Web Application Development

Last update: January 29, 2015

Contents

Introduction

A web application is constructed in 3 distinct technologies.

- HTML – It is used to construct web pages
- Client site scripts – They are used to implement programming logic and manipulate HTML elements, i.e. texts, images, java applets, ActiveX controls, etc. Usually JavaScript, VB Script, and other scripting languages can be used.
- Server site programming languages – They are used to do server site operations, such as database accessing, email sending, web service accessing, etc. For example, CGI Scripting, PHP, Pearl, C/C++, C#, VB, Java Servlets, etc.

Learning 3 technologies in order to develop web sites and web applications can be too much of a burden for some of us.

Limnor Studio uses a visual and codeless programming approach to develop computer software. It can be used to develop GUI applications, Windows services, web services, kiosk applications, screensavers, installers, console applications, software libraries, etc. It can also be used to develop web sites and web applications in the same visual and codeless programming approach as doing other kinds of software.

You use one programming technology instead of 3 to develop web sites.

There is NOT a boundary between client side programming and server side programming. There is no need of handling client server communications and data exchanges.

Behind the scene the compiler of the Limnor Studio automatically generates HTML files, CSS files, client site scripting and server site programming, and automatically generates client server communications and performs necessary data uploads and downloads.

Thus you develop a web site in the same way as developing a standalone program. You focus on implementing your business logic instead of implementing web page and web server interactions.

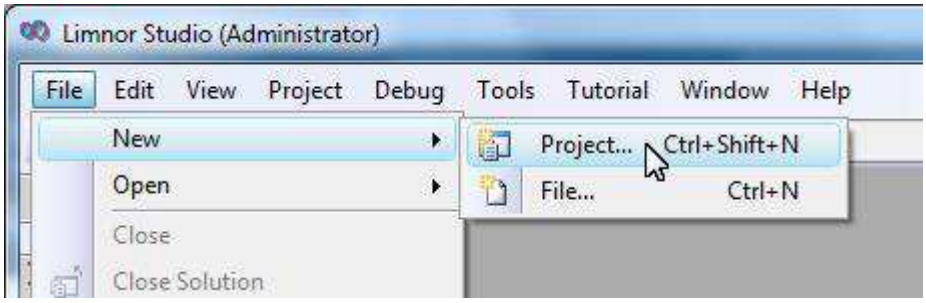
This document describes the process of developing web applications using Limnor Studio in a visual and codeless approach.

The contents of this document were tested on IE, Opera, Chrome, Safari, and FireFox.

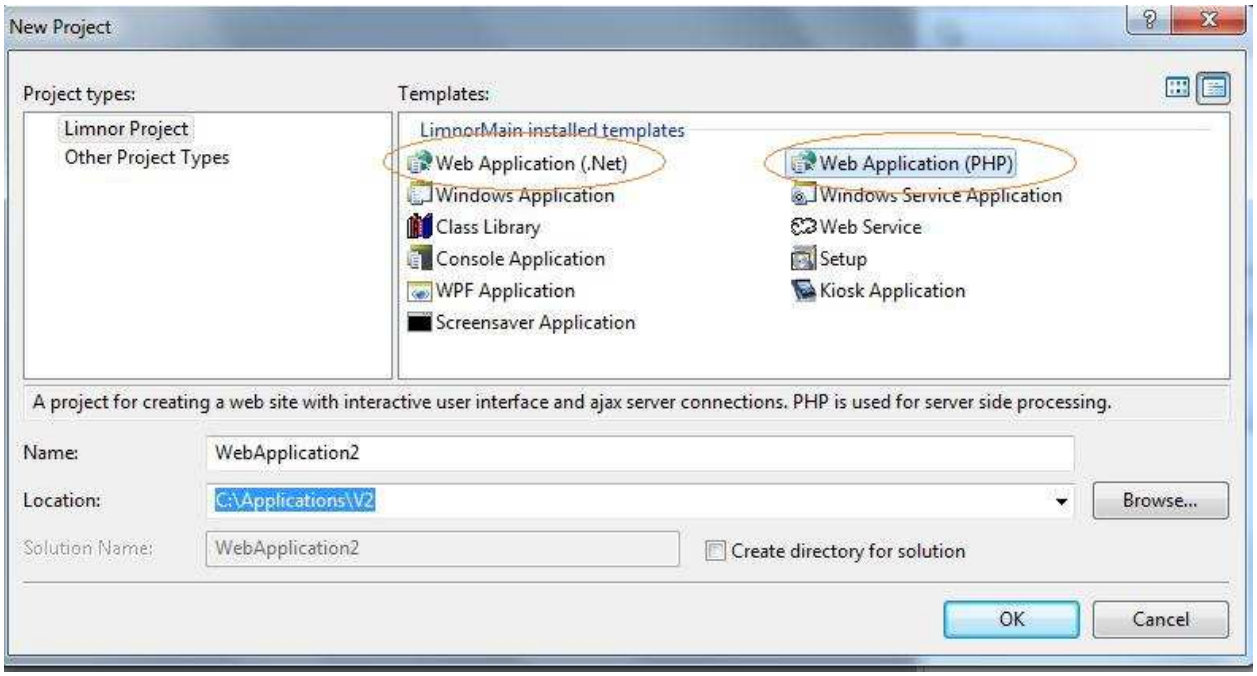
Create a project for web application

Select Web Application Project

Create a new project:



Select Web Application as the project type. Give a project name and specify project folder. You have a choice of selecting a Web Application (.Net) or a Web Application (PHP):



Note that if you are using PHP then it is better not to use spaces in your folder names. There are some discussions on the internet. For example, see http://www.iis-aid.com/articles/trouble_shooting/php_http_500_error

For Linux or for Windows

Limnor Studio currently supports PHP and .Net on the server side. So, technically it is not about Linux or Windows. Practically PHP is most likely being used in Linux machines and .Net is most likely being used in Windows machines.

Usually project type Web Application (PHP) is selected if you want to host your web application in a LAMP platform (Linux Apache MySql PHP); project type Web Application (ASPX) is selected if you want to host your web application in an IIS server.

No matter which project type you select the Visual Programming of the web application is exactly the same. Limnor Studio not only unifies the client/server programming, it also unifies web programming on different server processing technologies.

Some programming components in the Toolbox are only for PHP; some are only for .Net; and some are for both. Limnor Studio lets you know if you try to use incompatible components.

For .Net 3.5 or for .Net 4 (ASPX only)

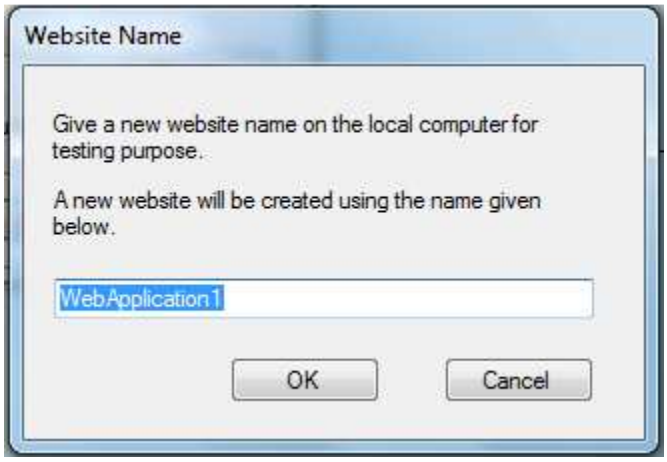
For a Web Application (ASPX) project you need to pay attention to your target IIS. If your IIS only supports .Net 3.5 then you need to use Limnor Studio for .Net 3.5 to develop your project. Do not use Limnor Studio for .Net 4. If you are developing PHP web projects then it does not matter which Limnor Studio edition you are using.

Limnor Studio will create a test web site for each web project you are developing so that you may test your developments. The test web sites are created under your computer's local host. The test web sites are created under an application pool named "Limnor Studio" or "Limnor Studio 5", depending on which Limnor Studio edition you are using. Limnor Studio for .Net 3.5 will create and use application pool "Limnor Studio" which uses .Net 2.0 as web handler. Limnor Studio for .Net 4 will create and use application pool "Limnor Studio 5" which uses .Net 4.0 as web handler.

Test web site

Suppose we select project type Web Application (PHP).

On clicking OK button, a dialogue box appears for specifying a web site name:



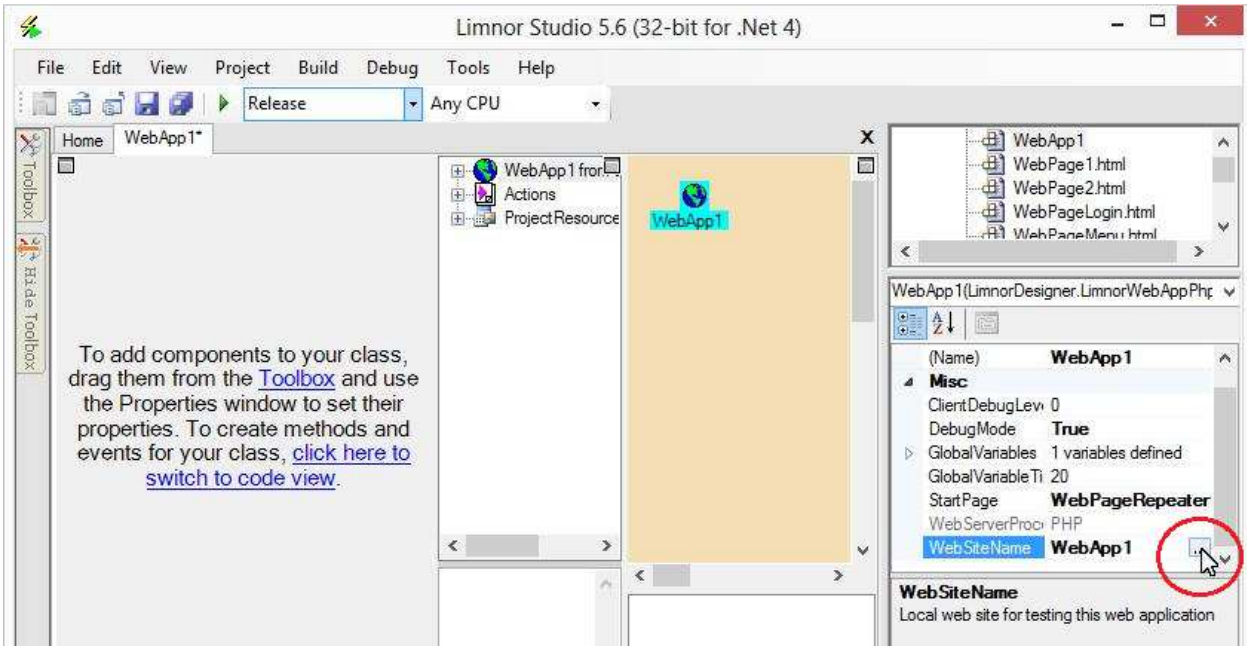
This web site name is for creating a testing web site on the development computer. This web site name does not have to be the same web site name for production.

Limnor Studio is running under Windows. Thus the test web site is created under the IIS server. You should enable IIS on your development computer. For how to enable IIS, see http://msdn.microsoft.com/en-us/library/ms181052(v=vs.80).aspx or search "enable IIS" in the Internet. If the IIS is not enabled then clicking the OK button will generate an error.

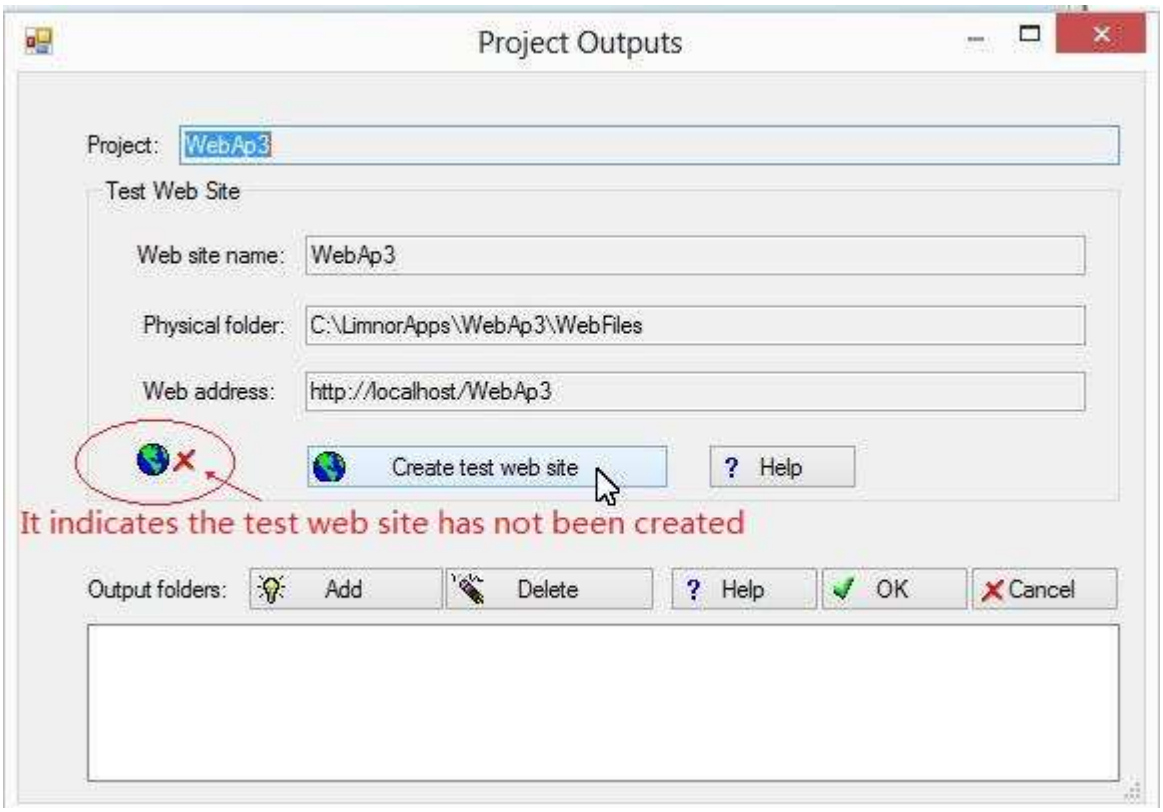
Your IIS server should support PHP. See http://php.net/manual/en/install.windows.php for installing PHP on IIS.

The web application the Limnor Studio compiler generated does not have to run under IIS. It may run under any web server supporting PHP, for example Apache server on a Linux box.

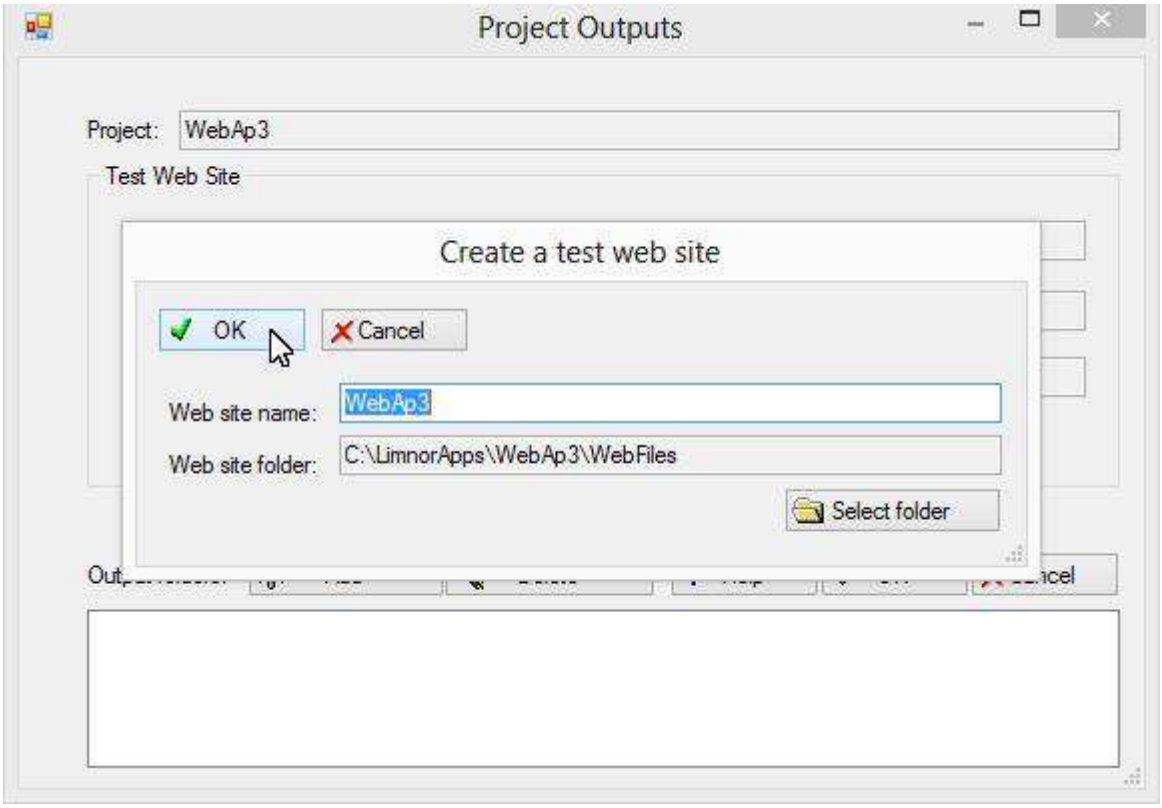
When you open a web page into the designers, if the web project has not a test web site then dialogues will appear to let you create a test web site. You may also change the test web site for a web project by modifying the WebSiteName property of the WebApp object:



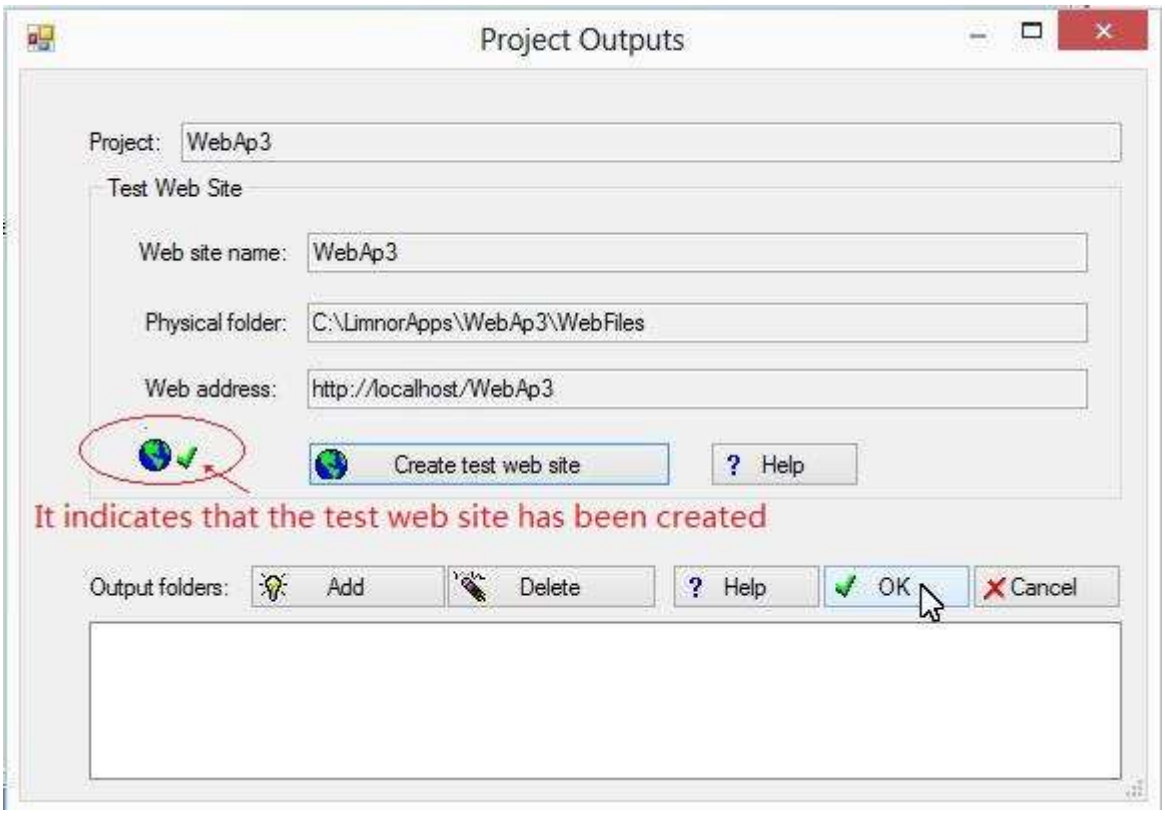
A dialogue box appears:



If the test web site for the project has not been created then you will see a red-X icon. You need to click "Create test web site" button to create a test web site for the project. A dialogue box appears to let you specify the web site name and the physical folder:



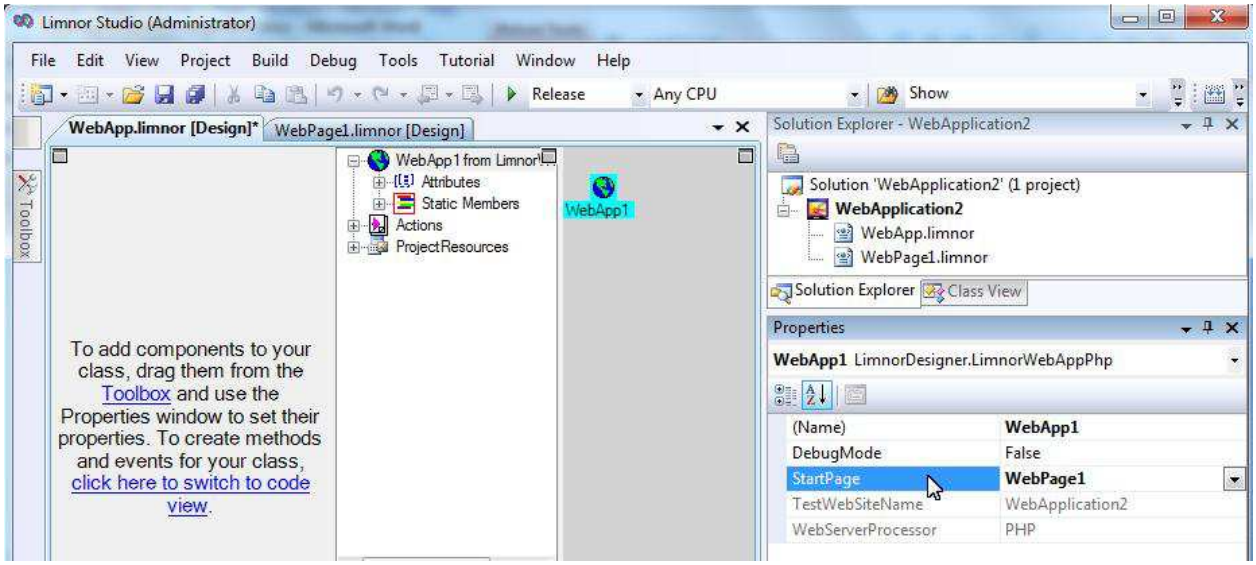
Click OK to create the test web site.



Once the web site is created, you can see a green check icon. Limnor Studio compiler will generate web files such as HTML files, CSS files, JavaScript files, PHP files for PHP web projects, or ASPX and .Net DLL files for ASPX web projects, and other supporting files. All the files will be generated in the "Physical folder" for the test web site so that the web pages can be tested. But you may also let Limnor Studio compiler copy the generated files to other folders, by adding "Output folders".

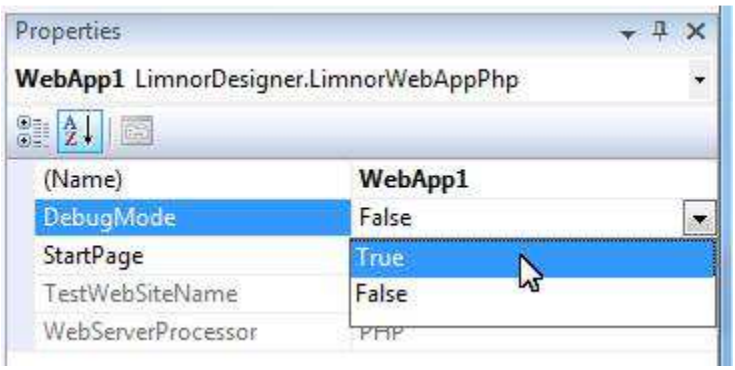
Web Application Class

The new web application has two classes created. WebApp class represents the web application.



The StartPage property indicates which web page will be opened when starting debugging by clicking a green arrow button on the toolbar of the Limnor Studio IDE. Note that this is just for testing. It is not the home page setting. Home page setting must be set through your web server. You may switch StartPage for testing different web pages.

Another important property is DebugMode. While developing and testing, you may set it to True:



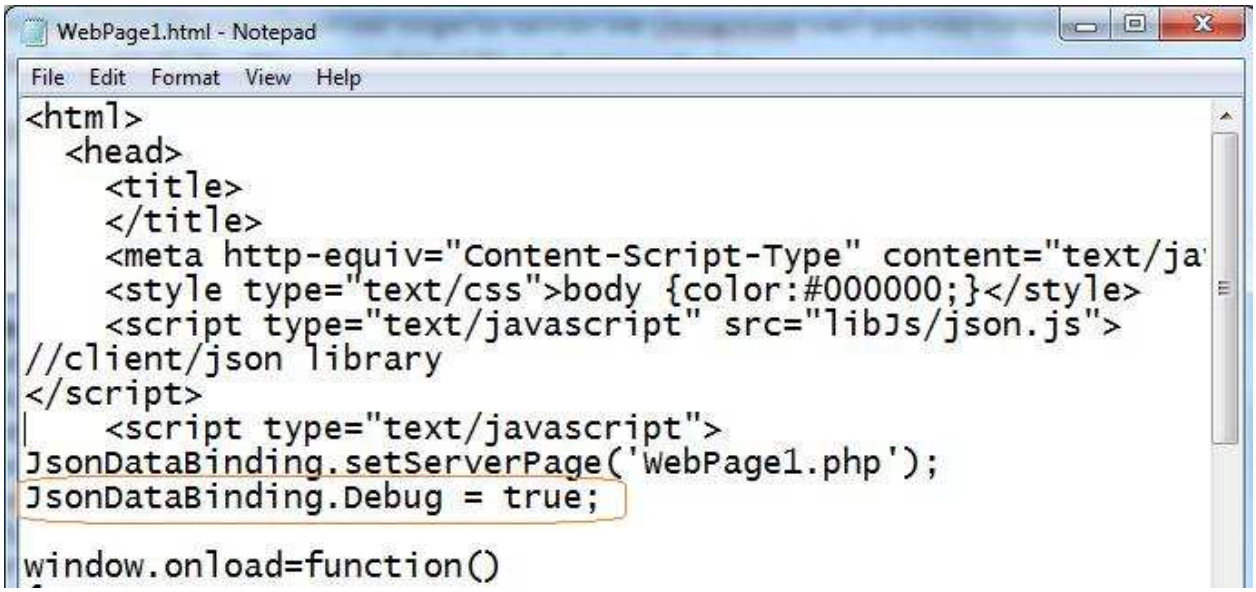
If DebugMode is True while running the web application in a browser then detailed client/server processing information, including requests sent from client pages to the web server, server responses to

the client, log in process, database operations, etc., are displayed. The information is displayed in a popup window. So, you must disable popup-blocker on your web browser if you turn on DebugMode.

When you are ready to publish your web application, turn off DebugMode and re-compile your web application.

