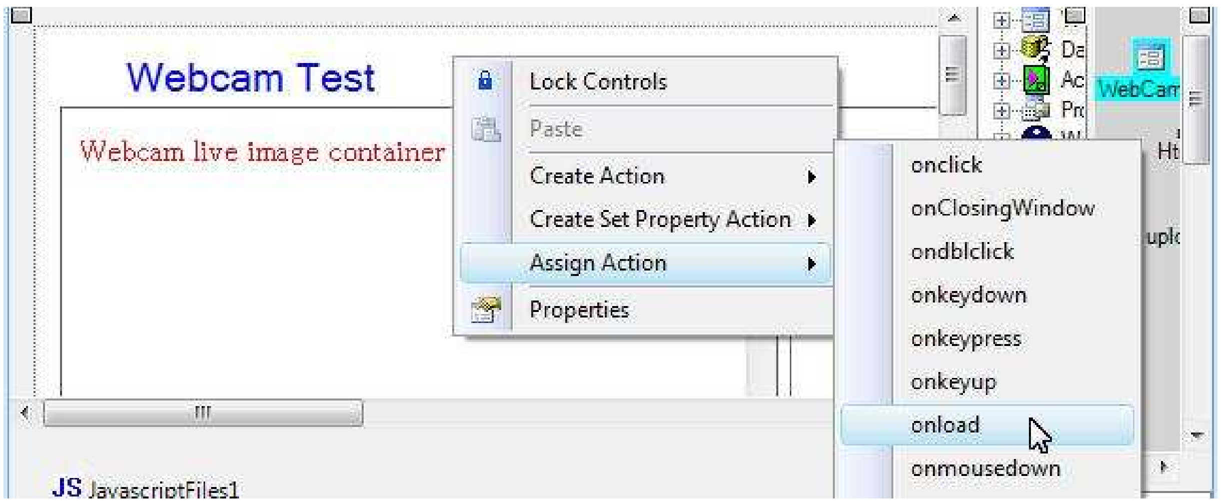
Be careful not put comments at the end of the code. It might damage the code following it.

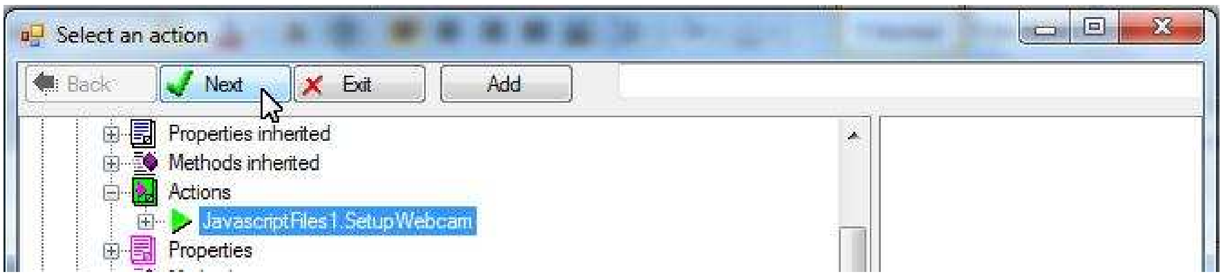
Rename the action to SetupWebcam:



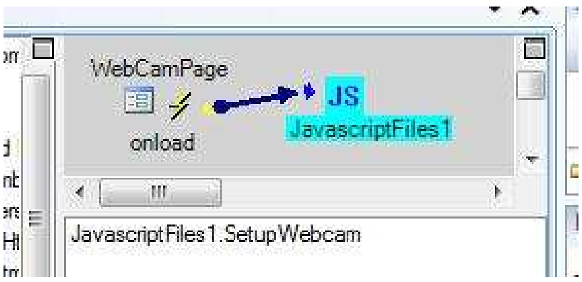
(ActionName)	JavascriptFiles1.SetupWebcam
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)
⊟ javascriptCode	webcam.set_api_url('saveUploadFile.php');
javascriptCode	webcam.set_api_url('saveUploadFile.ph

We may assign the action to onload event of the web page:



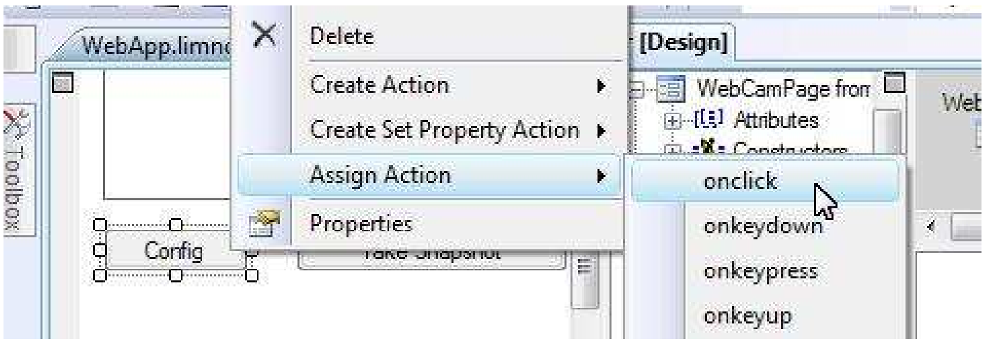


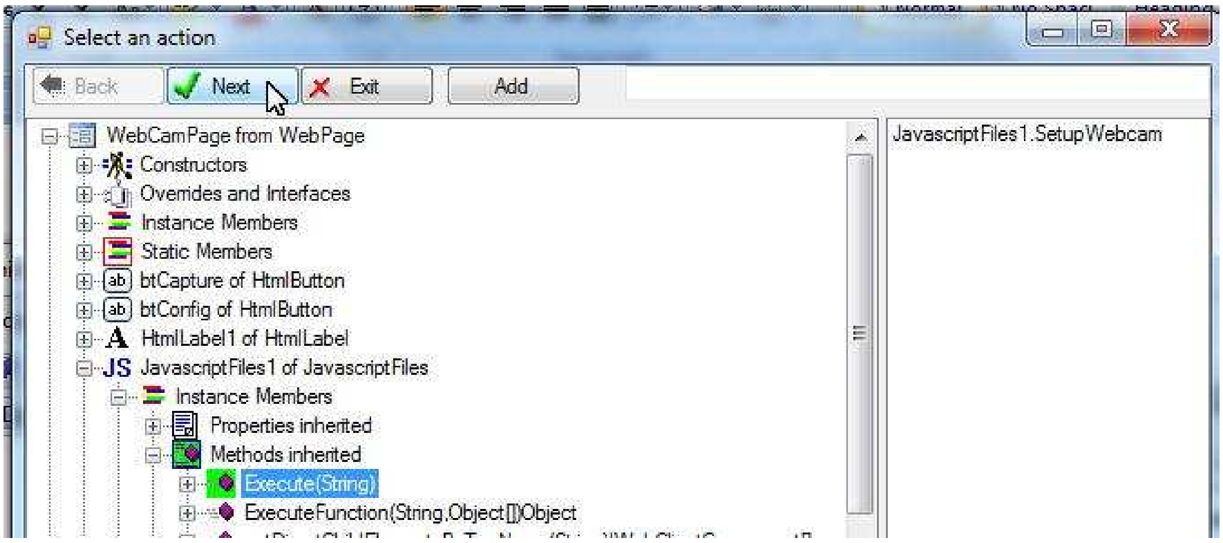
The action is assigned to onload:



Webcam Configure

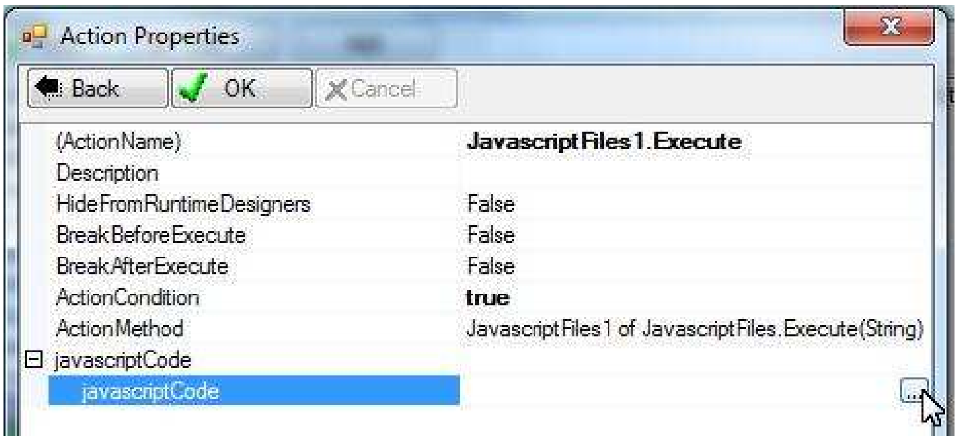
Jpegcam uses code `webcam.configure()` to do webcam configuration. We may execute this code on clicking configure button:



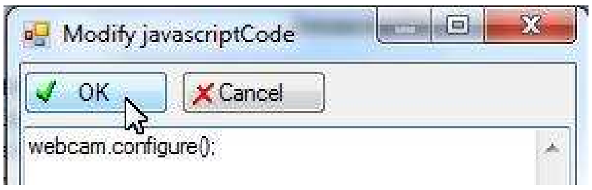


```
WebCamPage from WebPage
  Constructors
  Overrides and Interfaces
  Instance Members
  Static Members
  btCapture of HtmlButton
  btConfig of HtmlButton
  HtmlLabel1 of HtmlLabel
  JavascriptFiles1 of JavascriptFiles
    Instance Members
      Properties inherited
      Methods inherited
        Execute(String)
        ExecuteFunction(String,Object[])Object
```

JavascriptFiles1.SetupWebcam

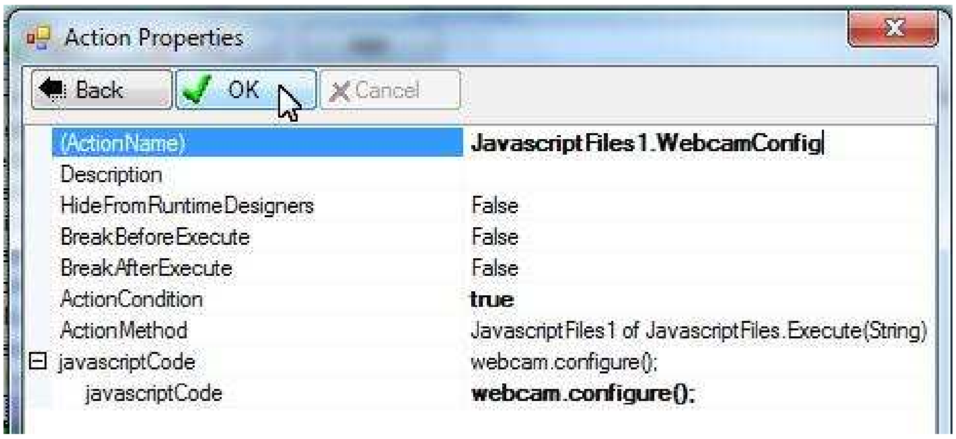


(ActionName)	**JavascriptFiles1.Execute**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)
javascriptCode	
javascriptCode	



```
webcam.configure();
```

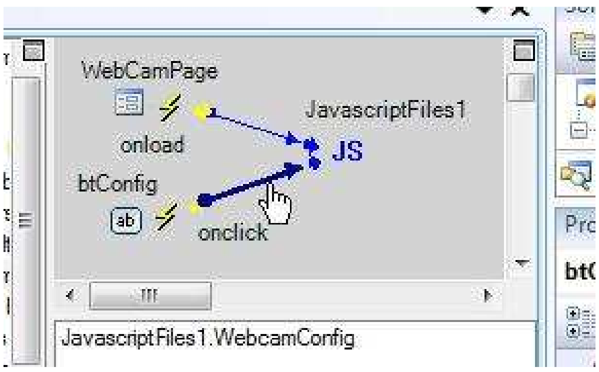
Rename the action to WebcamConfig:



Action Properties

(ActionName)	JavascriptFiles1.WebcamConfig
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)
javascriptCode	webcam.configure();
javascriptCode	webcam.configure();

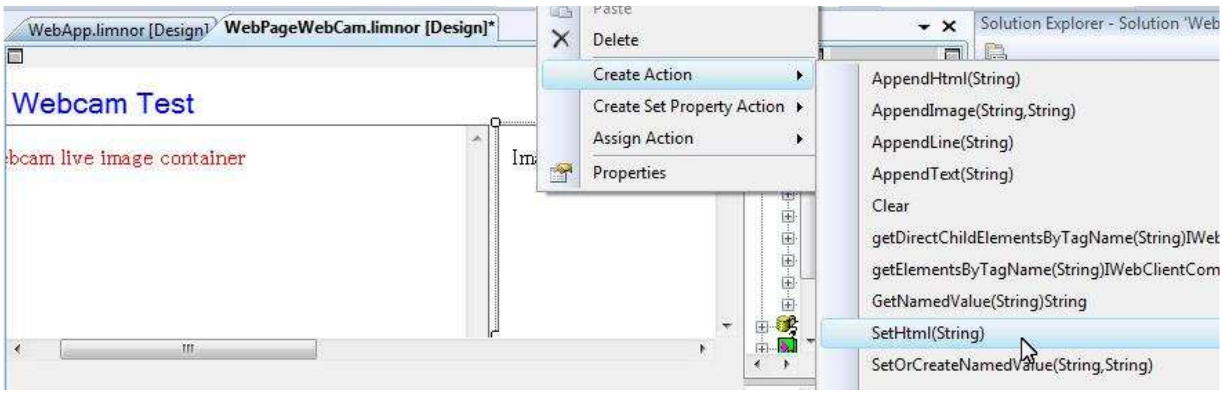
The action is created and assigned to the button:

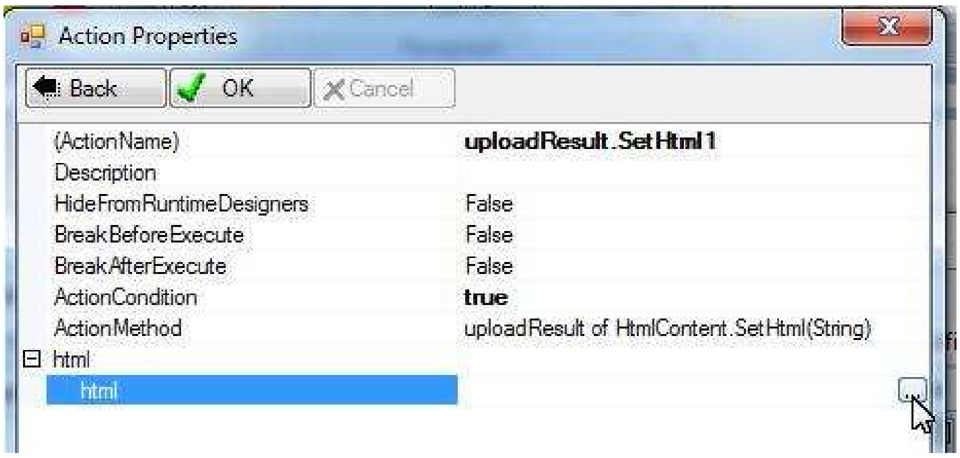


Take webcam snapshot and upload to web server

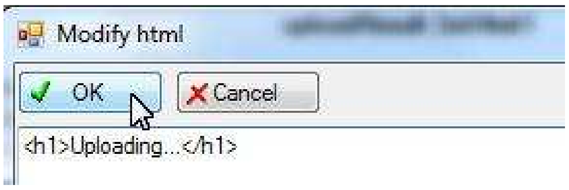
Jpegcam uses code `webcam.snap()` to take snapshot and upload it to web server. We may execute this code on clicking “Take Snapshot” button.

Before taking snapshot, we may show waiting message on the web page:


WebApp.limnor [Design]
WebPageWebCam.limnor [Design]*
Webcam Test
bcam live image container
Paste
Delete
Create Action
Create Set Property Action
Assign Action
Properties
Solution Explorer - Solution 'Web
AppendHtml(String)
AppendImage(String,String)
AppendLine(String)
AppendText(String)
Clear
getDirectChildElementsByTagName(String)IWeb
getElementsByTagName(String)IWebClientCom
GetNamedValue(String)String
SetHtml(String)
SetOrCreateNamedValue(String,String)

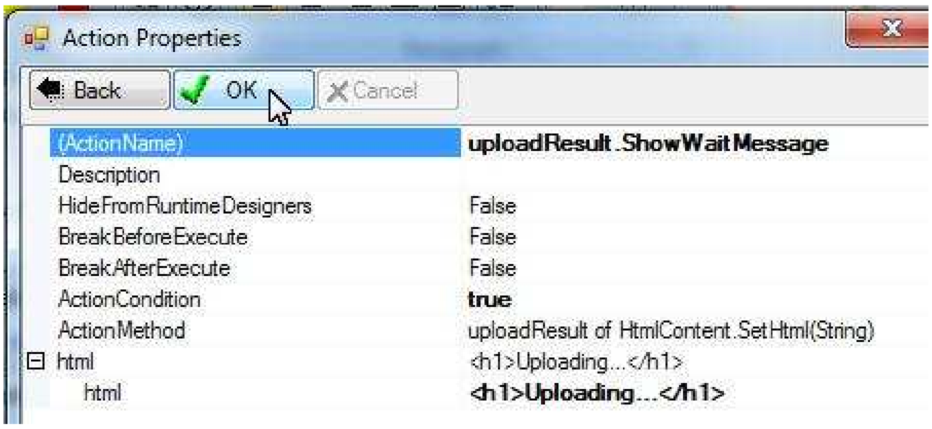


(ActionName)	**uploadResult.SetHtml1**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	uploadResult of HtmlContent.SetHtml(String)
⊟ html	
html	



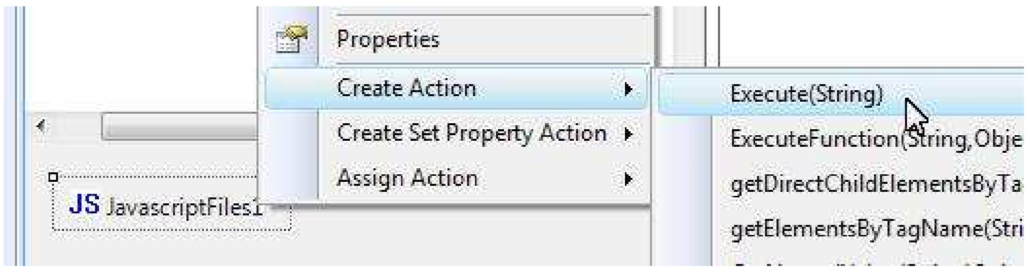
```
<h1>Uploading...</h1>
```

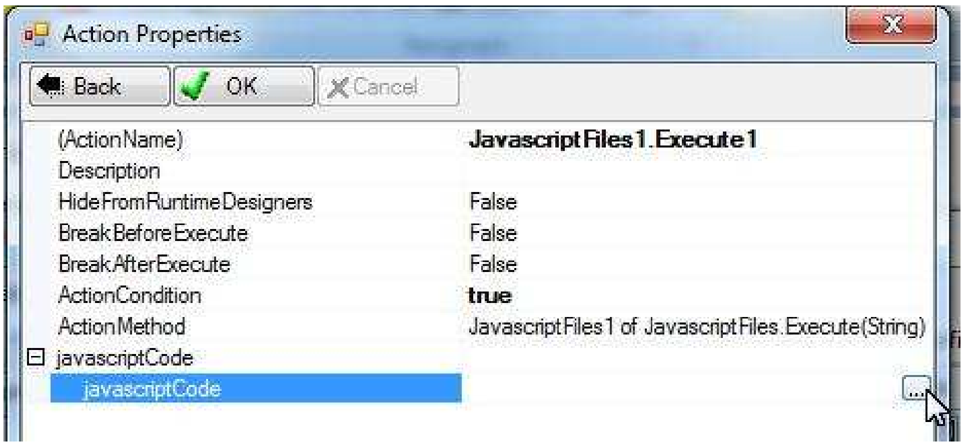
Rename the action and click OK:



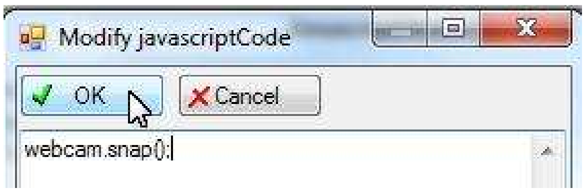
(ActionName)	**uploadResult.ShowWaitMessage**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	uploadResult of HtmlContent.SetHtml(String)
⊟ html	<h1>Uploading...</h1>
html	**<h1>Uploading...</h1>**

Now we may create “snap” action:


Properties
Create Action
Create Set Property Action
Assign Action
Execute(String)
ExecuteFunction(String,Obje
getDirectChildElementsByTa
getElementsByTagName(Stri
JS JavascriptFiles1

