

# Use JavaScript Files

---

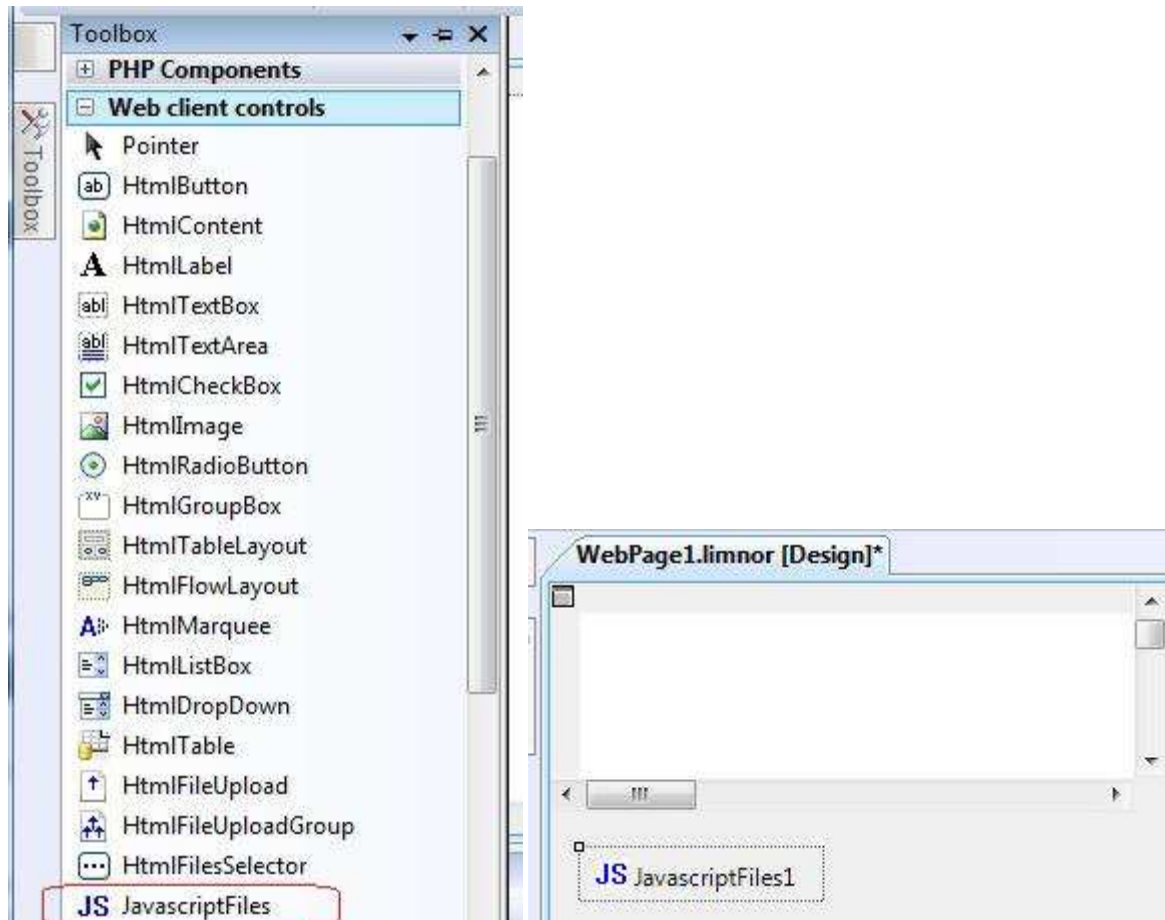
## Contents

Introduction .....	1
Include JavaScript Files .....	2
Access JavaScript Variables.....	4
Execute JavaScript Functions .....	8
Test.....	10
Example: Use CKEditor.....	12
Sample scenario .....	12
Data entry form.....	13
Integrate CKEditor into web application.....	14
Use JavaScriptFiles component.....	15
Execute CKEditor functions.....	15
“Execute” action .....	16
“ExecuteFunction” action .....	17
Close editor .....	19
Cancel editor .....	20
Assign actions to buttons.....	20
Data loading, record navigation and update .....	21
Test.....	23
Feedbacks .....	25