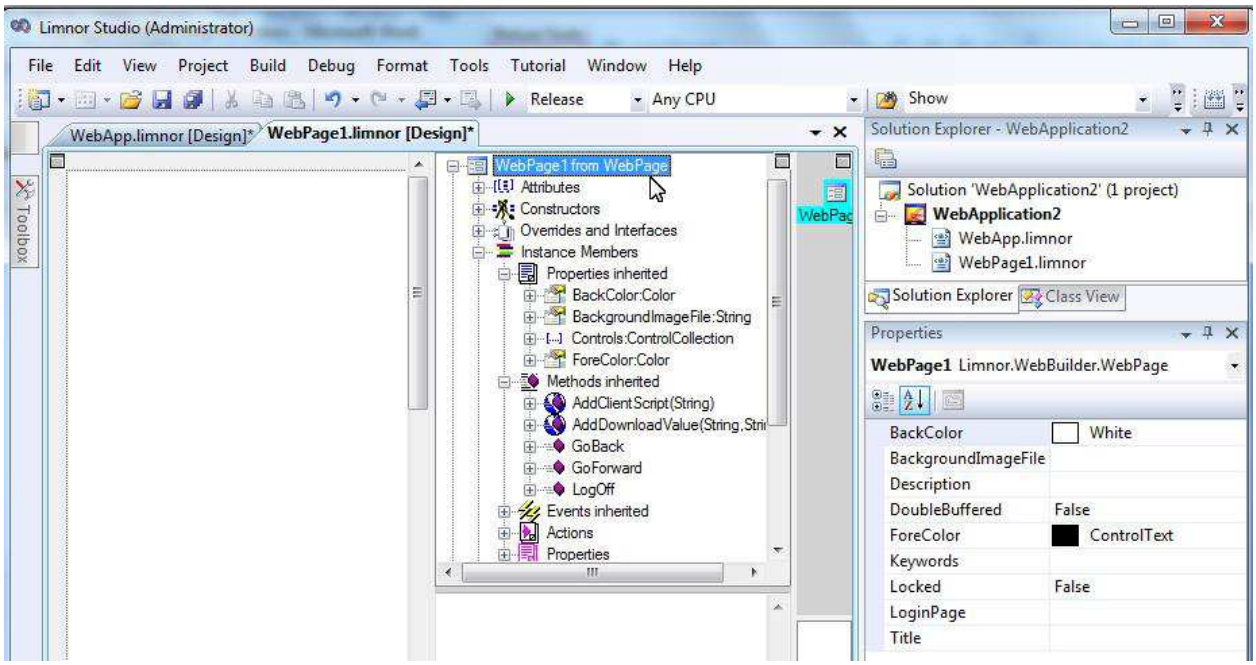
If you published your web application but you want to debug some web pages then you may manually turn on/off the DebugMode by editing an html file and adding/remove the following line

JsonDataBinding.Debug = true;



Web Page Class

The other class created in the new web application is an empty web page named WebPage1:



Develop Web Page

Limnor Studio provides two web page editors:

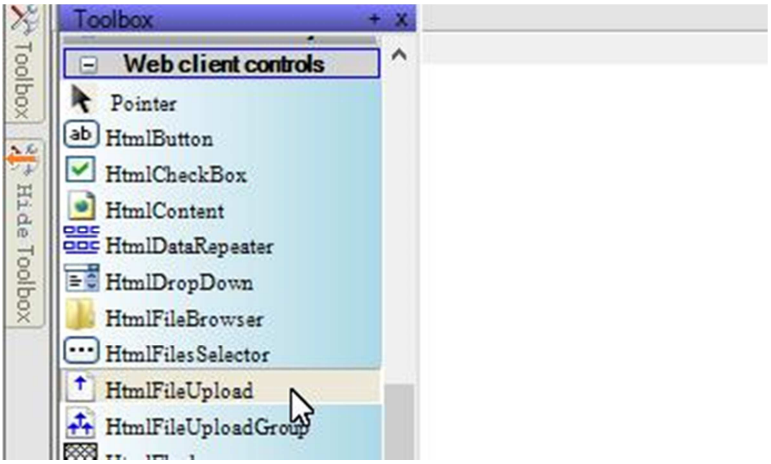
- Web form editor – it is very much like the form designer for standalone Windows Form application. Its look and feel is almost identical to the form designer. It adds components from the toolbox to the web page and manipulates the page components.
- Visual HTML editor – it is a visual HTML document editor. It combines word-processing, HTML elements manipulations, visual automatic CSS creations and other editing capabilities. It supports data-binding in the same way as the components added via the web form editor.

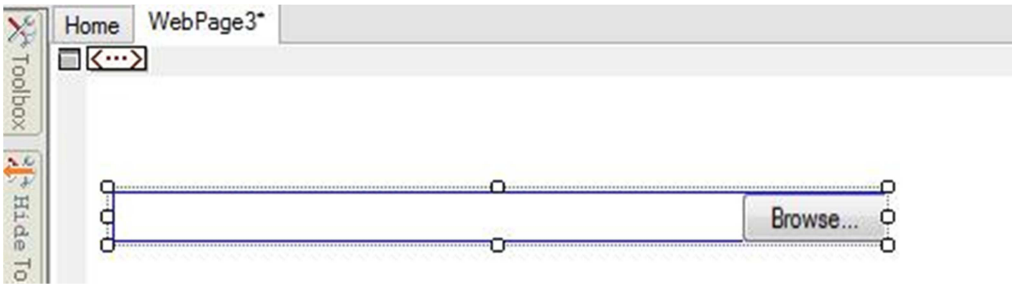
You may switch between the two editors. The results of the two editors are combined into the final web page.

Switch Web Editors

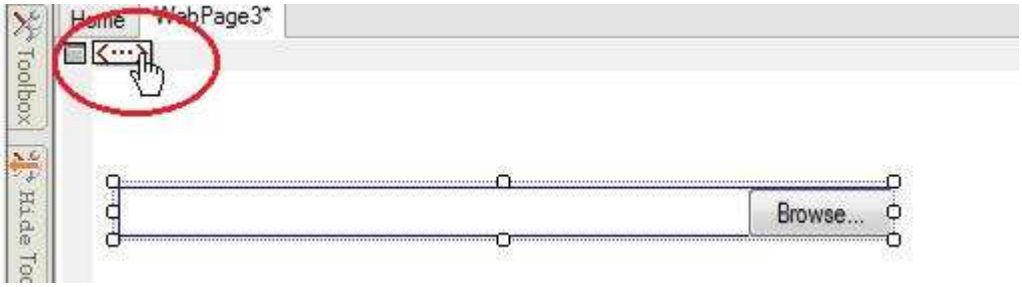
In Limnor Studio IDE, you may click  to switch to the Visual HTML Editor; you may click  to switch to the Web Form Editor.

Suppose we are in the Web Form Editor, and add a file upload component to the page:

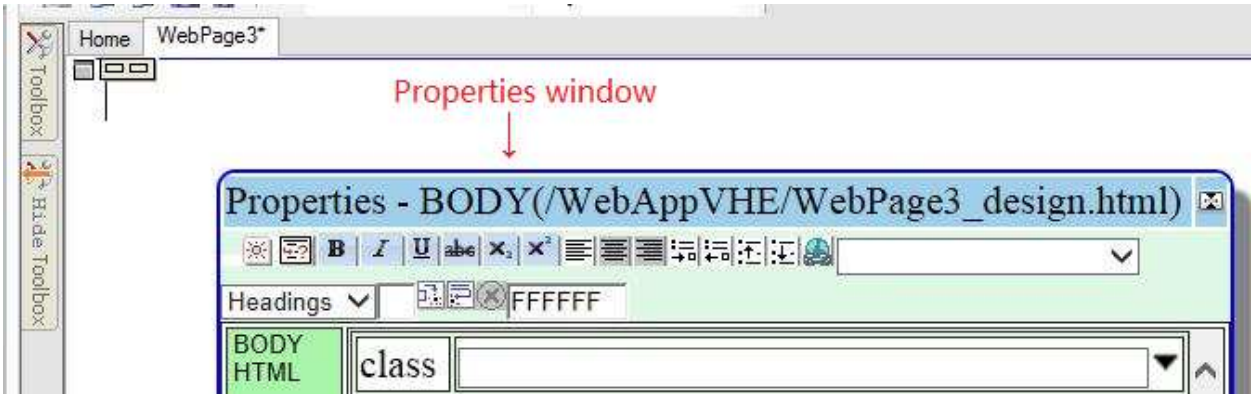




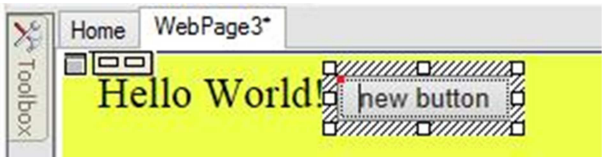
Now we click  to switch to the Visual HTML Editor:



The Visual HTML Editor replaces the Web Form Editor. Note that 1) a properties window appears; 2) the components added via the Web Form Editor are hidden, in this sample, the file upload component is hidden.

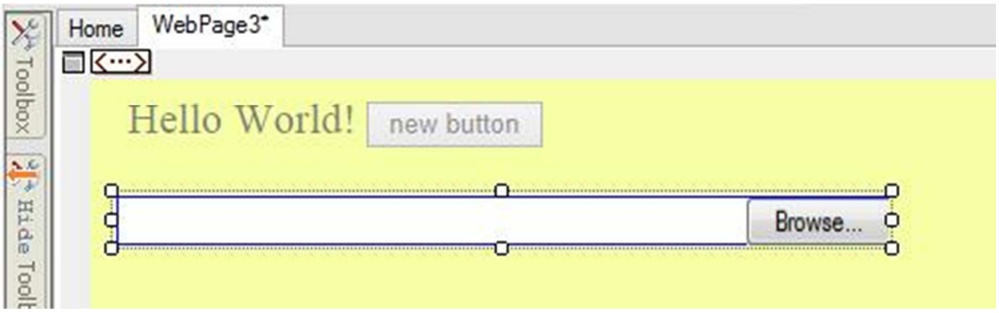


Let's make some web page editing. For example, type some words in the Visual HTML Editor, add a button and set the page background color:

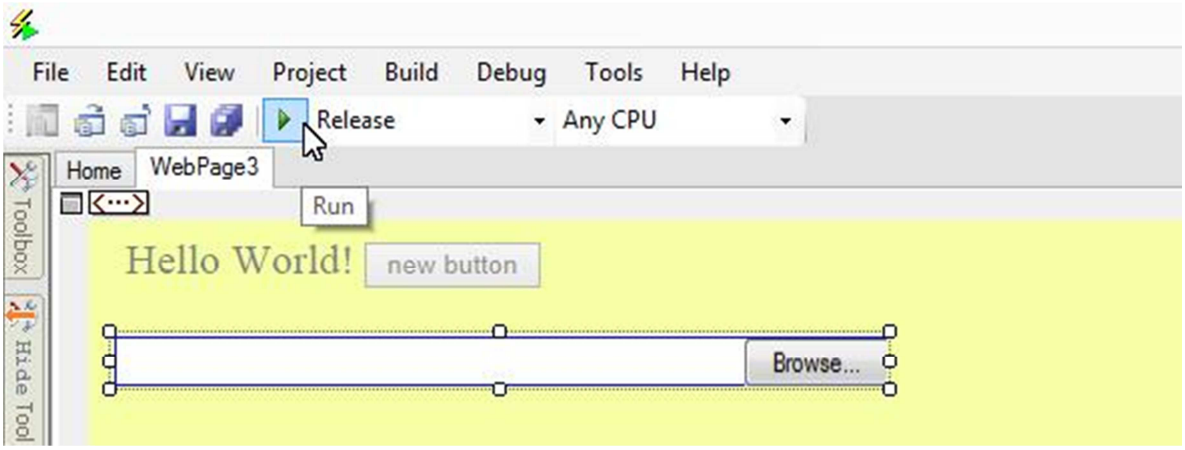


Now let's click  to switch to the Web Form Editor:

The Web Form Editor replaces the Visual HTML Editor. Note that 1) the components previously added via the Web Form Editor re-appear, in our sample, the file upload component appears; 2) the editing made via the Visual HTML Editor appears as dimmed background:



Click the Run button to test the web page.



We can see that contents added via both the Web Form Editor and the Visual HTML Editor appear.

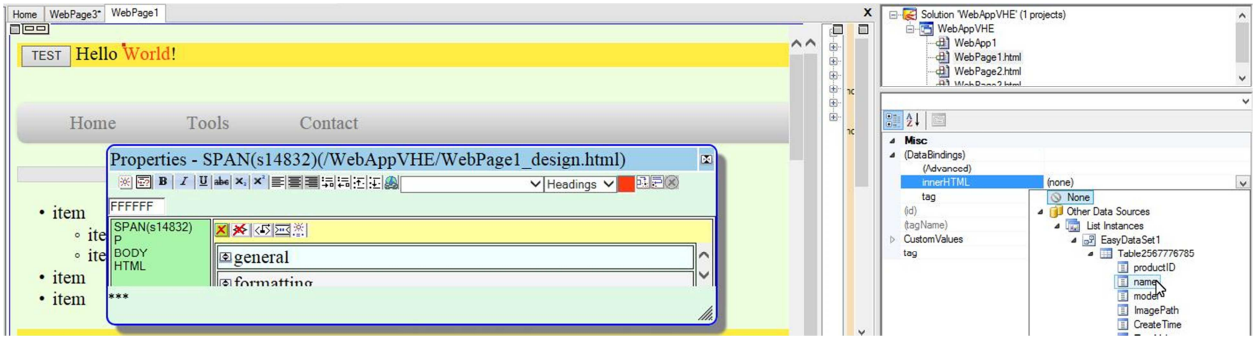
Use Visual HTML Editor

Users' Guide

See http://www.limnor.com/home1/AA/AA/htmlEditor.html for a Users' Guide of the Visual HTML Editor. The Users' Guide was completely created using the Visual HTML Editor via Web Home Service (http://www.limnor.com/home1/AA/AA/webhomehelp.html).

Data-binding

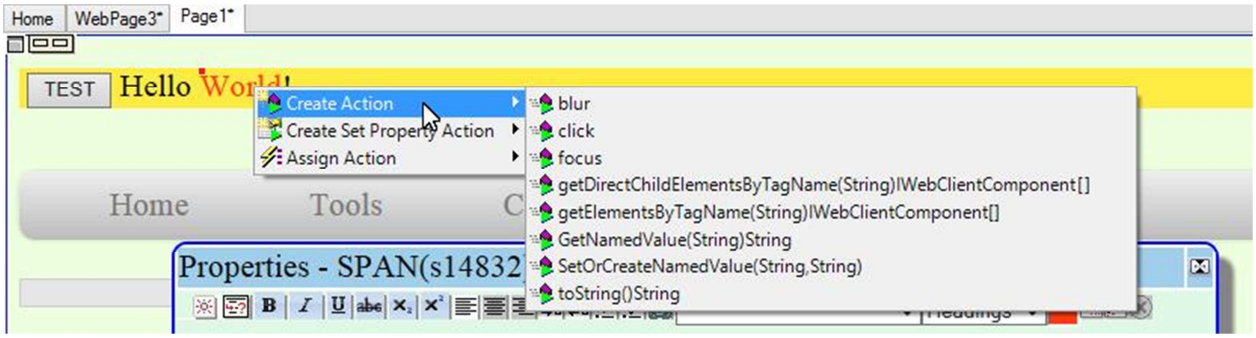
To set data-binding to elements, select an element, set its "DataBindings" property in the IDE:

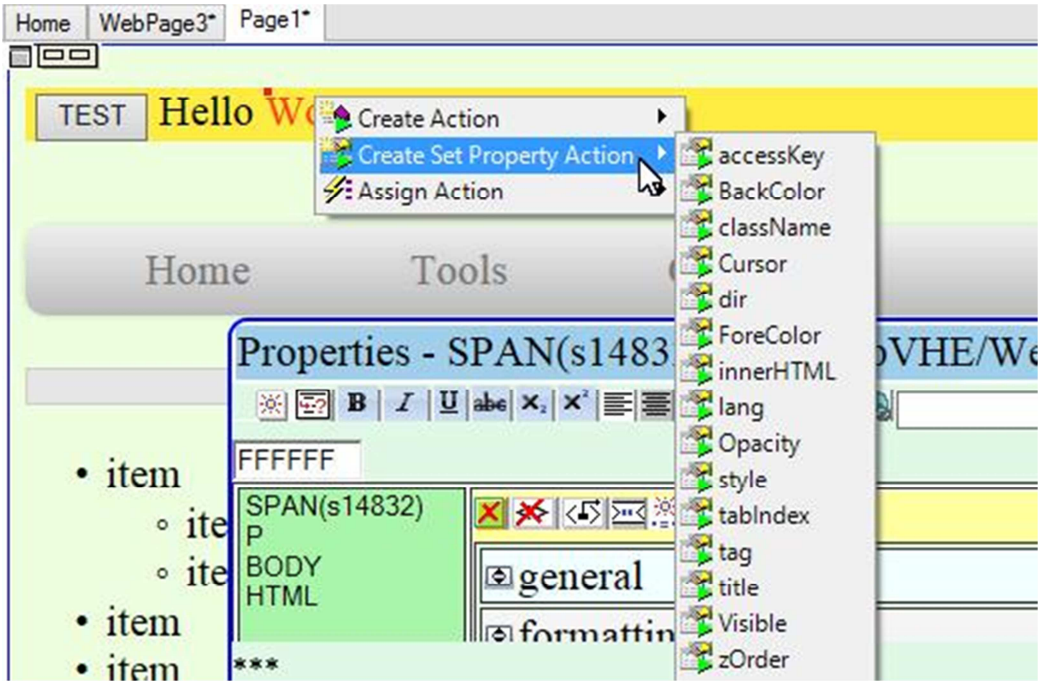


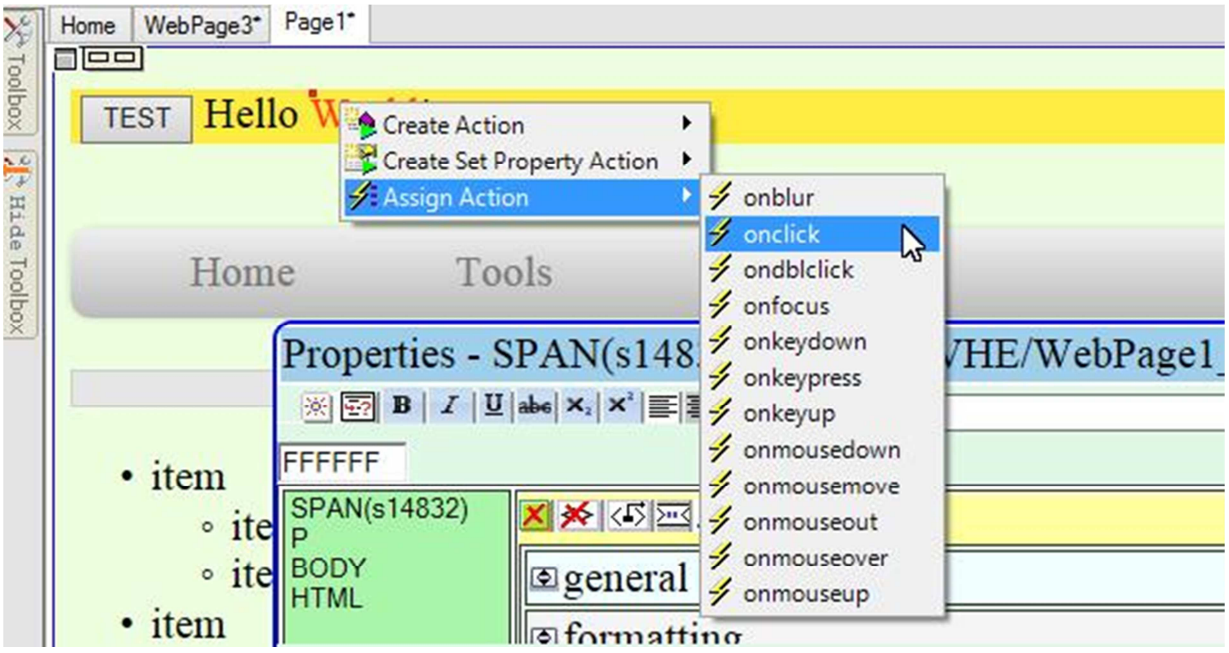
In the above page, the word "World" is set the red color, and thus belongs to a "SPAN" element. Clicking the world "World" selects the SPAN element. Its DataBindings property can be used to bind data to its InnerHtml and tag properties. All elements have a CustomValues property. You may create your own named values via CustomValues property. All named values you created will appear under DataBindings and can be data-bound. The data sources are EasyDataSet components. You must add EasyDataSet components in the Web Form Editor. See http://www.limnor.com/support/webDatabaseProgramming.pdf for database programming.

Element programming

Right-click an element in the web page, a context menu allows you to do programming for the element. If you already familiar with programming in Limnor Studio for other types of projects then you will see that it is the exact the same programming approach by creating method execution actions; creating property-setting actions; and creating event-actions assignments.


Home
WebPage3*
Page1*
TEST
Hello World!
Create Action
Create Set Property Action
Assign Action
blur
click
focus
getDirectChildElementsByTagName(String)IWebClientComponent[]
getElementsByTagName(String)IWebClientComponent[]
GetNamedValue(String)String
SetOrCreateNamedValue(String,String)
toString()String
Home
Tools
Properties - SPAN(s14832


Home
WebPage3*
Page1*
TEST
Create Action
Create Set Property Action
Assign Action
accessKey
BackColor
className
Cursor
dir
ForeColor
innerHTML
lang
Opacity
style
tabIndex
tag
title
Visible
zOrder
Home
Tools
Properties - SPAN(s1483
FFFFFF
SPAN(s14832)
P
BODY
HTML
general
• item
• item
• item


Toolbox
Hide Toolbox
Home
WebPage3*
Page1*
TEST
Create Action
Create Set Property Action
Assign Action
onblur
onclick
ondblclick
onfocus
onkeydown
onkeypress
onkeyup
onmousedown
onmousemove
onmouseout
onmouseover
onmouseup
Home
Tools
Properties - SPAN(s148
FFFFFF
SPAN(s14832)
P
BODY
HTML
general
formatting
• item
• item
• item

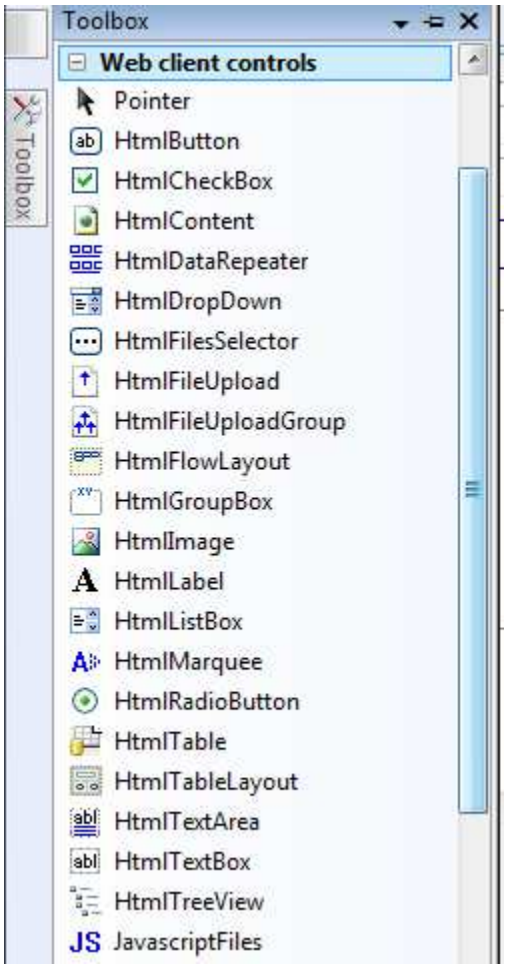
The above example, since it is right-clicked on the red word "World", the programming is done for the underlying element which is a SPAN element.

A section below shows a sample of creating an action and assigning the action to an event. The sample uses a component added via the Web Form Editor. But the process is the same for components added via the Visual HTML Editor.

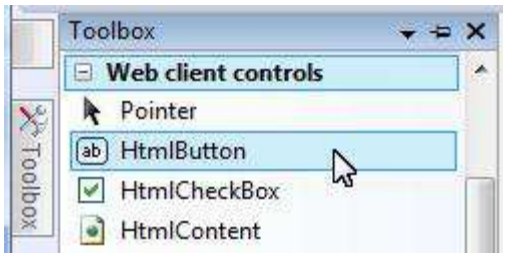
Use Web Form Editor

Visual/Client Components

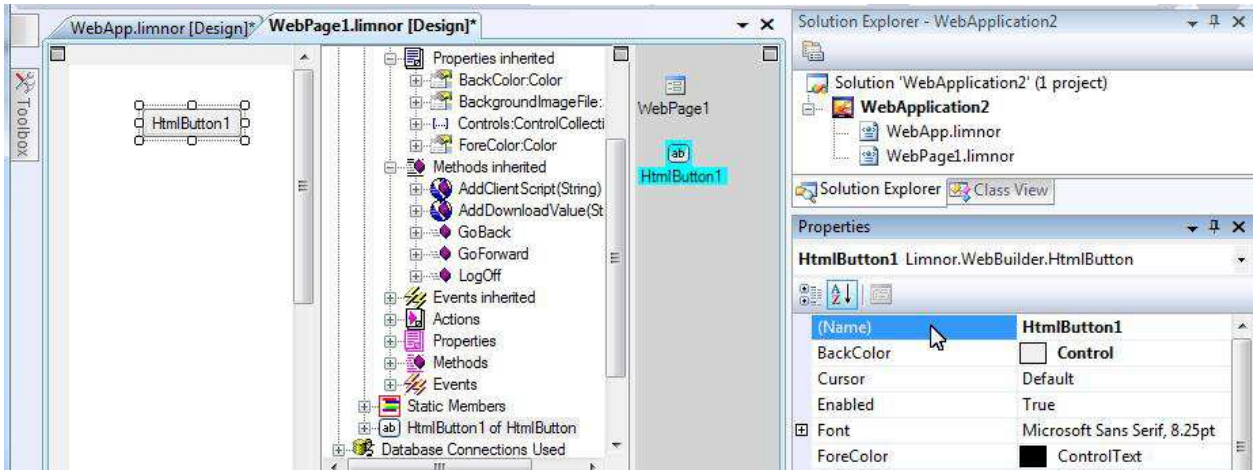
Visual components listed in the "Web client controls" can be added to a web page for constructing GUI:



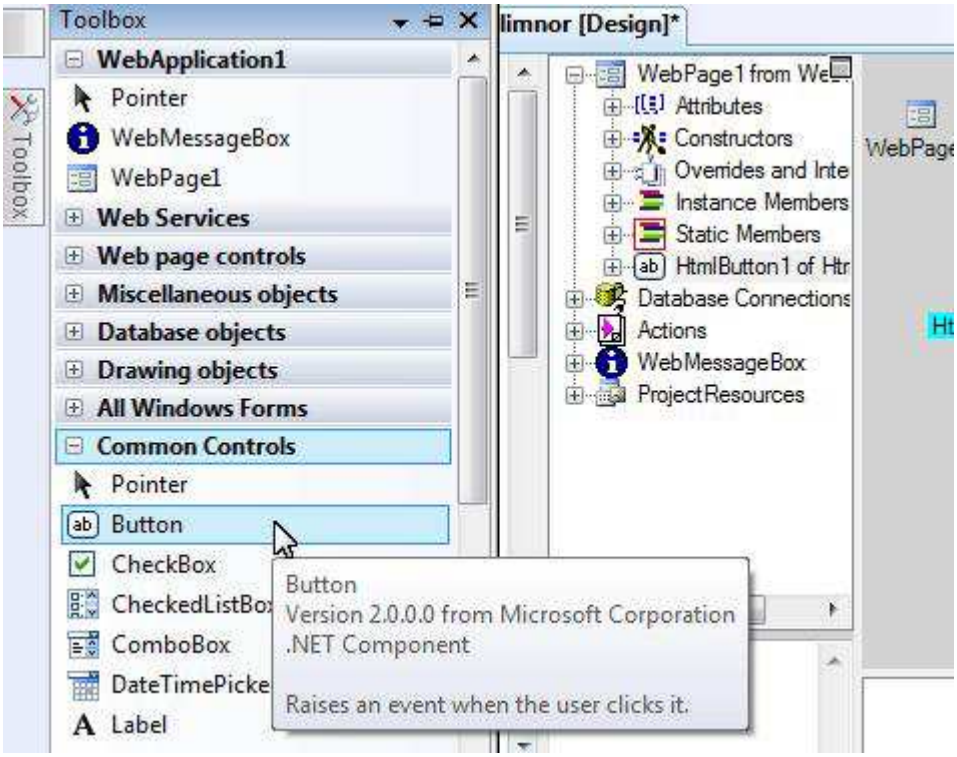
For example, we may add an HtmlButton to the web page:



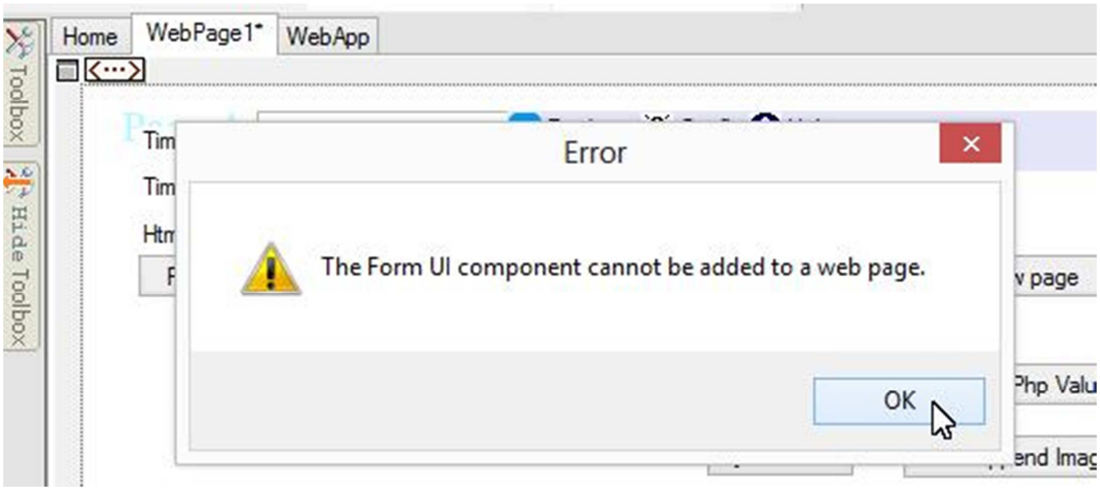
Drop HtmlButton to the web page:



Do not add visual components for standalone applications to a web page. For example, try drop a Microsoft .Net Button to the web page:

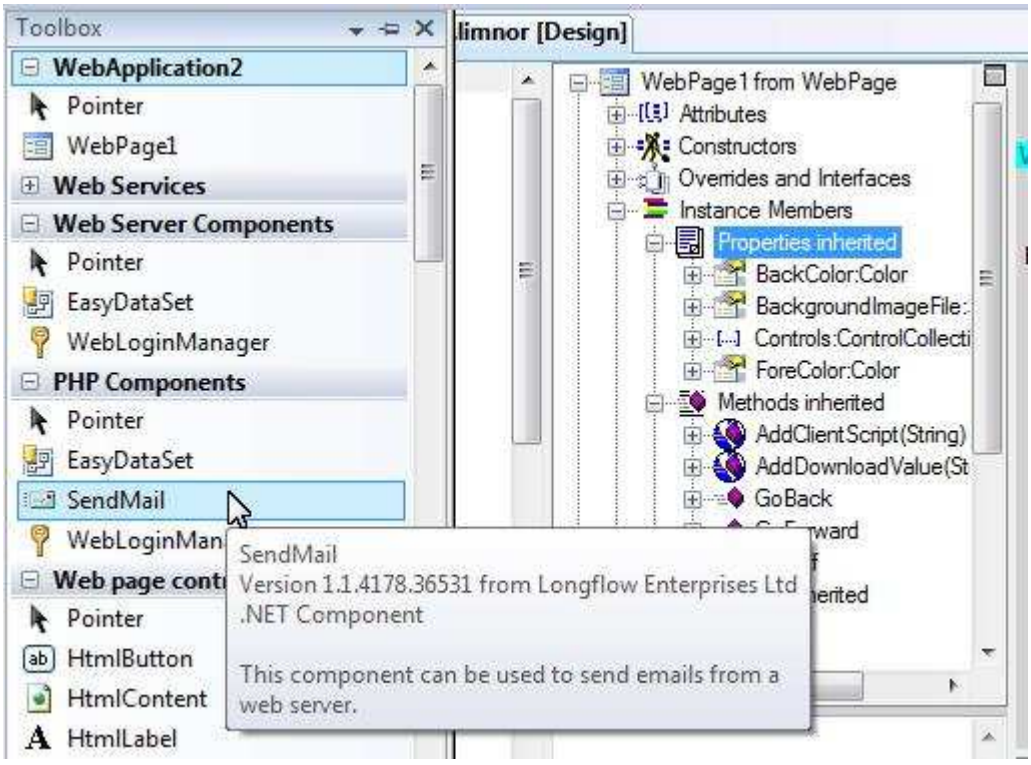


An error message box appears:

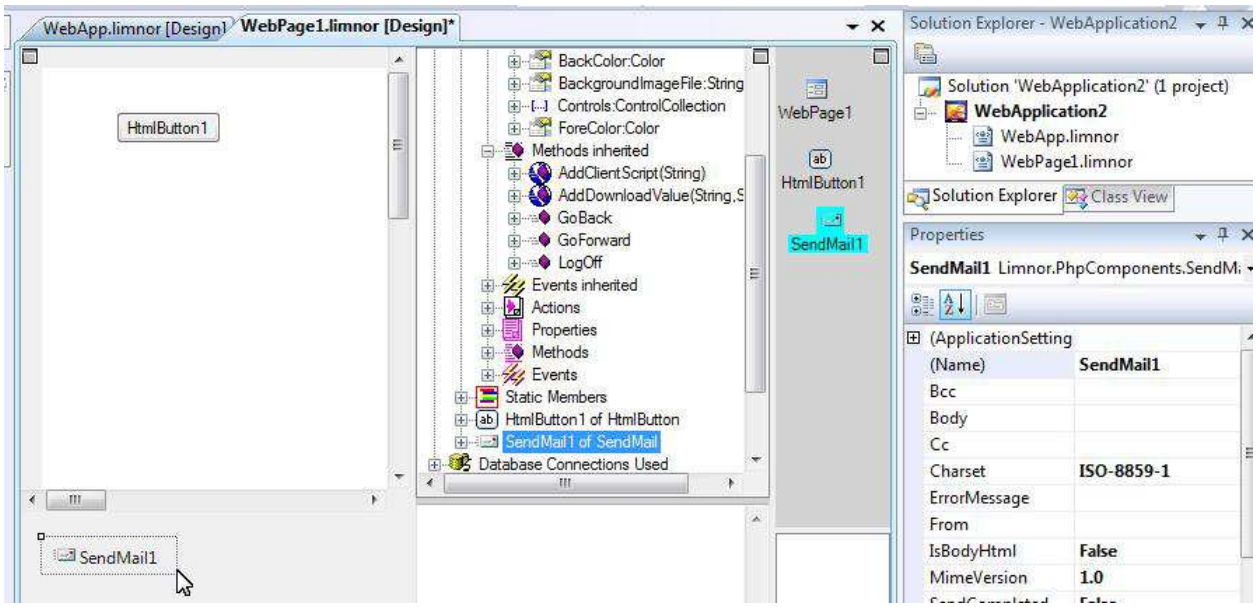


Server Components

Non-visual components can be added to the web page. Such components usually run at the web server. For example, we may add a SendMail component to the web page:



A SendMail instance is created in the web page:



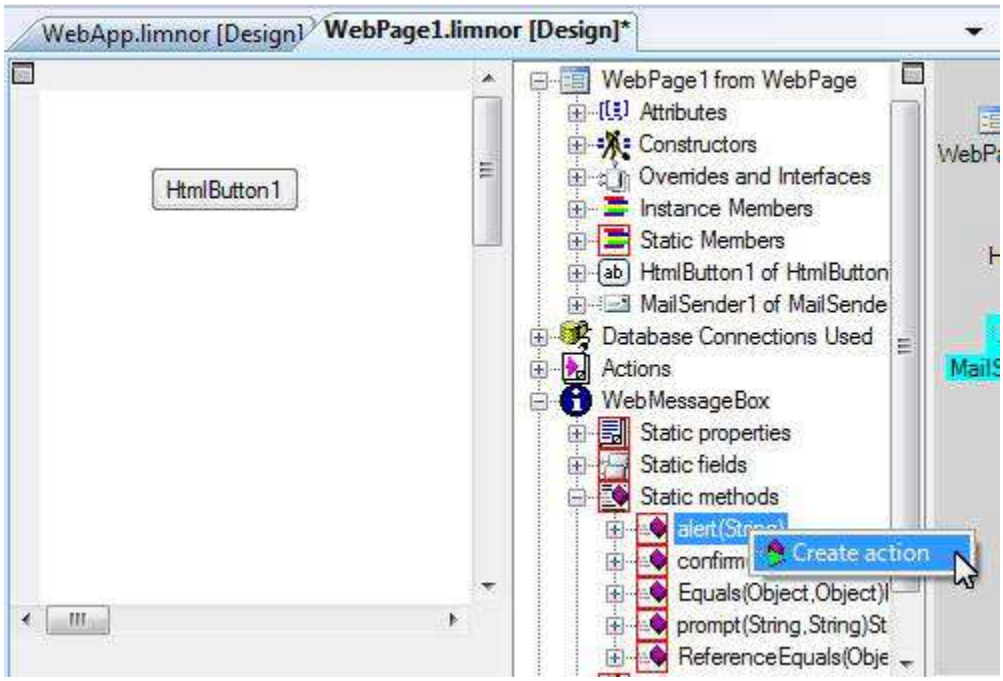
We call such components server components.

Create Action and Assign Action to Event

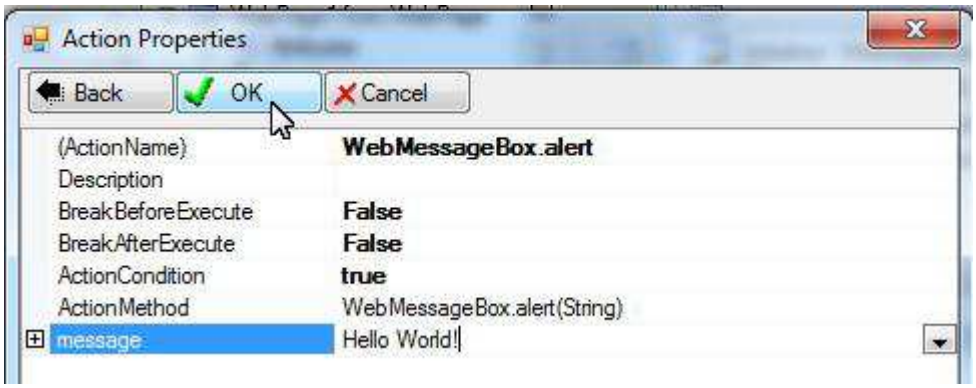
We use a message box to show the process. Suppose that we want to show "Hello World!" when the user clicks the button.

Create an alert action

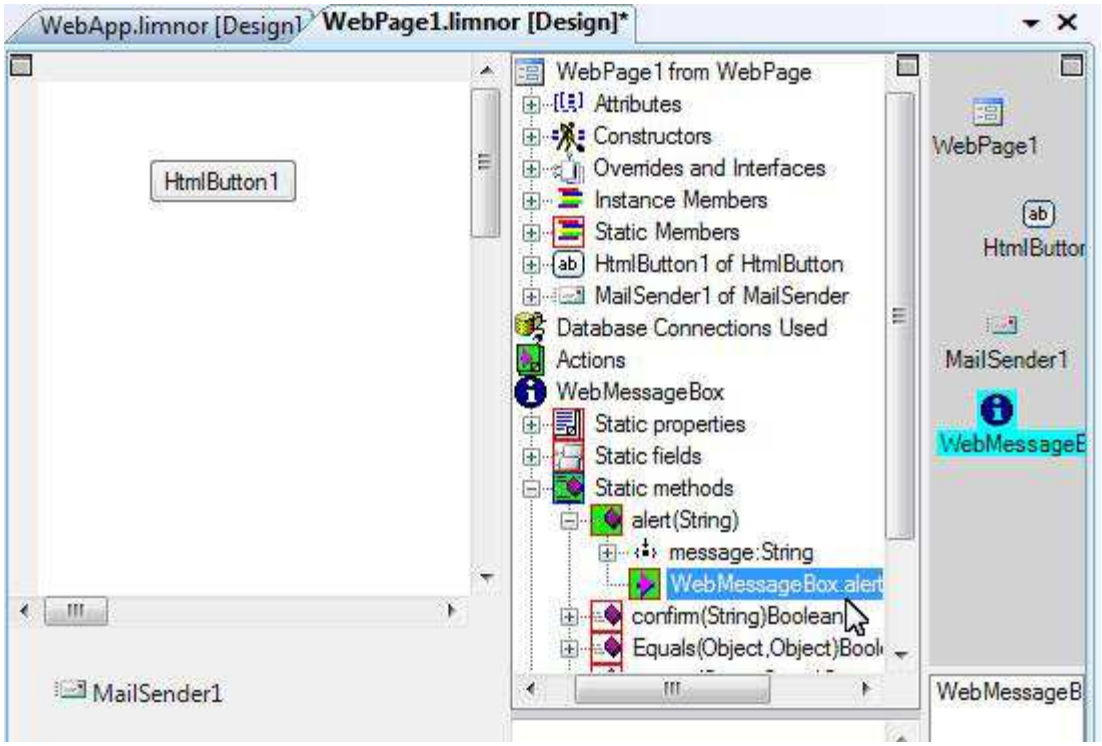
Right-click the "alert" method of the WebMessageBox class, choose "Create action":



Set message to Hello World!

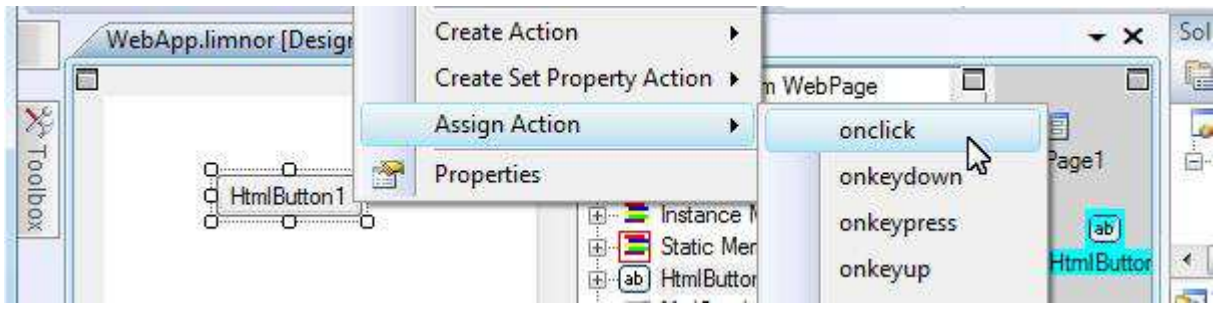


The action appears under the method:



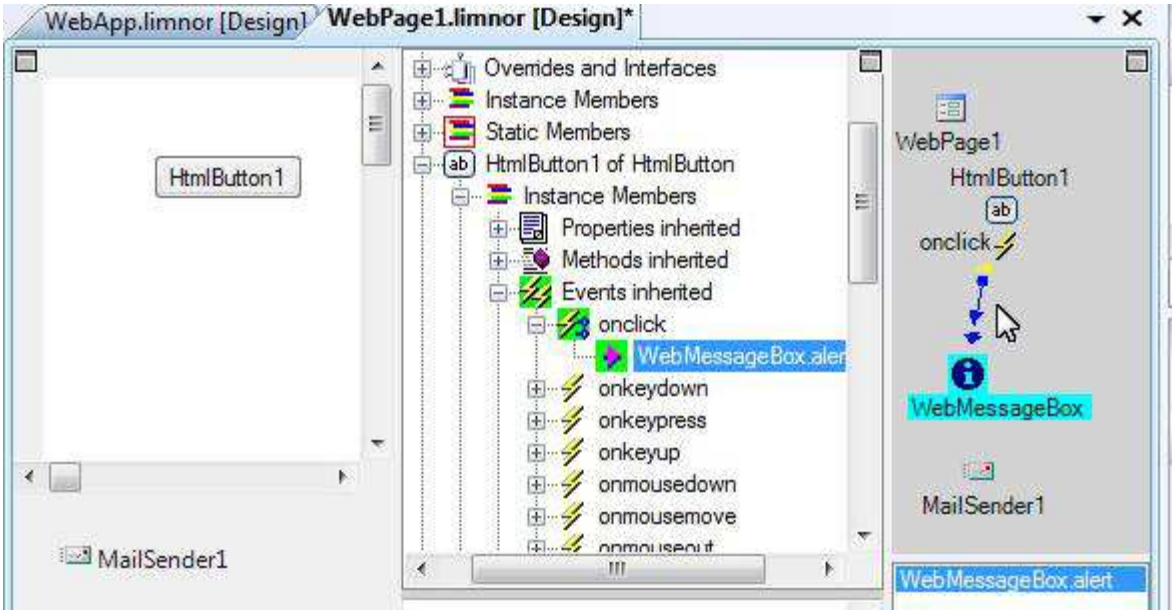
Assign action to event

Right-click the button; choose "Assign Action"; choose "onclick" event:



Select the action and click Next:

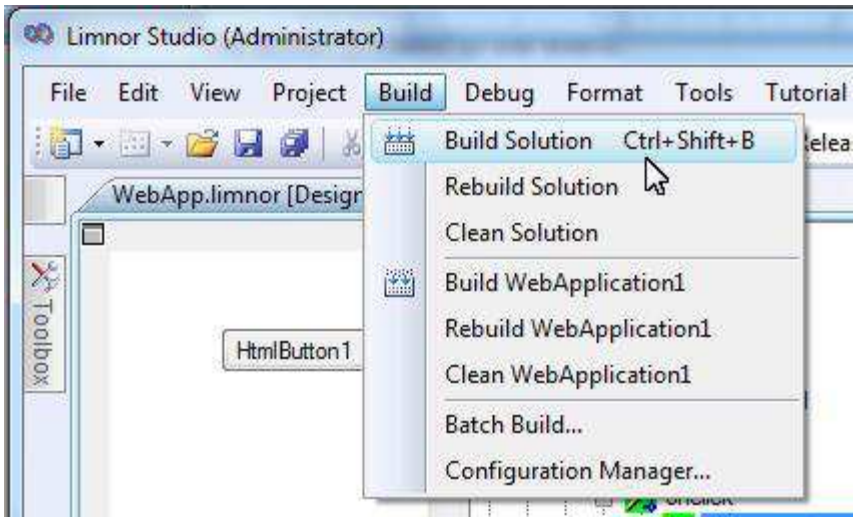
The action is linked to the event:



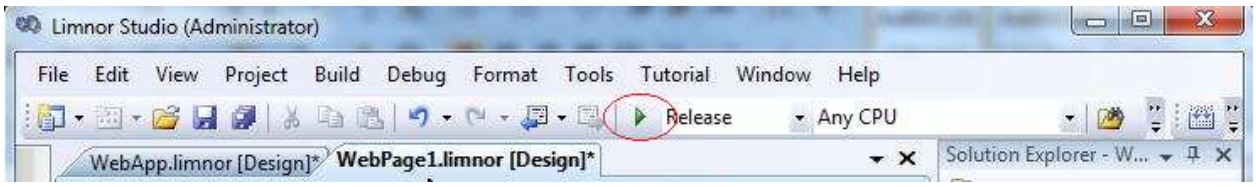
Compile and Test

Compile

If you just want to compile the project without test run then choose a Build menu:



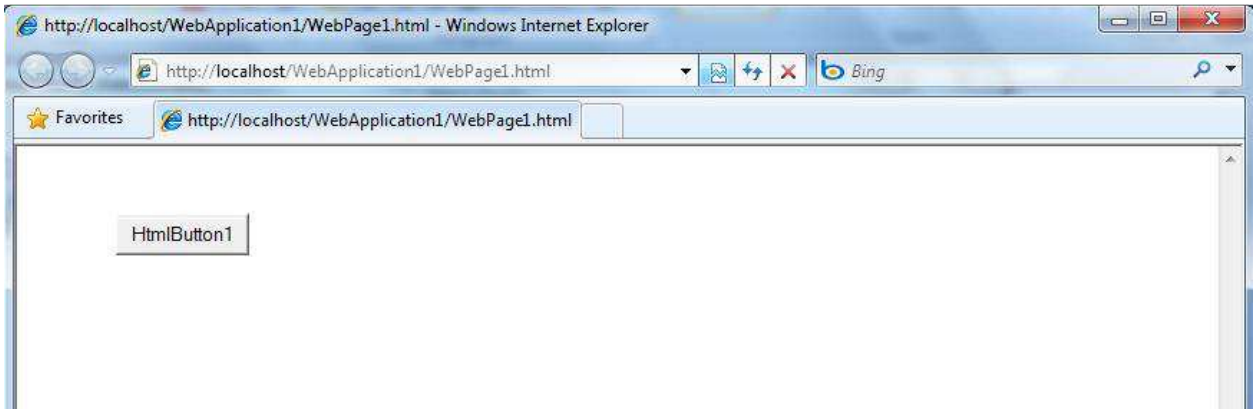
If you want to test run then click Run button . It will start compiling the project. If the compiling is successful then your default web browser will open the StartPage:



Note that clicking will only compile modified web pages. If you modified database connection, culture resources, and other global attributes then you need to select a build menu to re-compile every web page.

Test

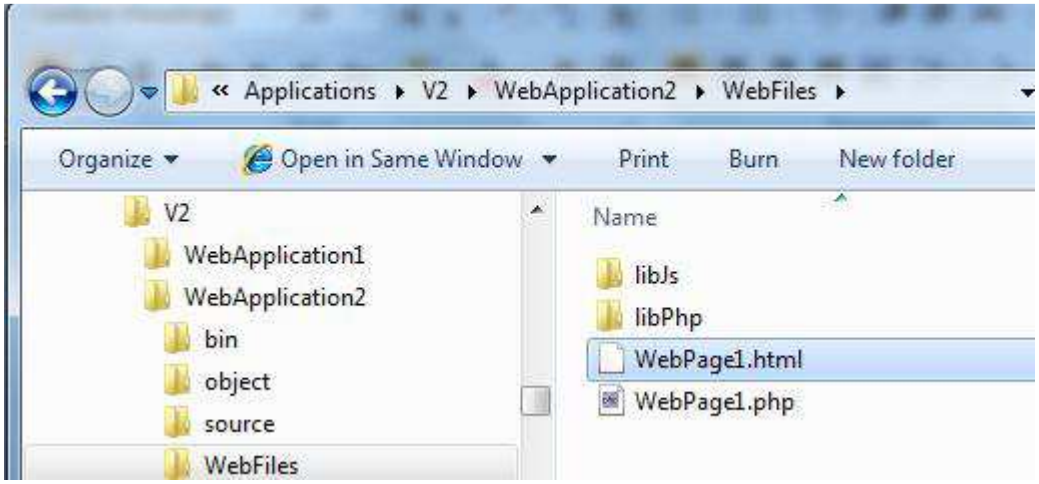
After compilation WebPage1 shows in the web browser:



Click the button, "Hello World!" appears:

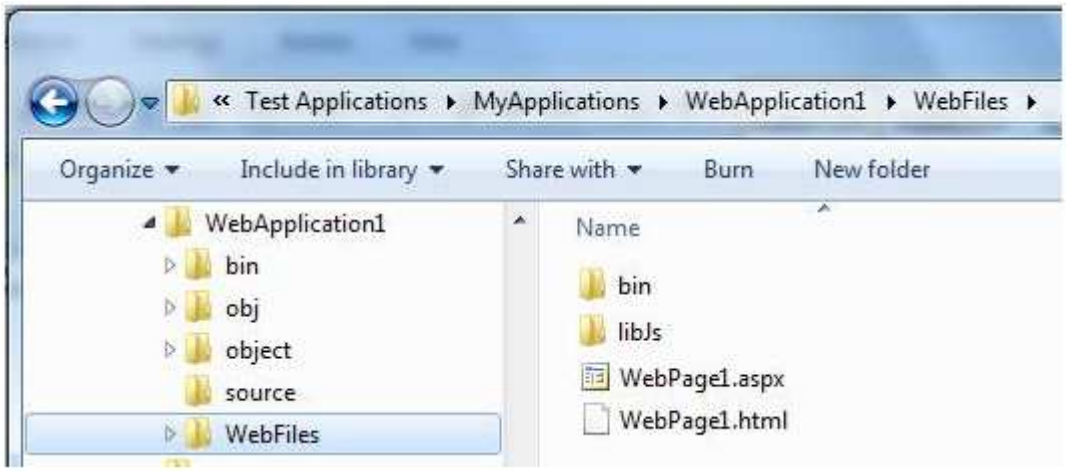
Compilation Results and Publish

The compilation results are saved in a folder named "webfiles" under the project folder:



To publish the web application, upload the files under WebFiles, including the sub-folders, to your web server.

If Web Application (.Net) is used then files will also be generated under the WebFiles folder:



To publish this web application, upload all files and folders in the WebFiles folder to your web server.

Internet and Web Applications

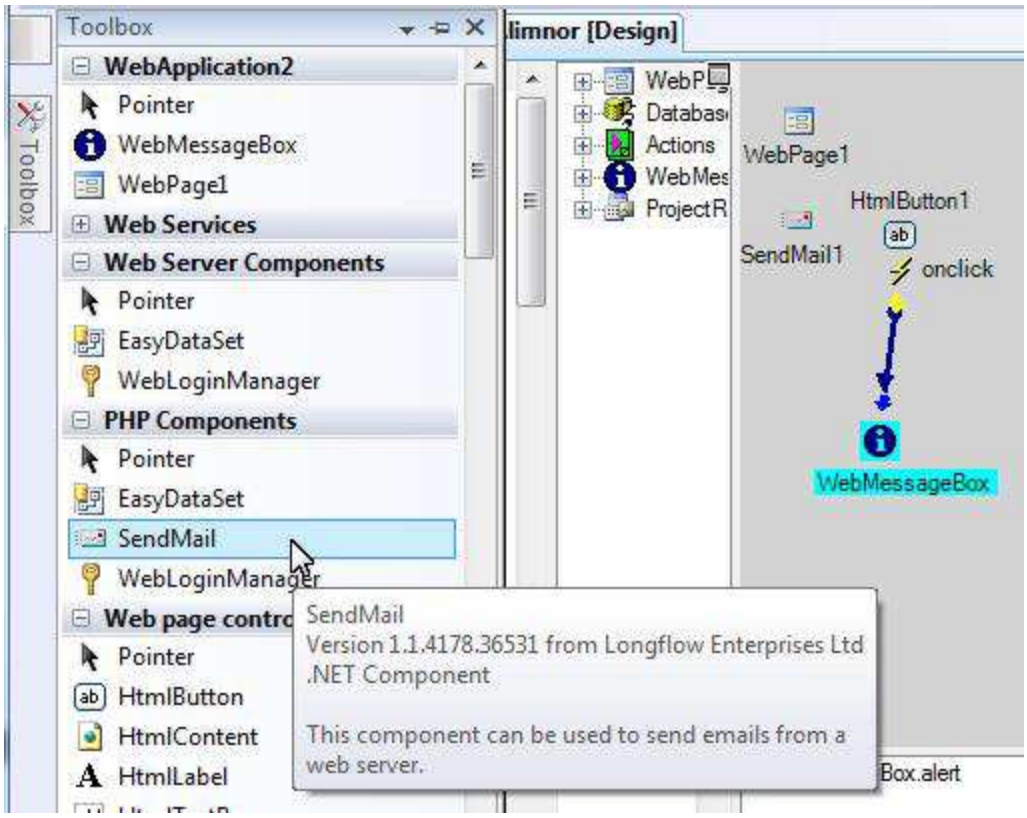
A web application runs at a web browser and a web server. The part that runs at a web browser is the client part. The part that runs at a web server is the server part.

User interface is the client part. Most other classes are server part.

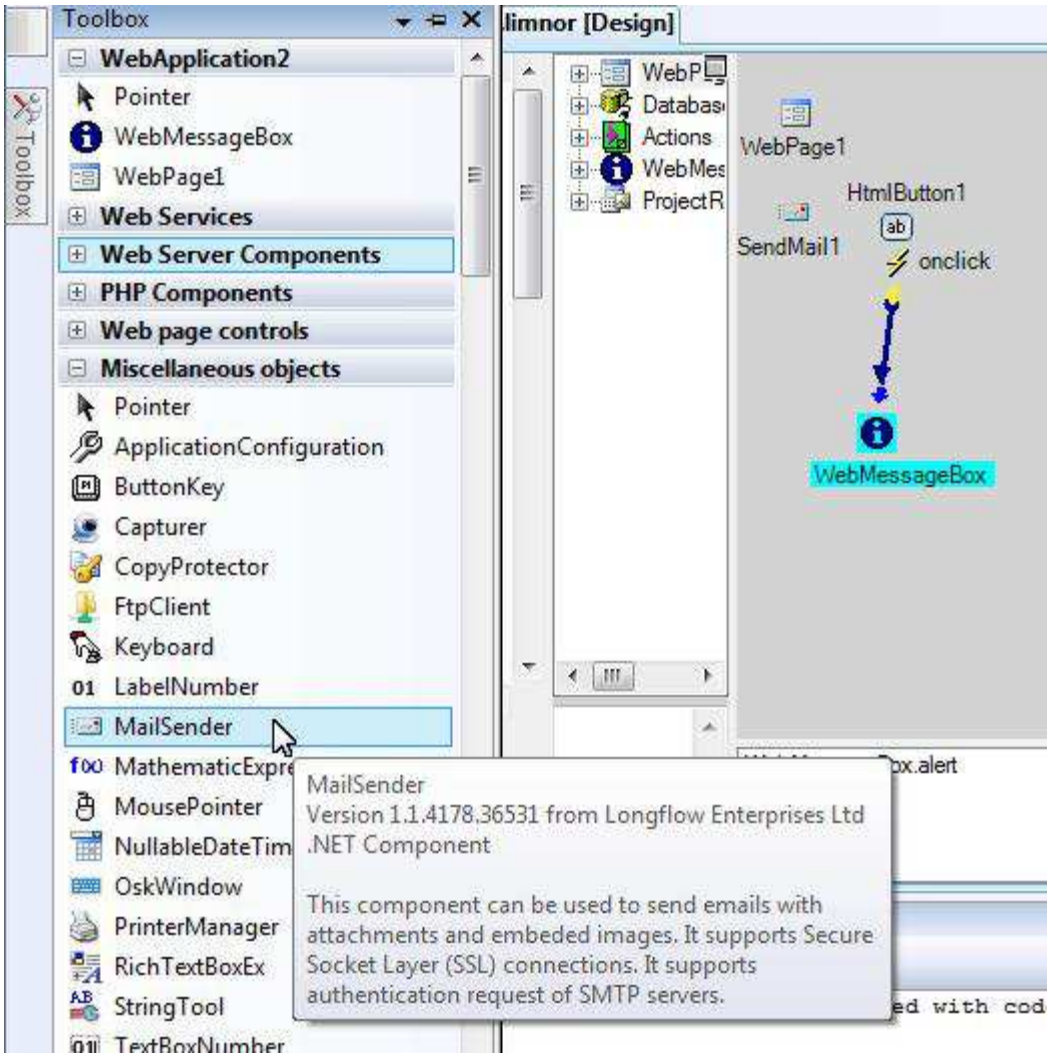
Although Limnor Studio handles client and server parts for you, it helps to know some basic operations behind the scene.