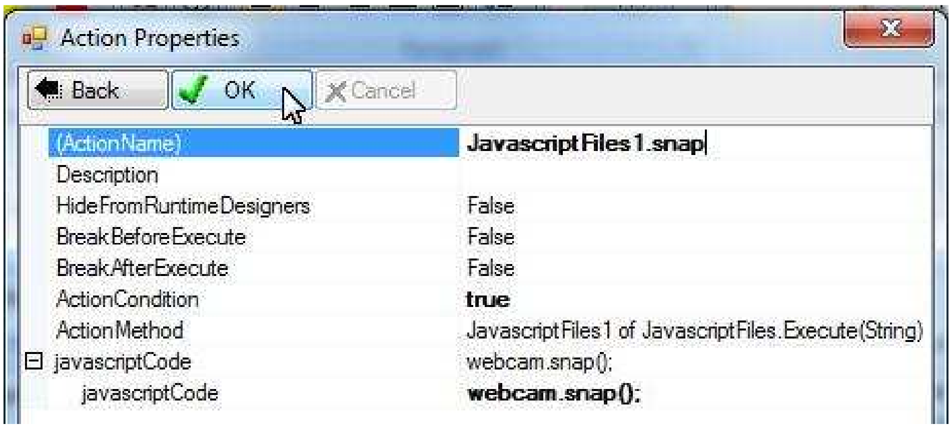


(ActionName)	JavascriptFiles1.Execute1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)
⊟ javascriptCode	
javascriptCode	



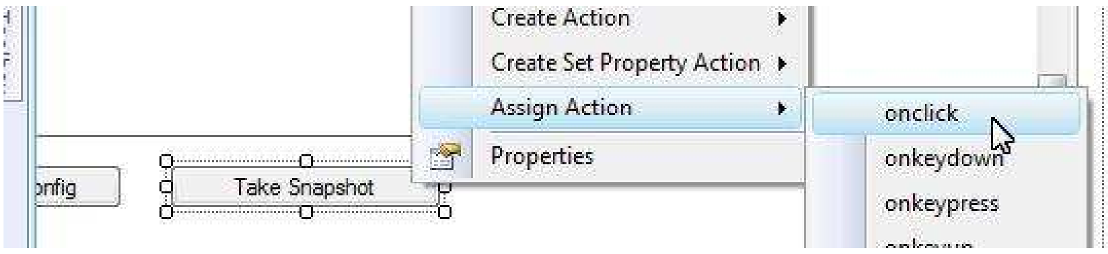
```
webcam.snap();
```

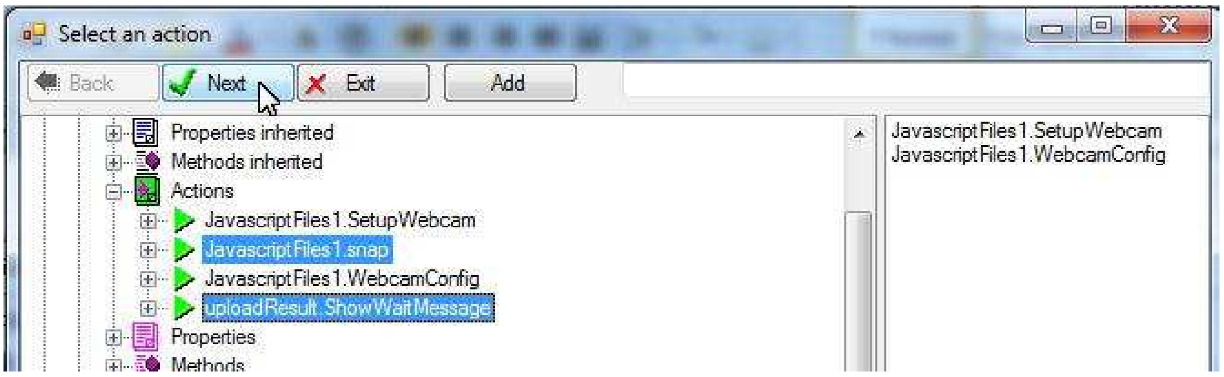
Rename the action and click OK:



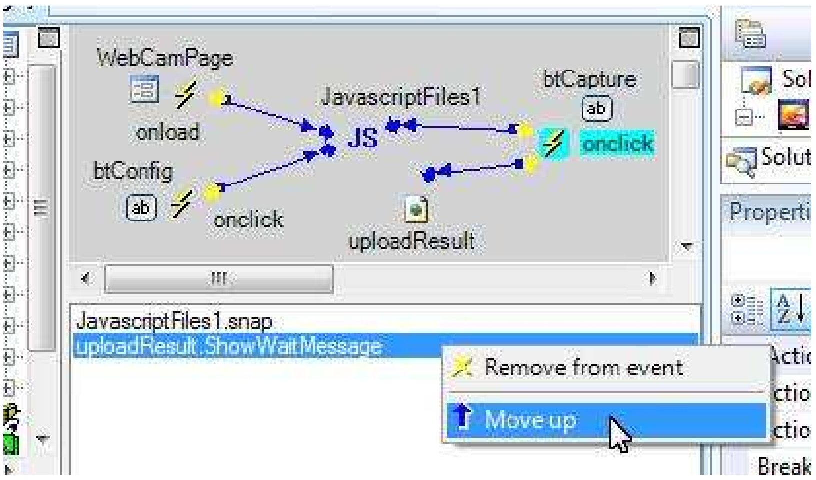
(ActionName)	JavascriptFiles1.snap
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)
⊟ javascriptCode	webcam.snap();
javascriptCode	webcam.snap();

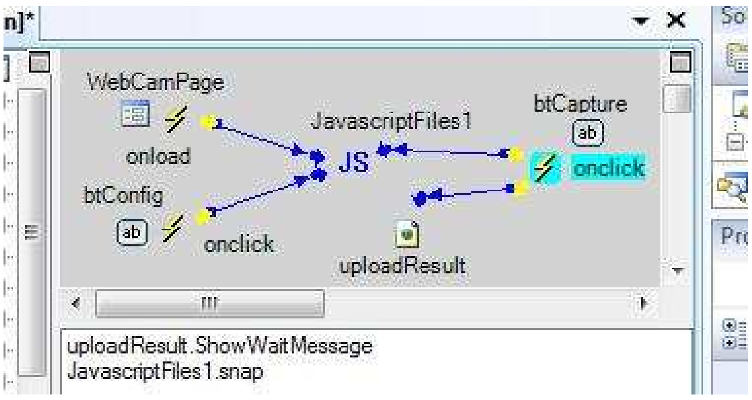
Assign the above two actions to the button:




Select an action
Back
Next
Exit
Add
Properties inherited
Methods inherited
Actions
JavascriptFiles1.SetupWebcam
JavascriptFiles1.snap
JavascriptFiles1.WebcamConfig
uploadResult.ShowWaitMessage
Properties
Methods
JavascriptFiles1.SetupWebcam
JavascriptFiles1.WebcamConfig

The actions are assigned to the button. Move the show message action up:

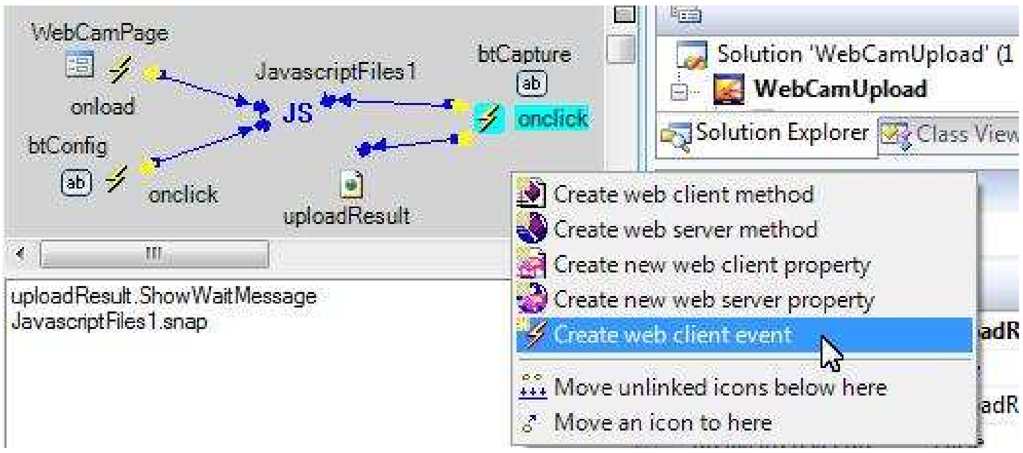




Use uploaded file

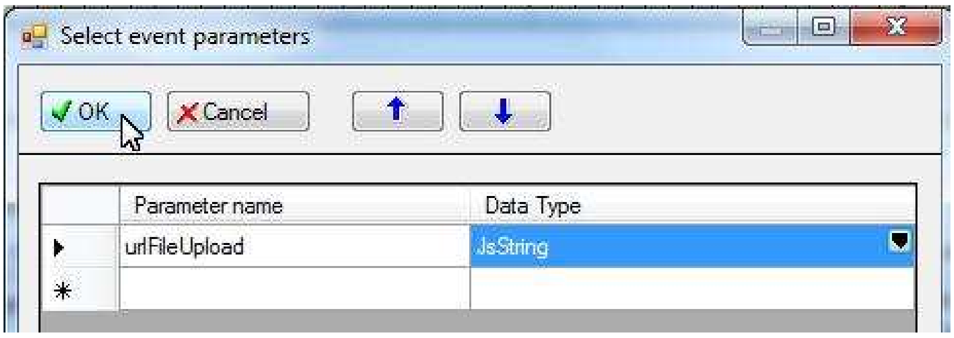
jpegcam uses a client method to handle file upload. The method takes a parameter which is a URL representing the uploaded file. We may create such a client method for it. But if we want to use server side functionality in our web application then we cannot do it directly in this method. The solution is to create a client site event.

Create a client event



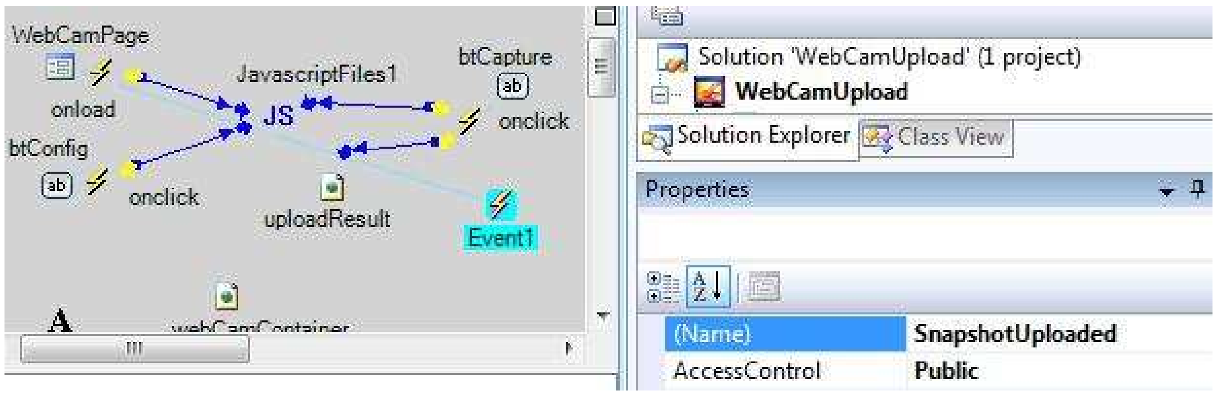
uploadResult.ShowWaitMessage
JavascriptFiles1.snap

Create an event parameter and name it urlFileUpload to represent uploaded file:



	Parameter name	Data Type
▶	urlFileUpload	JsString
*		

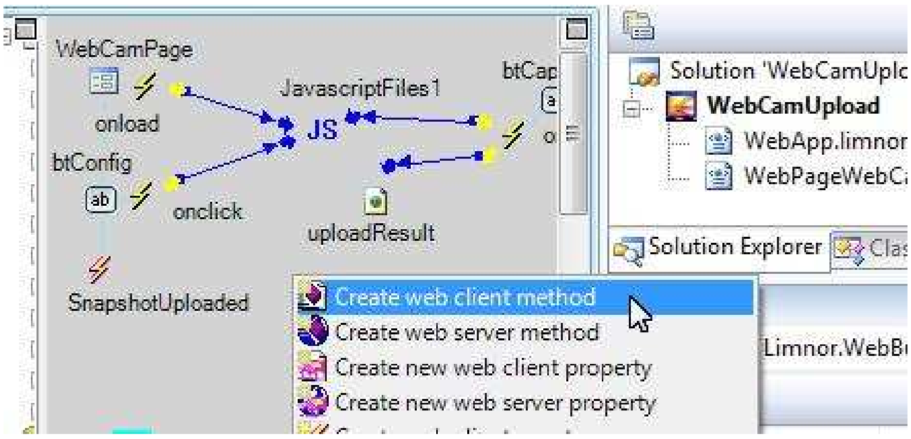
Rename the event from Event1 to SnapshotUploaded:



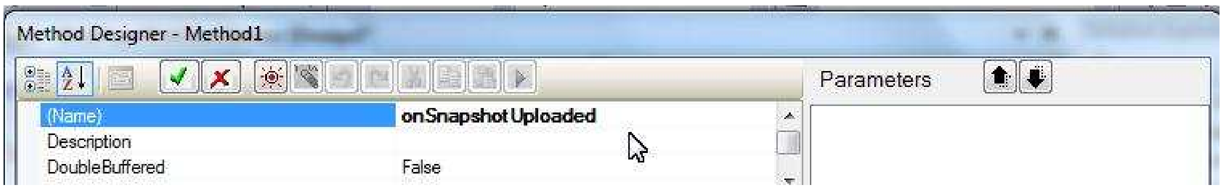
(Name)	SnapshotUploaded
AccessControl	Public

Create upload completion handler method

Now we may create a web client method to handle uploaded file:

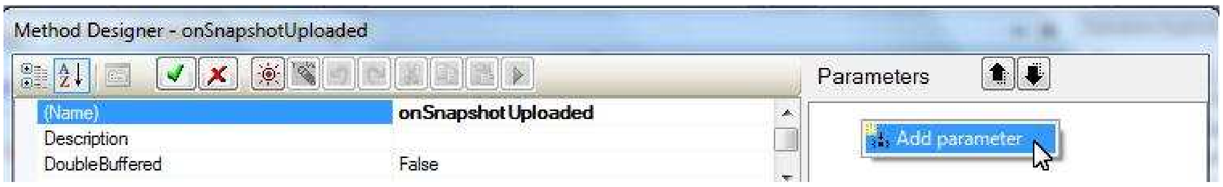


Rename the method name to onSnapshotUploaded:

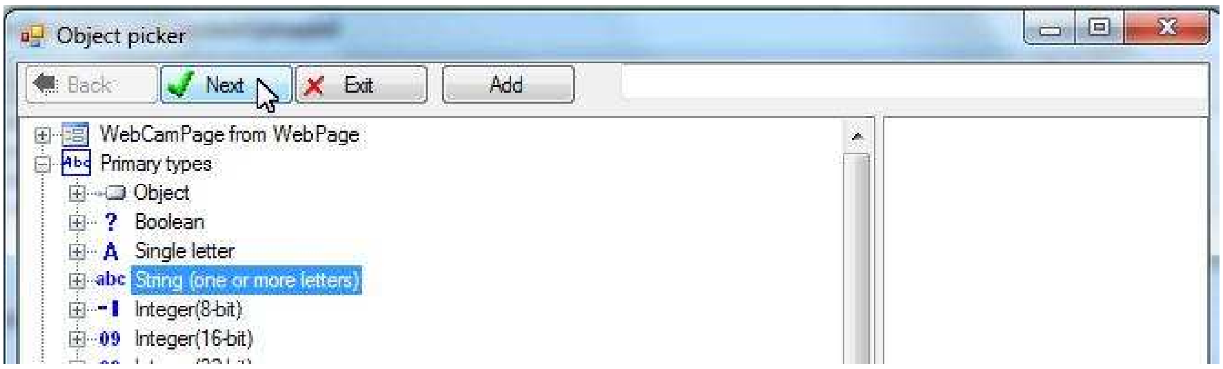


(Name) onSnapshotUploaded
Description
DoubleBuffered False

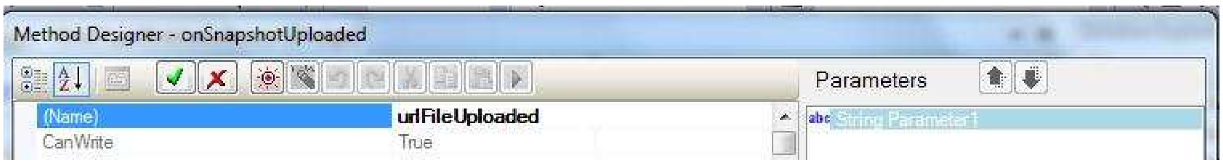
Add a parameter to the method, as required by the jpegcam library:



(Name) onSnapshotUploaded
Description
DoubleBuffered False



Rename the parameter to urlFileUploaded:



(Name) urlFileUploaded
CanWrite True

Handling file upload

In this sample, we execute following actions in this method:

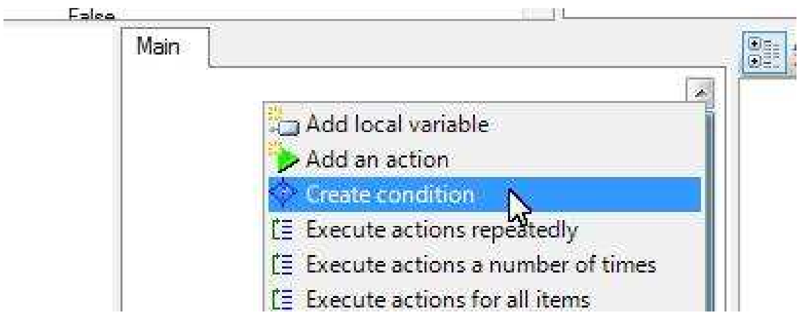
1. Show the URL of uploaded file on the web page or show error message
2. Display image using the URL of uploaded file
3. Fire event SnapshotUploaded

We may assign actions to event SnapshotUploaded; for example, insert a new database record to save the URL of uploaded file to a database.

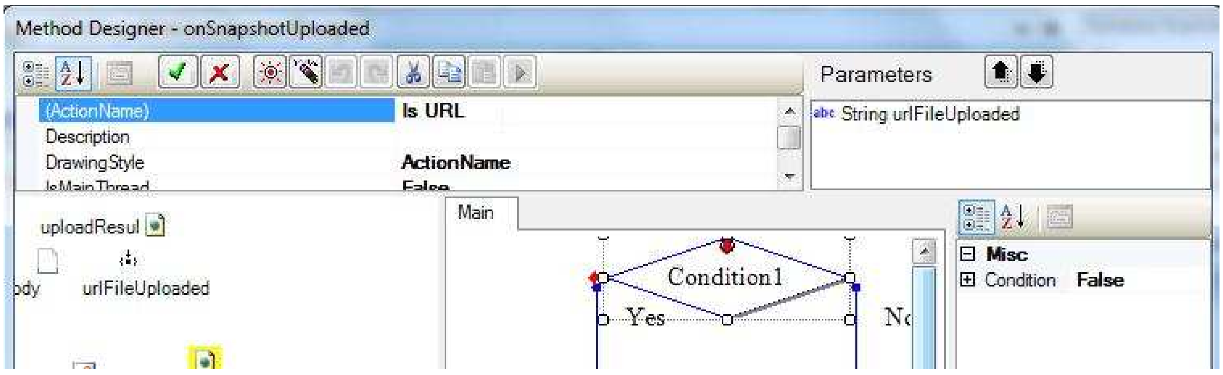
Let's create the above actions.