## Introduction

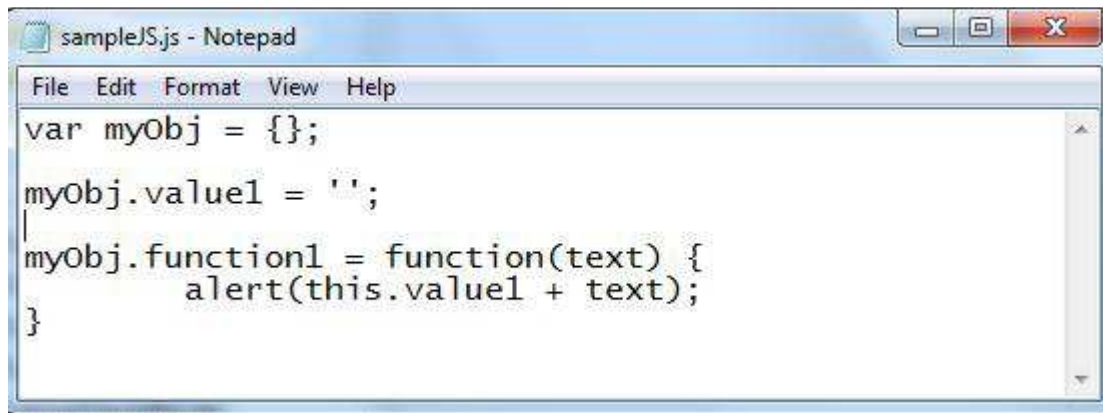
There are huge numbers of JavaScript libraries available on the internet. JavascriptFiles component is designed as an interface to include JavaScript files in web pages for visual and codeless programming. It can be used to read/write JavaScript values defined in the JavaScript files. It can be used to execute functions defined in the JavaScript files.

This document assumes you are familiar with Limnor Studio visual programming. For an introduction to Limnor Studio visual programming, see <http://www.limnor.com/support/UsersGuideForBeginners.pdf>. For information on web application development, see <http://www.limnor.com/support/WebApplicationDevelopment.pdf>. For information on web database application development, see <http://www.limnor.com/support/webDatabaseProgramming.pdf>.

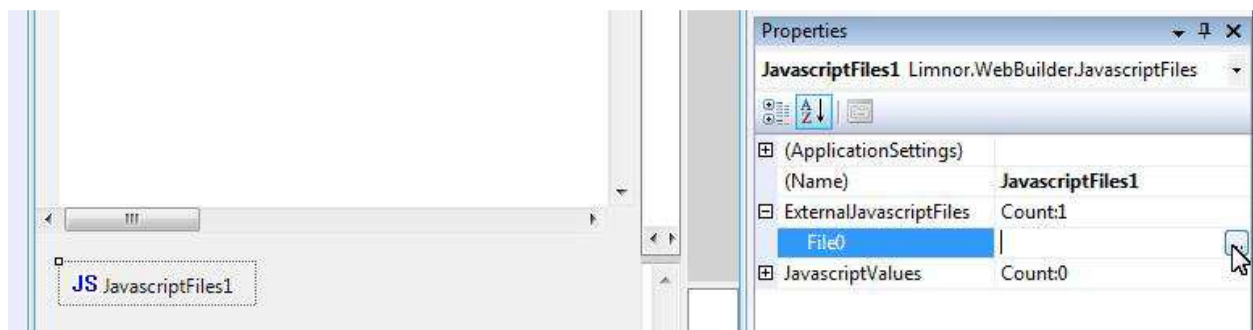


## Include JavaScript Files

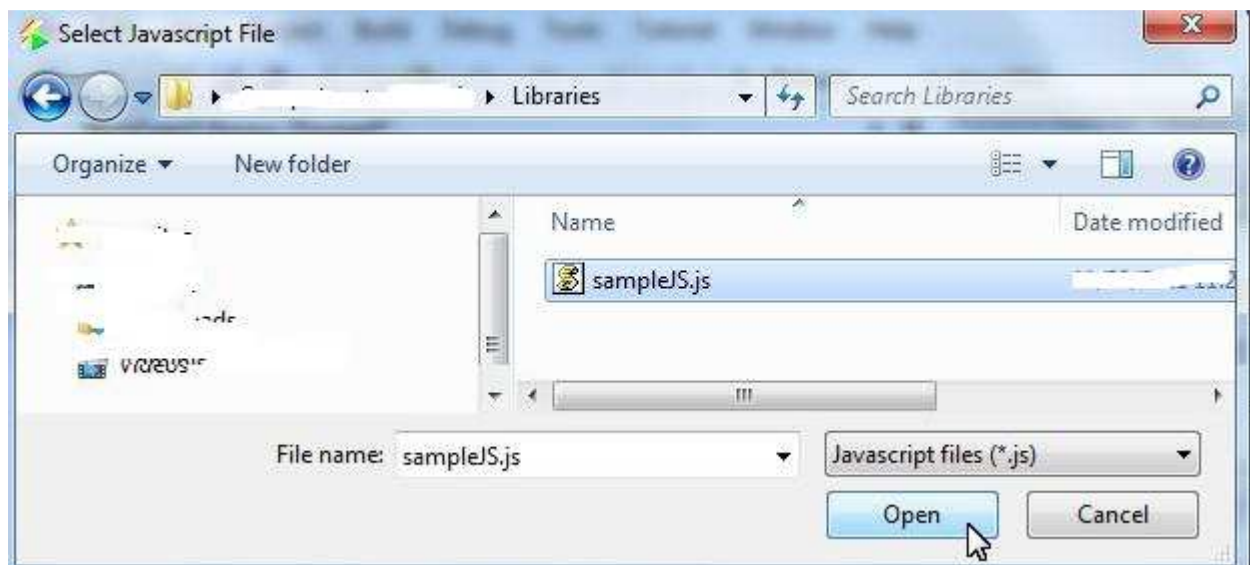
Suppose we have a simple JavaScript file as shown below.