Let's use a simple web mail sample to illustrate how client side and server side work together in a web application.

If you are developing a PHP web application then you may use SendMail component under PHP Components:



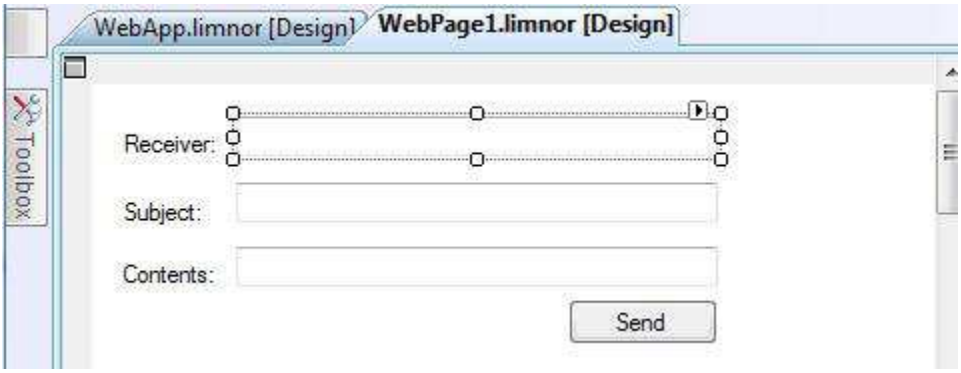
If you are developing a .Net web application then you may use MailSender:



Below we will develop a .Net web application and use MailSender to make the sample.

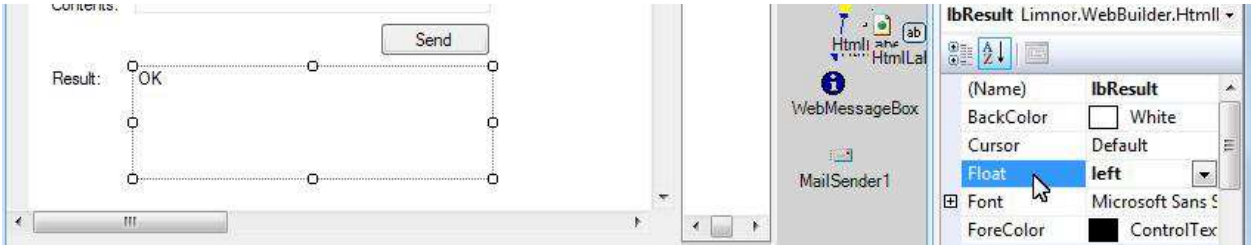
Web interface sample

We use HtmlTextBox components to allow the user to enter email contents:



We use an HtmlLabel to show email sending result. If the email sending succeeds then we want it to show OK; if the email sending fails then we want it to show the error message.

We set the Float property of the HtmlLabel to "left" so that the label can be resized:



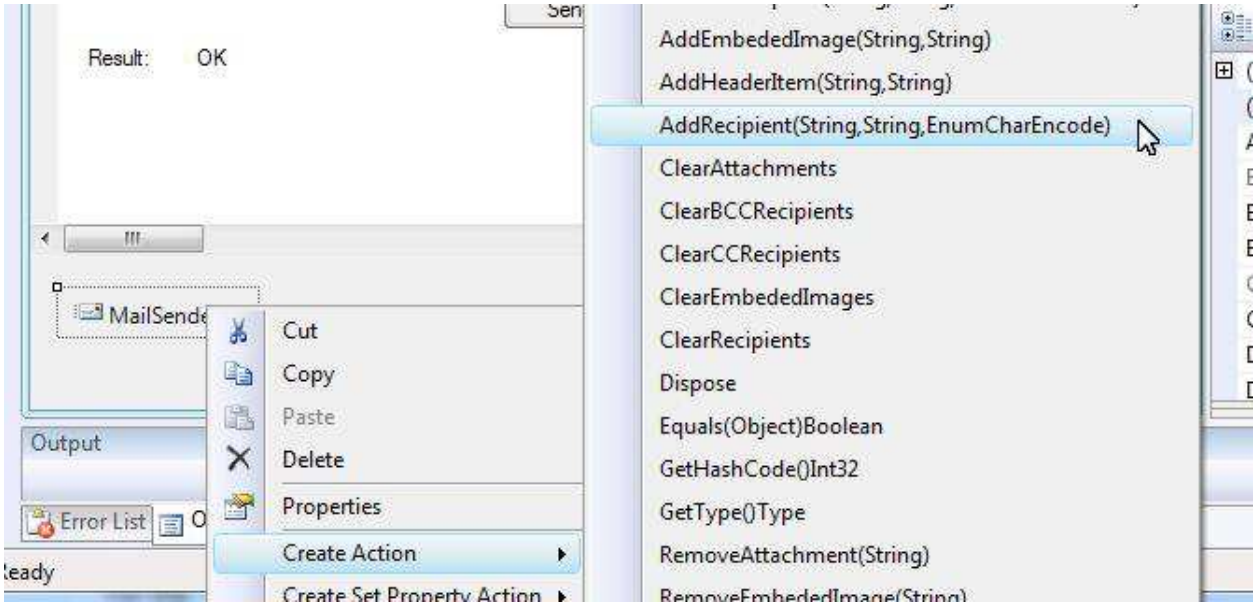
Examples of making actions

We may use a MailSender component to send emails. We may create actions to set the properties of the MailSender component to the values entered by the user. We may create a Send action to send the email. We may assign the ErrorMessage property of the MailSender to the label on the web page.

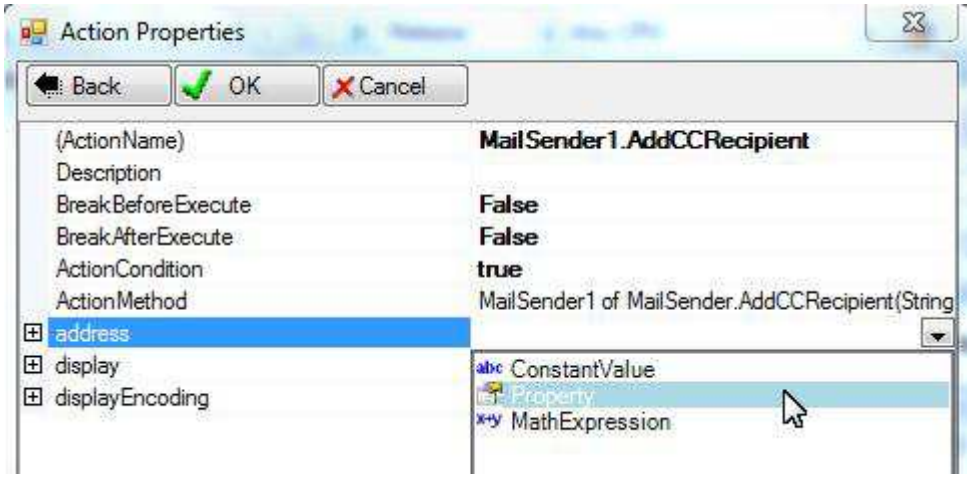
Below, we will create actions to accomplish the tasks outlined above.

Action - Set email receiver

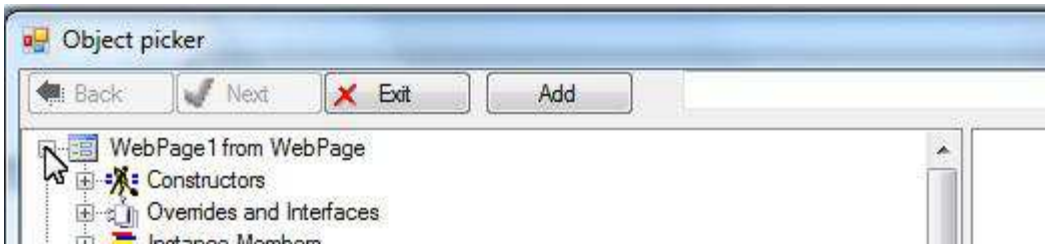
Right-click the MailSender; choose "Create Action"; choose "AddRecipient":

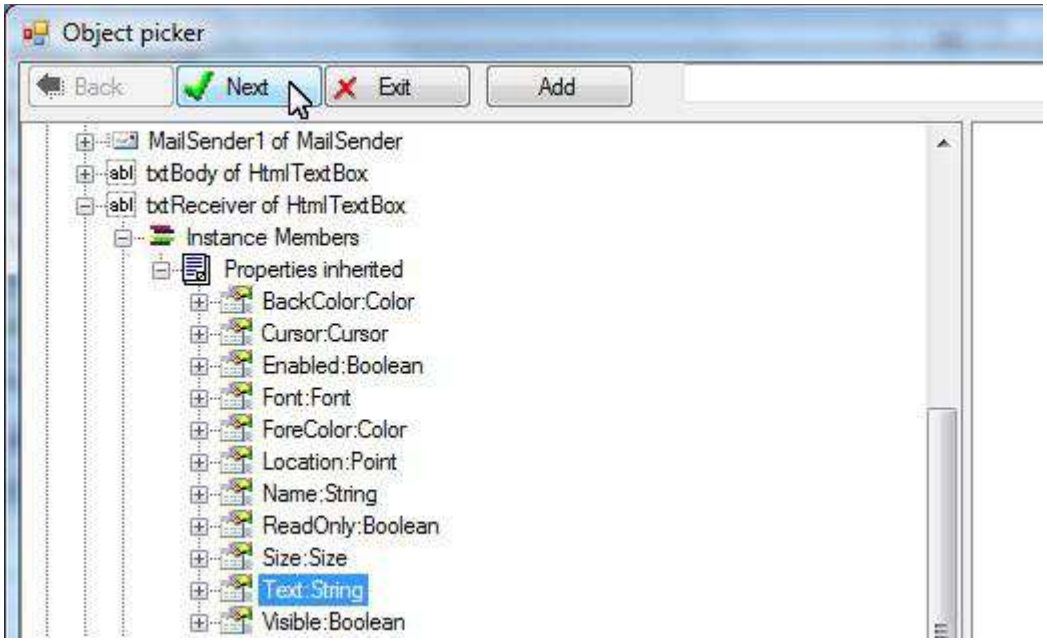


For the "address" parameter of the action, select "Property":

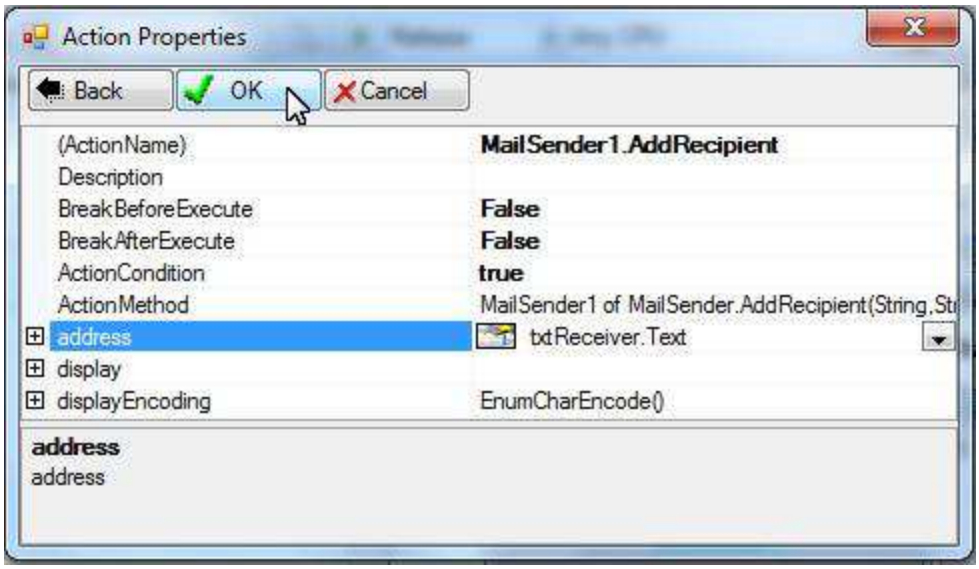


Select the Text property of the receiver text box:



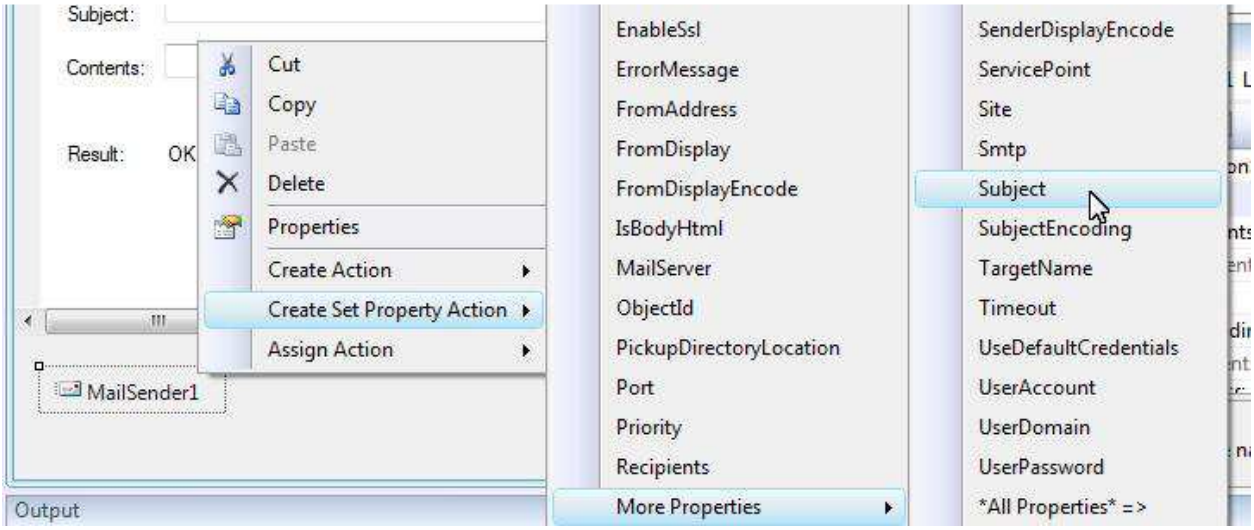


Click OK to finish creating this action:

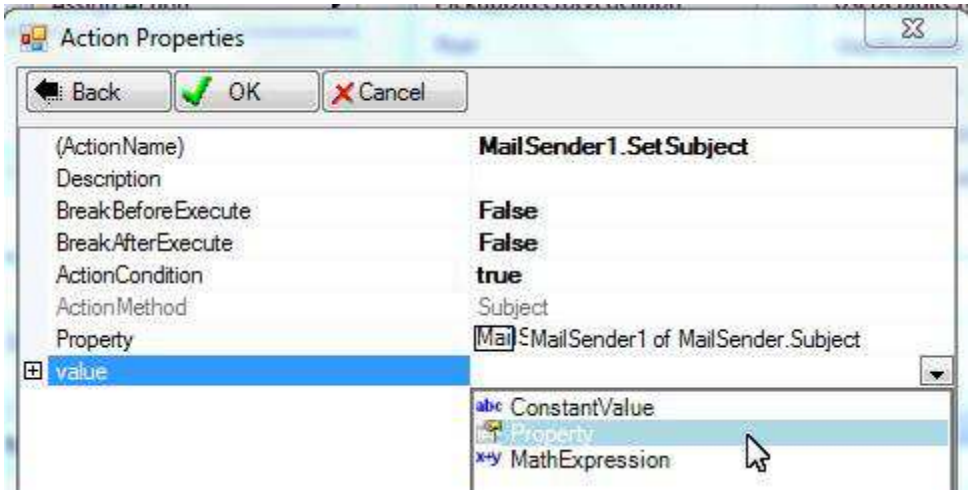


Action - Set email subject

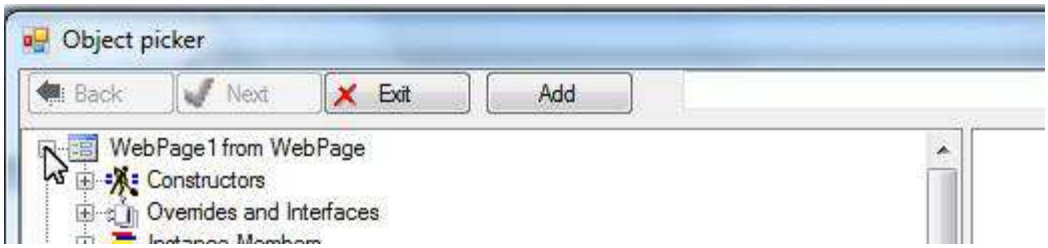
Right-click the MailSender; choose "Create Set Property Action"; choose "Subject" property:

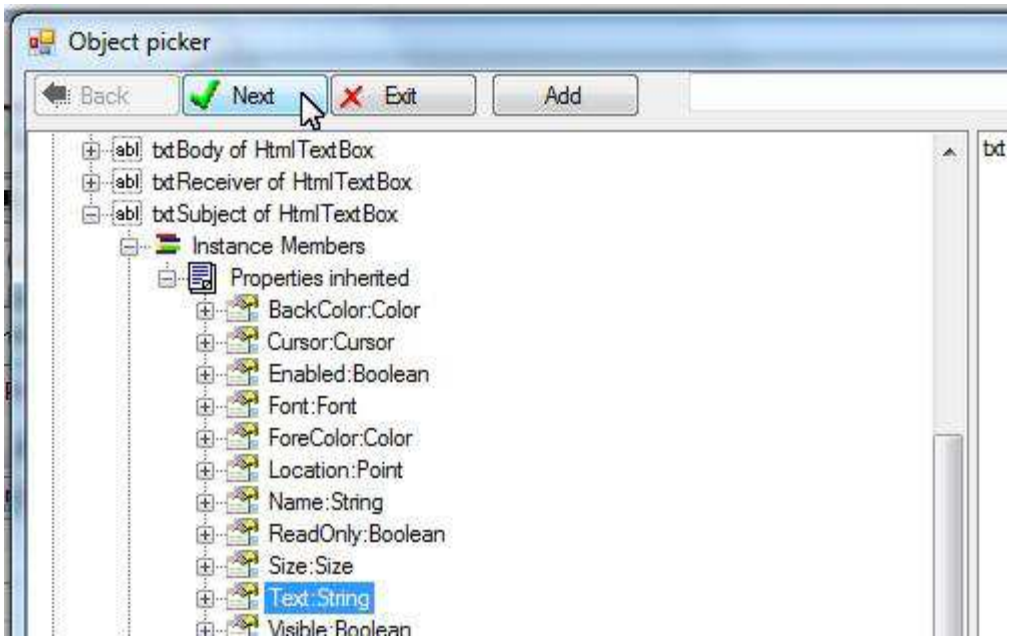


For the "value" parameter of the action, select "Property":

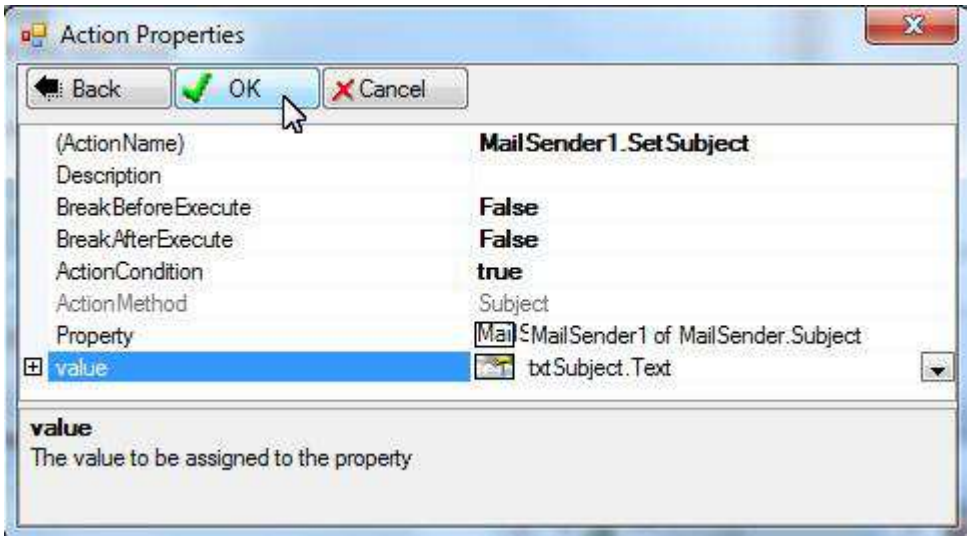


Select the Text property of the "Subject" text box:



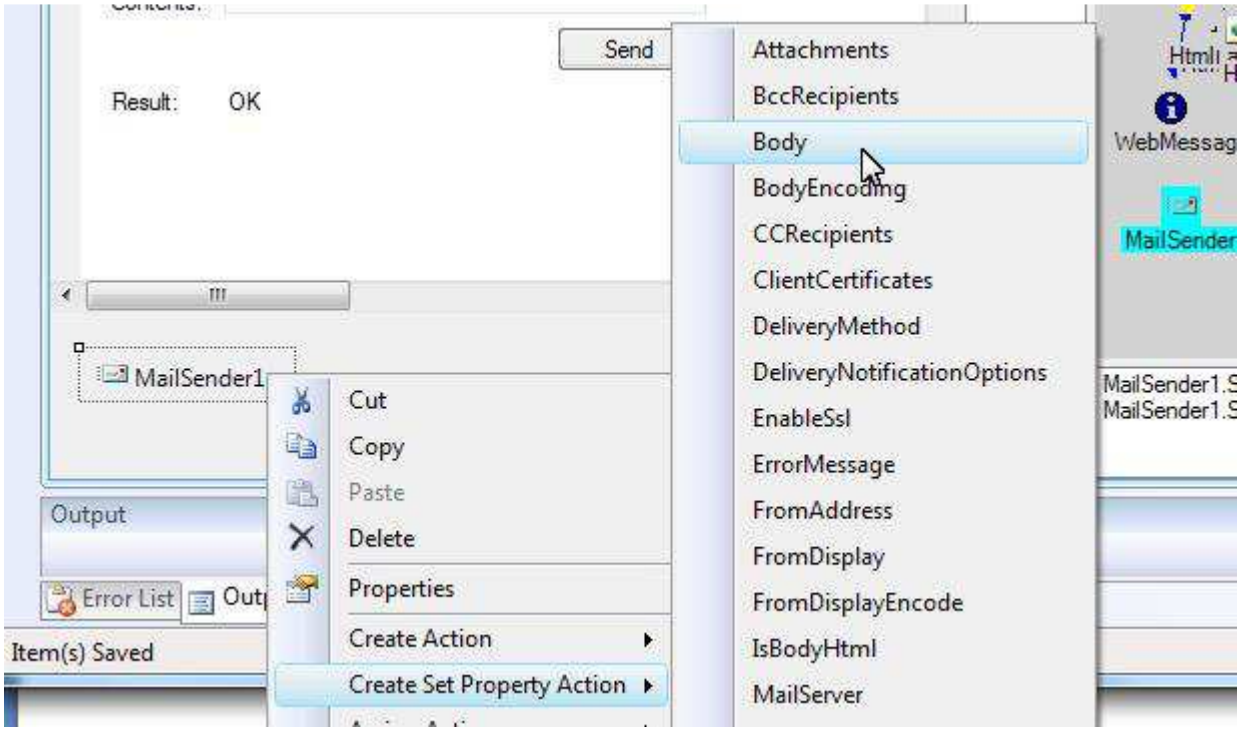


Click OK to finish creating this action:

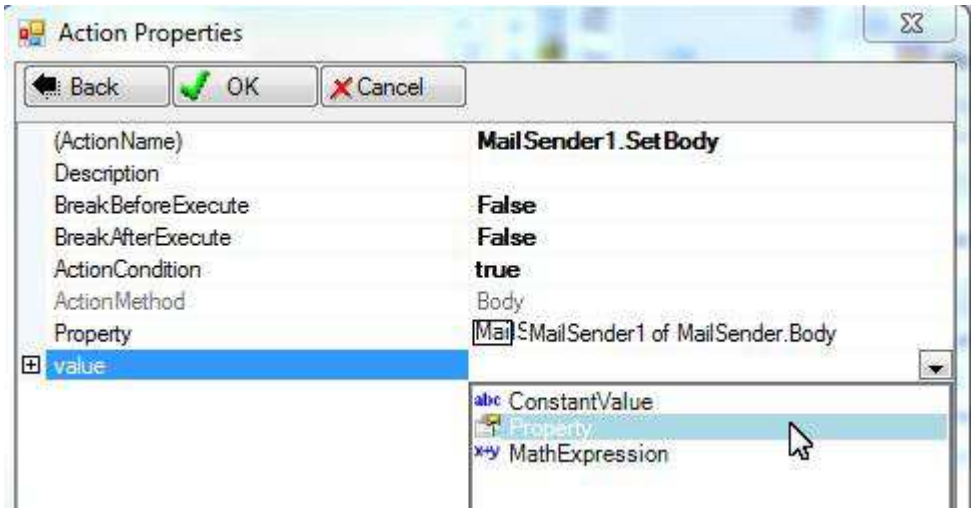


Action - Set email body

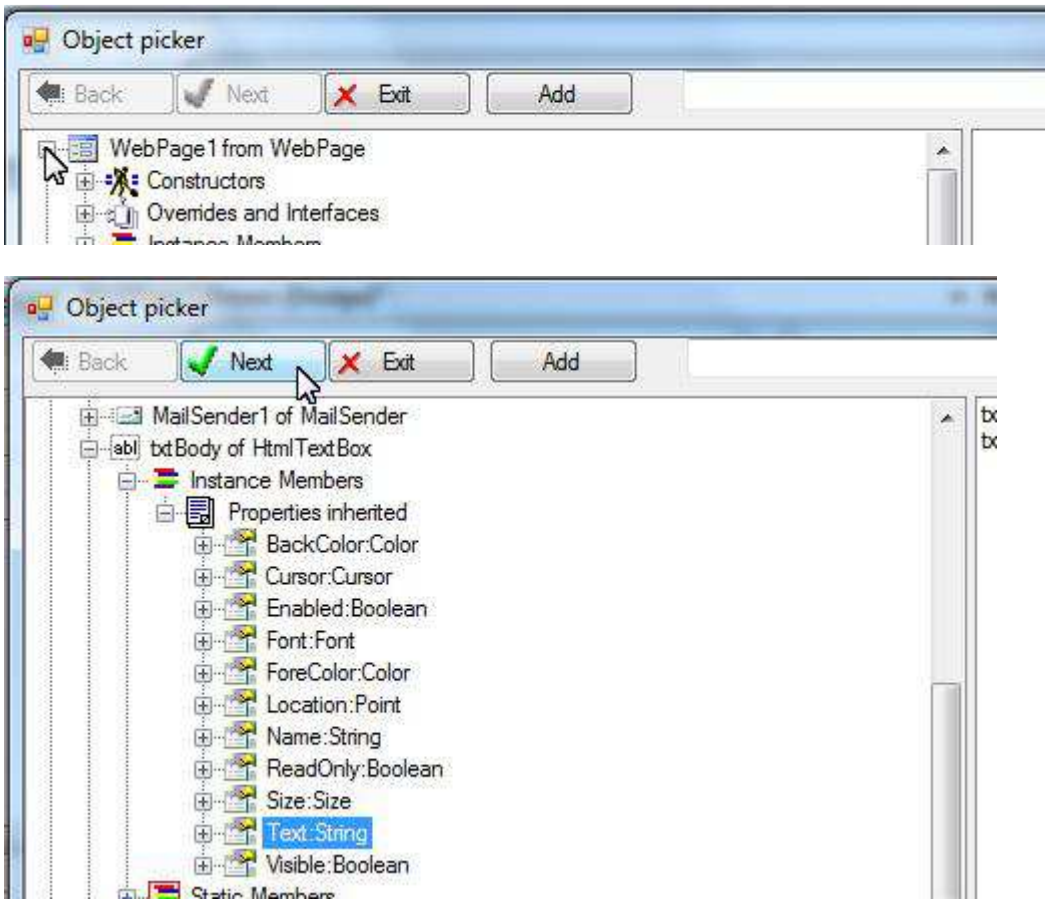
Right-click the MailSender; choose "Create Set Property Action"; choose "Body" property:



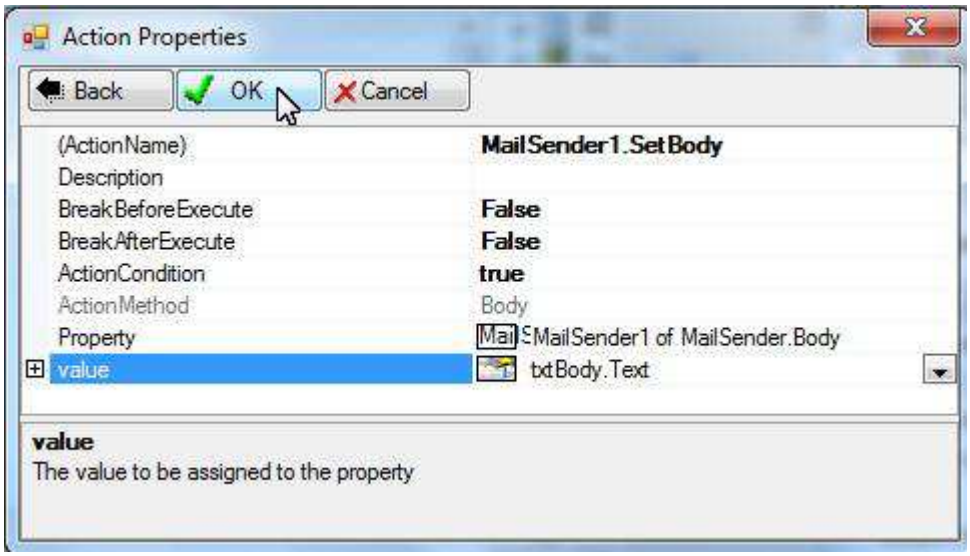
For the "value" parameter of the action, select "Property":



Select the Text property of the "Body" text box:

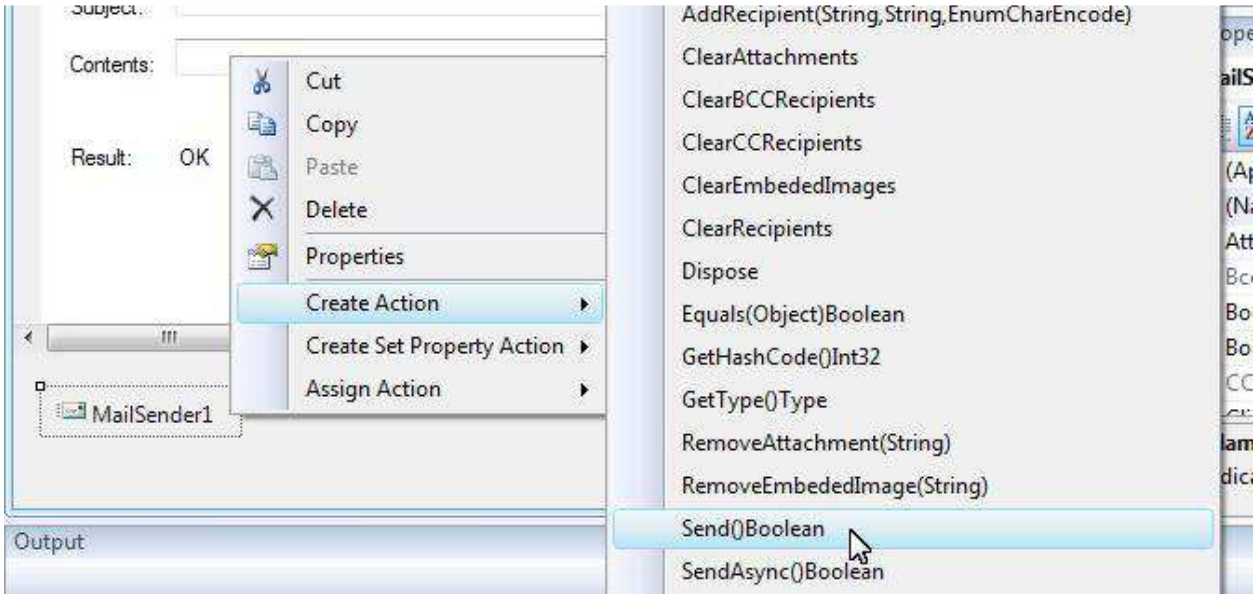


Click OK to finish creating this action:

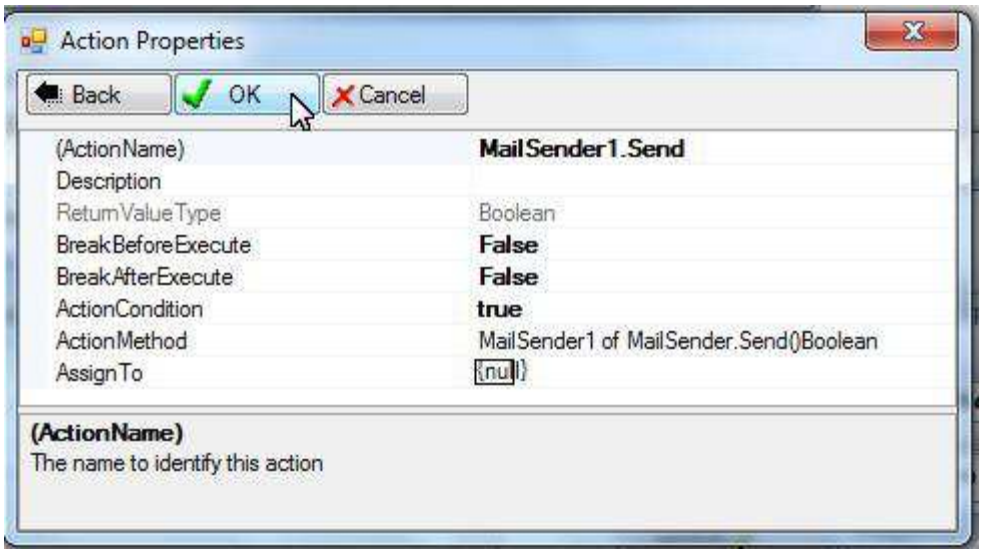


Action - Send email

Right-click MailSender; choose "Create Action"; choose "Send" method:

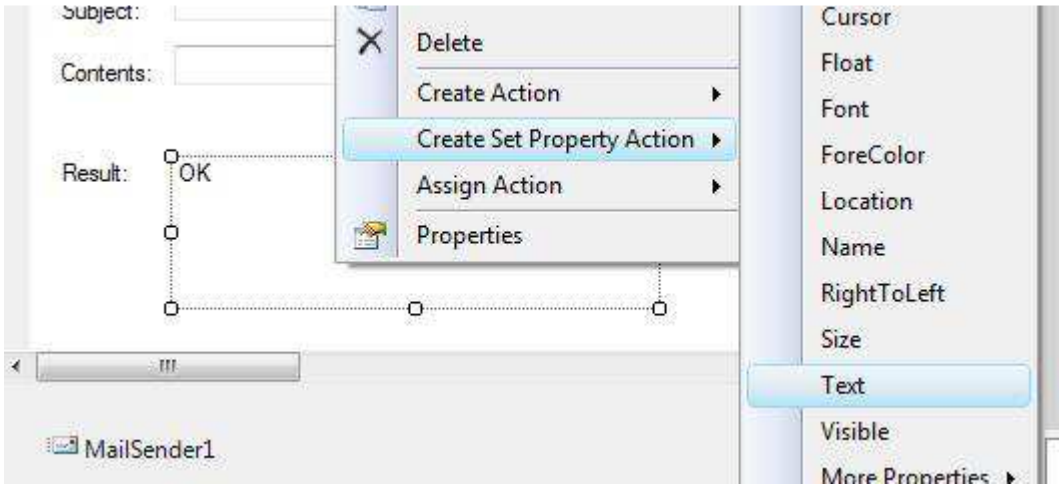


Click OK to finish creating this action:

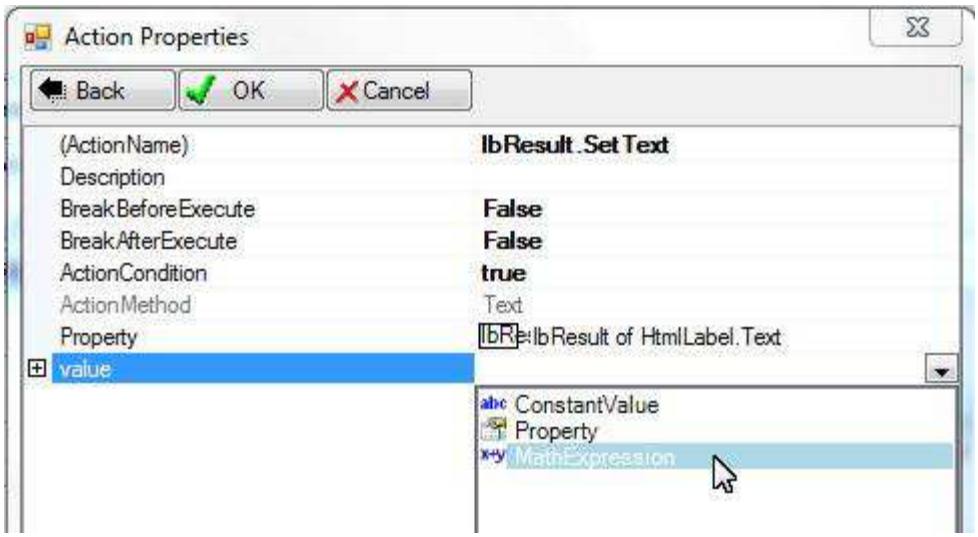


Action – Set results

Right-click on the label for showing results; choose "Create Set Property Action"; choose "Text":

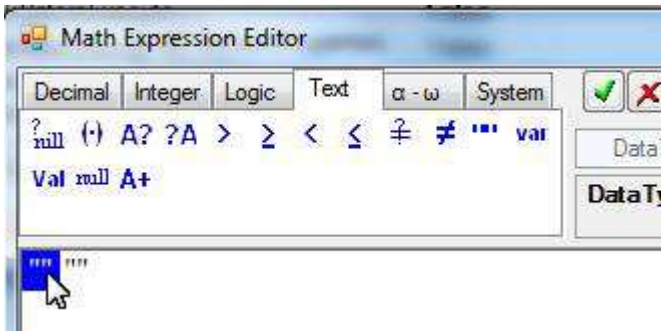


For the "value" parameter of the action, this time we make it more complex, select "MathExpression":

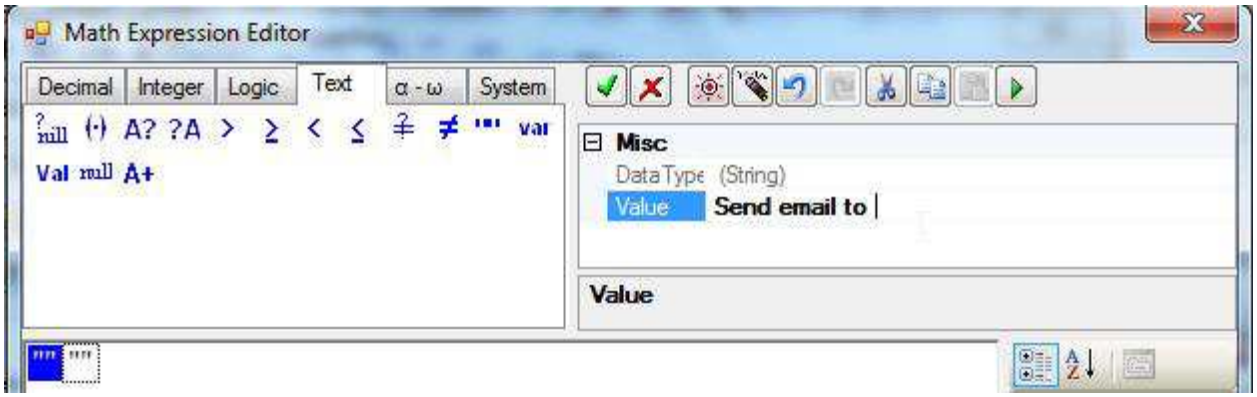


Click A+ to add string concatenation expression:

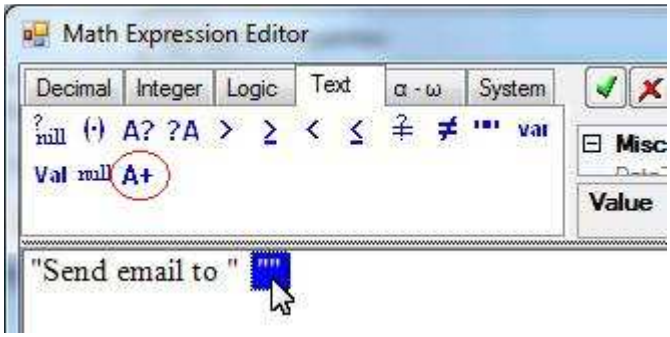
Select the first element:



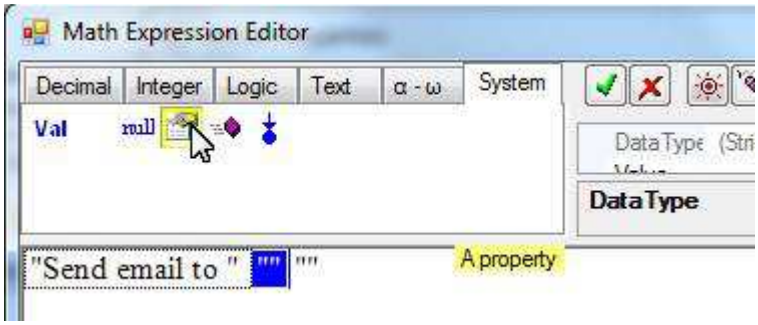
Type in some text:



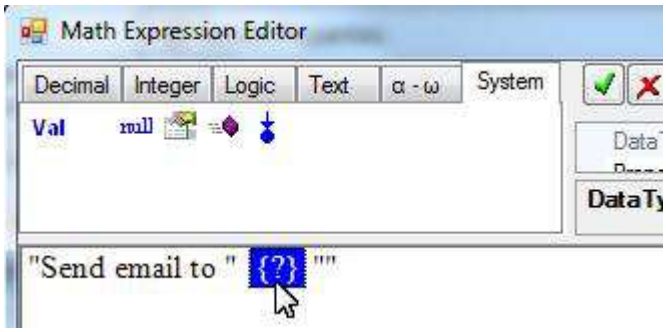
Select the second item; click A+ to add one more string:



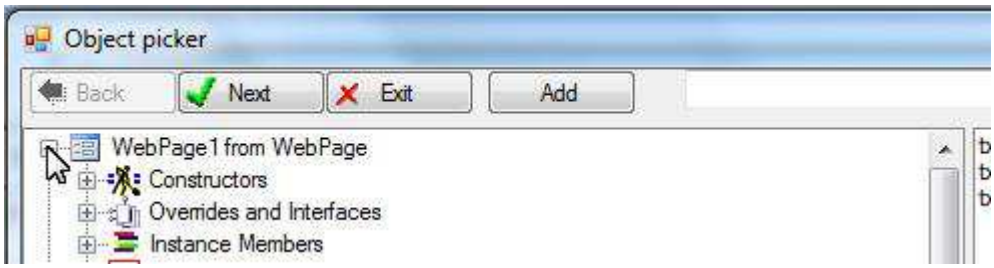
Select the second element. Click the Property icon:

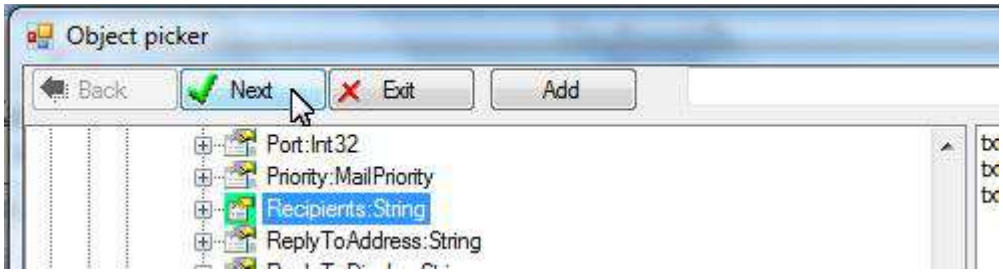


Double-click the Property element:



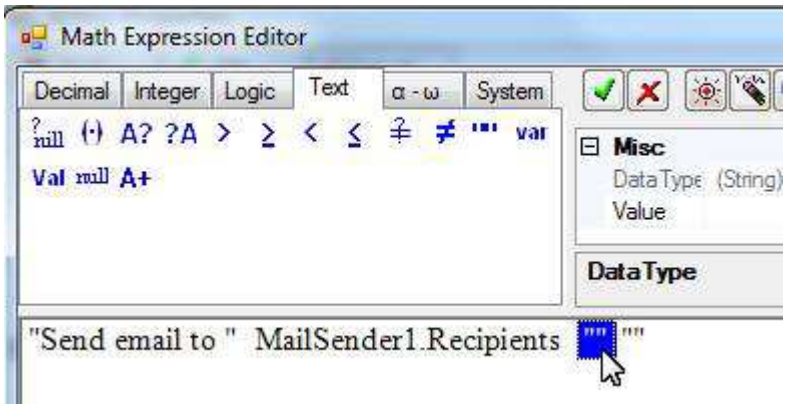
Select the Recipients property of the MailSender:



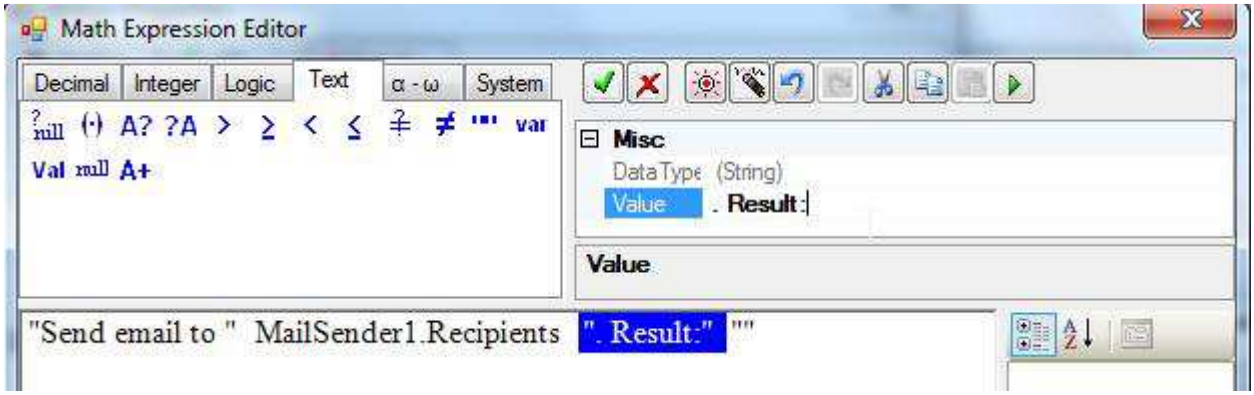


Select the last element; click A+ to add one more string:

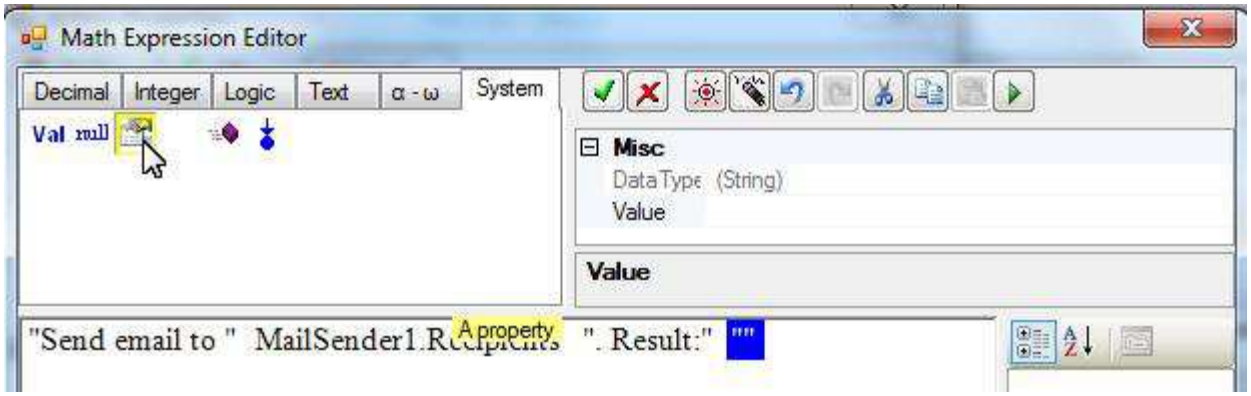
Select the element following the Recipients:



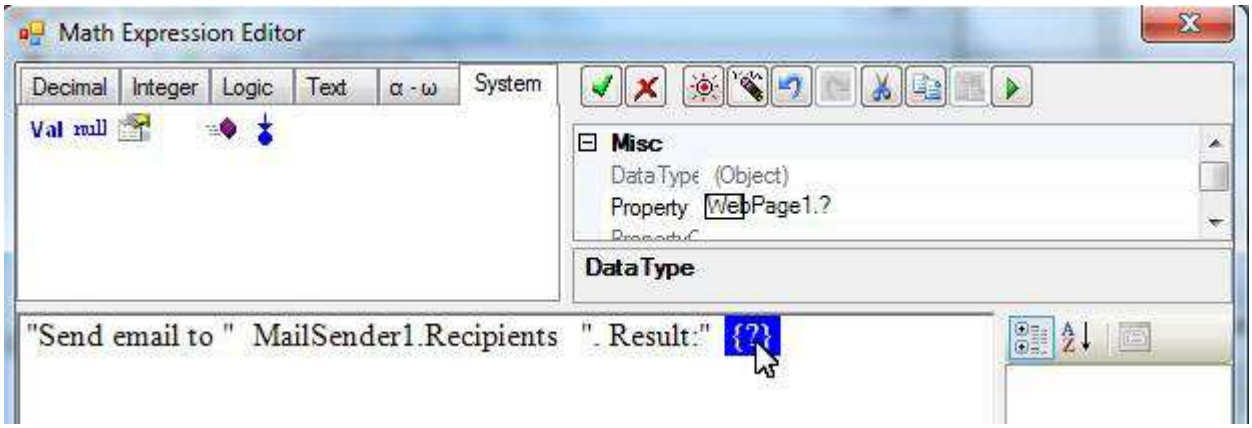
Type some text:



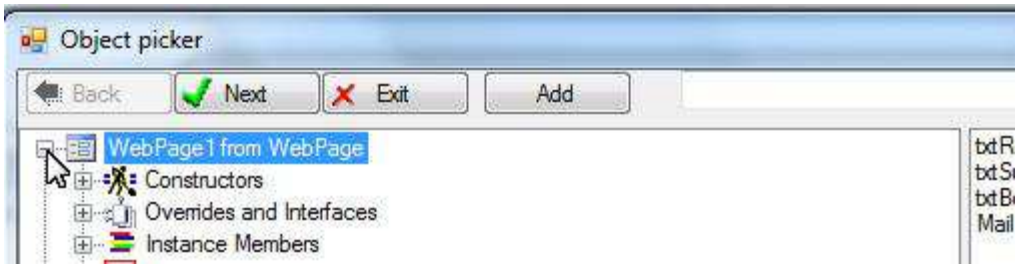
Select the last element; click the Property icon:

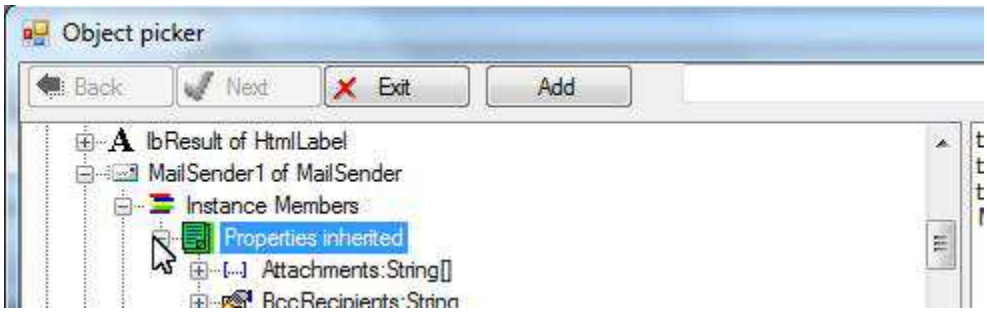


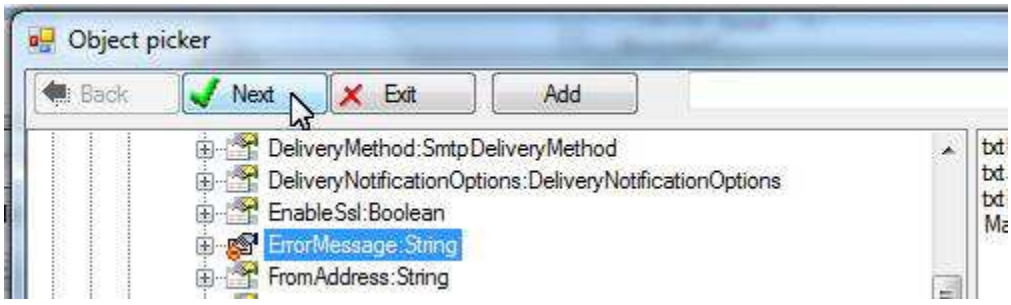
Double-click the Property element:



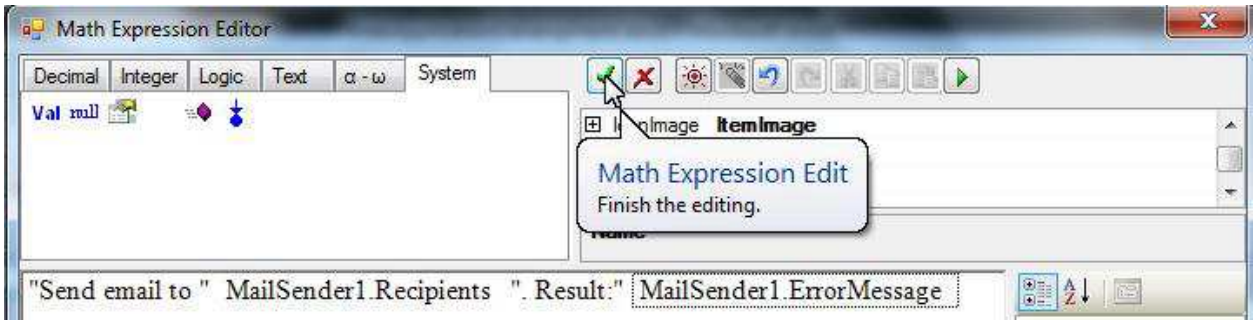
Select the ErrorMessage property of the MailSender:



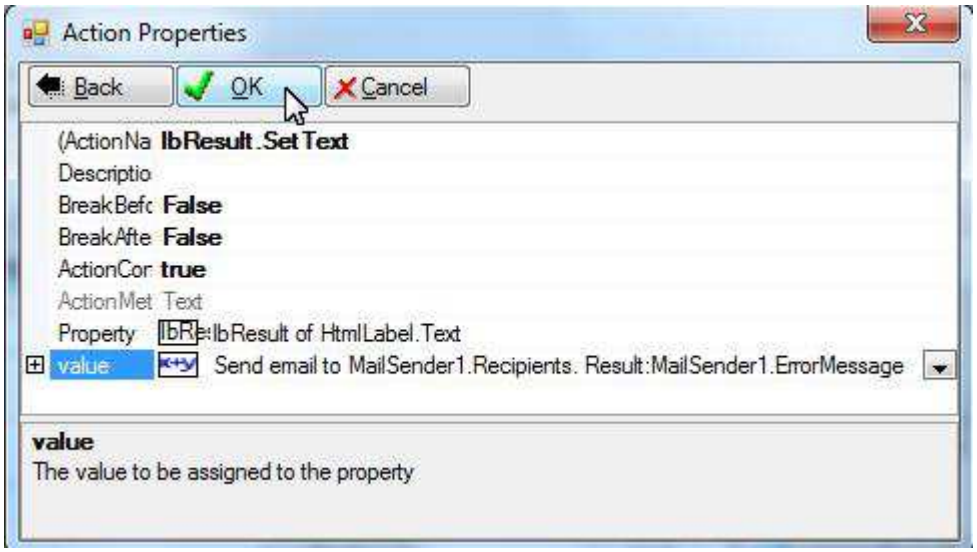




This is the expression we formed:

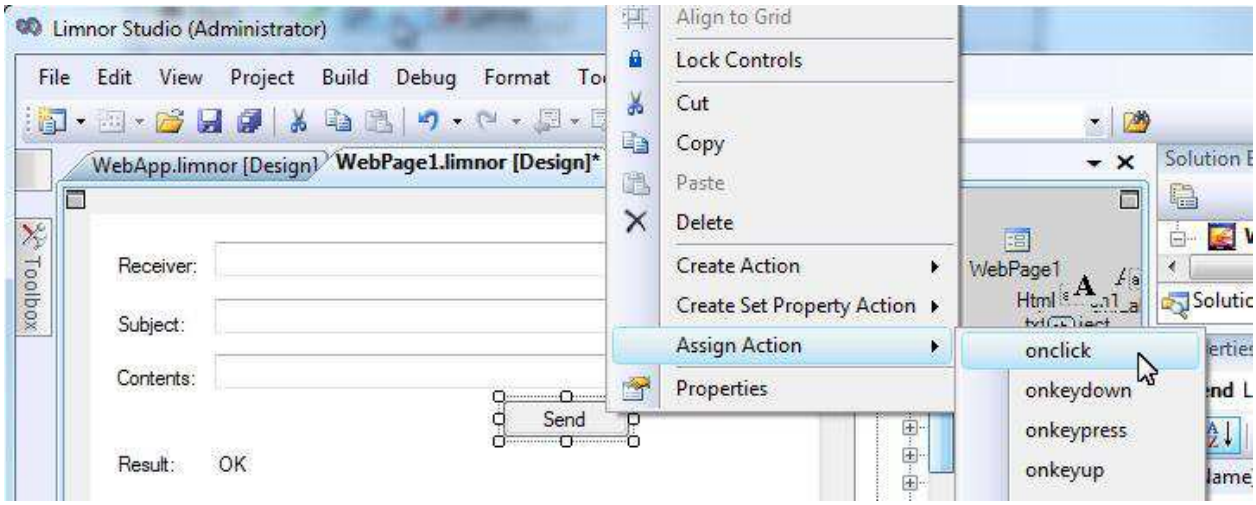


Click OK to finish creating this action:

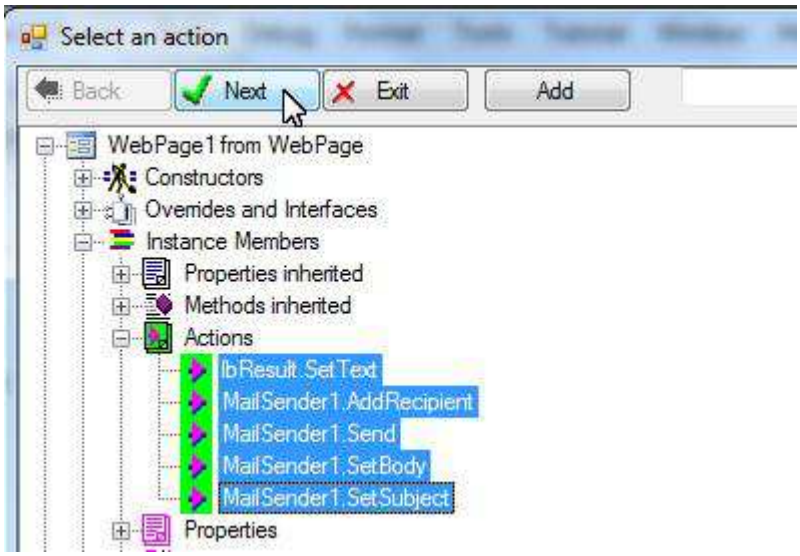


Assign actions to event

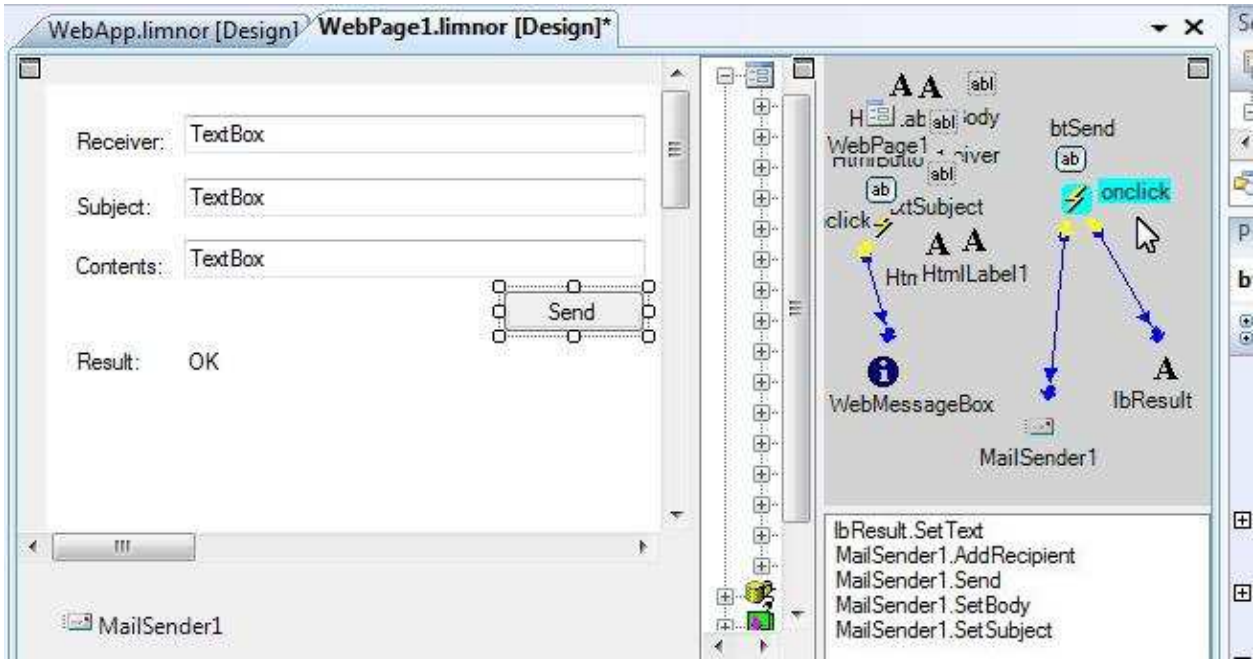
Right-click the button; select "Assign Action"; choose "onclick" event:



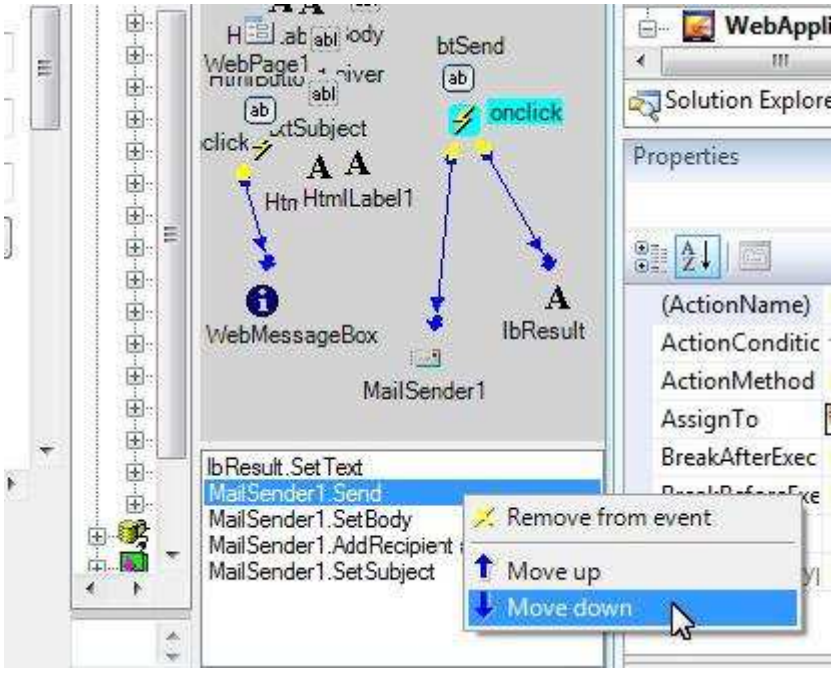
Select the actions to be used:

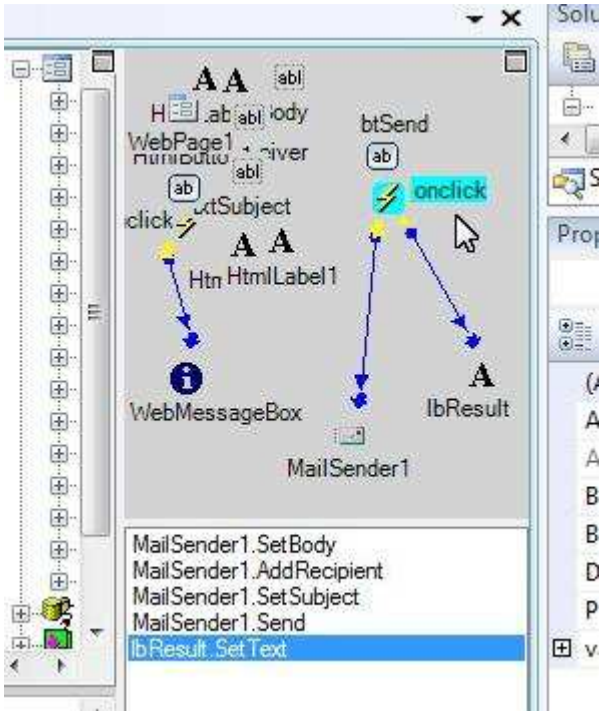


The actions appear under the event:



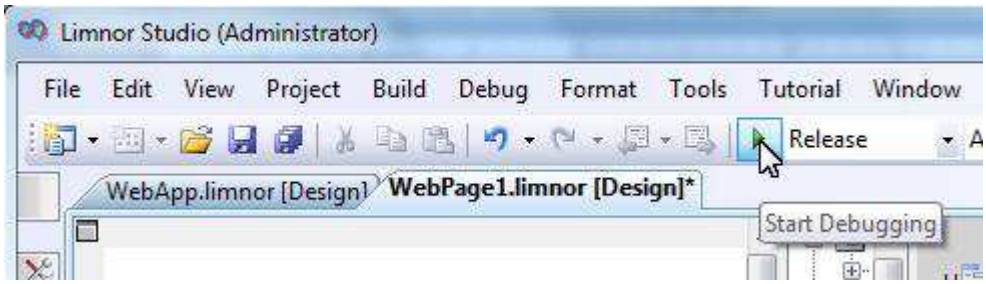
The actions executed in the order of the action list from top to bottom. We need to move the action MailSender1.Send action down after all the set property actions; and move action lbResult.SetText to the last.



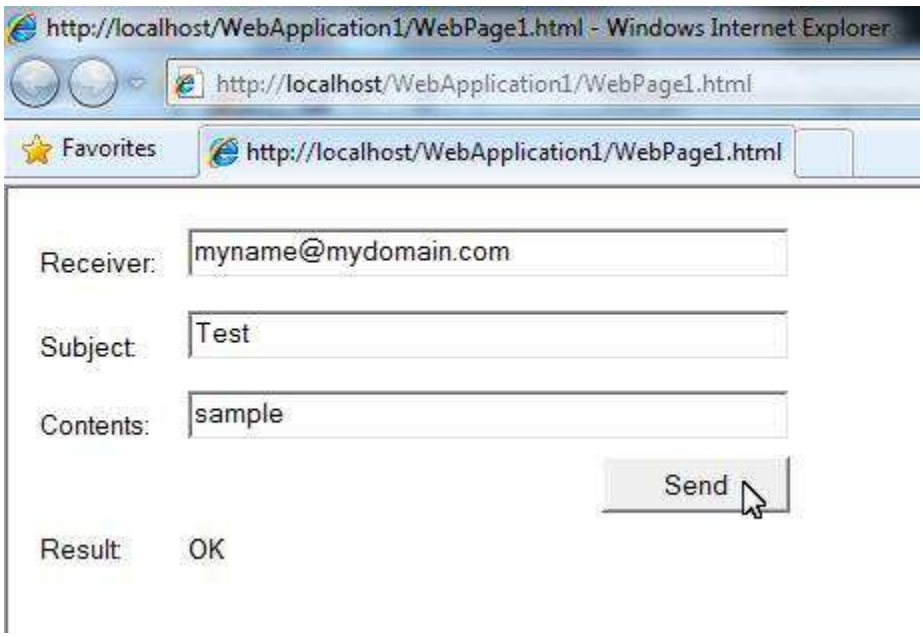


Test the sample web application

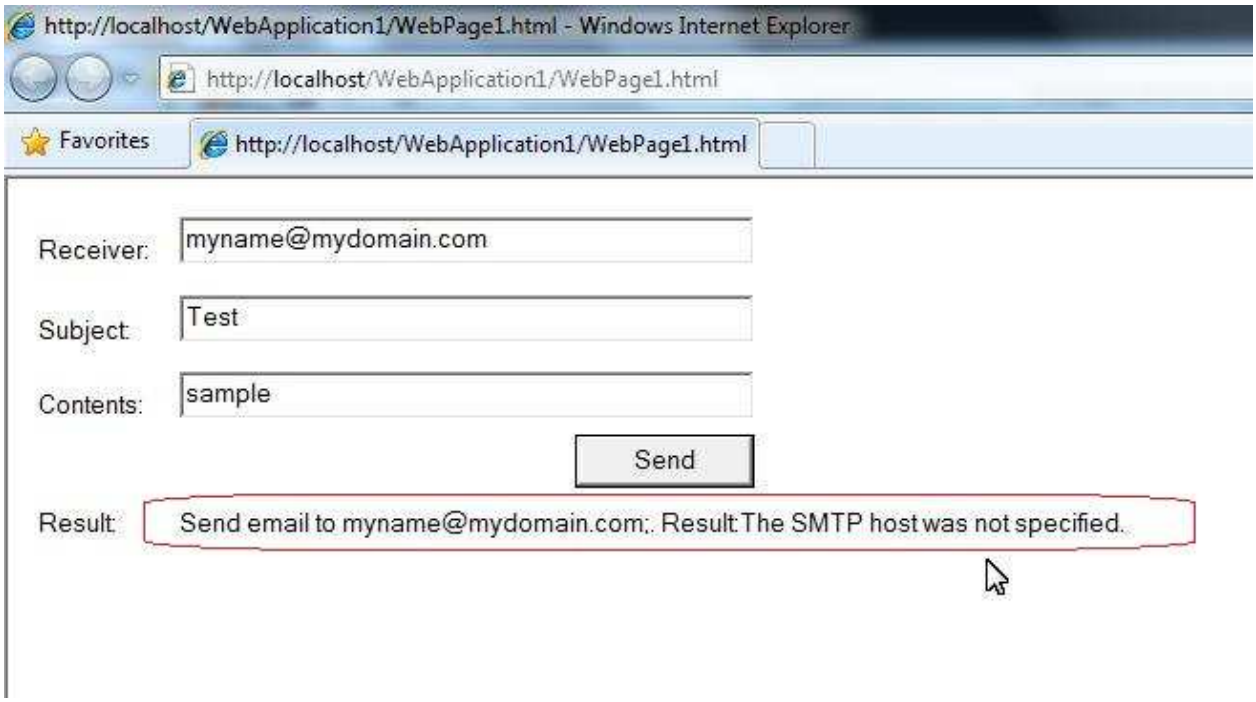
Click the Run button to test the web application:



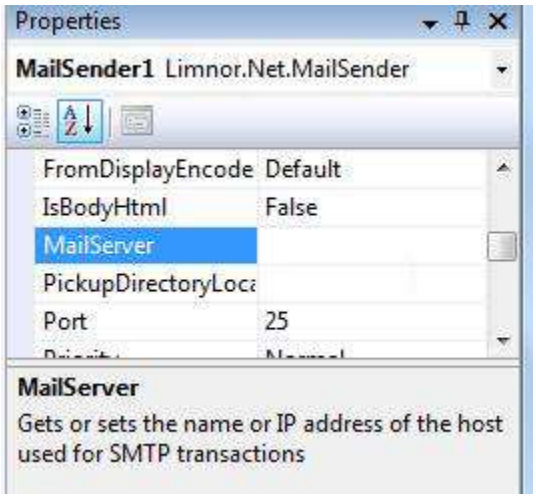
The compiling starts. After compiling, the web application is launched in the web browser. We may enter some test data in the web page:



Click Send button and wait for some time. The results are displayed:

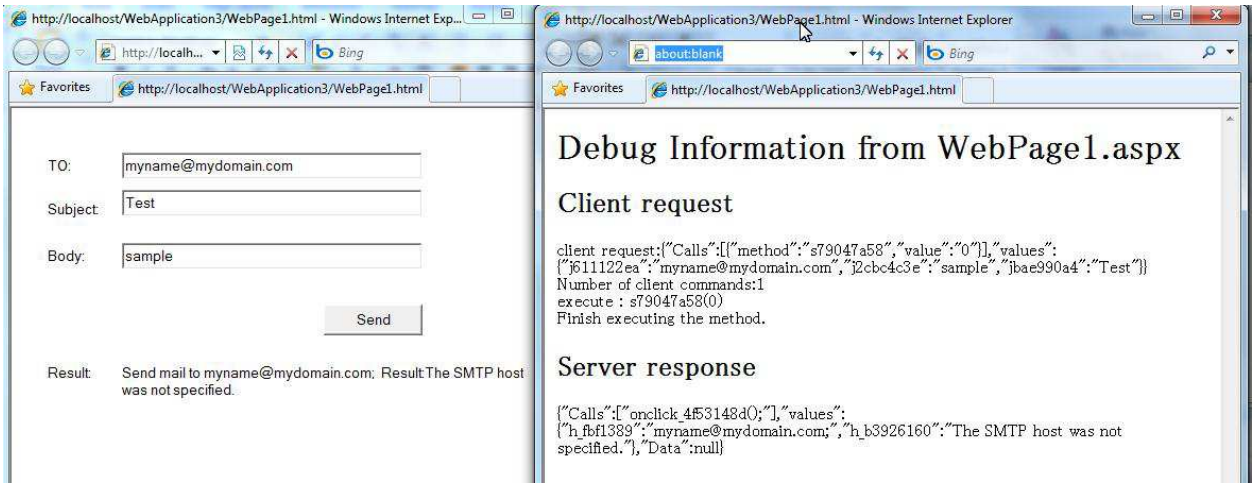


The result tells us that "The SMTP host was not specified". It is telling the truth, we did not set the MailServer property of the MailSender object when we were developing this web application:



Debug information

If we turn on DebugMode then a web page will appear showing the client and server communications:



If you do not close the debug page then new information will keep appending to the end of the page. If you close the debug page then it will get re-created when new debugging information arrives, but the previous information will not be restored.

Note that the debug page is a pop up window. Your browser most likely blocks pop up window. Some kinds of browsers will nicely ask your permission to allow the pop up window, such as IE. Some browsers will give you confusing error messages, such as Safari. Be sure to turn off pop up blocking for your test web site if you turn on debug mode.

How a web application works

We have seen our sample web application working by seeing the data flowing between the client and server.

If you have used Limnor Studio to do programming other than web applications then you can see that the programming of a web application as we just did is just like programming other kinds of projects.

But behind the scene a web application works very much differently than a standalone application. In this section, we describe how this sample web application runs. Some basic understanding of how a web application runs may help when doing high level design of a web project.

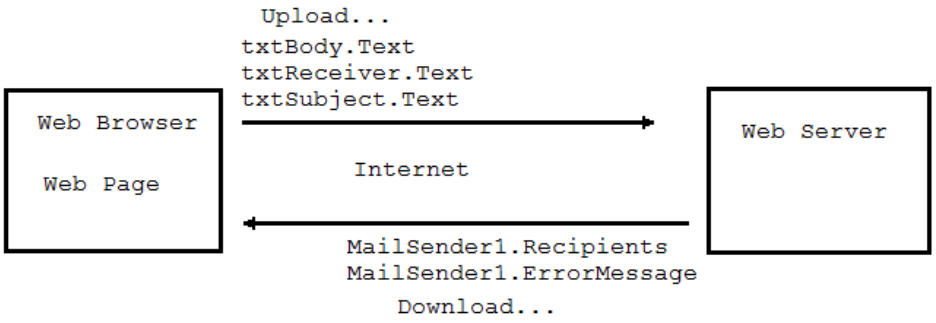
In this sample, clicking the Send button triggers following actions:

1. MailSender1.SetBody
2. MailSender1.AddRecipient
3. MailSender1.SetSubject
4. MailSender1.Send
5. lbResult.SetText

Behind the scene the following things happen automatically:

1. A MailSender object, named MailSender1, is created in the web server

2. The values the user entered in the web page through the web browser are needed by actions MailSender1.SetBody, MailSender1.AddRecipient and MailSender1.SetSubject. Therefore these values are uploaded automatically from the web browser to the web server
3. Using the values uploaded from the client, actions MailSender1.SetBody, MailSender1.AddRecipient and MailSender1.SetSubject are executed at the web server.
4. Action MailSender1.Send is executed at the web server. This action fails due to the fact that the MailServer property is not set. The ErrorMessage property of MailSender1 holds the error message.
5. The last action to be executed is lbResult.SetText. It sets the text of lbResult to an expression formed as "Send email to " + MailSender1.Recipients + ". Result: " + MailSender1.ErrorMessage. Since this expression uses two values, MailSender1.Recipients and MailSender1.ErrorMessage, from the web server, these two values are downloaded automatically from the web server to the web browser.
6. Action lbResult.SetText is executed in the web browser, using the values downloaded from the web server.



The result displayed in the web page is

Send email to myname@mydomain.com;. Result:The SMTP host was not specified.

where the email address and the error message are downloaded from the web server.

Note that the email address is entered in the web page and uploaded to the web server. It proved that the data are indeed traveling from web browser to web server (upload) and from web server to web browser (download).

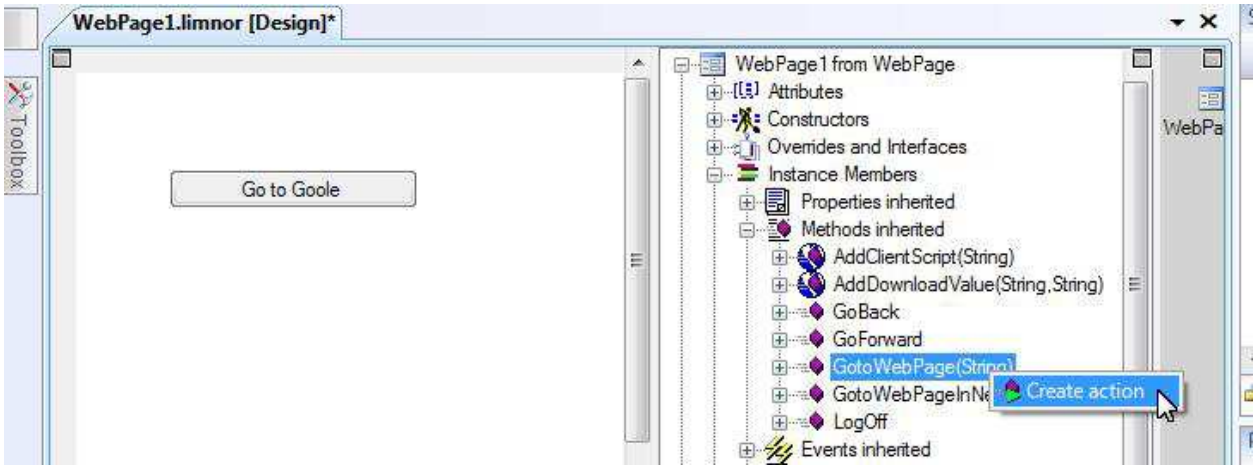
Page Parameters (Get and POST)

In the above example, data uploading is done via Post method asynchronously.

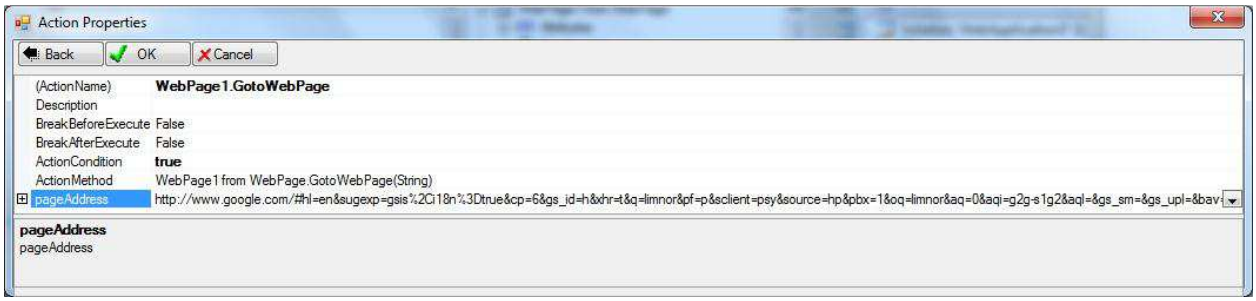
But web page parameters are sent using Get method.

Send page parameters

Methods "GotoWebPage" and "GotoWebPageInNewWindow" can be used to send page parameters.



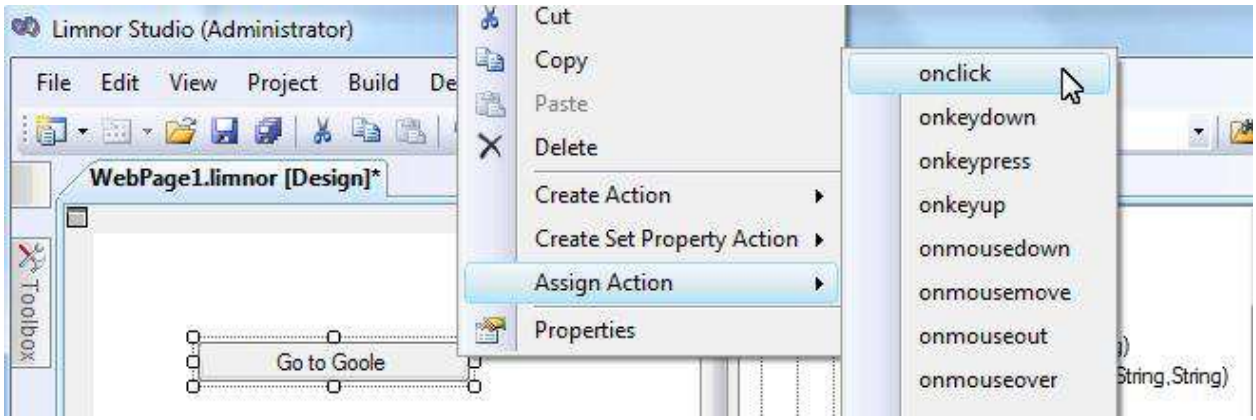
We may use ?{name}={value}&{name}={value}&… to pass page parameters:

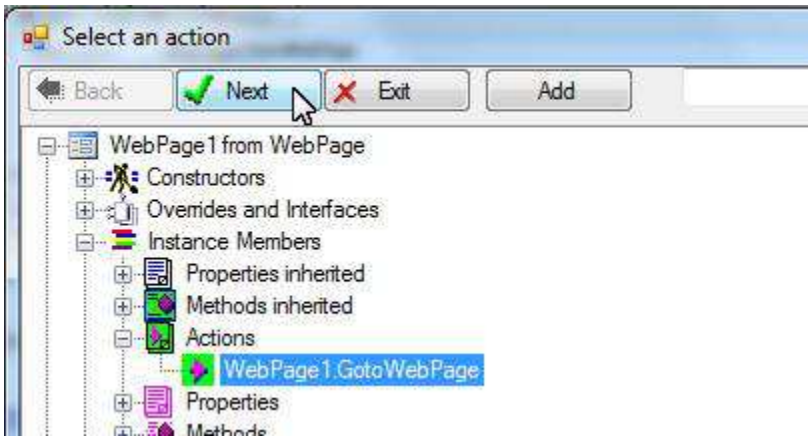


In the above example, we used following page address:

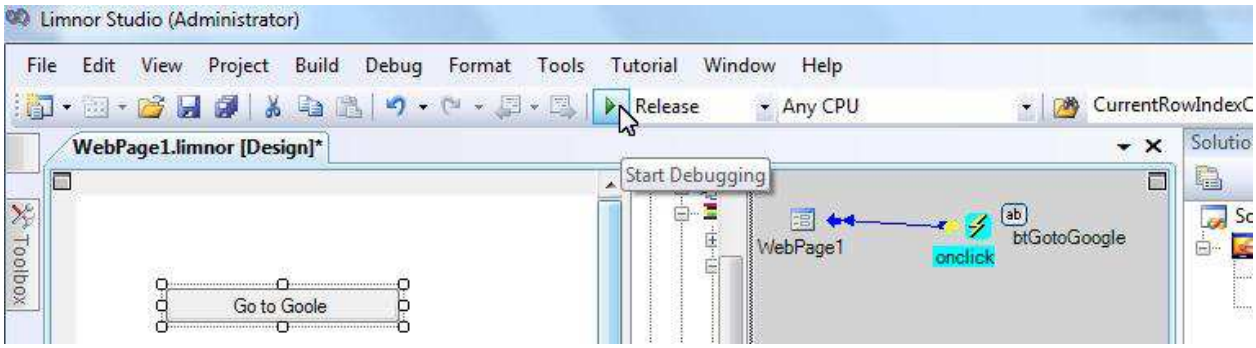
http://www.google.com/#hl=en&sugexp=gsis%2Ci18n%3Dtrue&cp=6&gs_id=h&xhr=t&q=limnor&pf=p&sclient=psy&source=hp&pbx=1&oq=limnor&aq=0&aqi=g2g-s1g2&aql=&gs_sm=&gs_upl=&bav=on.2,or.r_gc.r_pw.&fp=2ea25270560d40fb&biw=1280&bih=688