Check URL

First, we need to check to see if the parameter, urlFileUploaded is really an URL. An URL should start with "http://". In test.html, it uses JavaScript expression, `match(/(http\:\/\/\S+)/)`, to do this checking. We can use JavascripFiles.Execute action execute this expression:

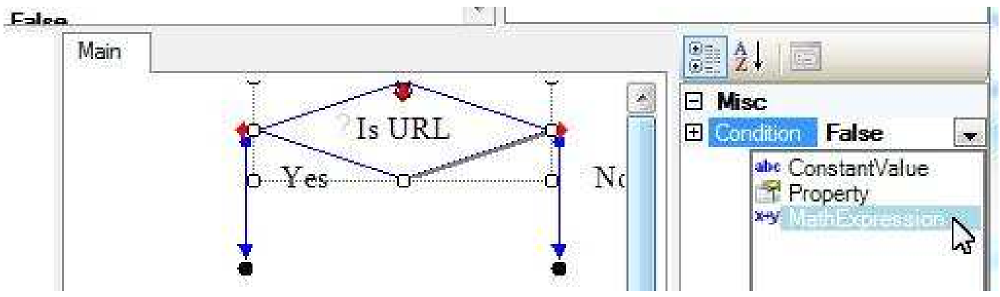

Main
Add local variable
Add an action
Create condition
Execute actions repeatedly
Execute actions a number of times
Execute actions for all items

Rename the condition:

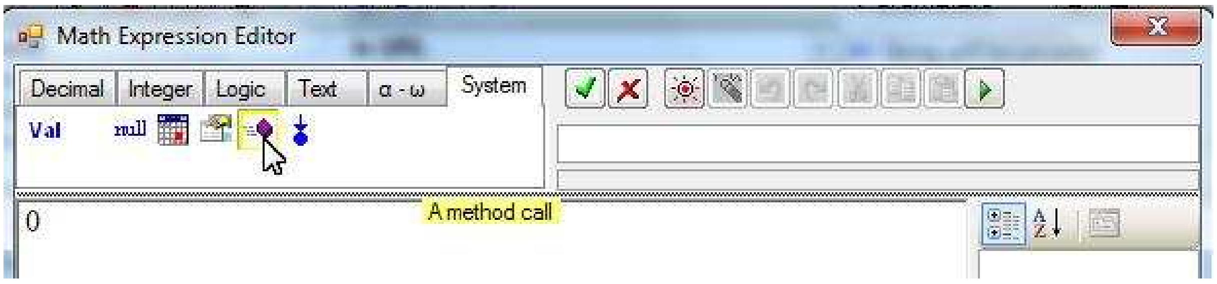


(ActionName)	Is URL
Description	
DrawingStyle	ActionName
IsMainThread	False
Condition	False

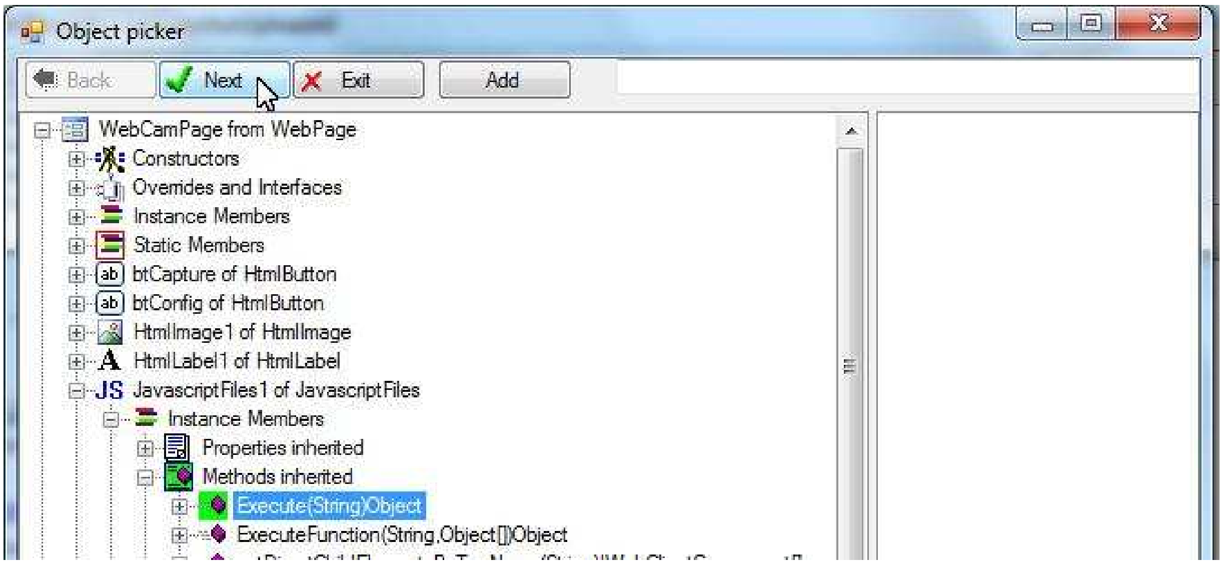
Set the condition to an expression:



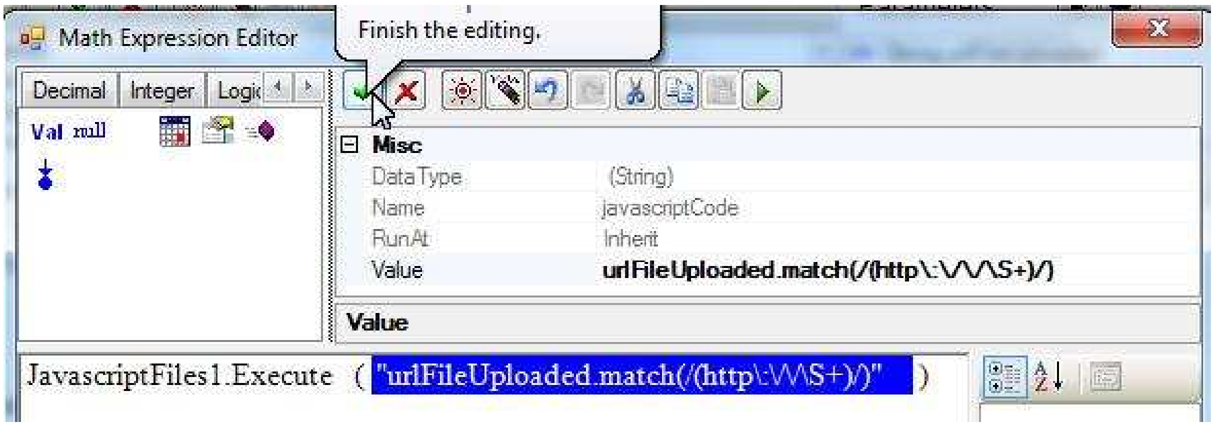
Use a method in the expression:



Select Execute method:



Enter urlFileUploaded.`match(/(http\:\/\/\S+)/)` as the script to be executed:



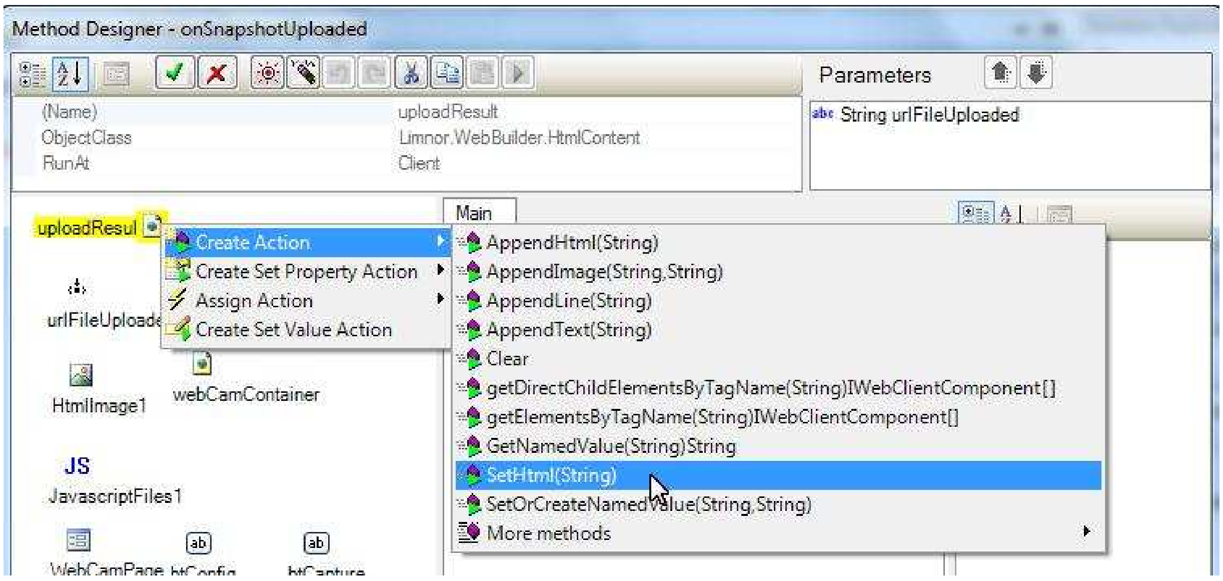
Misc

DataType	(String)
Name	javascriptCode
RunAt	Inherit
Value	urlFileUploaded.match(/(http\:\/\/\S+)/)

Value

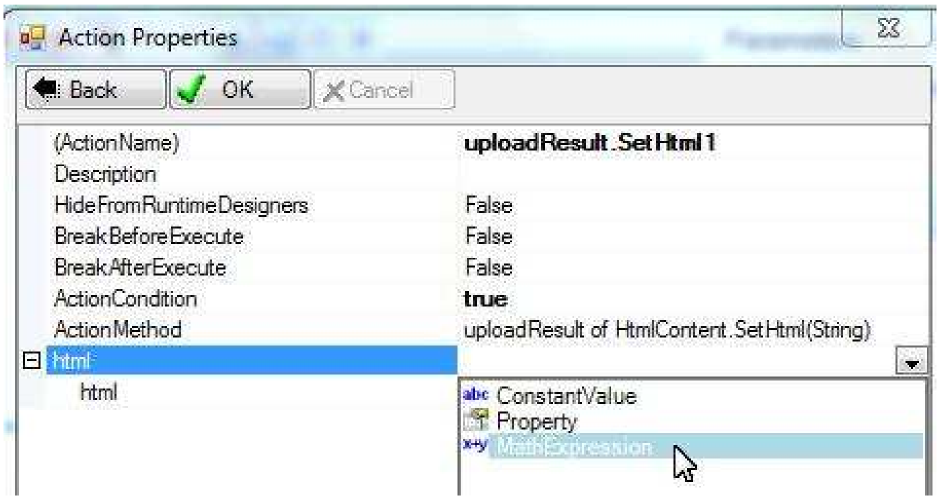
```
JavascriptFiles1.Execute ( "urlFileUploaded.match(/(http\:\/\/\S+)/)" )
```

Create a show "Success" message

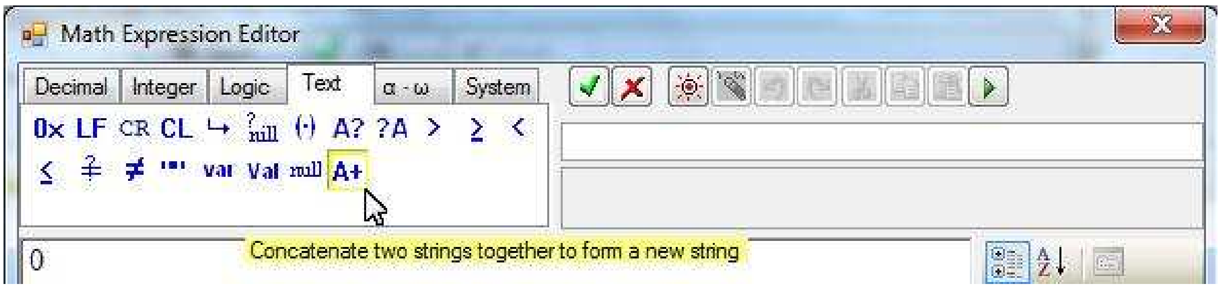

Method Designer - onSnapshotUploaded
Parameters
Main
uploadResul
urlFileUploade
HtmlImage1
webCamContainer
JS
JavascriptFiles1
WebCamPage
btConfig
btCapture
Create Action
Create Set Property Action
Assign Action
Create Set Value Action
AppendHtml(String)
AppendImage(String,String)
AppendLine(String)
AppendText(String)
Clear
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
SetHtml(String)
SetOrCreateNamedValue(String,String)
More methods

(Name)	uploadResult
ObjectClass	Limnor.WebBuilder.HtmlContent
RunAt	Client

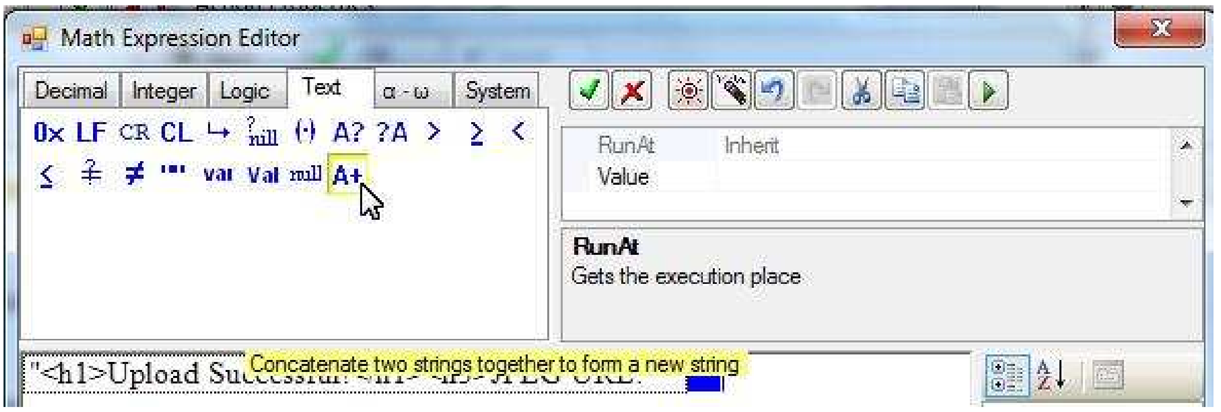
String urlFileUploaded



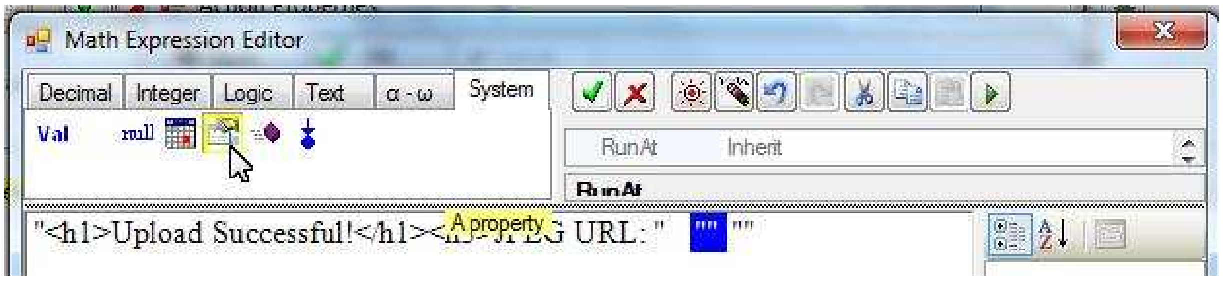
(ActionName)	uploadResult.SetHtml1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	uploadResult of HtmlContent.SetHtml(String)
html	
html	



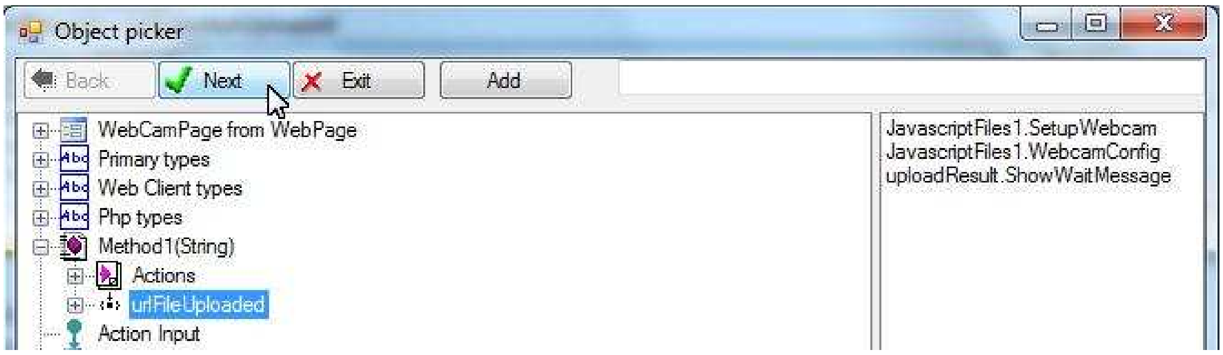
Enter some html message in the first string item .Select the second stringitme and click A+ :



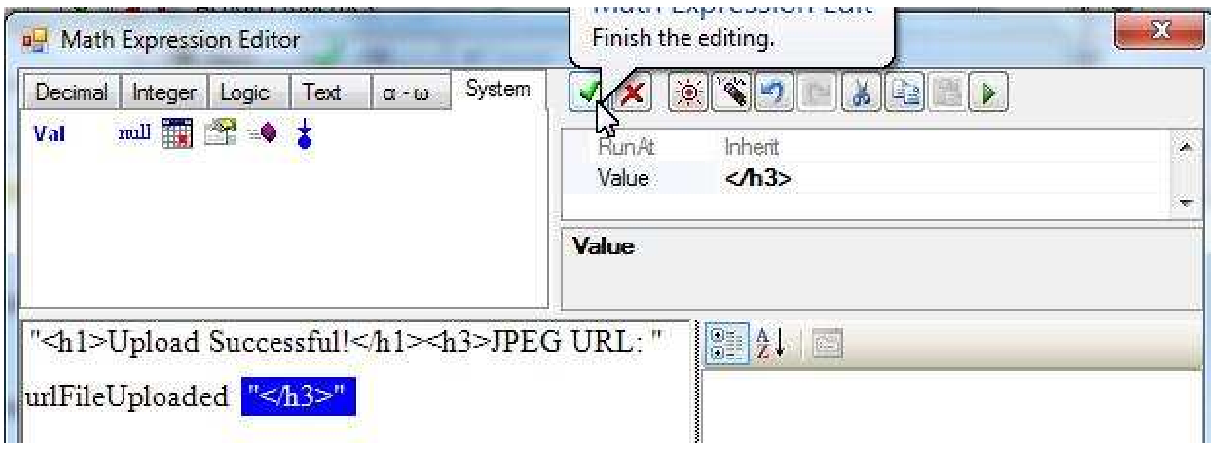
Select the second string item and click Property icon:



Select urlFileUploaded:



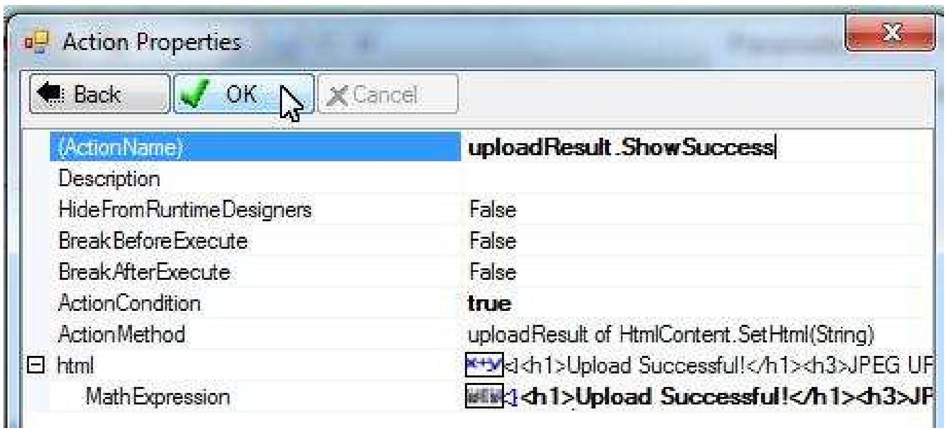
Enter html in the 3rd string item and finish the editing:



RunAt	Inherit
Value	</h3>

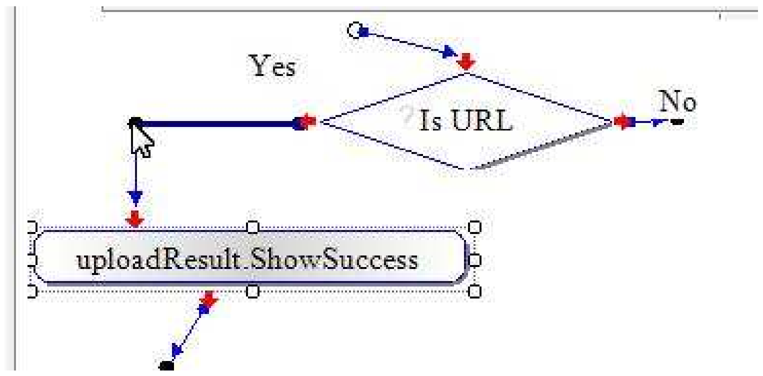
"<h1>Upload Successful!</h1><h3>JPEG URL: "
urlFileUploaded "</h3>"

Rename the action and click OK:

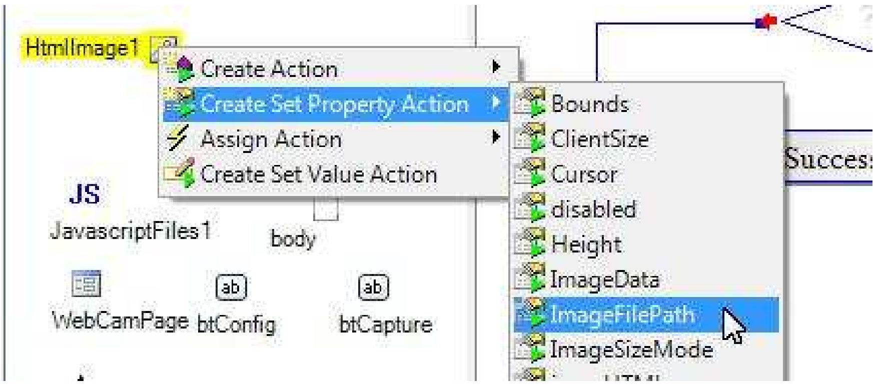


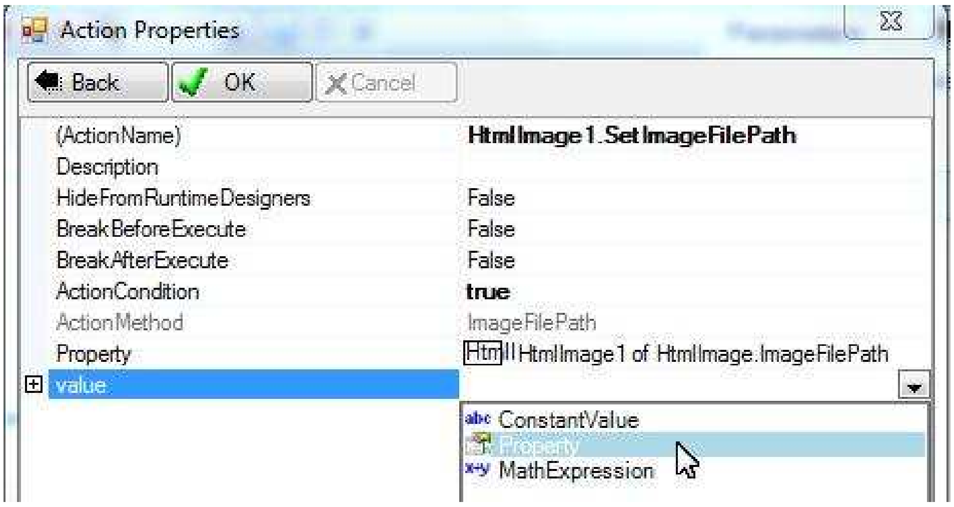
(ActionName)	uploadResult.ShowSuccess
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	uploadResult of HtmlContent.SetHtml(String)
html	<h1>Upload Successful!</h1><h3>JPEG UR
MathExpression	<h1>Upload Successful!</h1><h3>JP