Assign the action to the button:

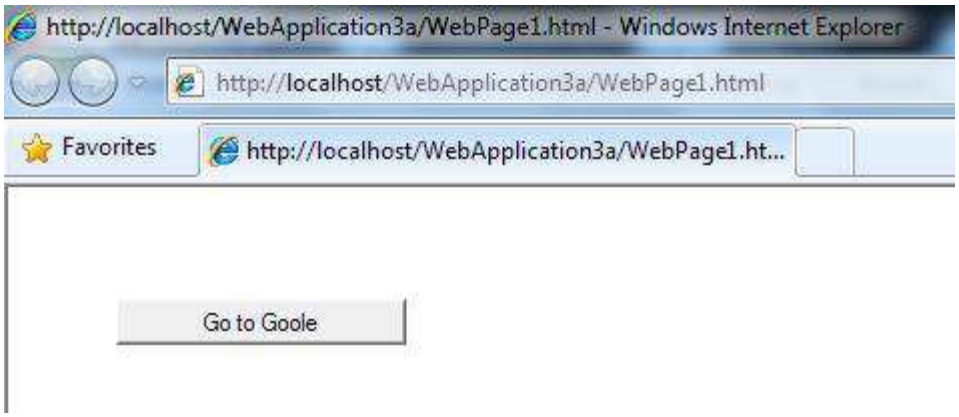




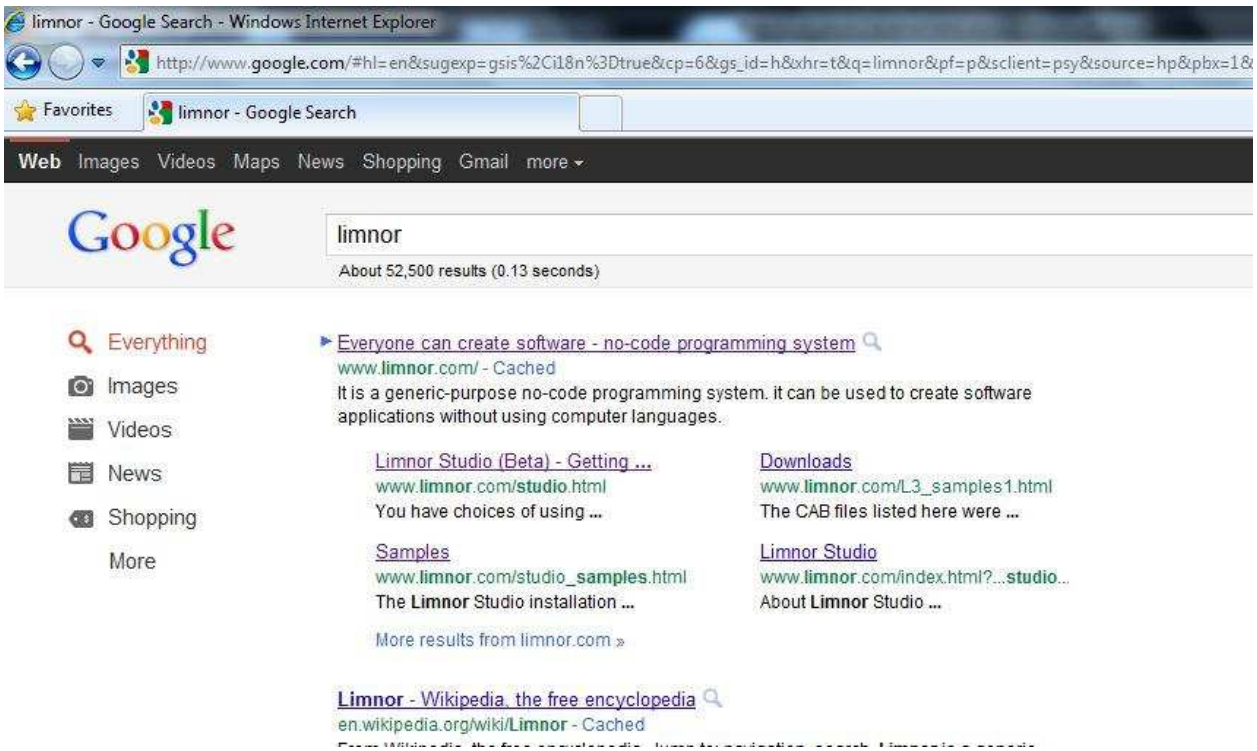
Test this web application:



The web page appears:



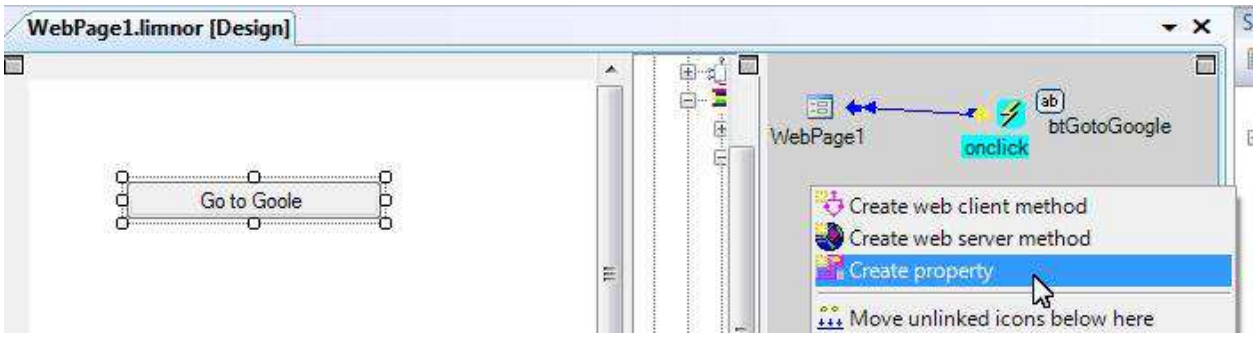
Click the button. The Google web site is visited with the page parameters used:



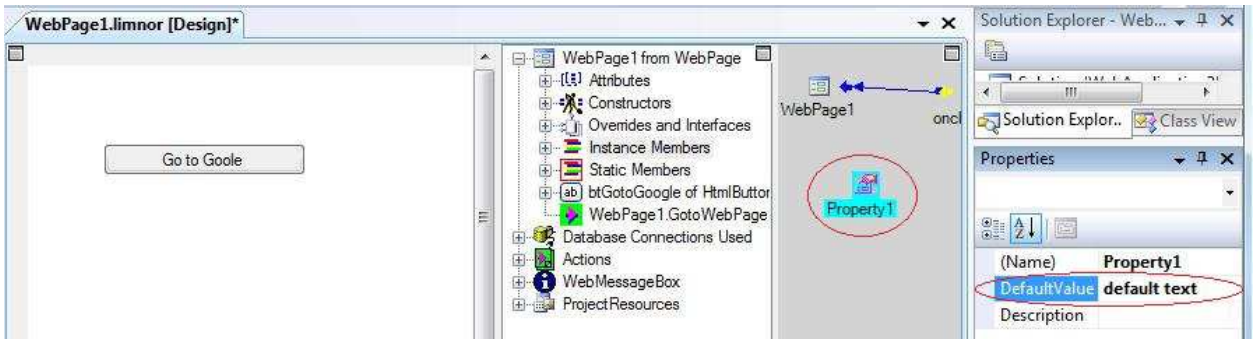
Use Page Parameters

In our web pages, we may also make use of the page parameters the visitors passed in.

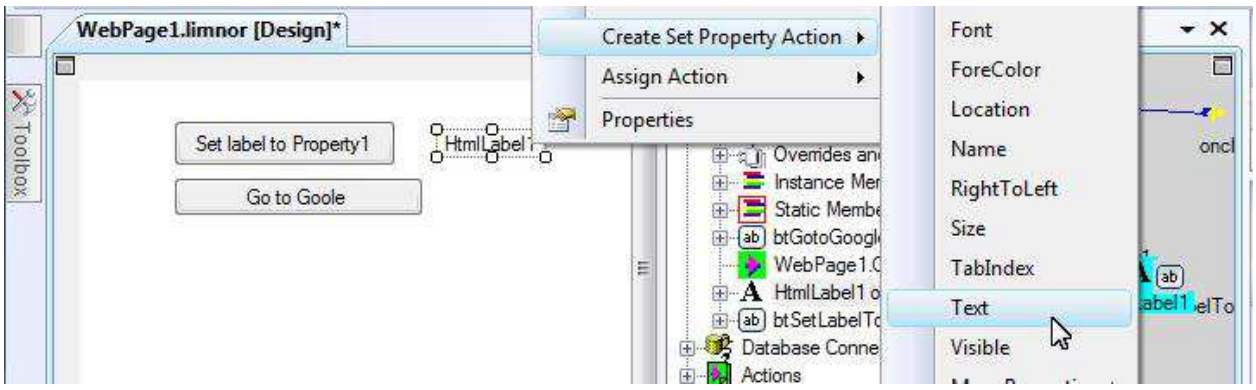
Create a new property on the page:



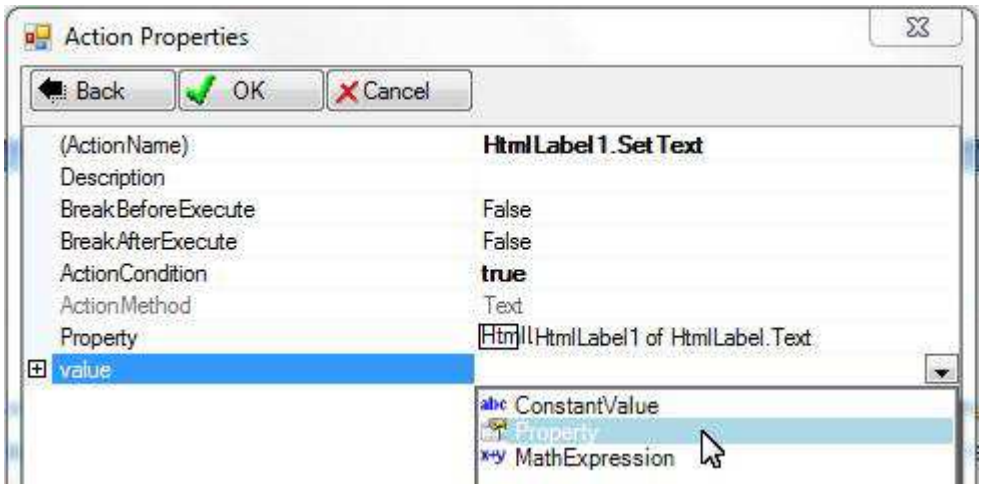
Give the new property a value:

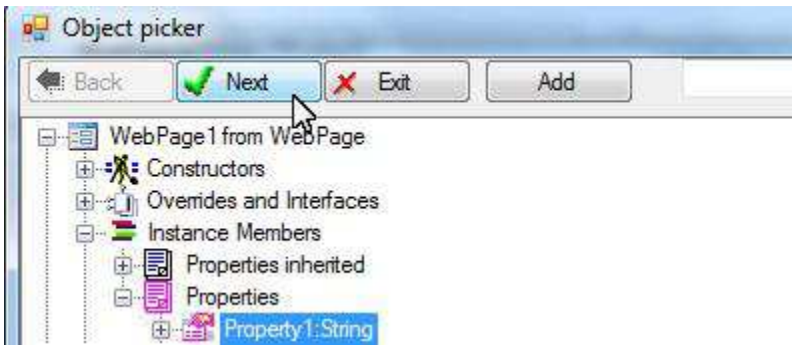


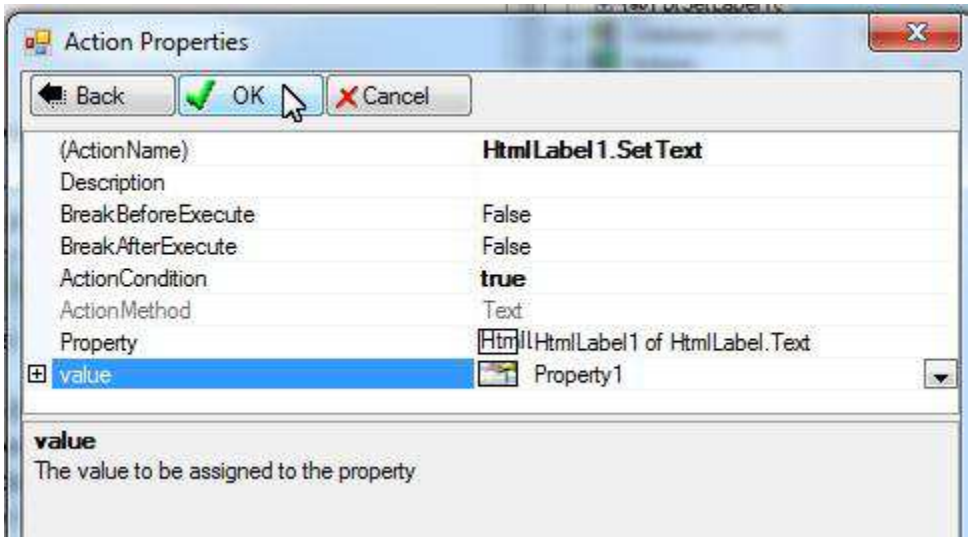
This property can be used anywhere in the programming. Let's create an example of using it by assigning its value to a label. Add a label to the page. Add a button to the page. When the button is clicked we want to assign the property to the label.



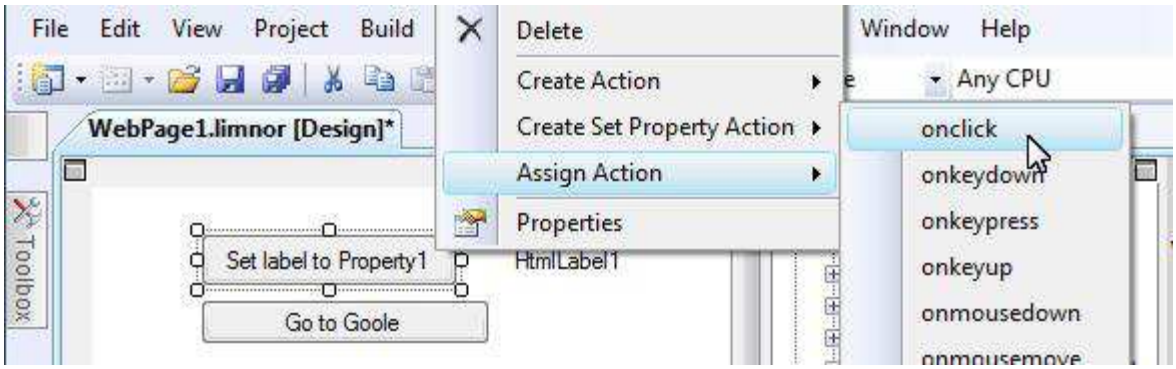
Select the Property1:

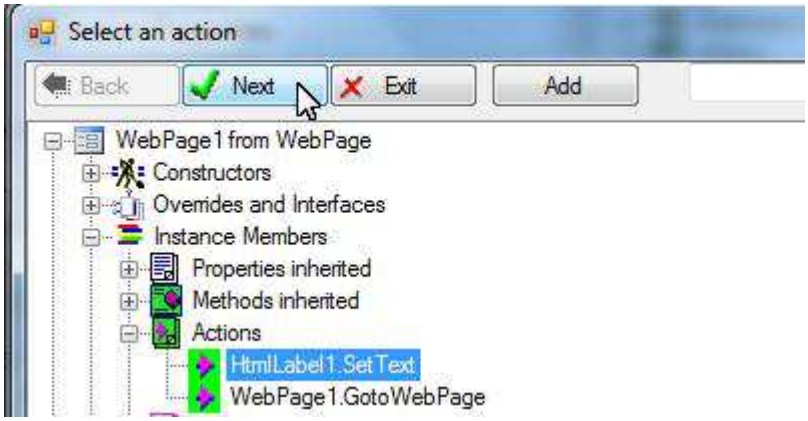




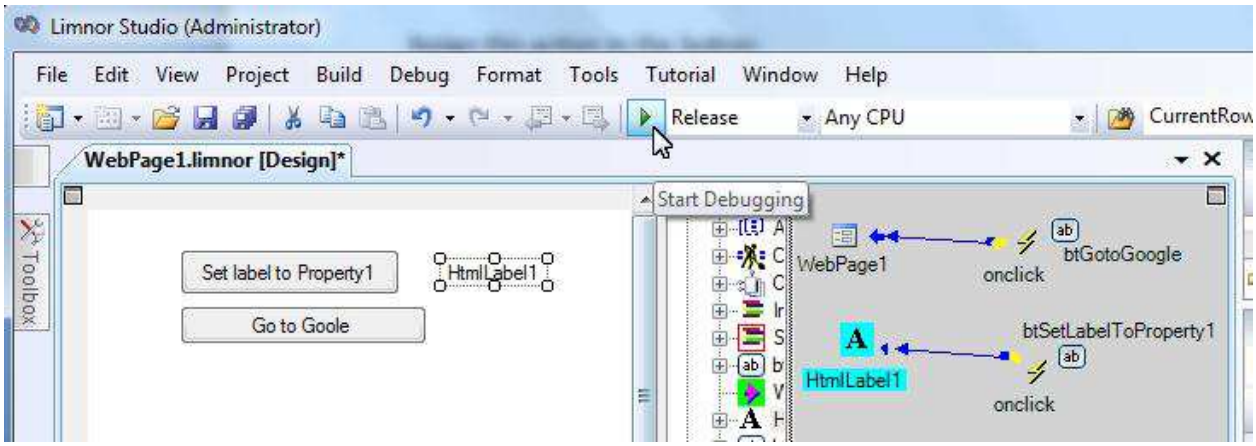


Assign this action to the button:

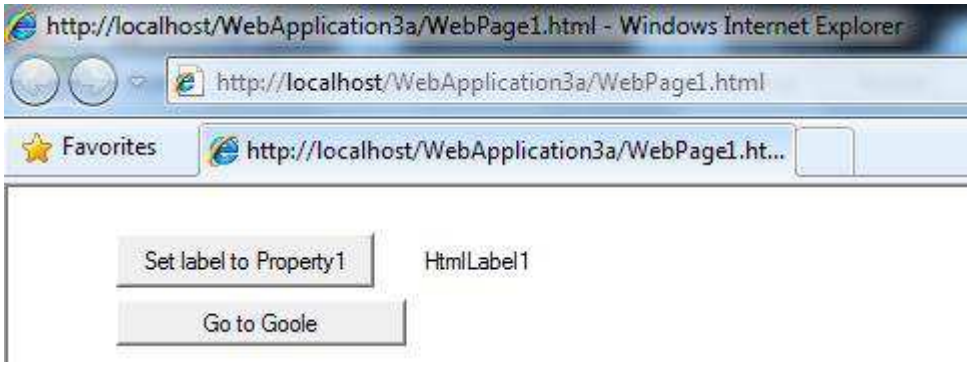




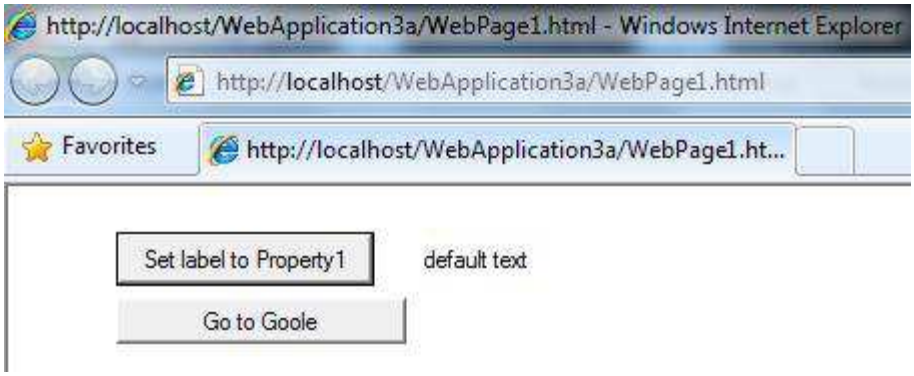
We may test the web application:



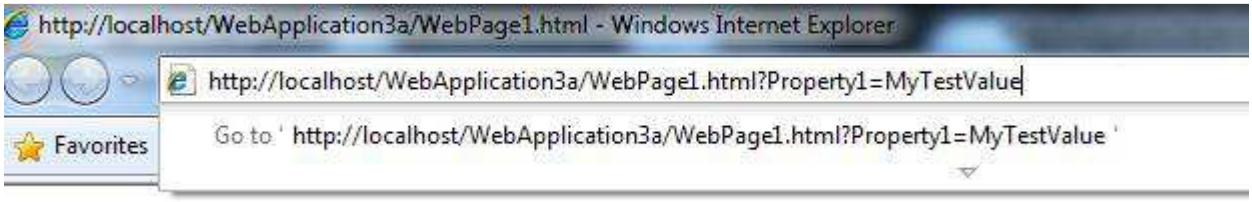
The web page appears:



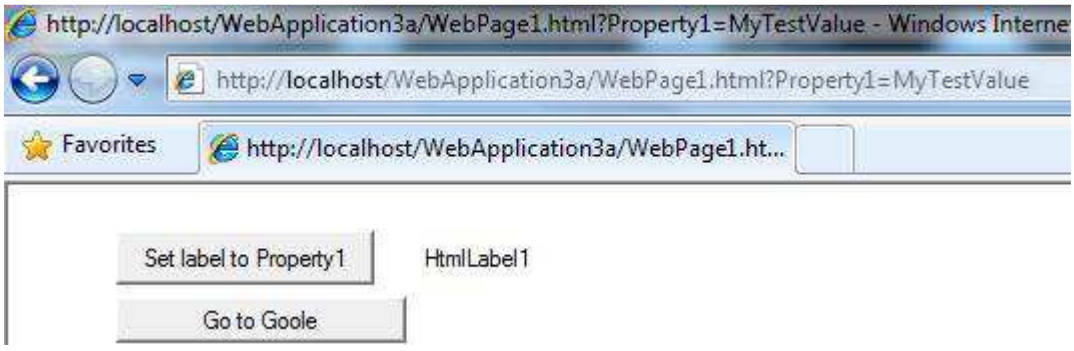
Click the button. The value of Property1 appears:



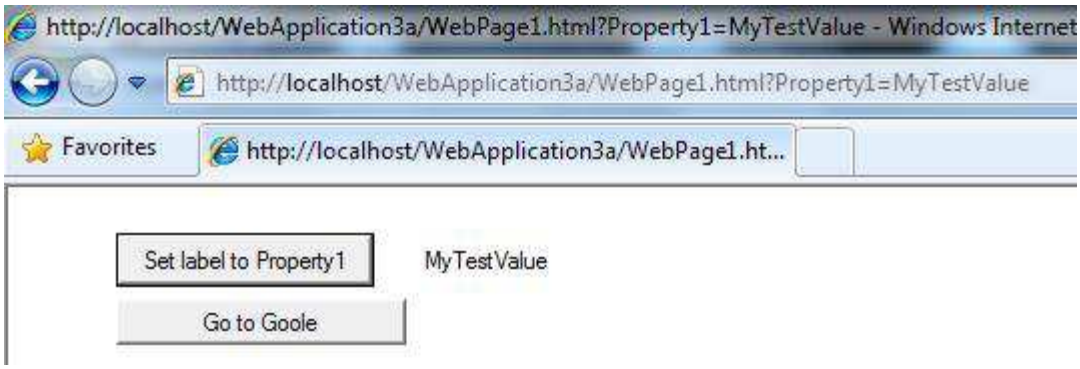
Now we try to use the page parameter. Enter "MyTestValue" as page parameter:



The page appears:



Click the button. We see that our page parameter appears:



Database Programming

EasyDataSet and EasyUpdator are database components for database programming. They can be used for both PHP and .NET web applications.

For a .NET web application, all databases with ADO.NET drivers can be used.

For a PHP web application, currently only MySql database is supported.

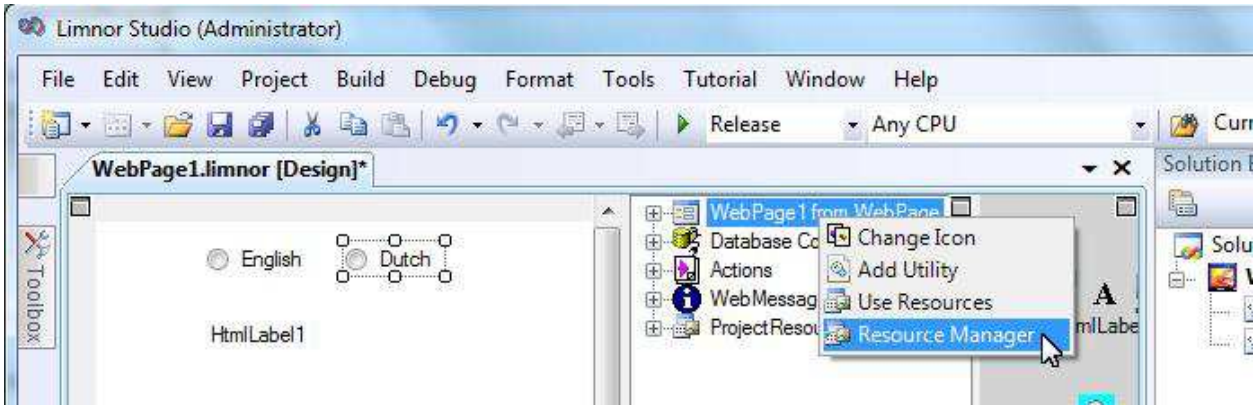
See http://www.limnor.com/support/webDatabaseProgramming.pdf for details of using databases in web applications.

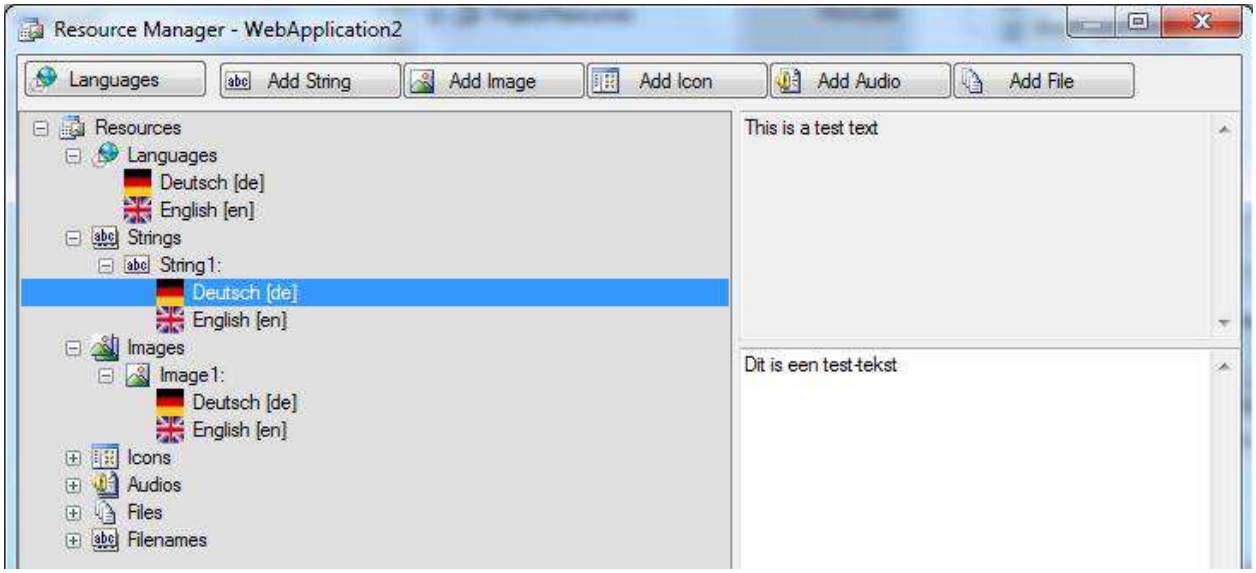
Multiple-Language Web Site

A web site is visible globally by people speaking different languages. You may want to let your web site visitors select display languages.

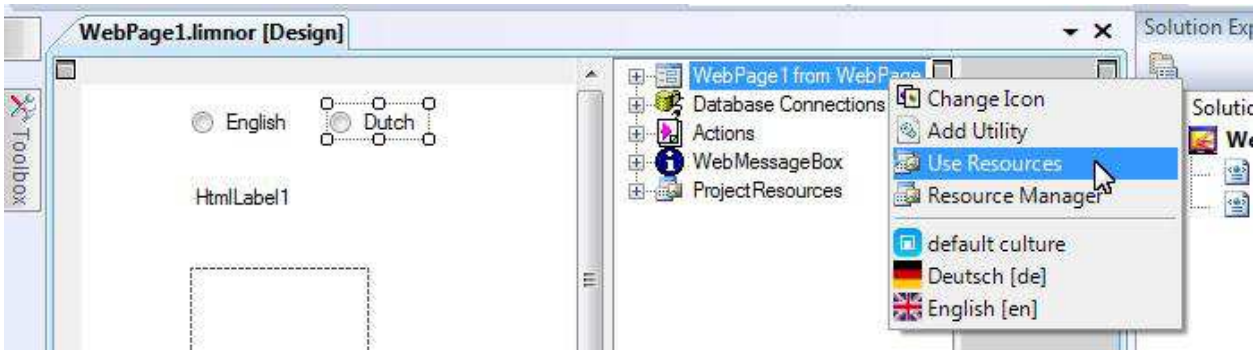
In Limnor Studio, creating a multi-language web site is in exactly the same way as creating a multi-language standalone application. For details, see http://www.limnor.com/support/Limnor%20Studio%20-%20User%20Guide%20-%20Resource%20Manager.pdf

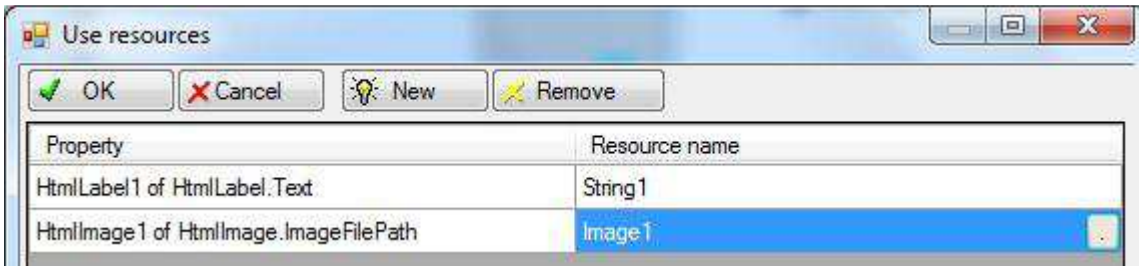
In summary, you use the Resource Manager to create resources in multiple languages:



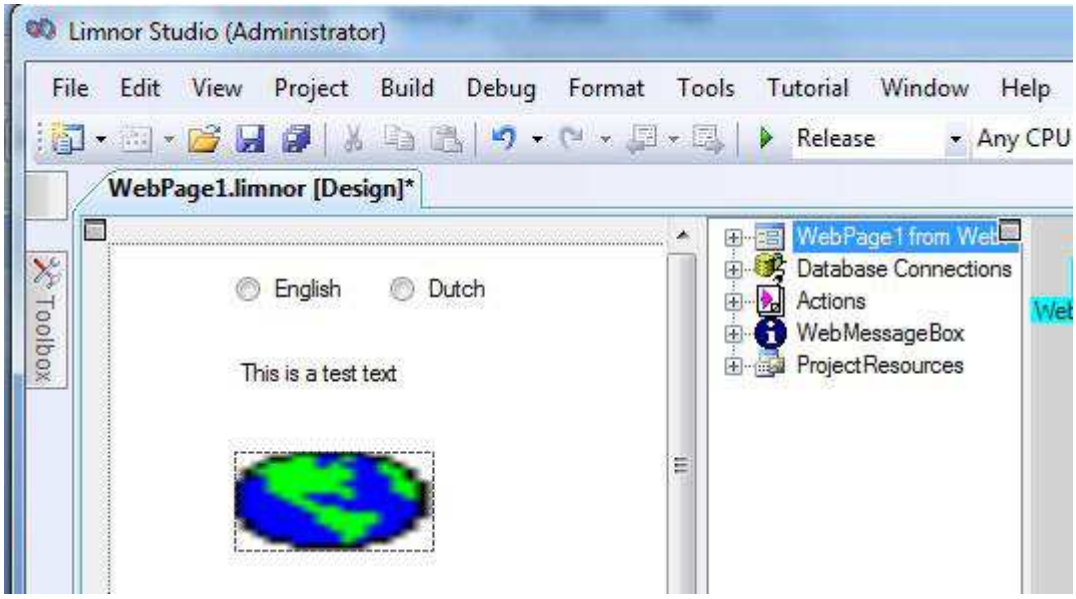


Map properties to the resources:

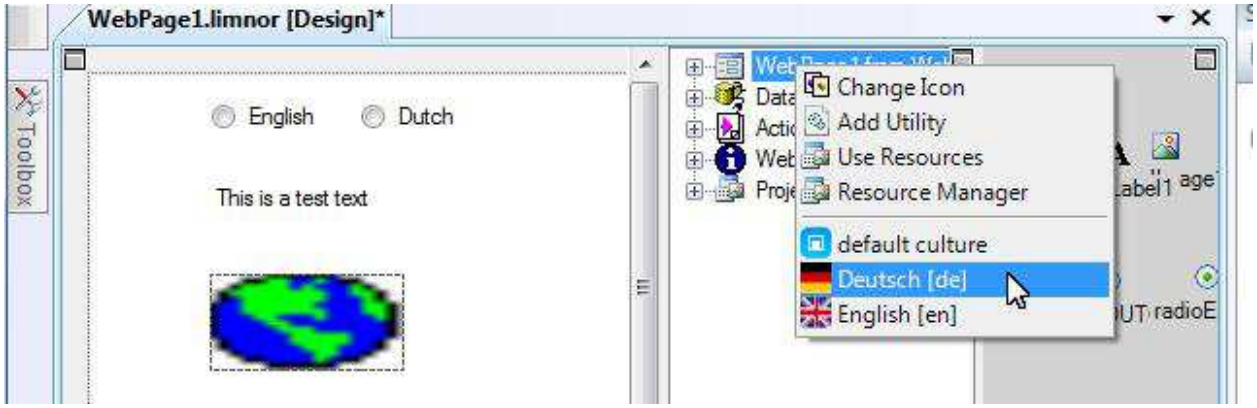


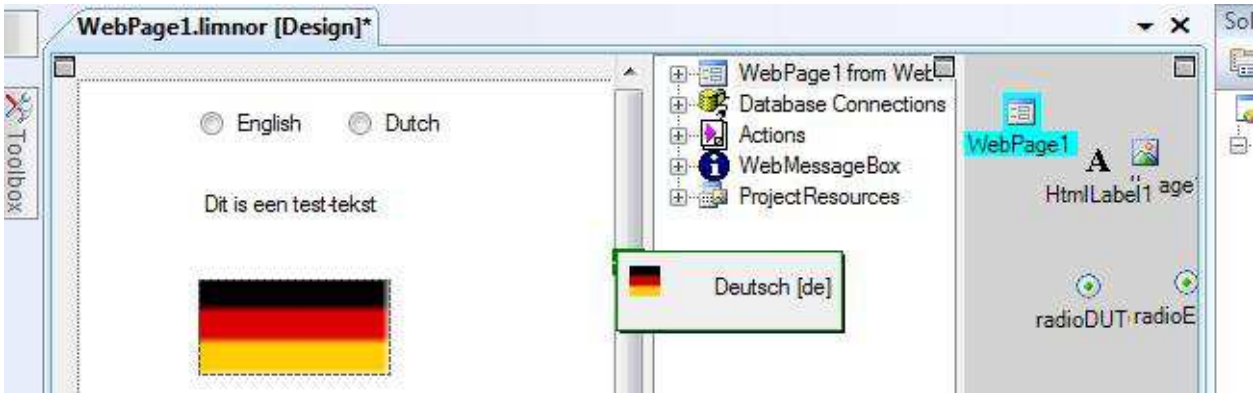


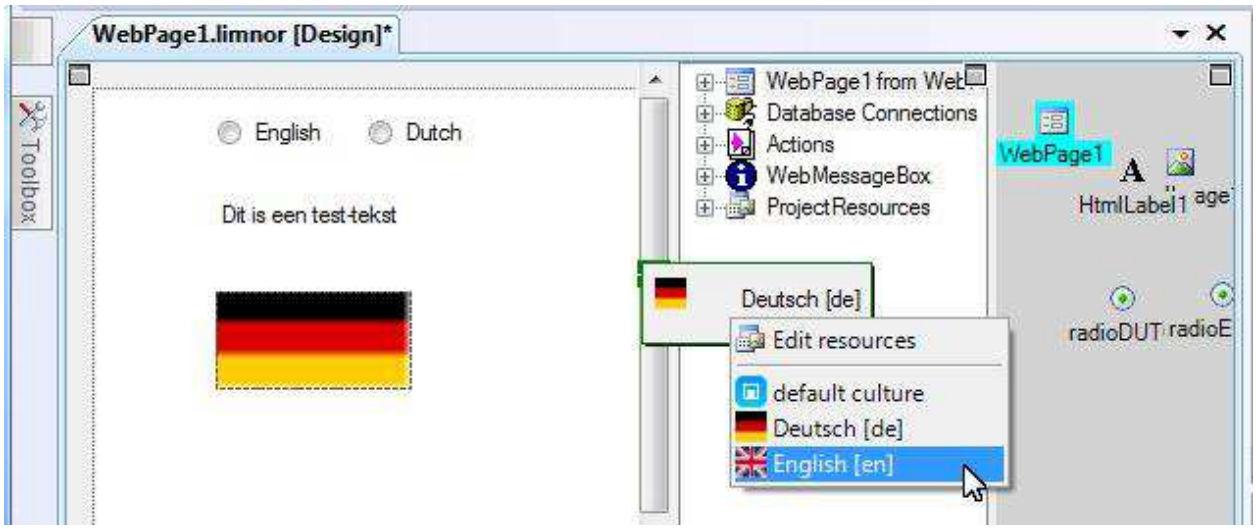
Once the property-resource mappings are established, the resources are used in the web page:



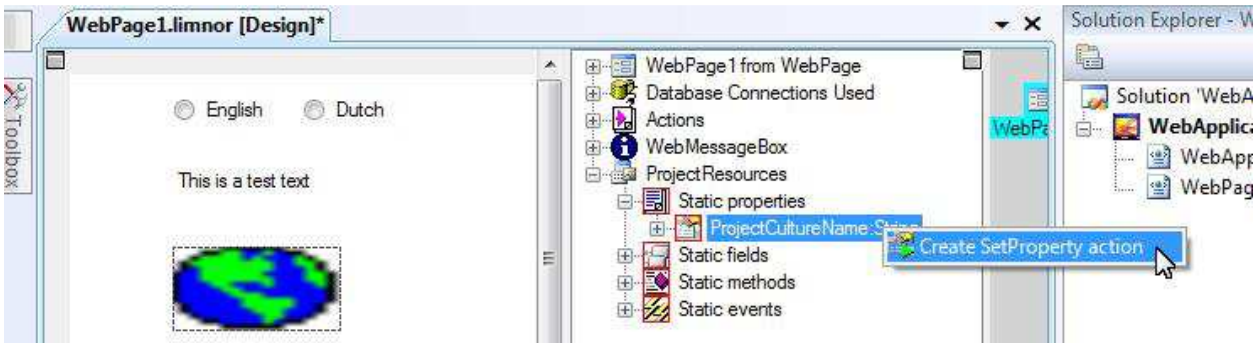
You may switch language to see the web page in different languages:

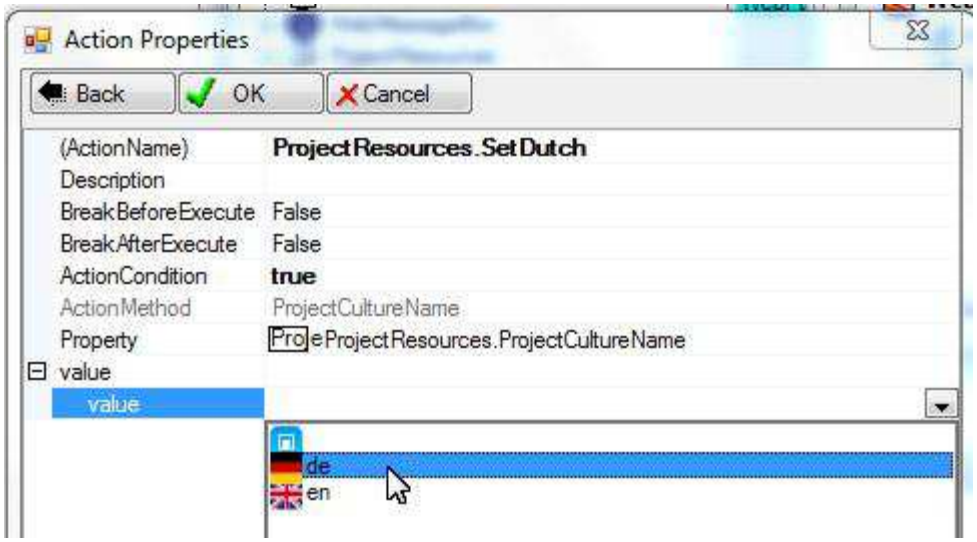




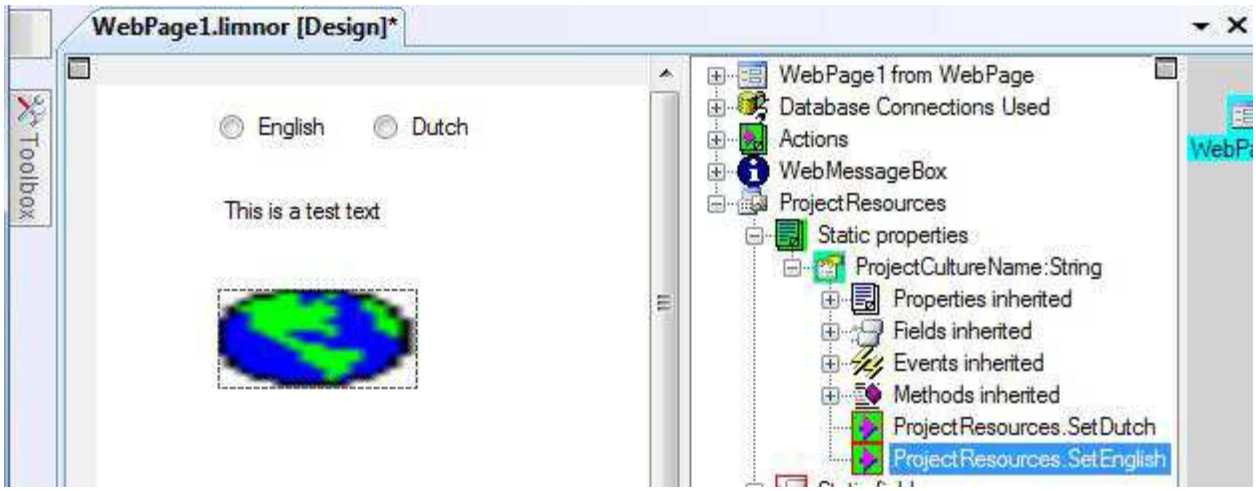


To let web visitors switch languages, create actions to set ProjectCultureName:

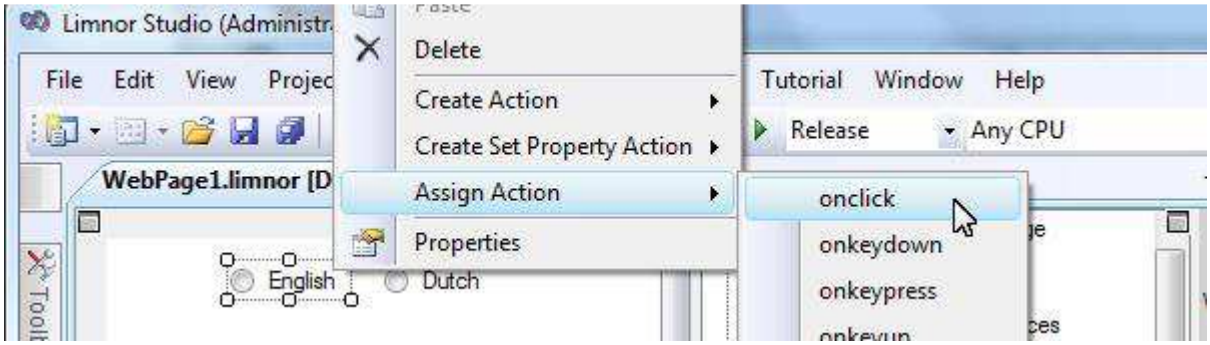


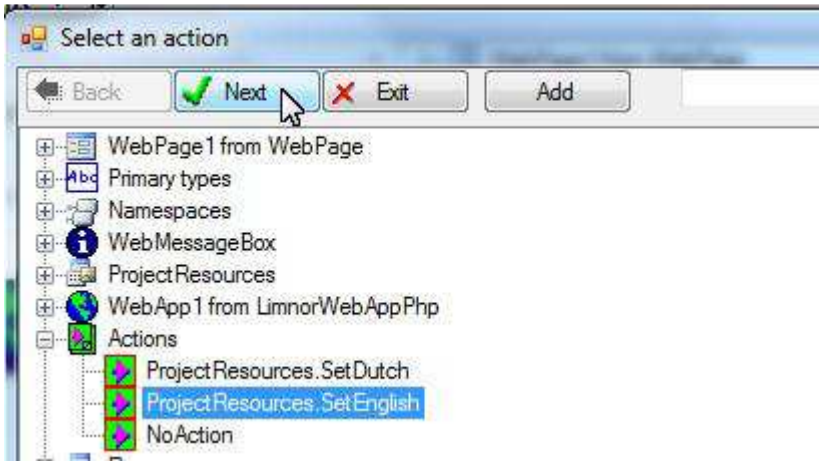


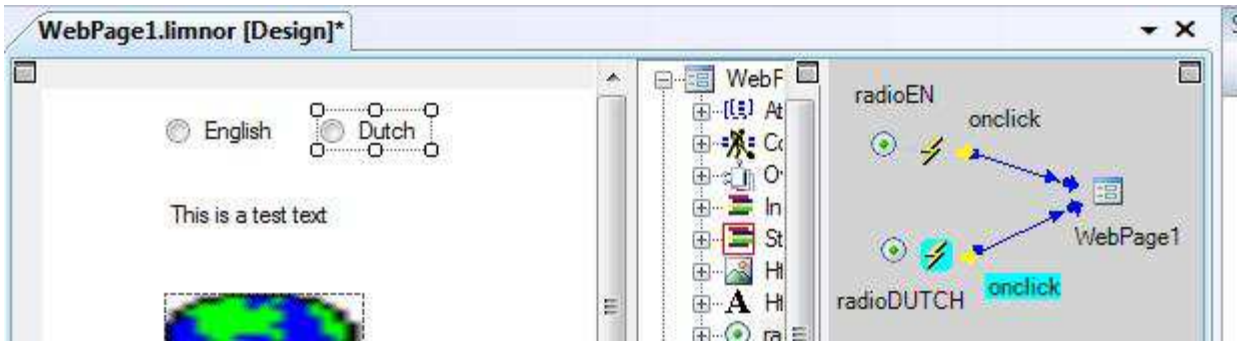
Create such an action for each language:



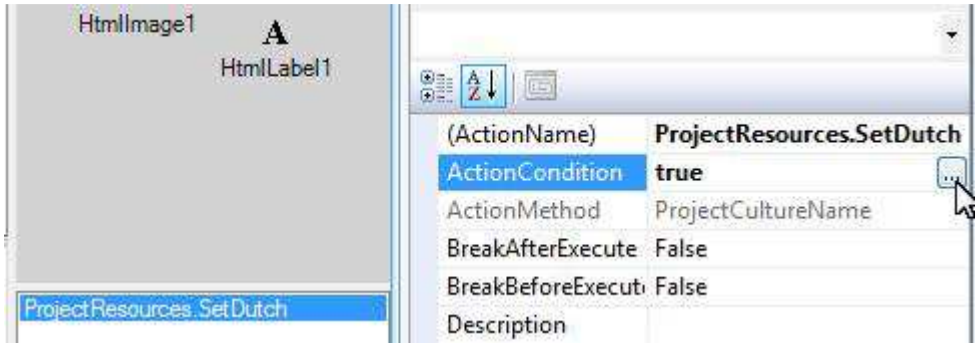
Suppose we use radio buttons to let the web visitors switch languages. We assign the actions to corresponding radio buttons:



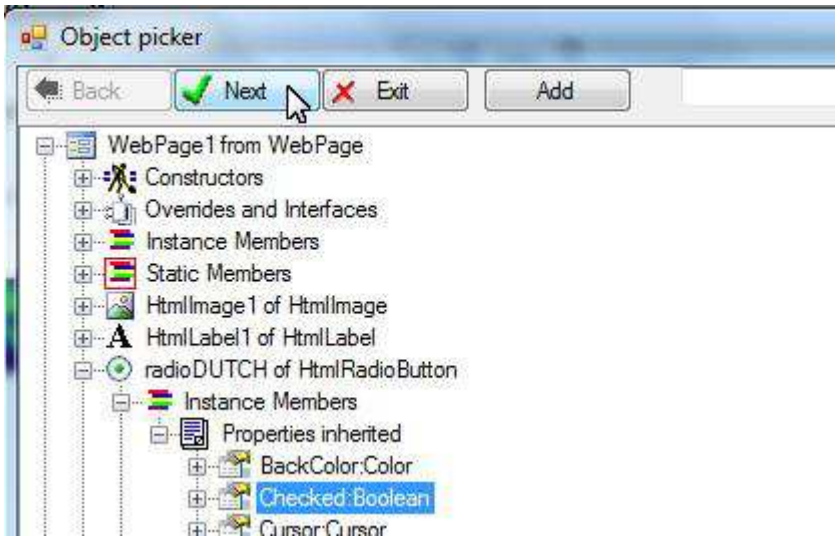


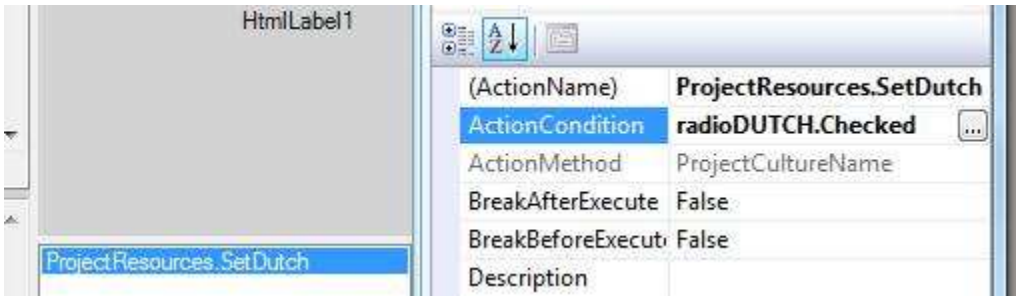


We need to set ActionCondition so that each language-switching action only executes when the corresponding radio button is checked:

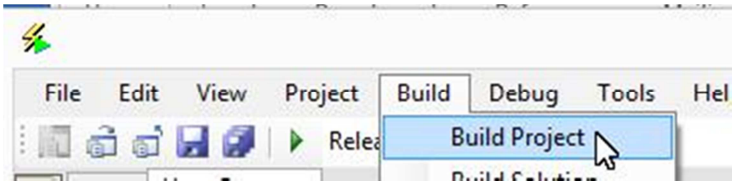


Select the Checked property of the radio button:

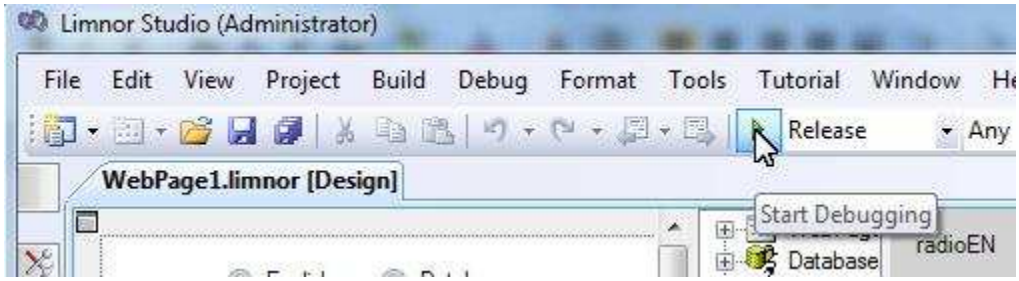




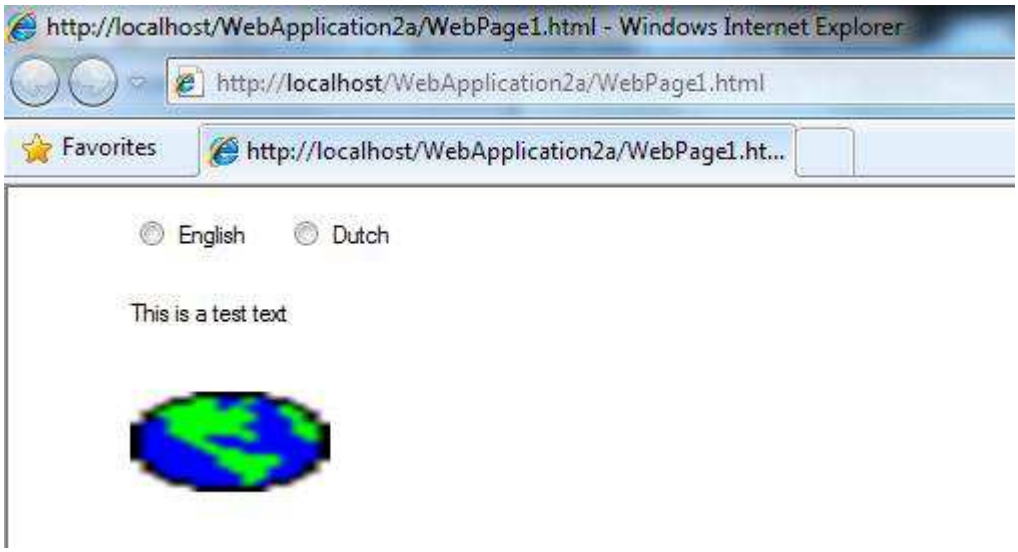
Note that before clicking the Run button to test the web application, you must choose a Build menu to compile the whole project so that the new culture resources can be generated.



Let's test the web application:

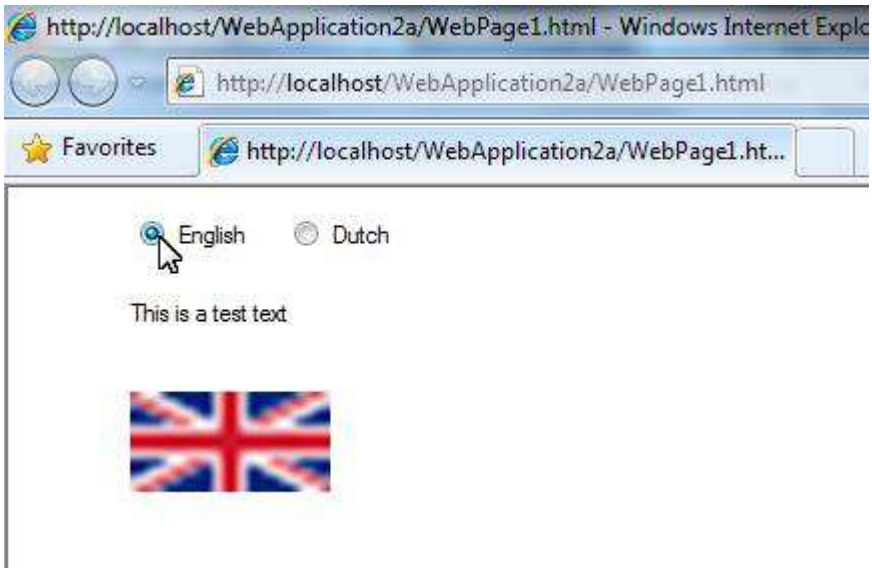


The web page appears in the web browser:



Click the radio button "Dutch". The text and image change to Dutch resources immediately:

Click the radio button "English". The text and image change to English resources immediately:



Use Arbitrary PHP Code and PHP Files

The contents of this section are moved to another file. See
http://www.limnor.com/support/UsePhpCode.pdf

More documents

http://www.limnor.com/support/WebApplicationDevelopment.pdf

http://www.limnor.com/support/WebEditors.pdf

http://www.limnor.com/support/WebHtmlEditorProgramming.pdf

http://www.limnor.com/support/webDatabaseProgramming.pdf

http://www.limnor.com/support/WebDataRepeater.pdf

http://www.limnor.com/support/UsePhpCode.pdf

http://www.limnor.com/support/WebFileupload.pdf

http://www.limnor.com/support/WebEmail.pdf

http://www.limnor.com/support/WebTreeView.pdf

http://www.limnor.com/support/UseJavaScriptFiles.pdf

http://www.limnor.com/support/WebMenu.pdf

http://www.limnor.com/support/WebDialogAndChildPage.pdf

http://www.limnor.com/support/WebDateTime.pdf

http://www.limnor.com/support/HowToUseLargePageSizeAtDesignTime.pdf

http://www.limnor.com/support/WebImageDataBinding.pdf

http://www.limnor.com/support/WebEventBubbling.pdf

http://www.limnor.com/support/HandleLargeNumberOfWebElements.pdf

http://www.limnor.com/support/MultipleWebPageLogins.pdf

http://www.limnor.com/support/WebCamInWebPage.pdf

http://www.limnor.com/support/GoogleMaps.pdf

http://www.limnor.com/support/GoogleMapsControl.pdf

Please visit http://www.limnor.com for new documents.