Link the action to Yes port:

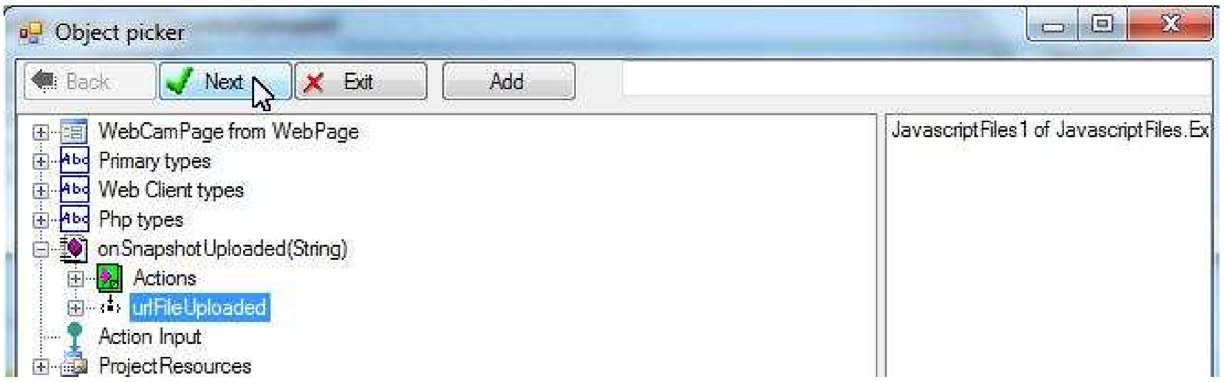


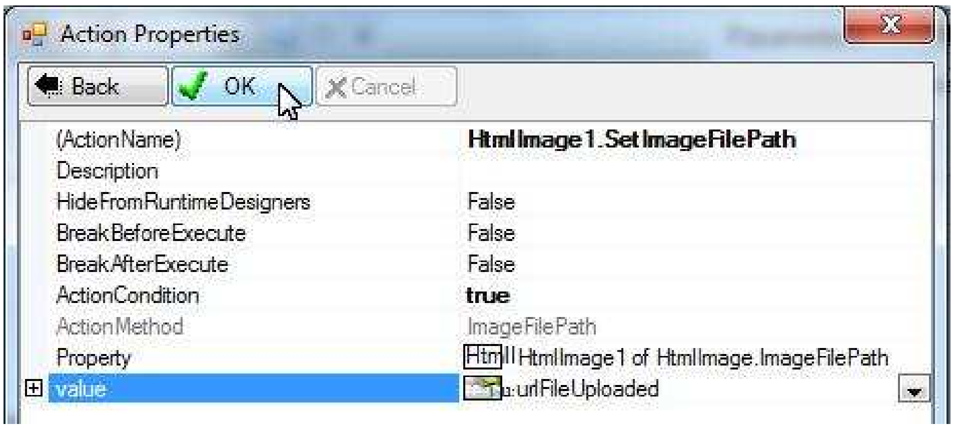
Create an action to show image using the URL as the image source





(ActionName)	HtmlImage1.SetImageFilePath
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ImageFilePath
Property	HtmlImage1 of HtmlImage.ImageFilePath
value	

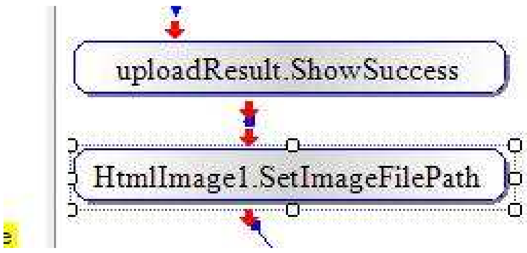




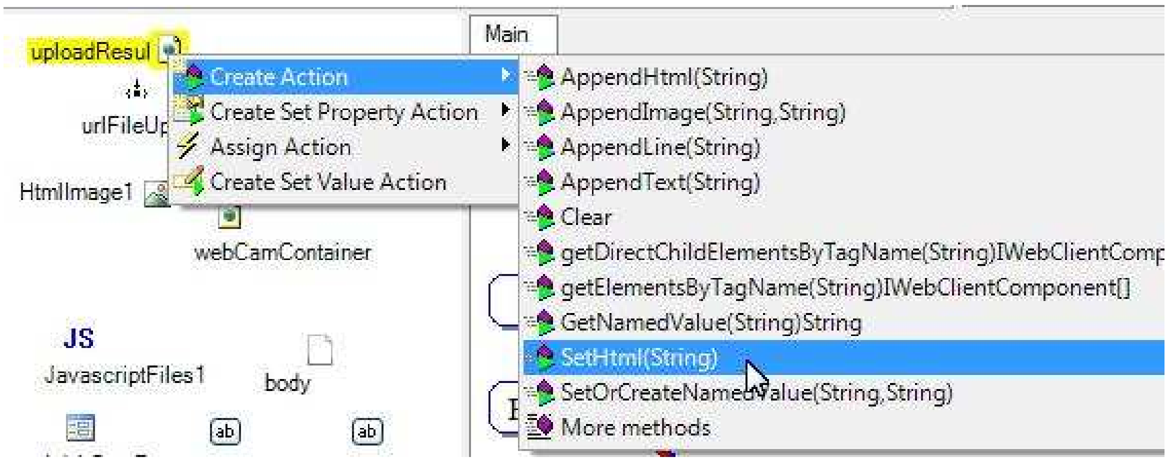
Action Properties

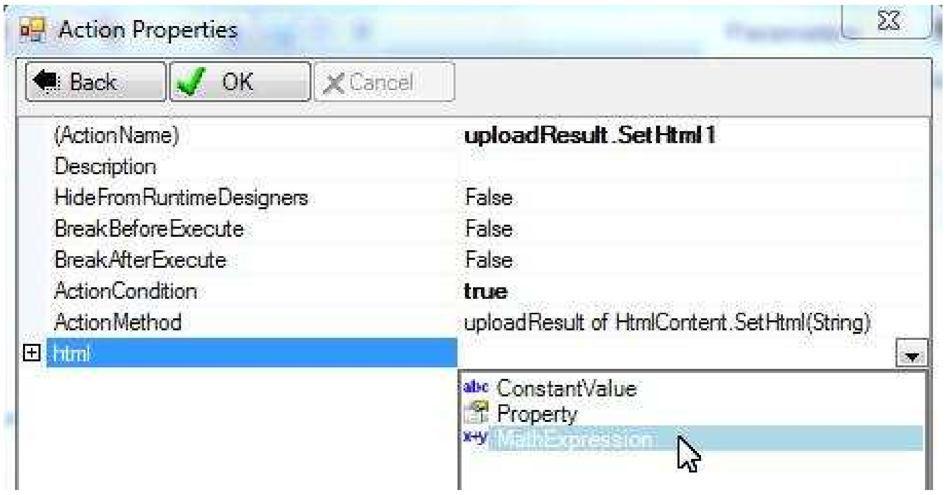
(ActionName)	HtmlImage1.SetImageFilePath
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ImageFilePath
Property	HtmlImage1 of HtmlImage.ImageFilePath
value	urlFileUploaded

Link the action to the last action:

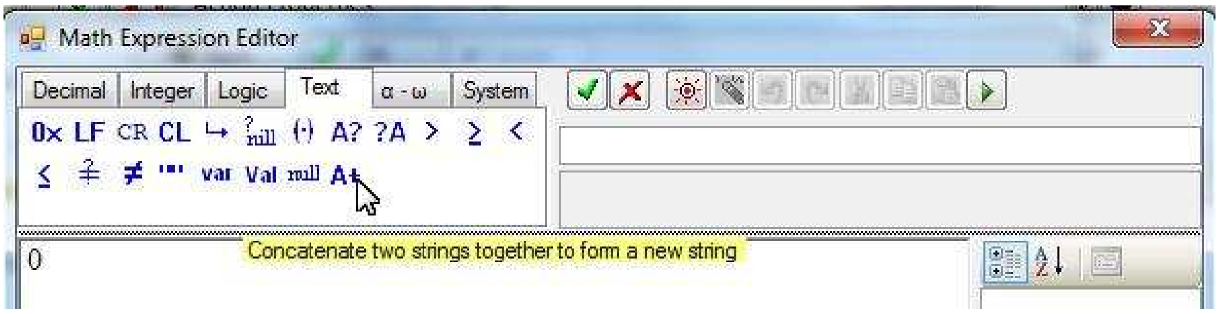


Create an action to show error message


uploadResul
urlFileUp
HtmlImage1
webCamContainer
JavascriptFiles1
body
Main
Create Action
Create Set Property Action
Assign Action
Create Set Value Action
AppendHtml(String)
AppendImage(String,String)
AppendLine(String)
AppendText(String)
Clear
getDirectChildElementsByTagName(String)IWebClientComp
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
SetHtml(String)
SetOrCreateNamedValue(String,String)
More methods

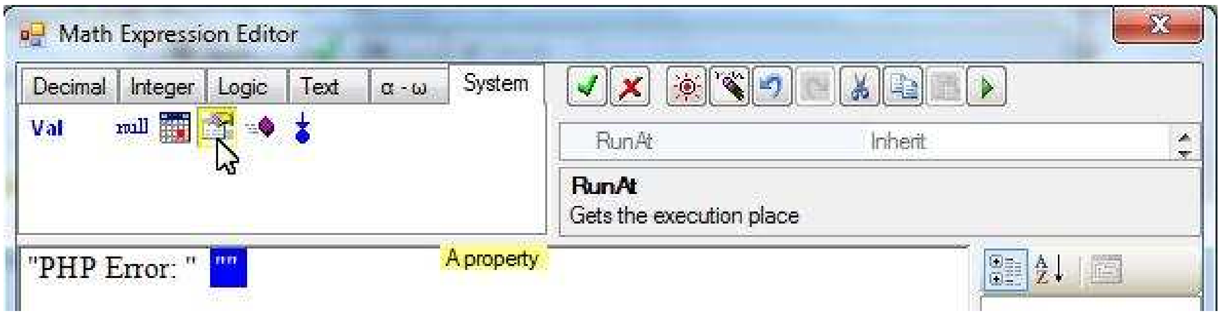


(ActionName)	uploadResult.SetHtml1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	uploadResult of HtmlContent.SetHtml(String)
html	



0

Concatenate two strings together to form a new string

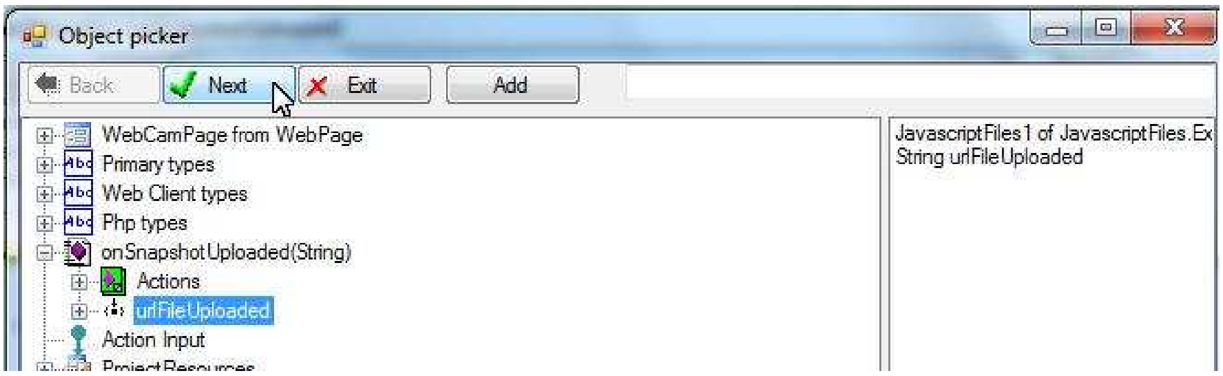


RunAt	Inherit

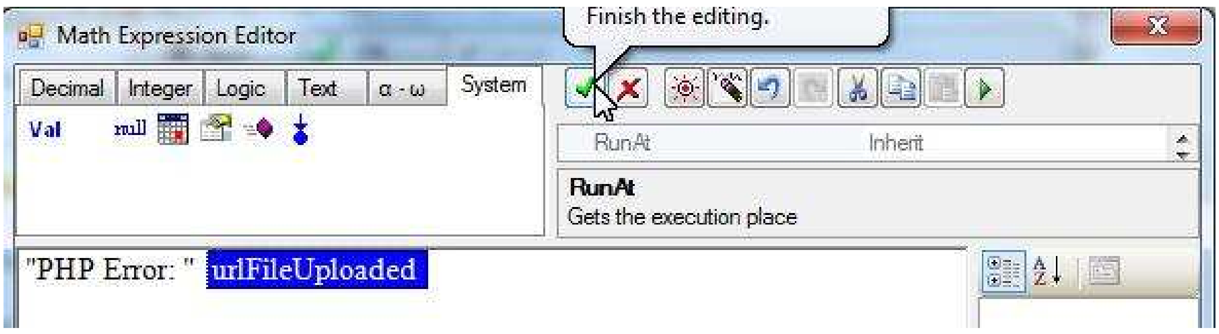
RunAt
Gets the execution place

"PHP Error: " ""

A property



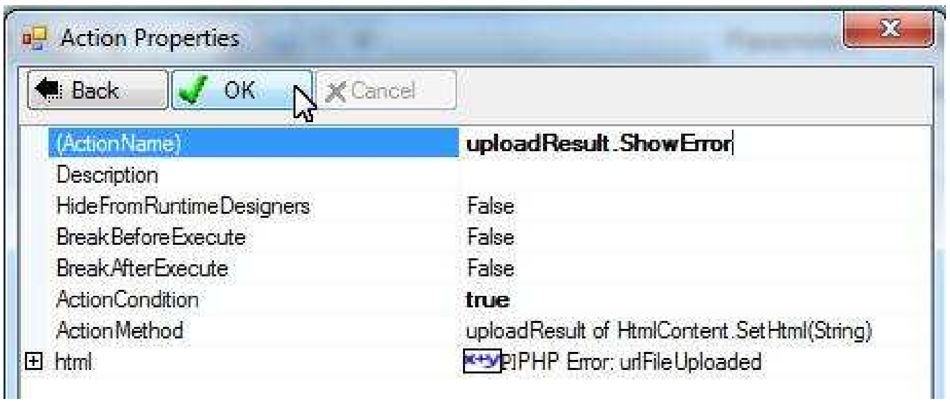
JavascriptFiles1 of JavascriptFiles.Ex
String urlFileUploaded



RunAt
Gets the execution place

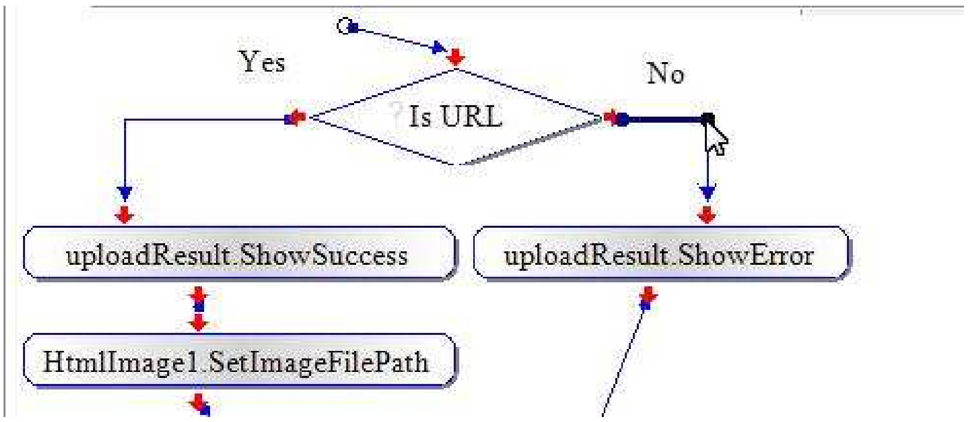
"PHP Error: " urlFileUploaded

Rename the action and click OK:



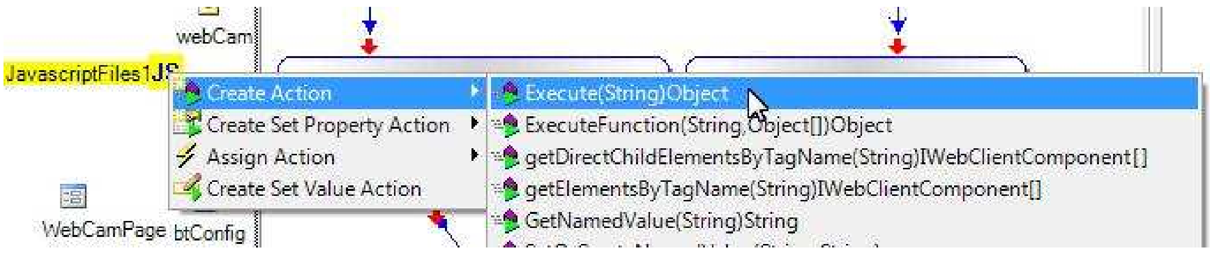
(ActionName)	uploadResult.ShowError
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	uploadResult of HtmlContent.SetHtml(String)
html	PHP Error: urlFileUploaded

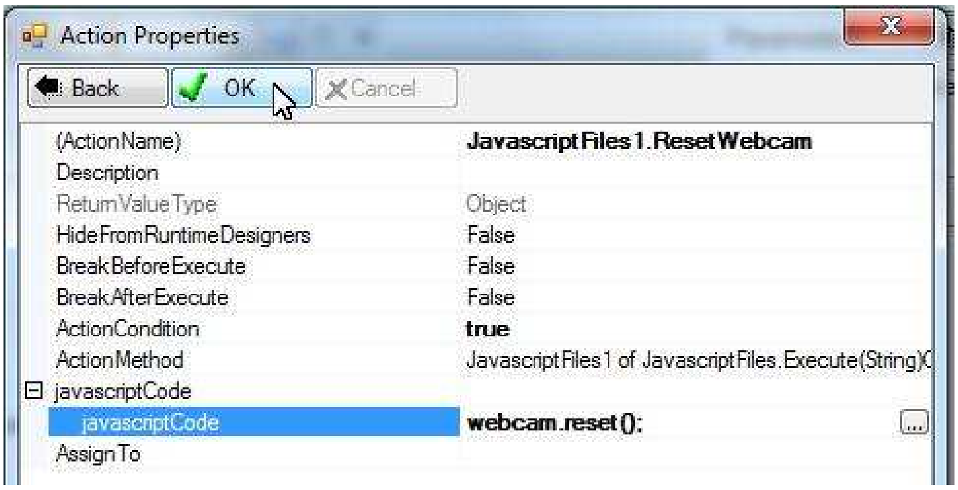
Link the action to No port:



Reset webcam

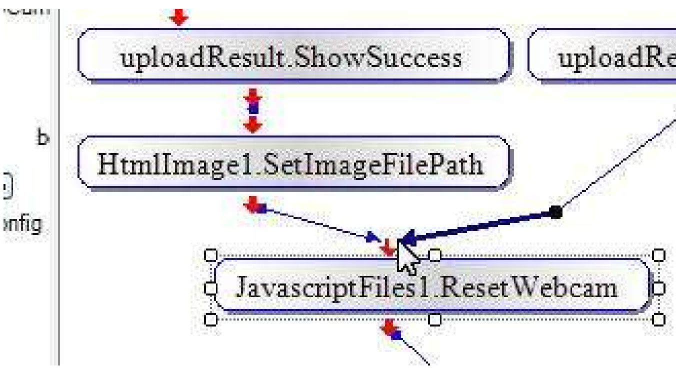
According to jpegcam sample, code `webcam.reset()` should be executed. Create an action to execute it:


webCam
JavascriptFiles1JS
Create Action
Create Set Property Action
Assign Action
Create Set Value Action
Execute(String)Object
ExecuteFunction(String,Object[])Object
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
WebCamPage
btConfig



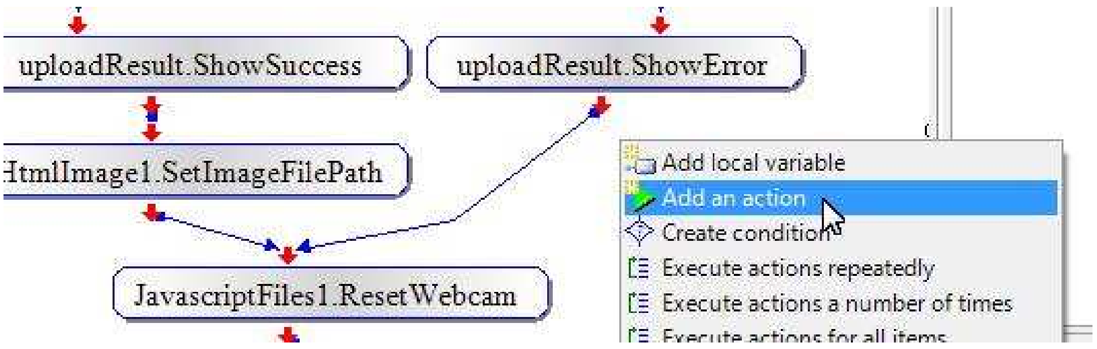
(ActionName)	JavascriptFiles1.ResetWebcam
Description	
ReturnValueType	Object
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)(
javascriptCode	
javascriptCode	webcam.reset();
AssignTo	

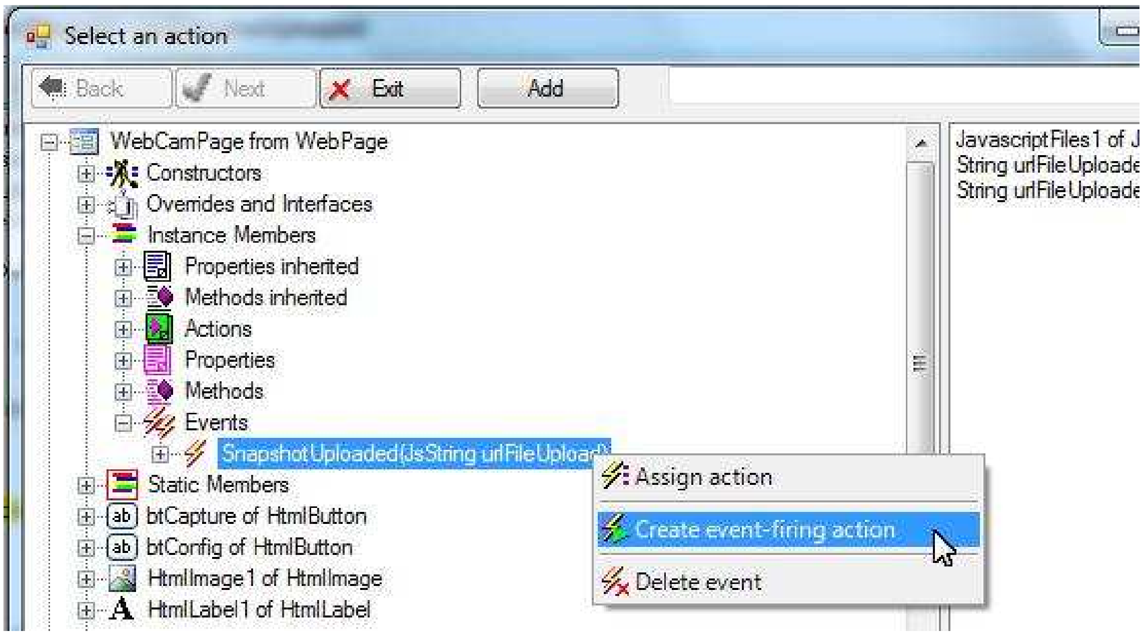
Link both action branches to this action:

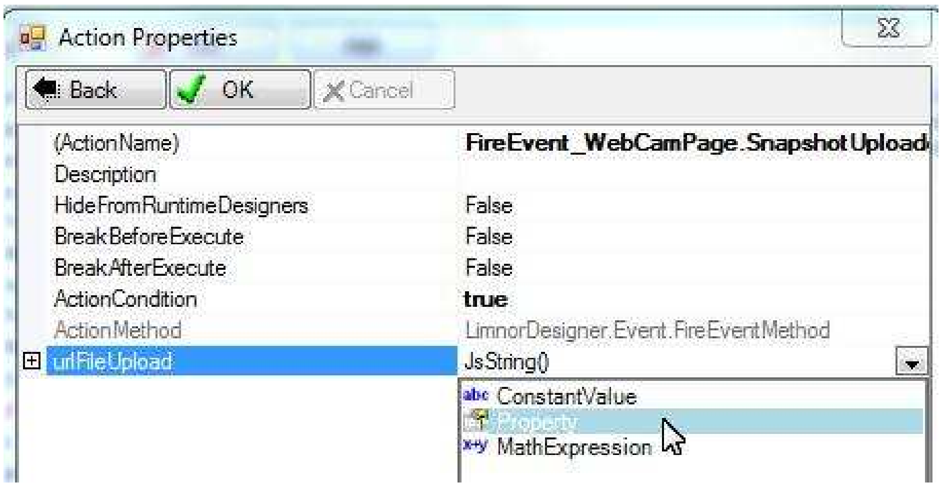


Fire event SnapshotUploaded

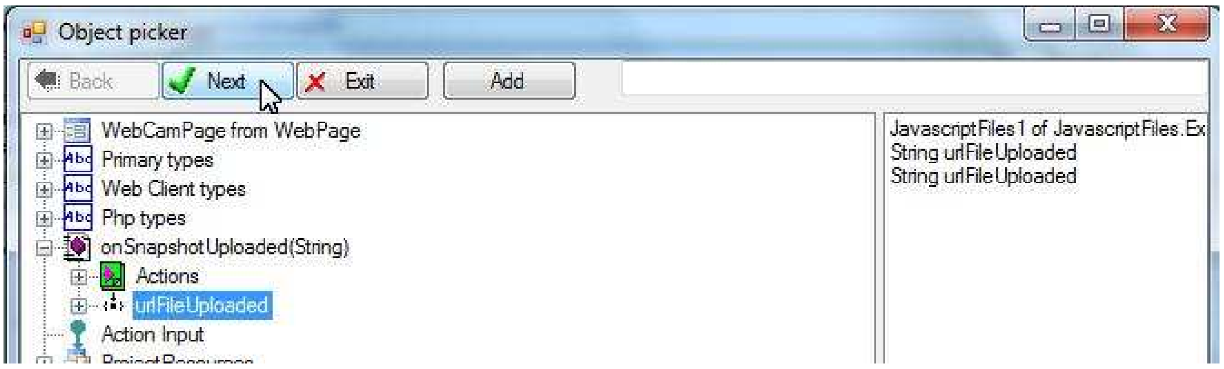
Create an action to fire event SnapshotUploaded so that all actions assigned to the event will be executed:


uploadResult.ShowSuccess
uploadResult.ShowError
HtmlImage1.SetImageFilePath
JavascriptFiles1.ResetWebcam
Add local variable
Add an action
Create condition
Execute actions repeatedly
Execute actions a number of times
Execute actions for all items



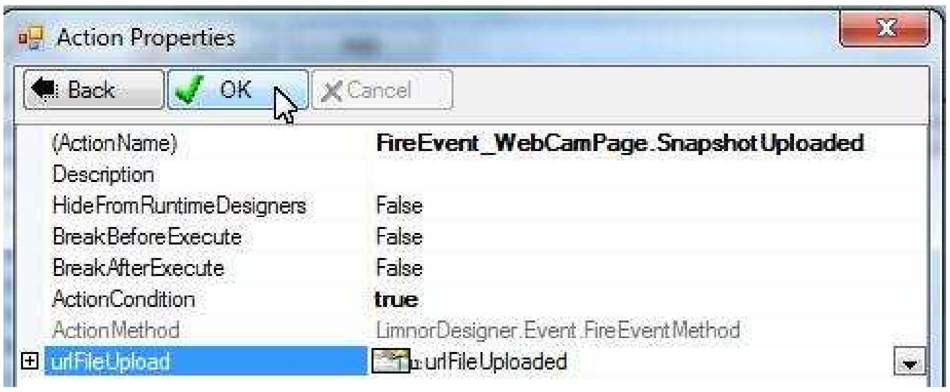


(ActionName)	FireEvent_WebCamPage.SnapshotUpload
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	LimnorDesigner.Event.FireEventMethod
urlFileUpload	JsString()

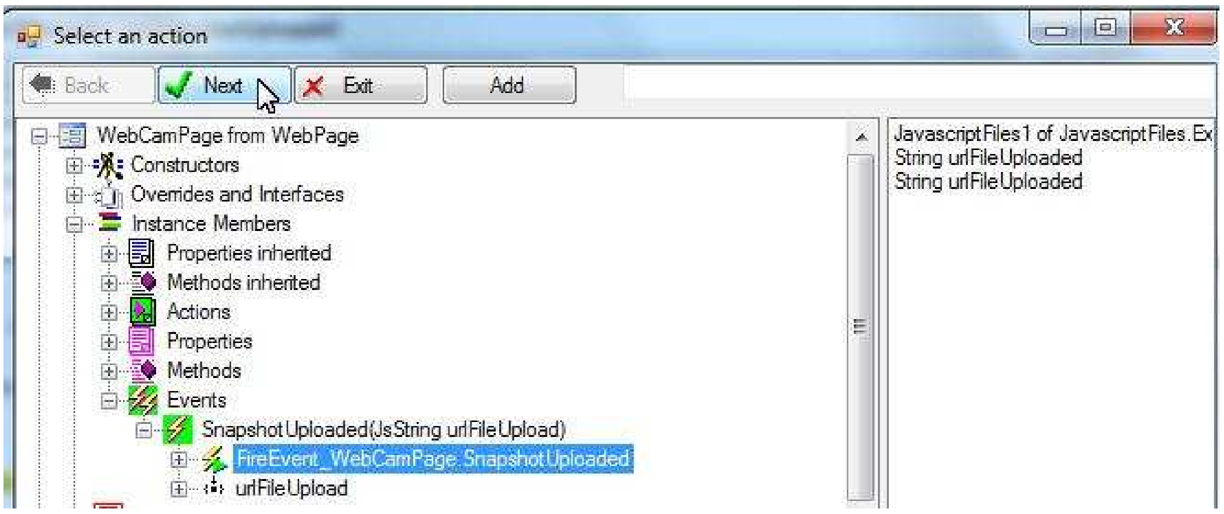


WebCamPage from WebPage
Primary types
Web Client types
Php types
onSnapshotUploaded(String)
Actions
urlFileUploaded
Action Input

JavascriptFiles1 of JavascriptFiles.Ex
String urlFileUploaded
String urlFileUploaded



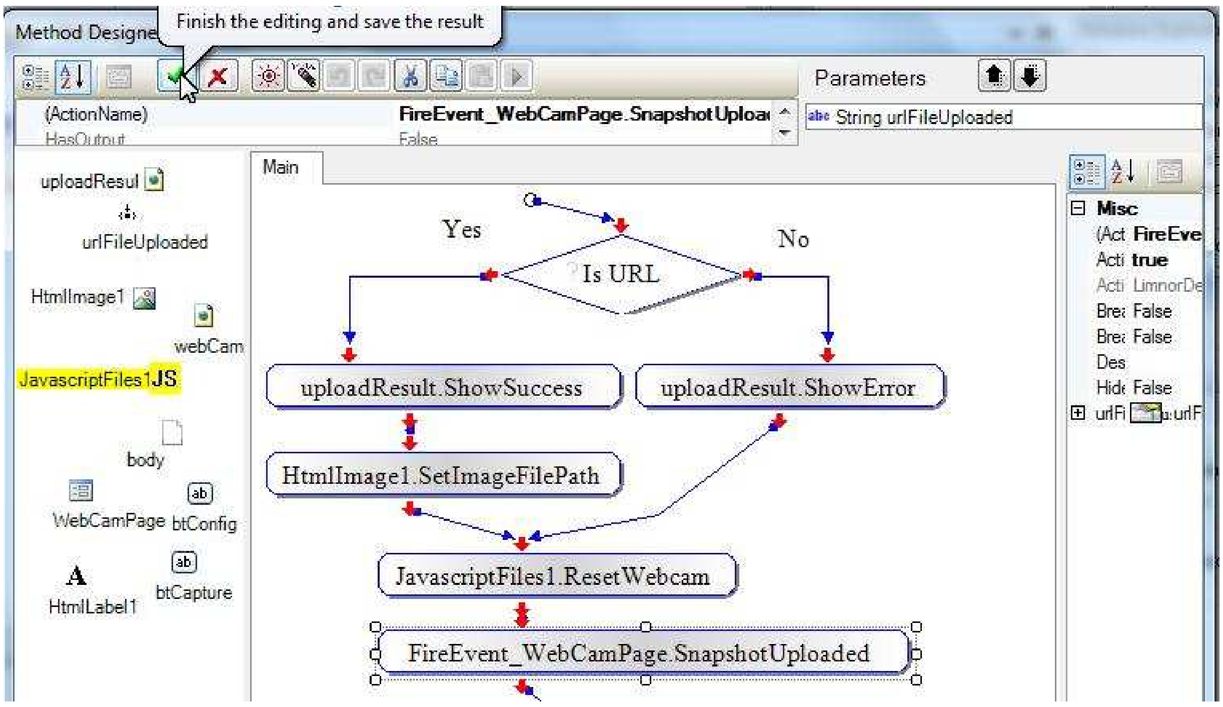
(ActionName)	FireEvent_WebCamPage.SnapshotUploaded
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	LimnorDesigner.Event.FireEventMethod
urlFileUpload	urlFileUploaded



WebCamPage from WebPage
Constructors
Overrides and Interfaces
Instance Members
Properties inherited
Methods inherited
Actions
Properties
Methods
Events
SnapshotUploaded(JsString urlFileUpload)
FireEvent_WebCamPage.SnapshotUploaded
urlFileUpload

JavascriptFiles1 of JavascriptFiles.Ex
String urlFileUploaded
String urlFileUploaded

Link it to the last action:

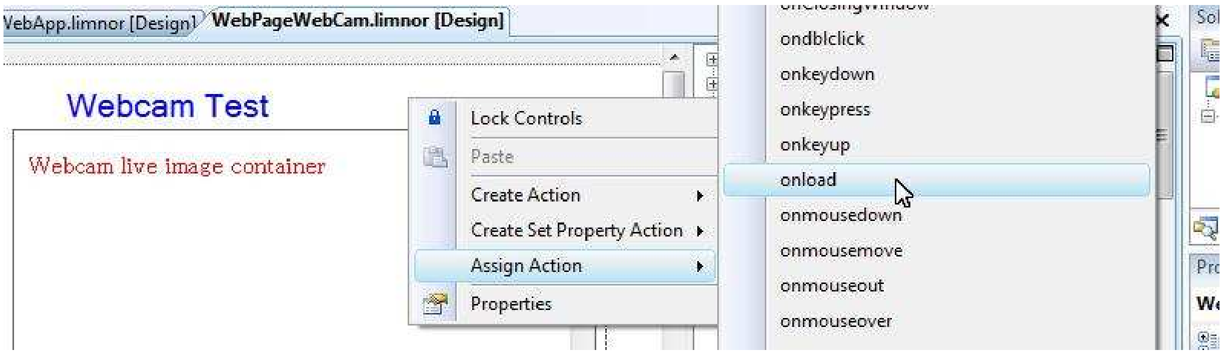


Hook Upload Handling

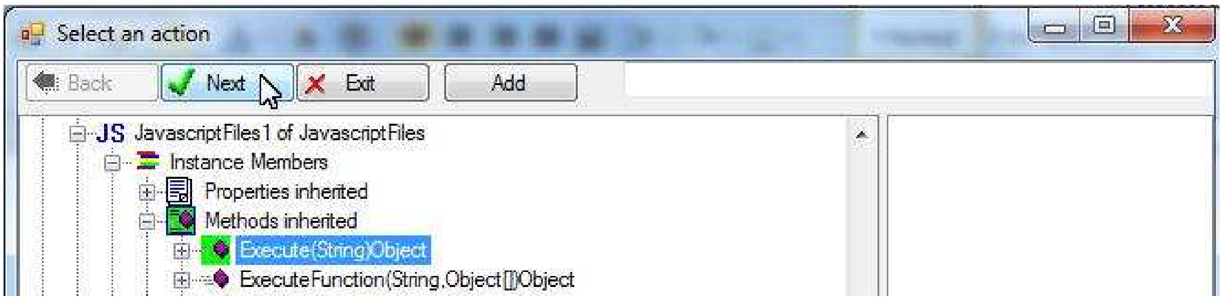
jpegcam uses such a code to hook upload handler:

```
webcam.set_hook( 'onComplete', handlerName );
```

In our case the handlerName is “onSnapshotUploaded”. We may execute the above code at the event of onload of the web page:

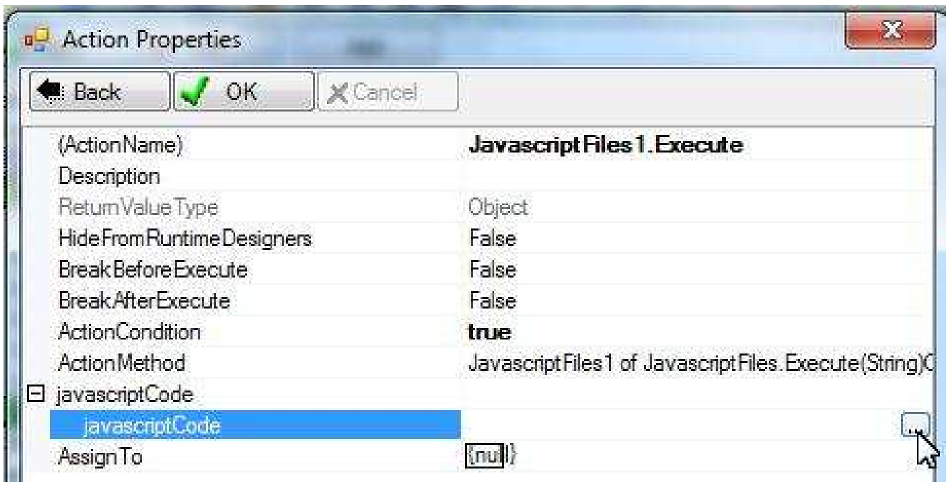


Select Execute method:

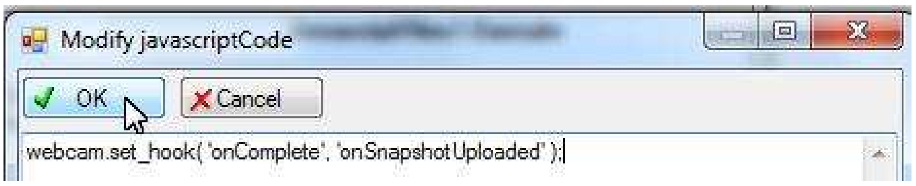


JS JavascriptFiles1 of JavascriptFiles
Instance Members
Properties inherited
Methods inherited
Execute(String)Object
ExecuteFunction(String,Object[])Object

Supply the JavaScript code:

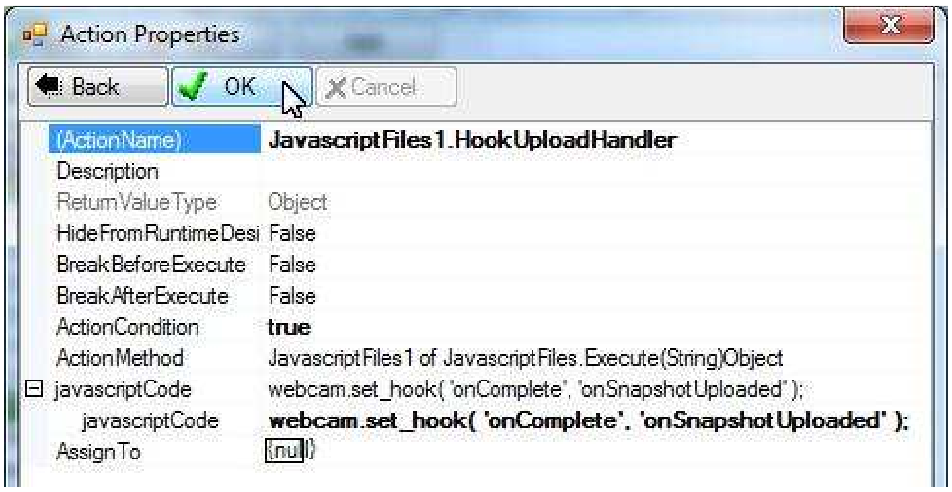


(ActionName)	JavascriptFiles1.Execute
Description	
ReturnValueType	Object
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)O
javascriptCode	
javascriptCode	
AssignTo	{null}



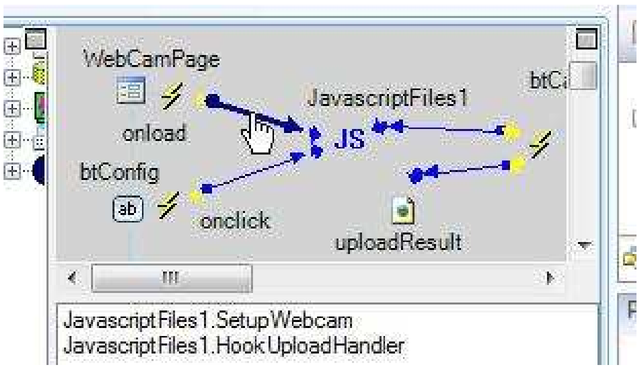
```
webcam.set_hook( 'onComplete', 'onSnapshotUploaded' );
```

Rename the action and click OK:



(ActionName)	JavascriptFiles1.HookUploadHandler
Description	
ReturnValueType	Object
HideFromRuntimeDesi	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	JavascriptFiles1 of JavascriptFiles.Execute(String)Object
javascriptCode	webcam.set_hook('onComplete', 'onSnapshotUploaded');
javascriptCode	webcam.set_hook('onComplete', 'onSnapshotUploaded');
AssignTo	{null}

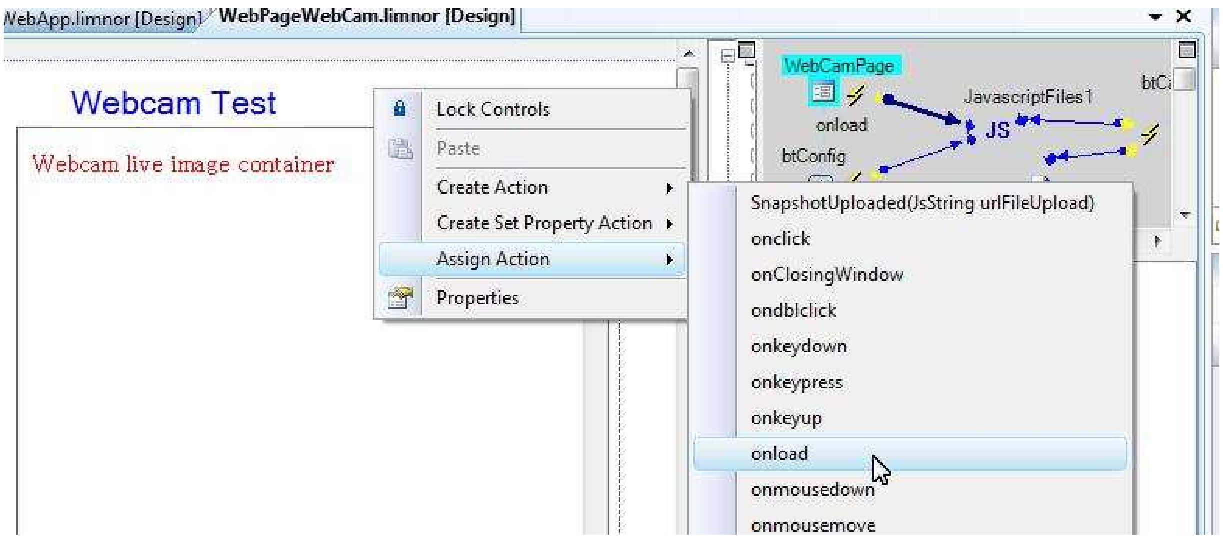
The action is created and assigned to the event onload:



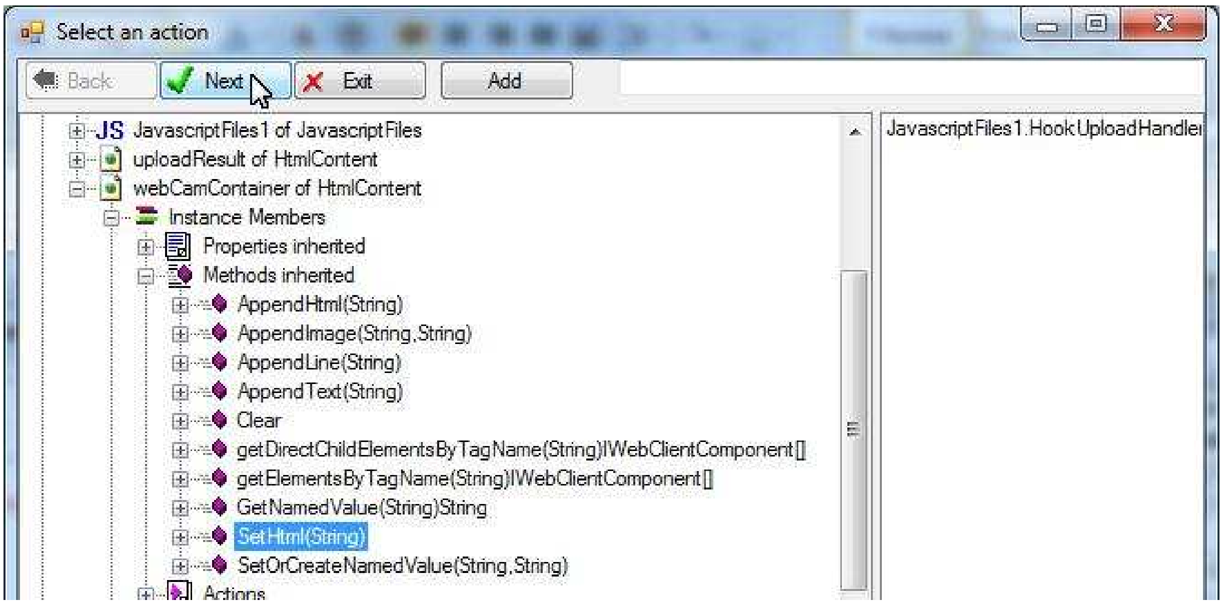
JavascriptFiles1.SetupWebcam
JavascriptFiles1.HookUploadHandler

Show Live Webcam

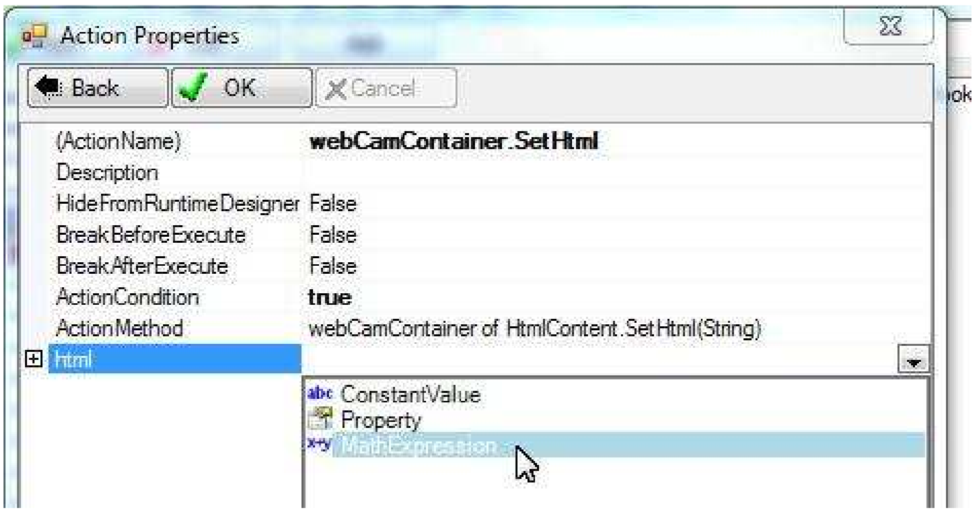
jpegcam uses code webcam.get_html(320, 240) to return HTML code which shows a Flash object. We may set it as the contents of our WebCamContainer component. We also do it at the event of onload of the web page:



Select SetHtml method of our webcam container:

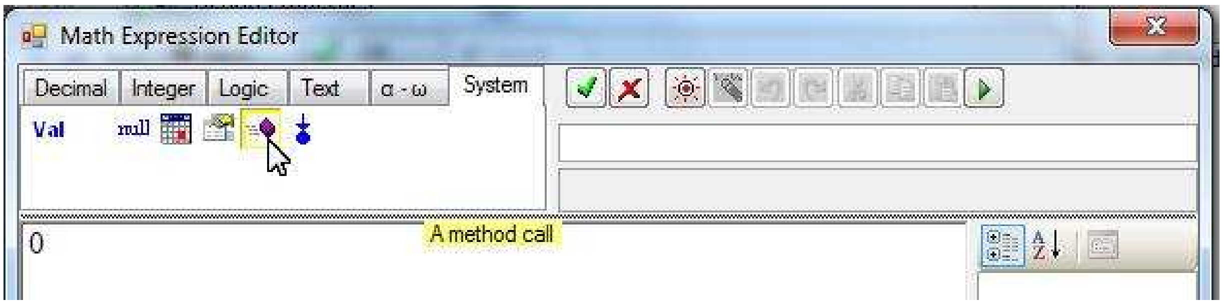

Select an action
Back
Next
Exit
Add
JavascriptFiles1 of JavascriptFiles
uploadResult of HtmlContent
webCamContainer of HtmlContent
Instance Members
Properties inherited
Methods inherited
AppendHtml(String)
AppendImage(String,String)
AppendLine(String)
AppendText(String)
Clear
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
SetHtml(String)
SetOrCreateNamedValue(String,String)
Actions
JavascriptFiles1.HookUploadHandler

Select Math Expression to execute the JavaScript code:

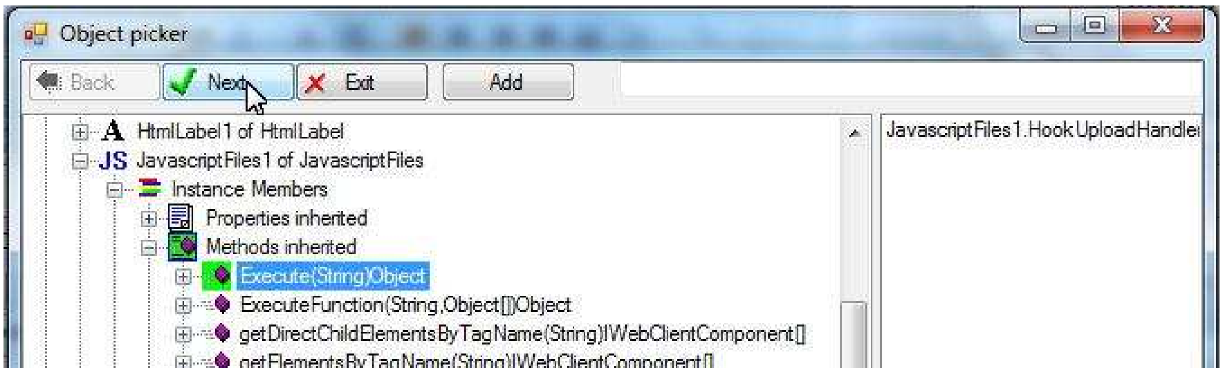


(ActionName)	webCamContainer.SetHtml
Description	
HideFromRuntimeDesigner	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	webCamContainer of HtmlContent.SetHtml(String)
html	

Select Method icon:



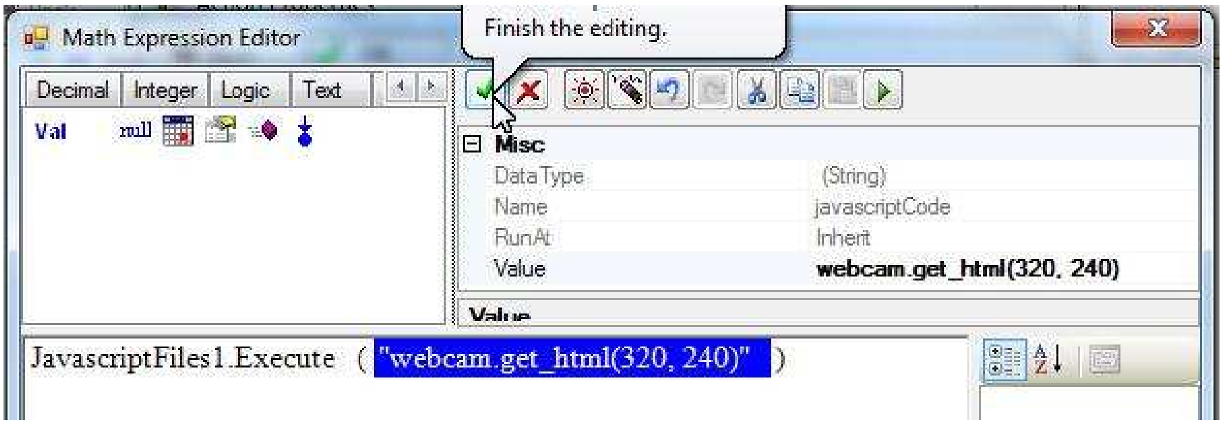
Select Execute method:



HtmlLabel1 of HtmlLabel
JavascriptFiles1 of JavascriptFiles
Instance Members
Properties inherited
Methods inherited
Execute(String)Object
ExecuteFunction(String,Object[])Object
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]

JavascriptFiles1.HookUploadHandler

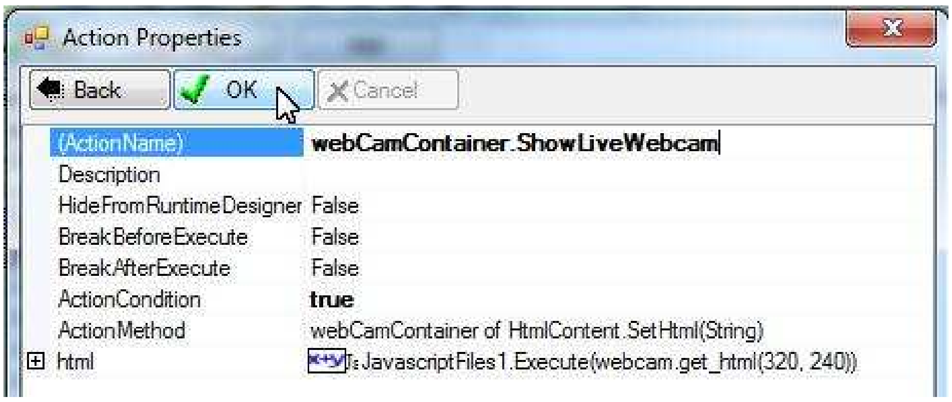
Supply the JavaScript code:



DataType	(String)
Name	javascriptCode
RunAt	Inherit
Value	webcam.get_html(320, 240)

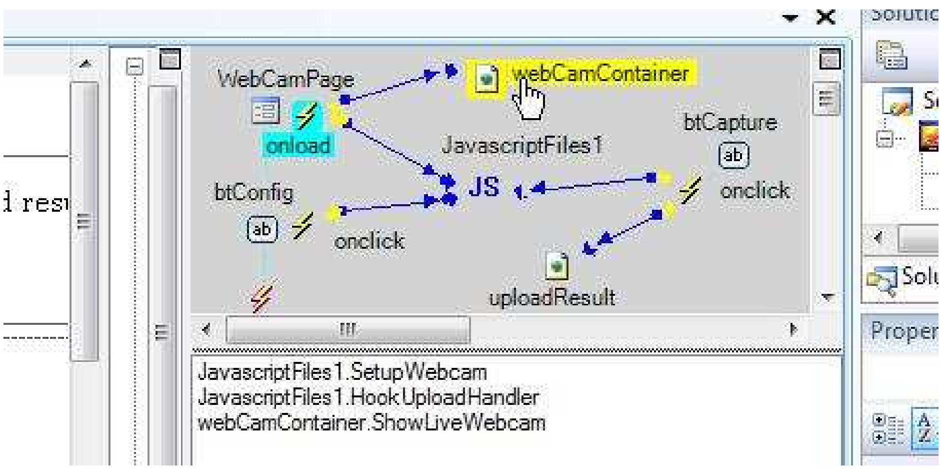
JavascriptFiles1.Execute ("webcam.get_html(320, 240)")

Rename the action and click OK:



(ActionName)	webCamContainer.ShowLiveWebcam
Description	
HideFromRuntimeDesigner	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	webCamContainer of HtmlContent.SetHtml(String)
html	JavascriptFiles1.Execute(webcam.get_html(320, 240))

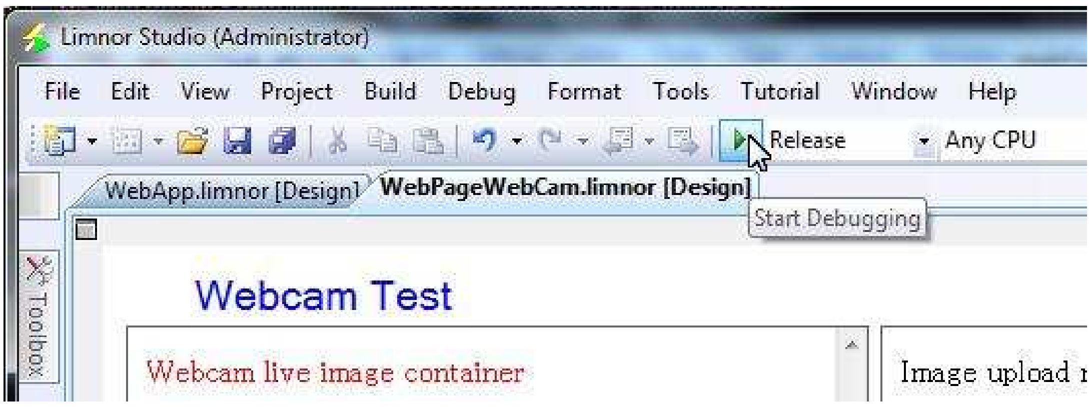
The action is assigned to onload:



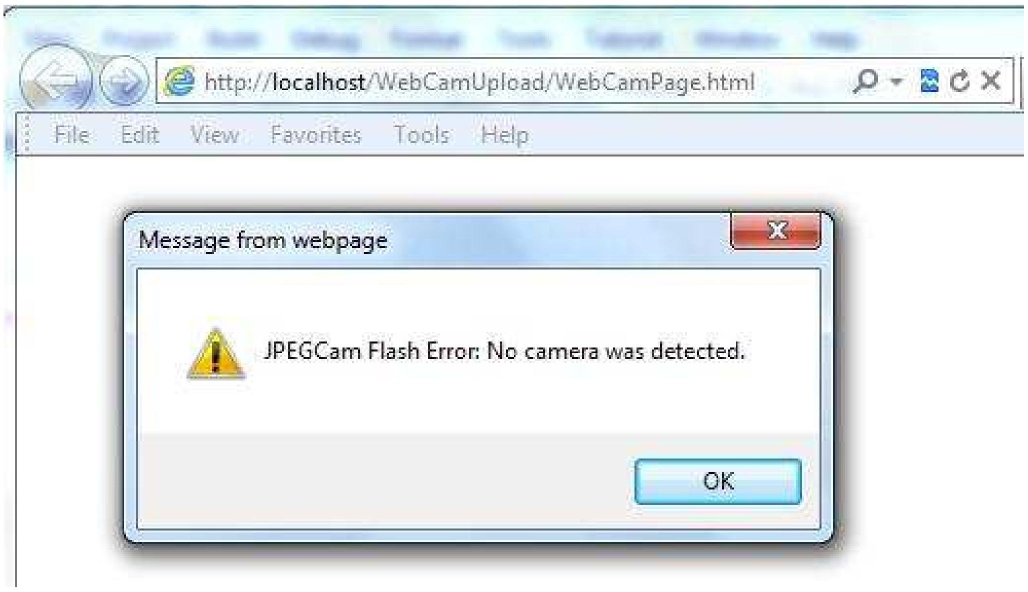
JavascriptFiles1.SetupWebcam
JavascriptFiles1.HookUploadHandler
webCamContainer.ShowLiveWebcam

Test

We may test the web application now.


Limnor Studio (Administrator)
File Edit View Project Build Debug Format Tools Tutorial Window Help
Release
Any CPU
WebApp.limnor [Design]
WebPageWebCam.limnor [Design]
Start Debugging
Toolbox
Webcam Test
Webcam live image container
Image upload r

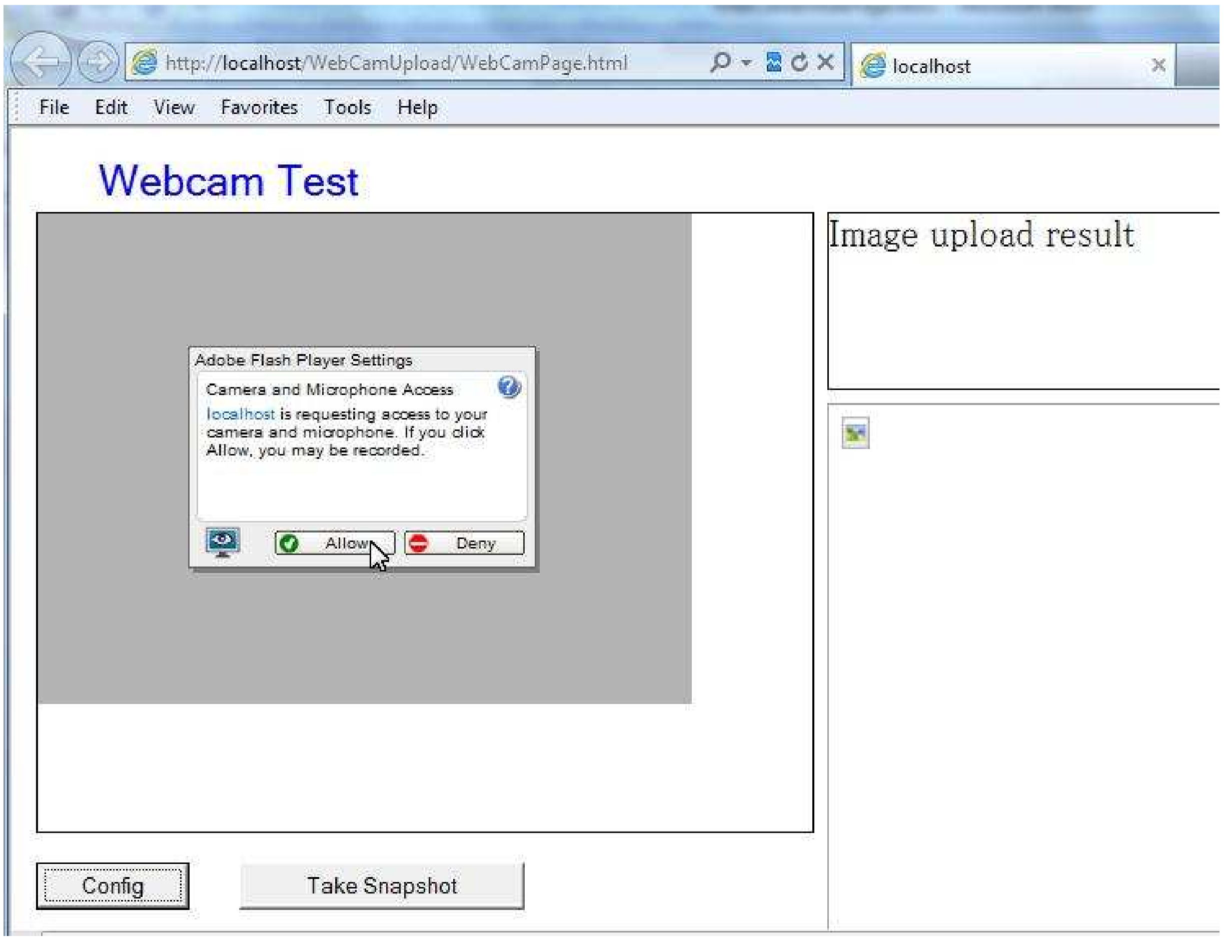
Note that a webcam should be available on the test computer. If a webcam does not exist then we will get an error:


http://localhost/WebCamUpload/WebCamPage.html
File Edit View Favorites Tools Help

Message from webpage

JPEGCam Flash Error: No camera was detected.

Insert a webcam and reload the web page. It asks permission to access the webcam:


http://localhost/WebCamUpload/WebCamPage.html
localhost
File Edit View Favorites Tools Help

Webcam Test

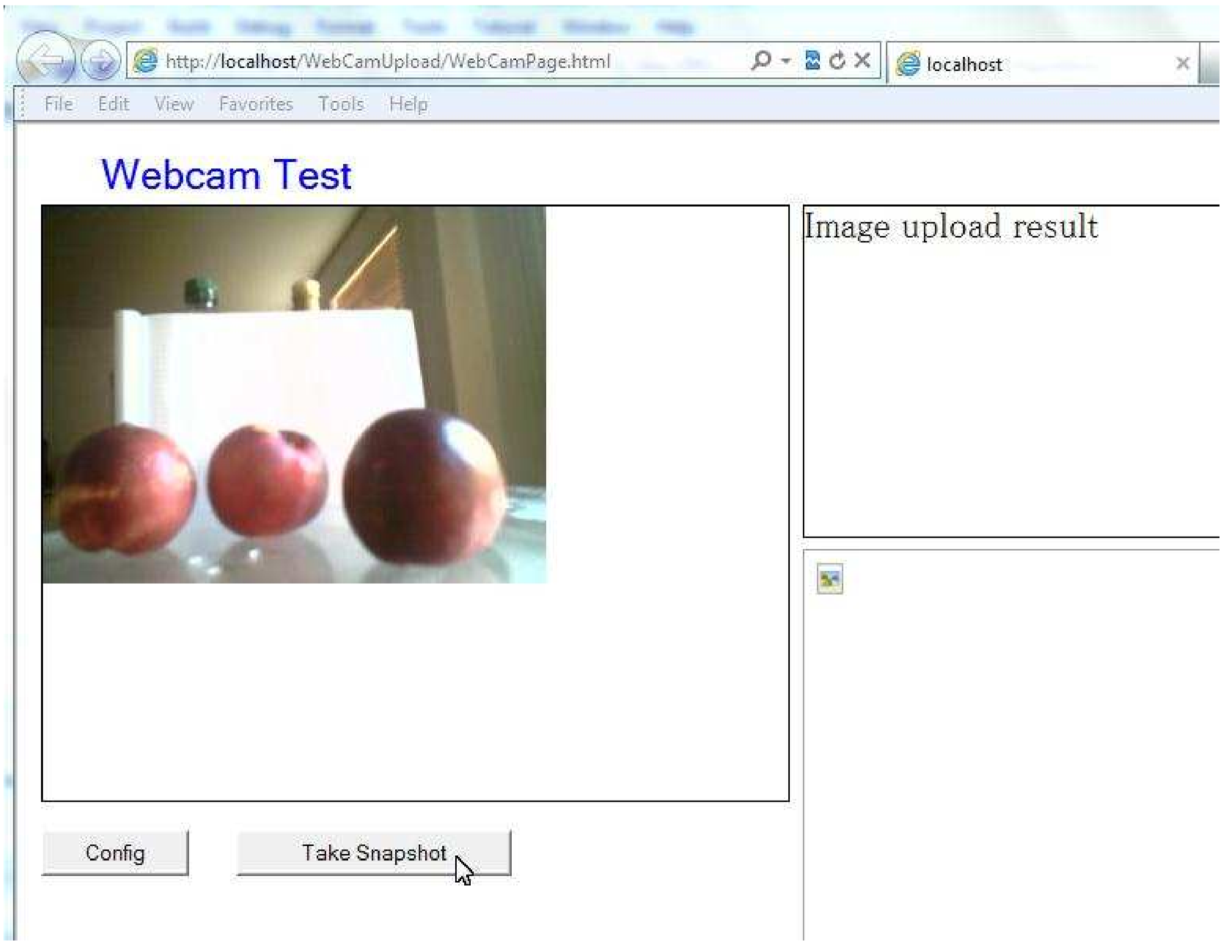
Adobe Flash Player Settings

Camera and Microphone Access

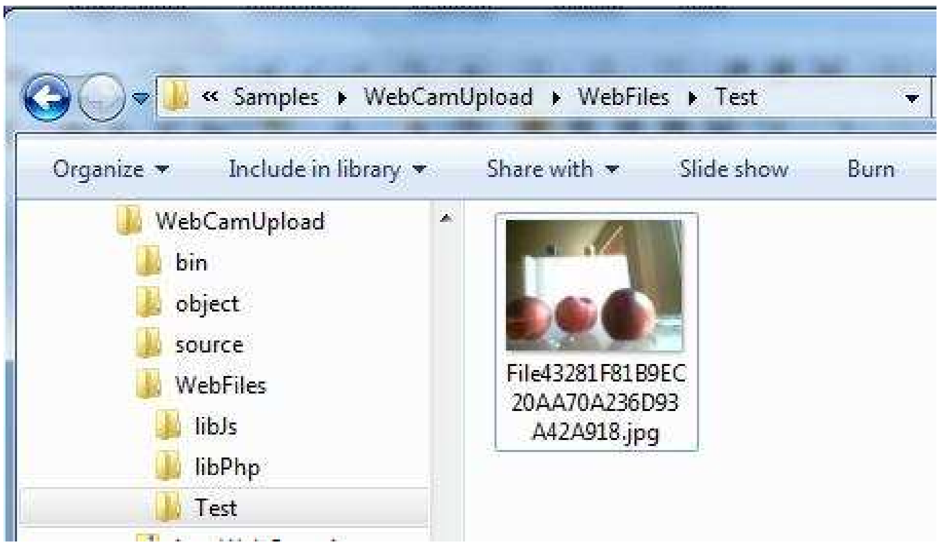
localhost is requesting access to your camera and microphone. If you click Allow, you may be recorded.

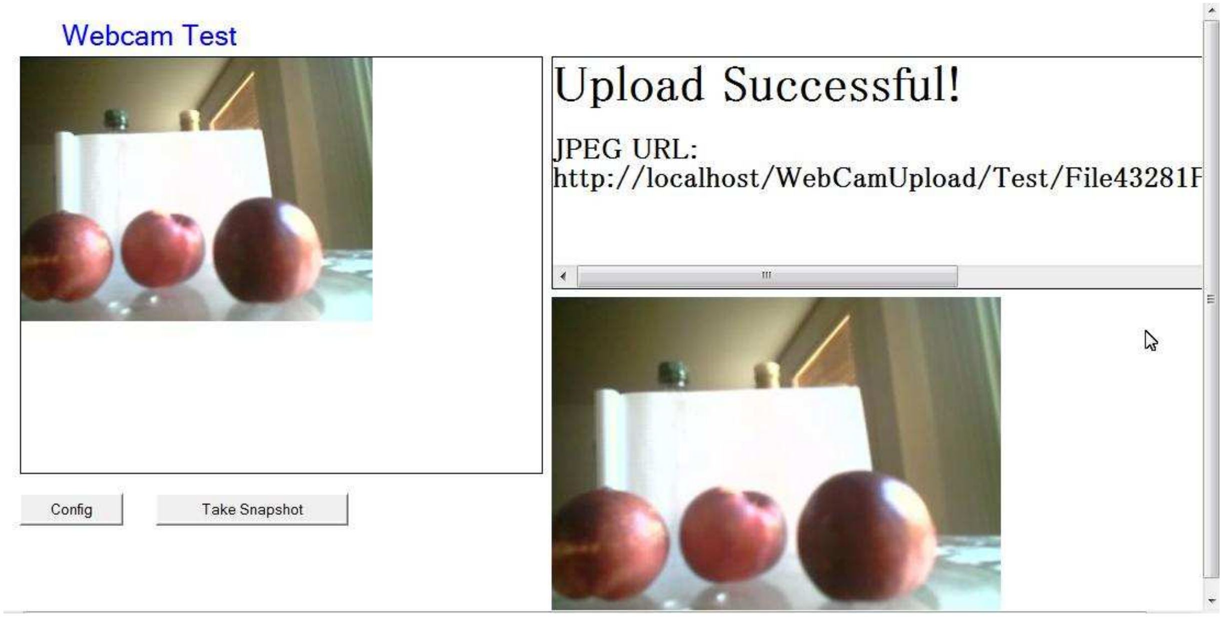
Image upload result

Click Allow. Webcam live image appears in the web page. Click “Take Snapshot” button:


http://localhost/WebCamUpload/WebCamPage.html
localhost
File Edit View Favorites Tools Help
Webcam Test
Image upload result
Config
Take Snapshot

A file is created in the web server and appears in the web browser:



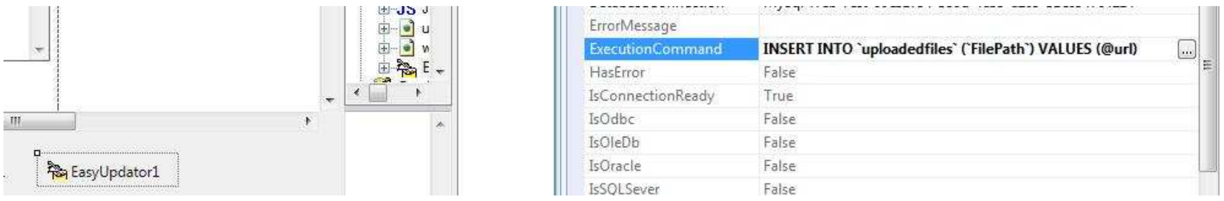


Upload Successful!

JPEG URL:
http://localhost/WebCamUpload/Test/File43281F

Save uploaded files to database

Suppose we have a table with a FilePath field to record file paths. We may use an EasyUpdater component to insert URL of uploaded file into the table.

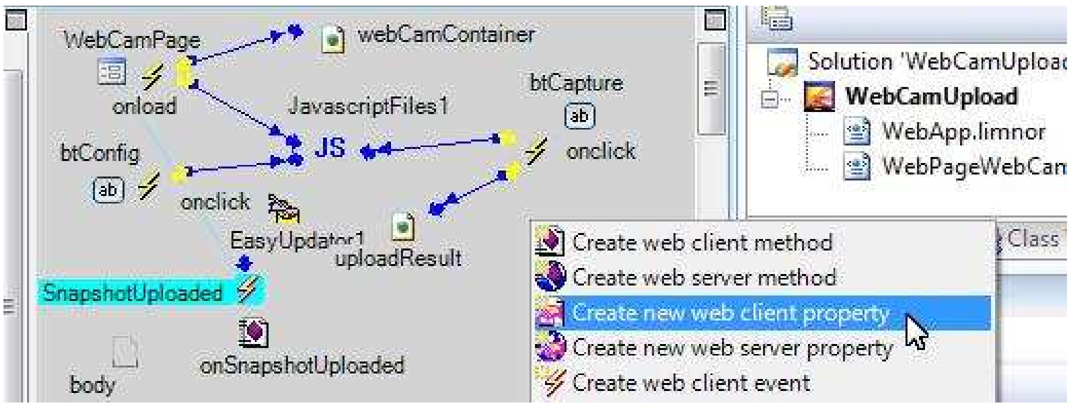


ErrorMessage	
ExecutionCommand	INSERT INTO `uploadedfiles` (`FilePath`) VALUES (@url)
HasError	False
IsConnectionReady	True
IsOdbc	False
IsOleDb	False
IsOracle	False
IsSQLSever	False

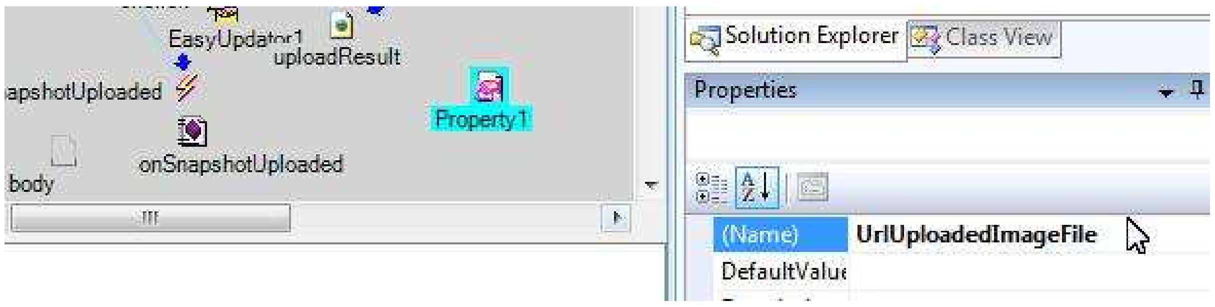
We may assign an ExecuteWithParameterValues action at the event SnapshotUploaded.

Remember Uploaded File URL

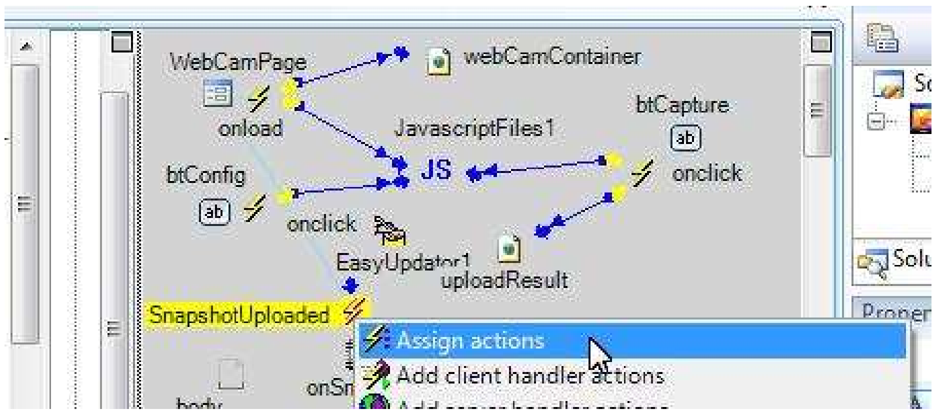
First, let's create a client property to remember the URL of uploaded file:

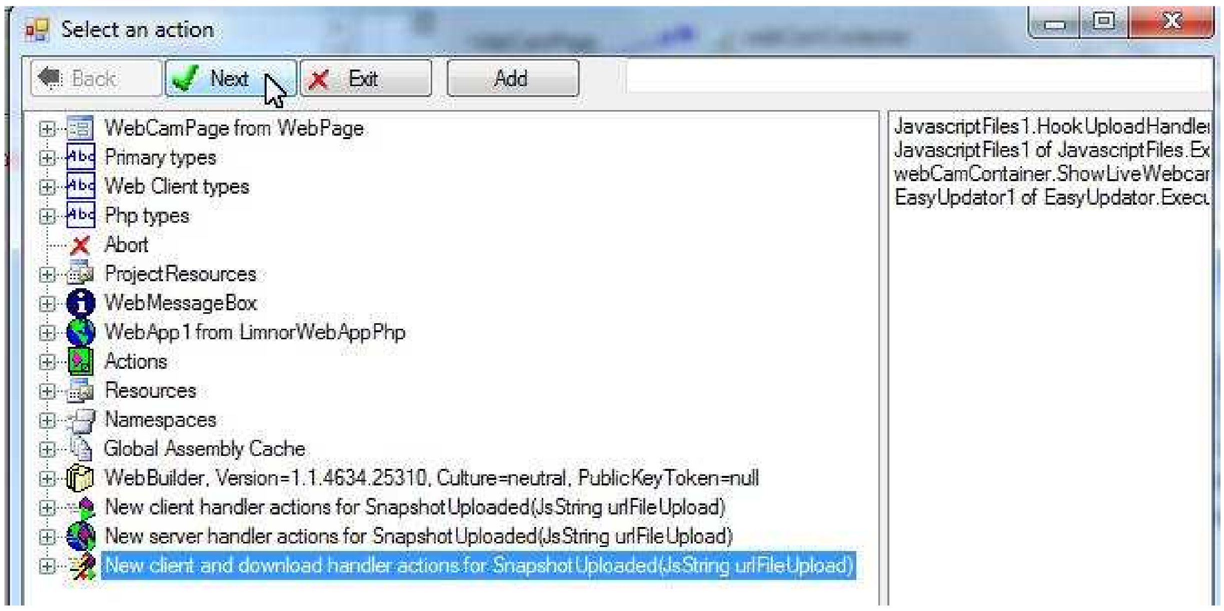


Rename it to UrlUploadedImageFile:



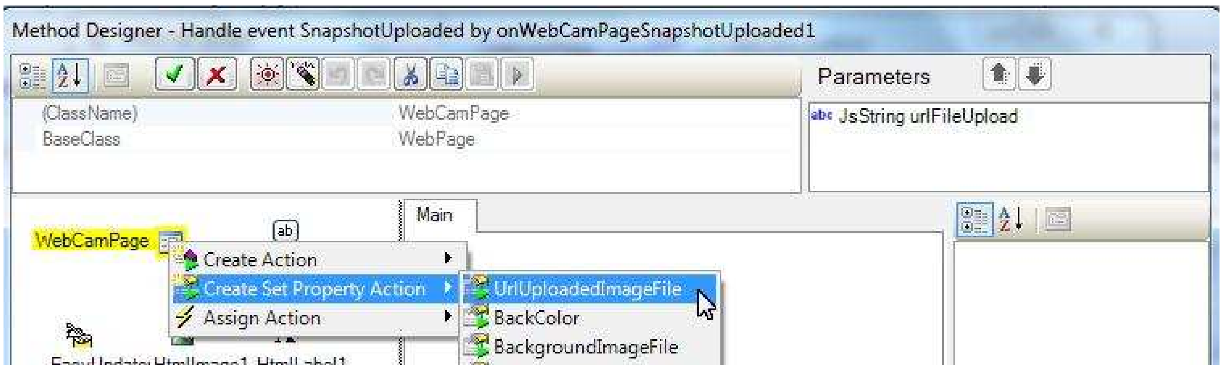
Set this property at the event of SnapshotUploaded:





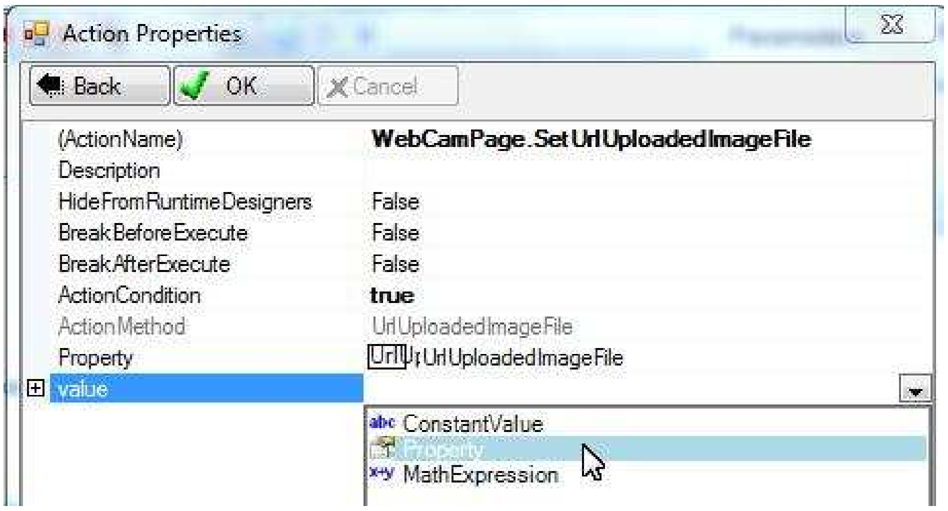
WebCamPage from WebPage
Primary types
Web Client types
Php types
Abort
ProjectResources
WebMessageBox
WebApp1 from LimnorWebAppPhp
Actions
Resources
Namespaces
Global Assembly Cache
WebBuilder, Version=1.1.4634.25310, Culture=neutral, PublicKeyToken=null
New client handler actions for SnapshotUploaded(JsString urlFileUpload)
New server handler actions for SnapshotUploaded(JsString urlFileUpload)
New client and download handler actions for SnapshotUploaded(JsString urlFileUpload)

JavascriptFiles1.HookUploadHandle
JavascriptFiles1 of JavascriptFiles.Ex
webCamContainer.ShowLiveWebcar
EasyUpdator1 of EasyUpdator.Execu

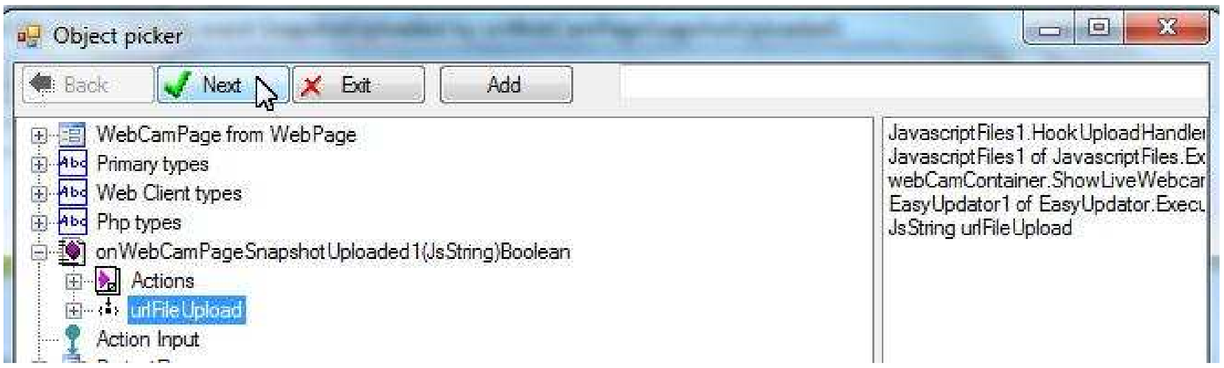

Method Designer - Handle event SnapshotUploaded by onWebCamPageSnapshotUploaded1
Parameters
Main
WebCamPage
Create Action
Create Set Property Action
Assign Action
UrlUploadedImageFile
BackColor
BackgroundImageFile

(ClassName)	WebCamPage
BaseClass	WebPage

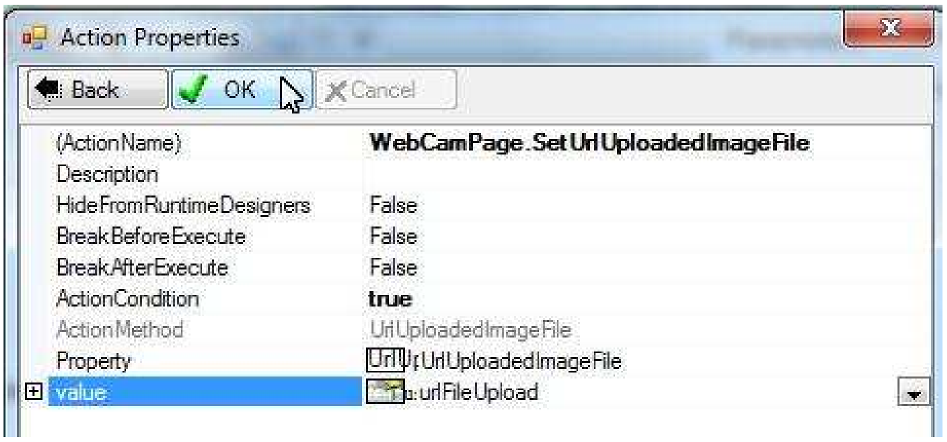
JsString urlFileUpload



(ActionName)	WebCamPage.SetUrlUploadedImageFile
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	UrlUploadedImageFile
Property	UrlUploadedImageFile
value	

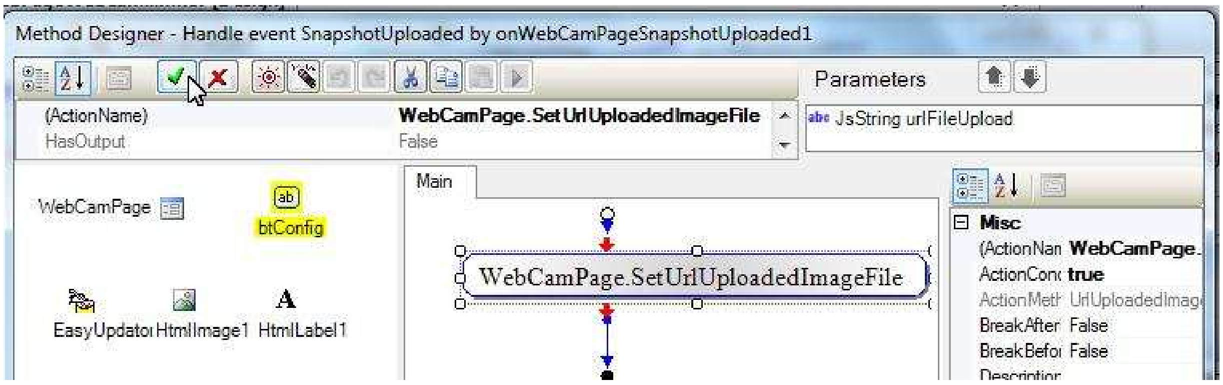


JavascriptFiles1.HookUploadHandle
JavascriptFiles1 of JavascriptFiles.Ex
webCamContainer.ShowLiveWebcar
EasyUpdator1 of EasyUpdator.Execu
JsString urlFileUpload



(ActionName)	WebCamPage.SetUrlUploadedImageFile
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	UrlUploadedImageFile
Property	Url UrlUploadedImageFile
value	urlFileUpload

The action appears in the method. Close the editor:

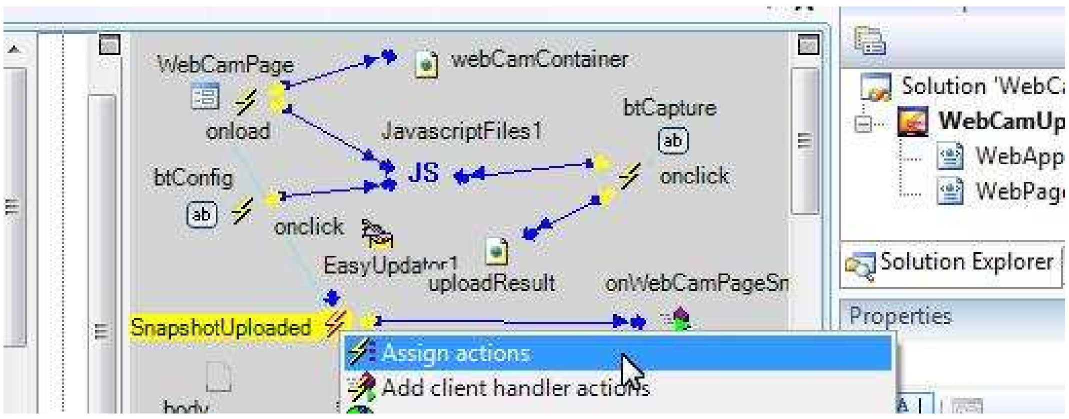

Method Designer - Handle event SnapshotUploaded by onWebCamPageSnapshotUploaded1

(ActionName)	WebCamPage.SetUrlUploadedImageFile
HasOutput	False

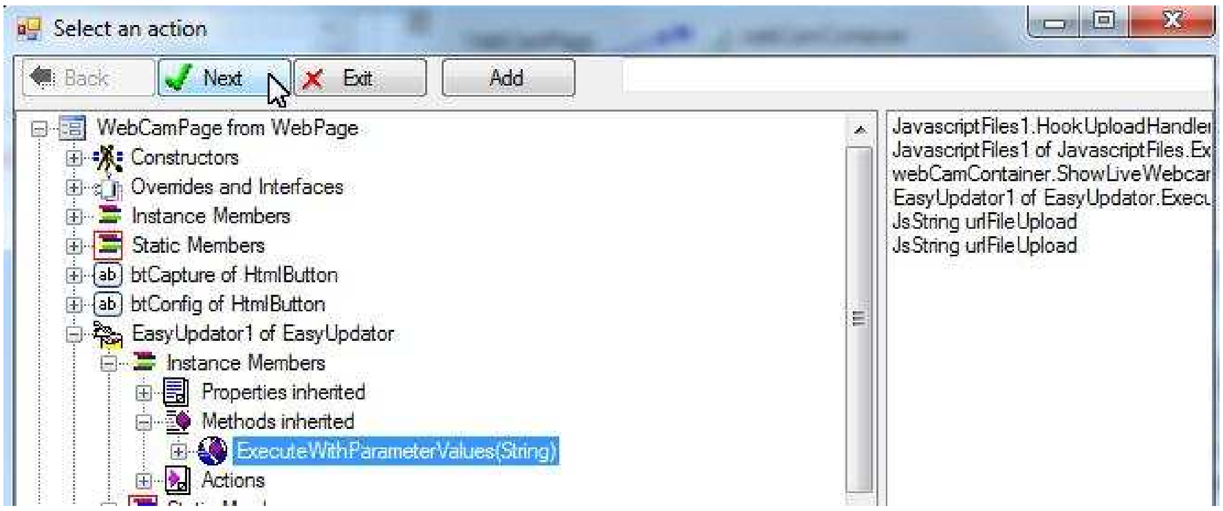
Parameters: JsString urlFileUpload

Execute Database Action

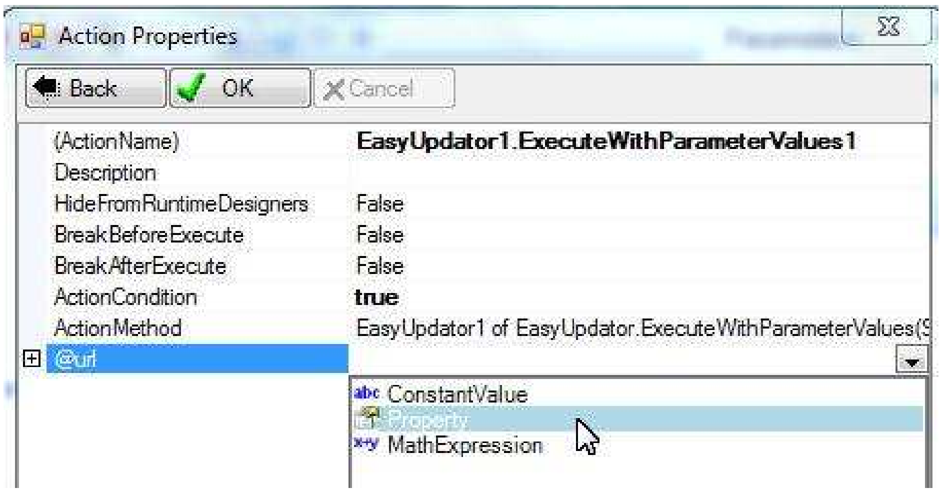
Now we may execute database updating action, also at the event of SnapshotUploaded:



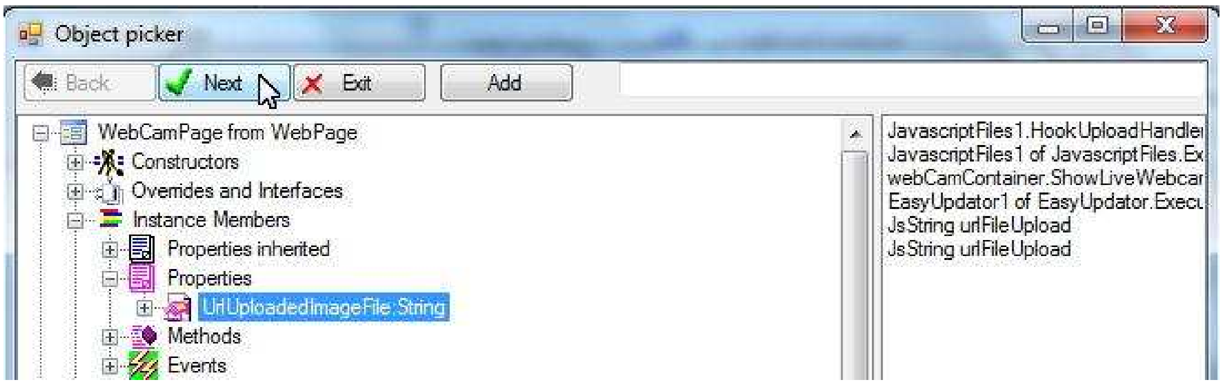
Select ExecuteWithParameterValues:

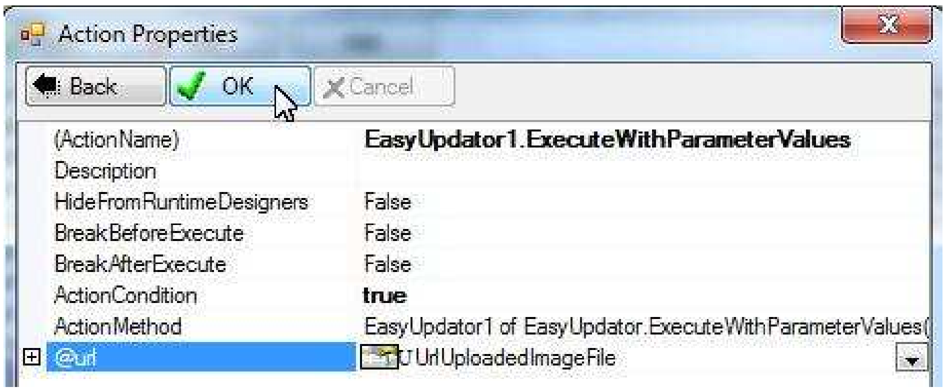


Provide the property value to @url:



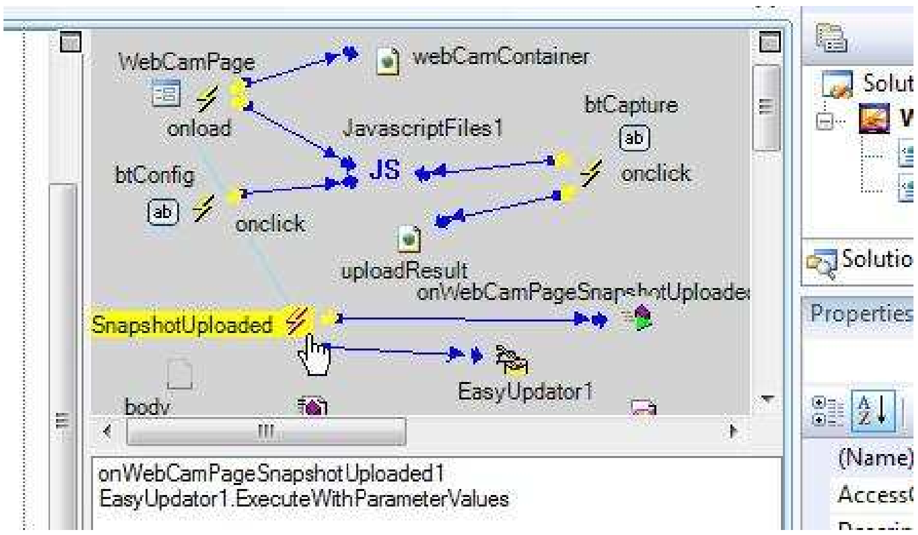
(ActionName)	EasyUpdator1.ExecuteWithParameterValues1
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyUpdator1 of EasyUpdator.ExecuteWithParameterValues(
@url	




Action Properties

(ActionName)	EasyUpdator1.ExecuteWithParameterValues
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyUpdator1 of EasyUpdator.ExecuteWithParameterValues(
@url	UrlUploadedImageFile

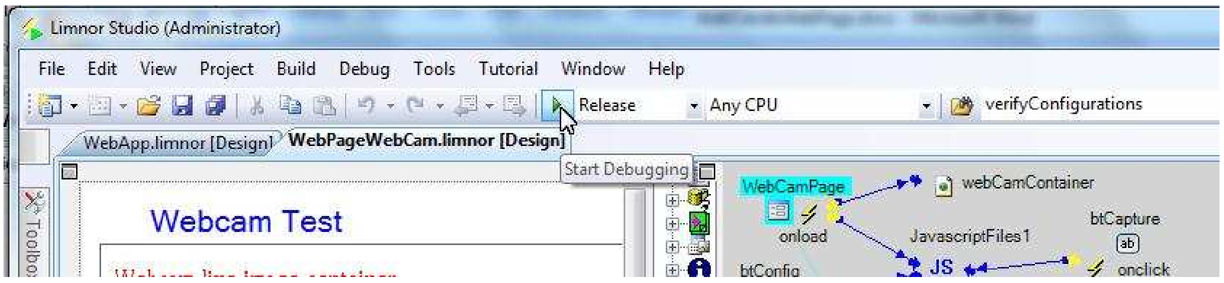
The action is created and assigned to the event:



onWebCamPageSnapshotUploaded1
EasyUpdator1.ExecuteWithParameterValues

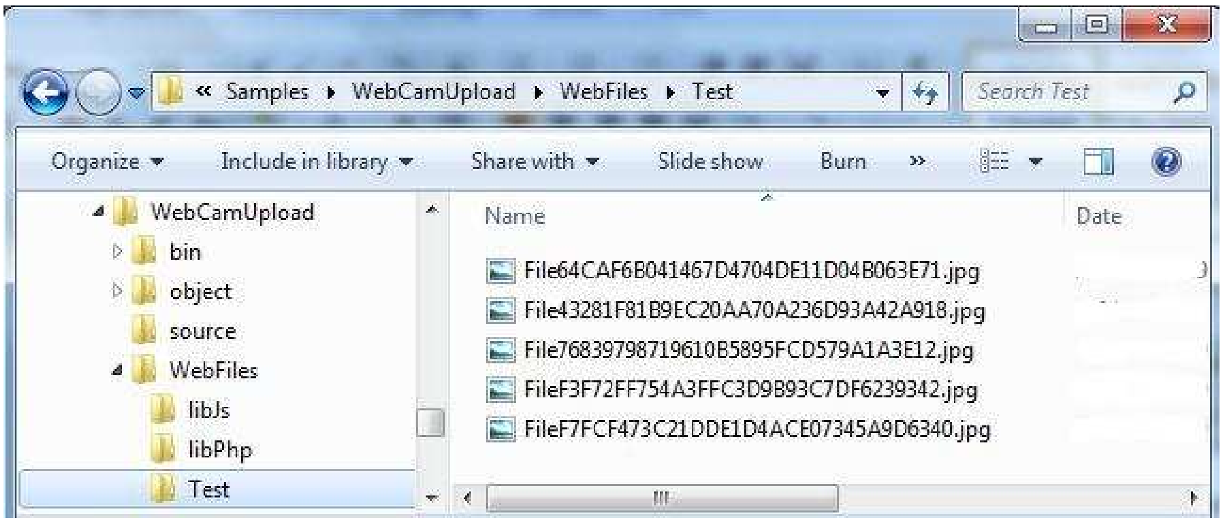
Test

Launch the web application:



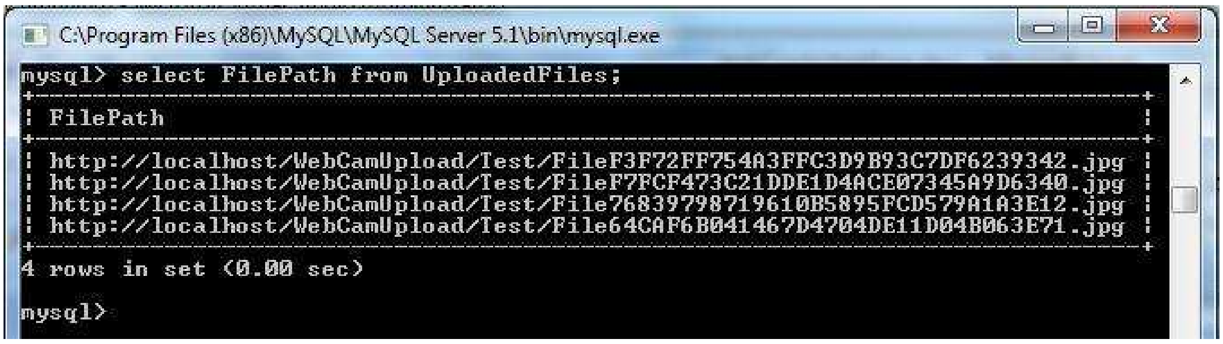
The web page appears. If the webcam is connected properly then live webcam image appears in the web page. Each click of the "Take Snapshot" button will capture one snapshot and upload it to the web server.

All files uploaded are saved under Test folder:



File64CAF6B041467D4704DE11D04B063E71.jpg
File43281F81B9EC20AA70A236D93A42A918.jpg
File76839798719610B5895FCD579A1A3E12.jpg
FileF3F72FF754A3FFC3D9B93C7DF6239342.jpg
FileF7FCF473C21DDE1D4ACE07345A9D6340.jpg

File path for each uploaded file is also saved in the database:



```
mysql> select FilePath from UploadedFiles;
+-----------------------------------------------------------------------------+
| FilePath                                                                    |
+-----------------------------------------------------------------------------+
| http://localhost/WebCamUpload/Test/FileF3F72FF754A3FFC3D9B93C7DF6239342.jpg |
| http://localhost/WebCamUpload/Test/FileF7FCF473C21DDE1D4ACE07345A9D6340.jpg |
| http://localhost/WebCamUpload/Test/File76839798719610B5895FCD579A1A3E12.jpg |
| http://localhost/WebCamUpload/Test/File64CAF6B041467D4704DE11D04B063E71.jpg |
+-----------------------------------------------------------------------------+
4 rows in set (0.00 sec)

mysql>
```