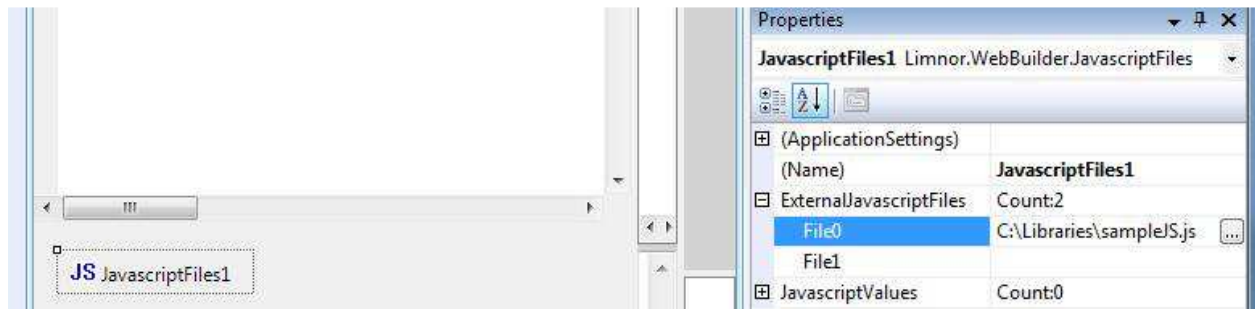
To include this file to the web page, click a “...” button under ExternalJavaScriptFiles:



Select the JavaScript file to be included:

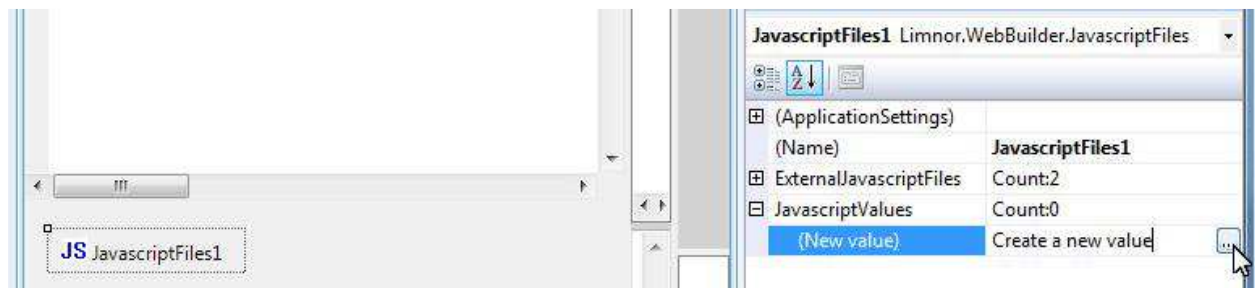


We may include as many JavaScript files as needed.

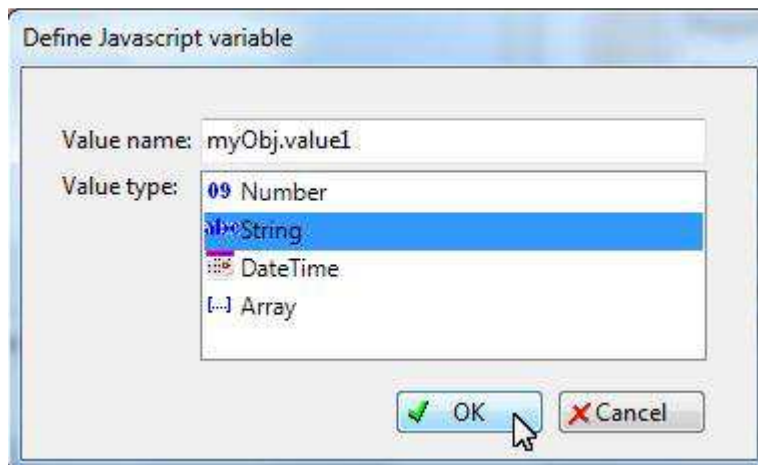


## Access JavaScript Variables

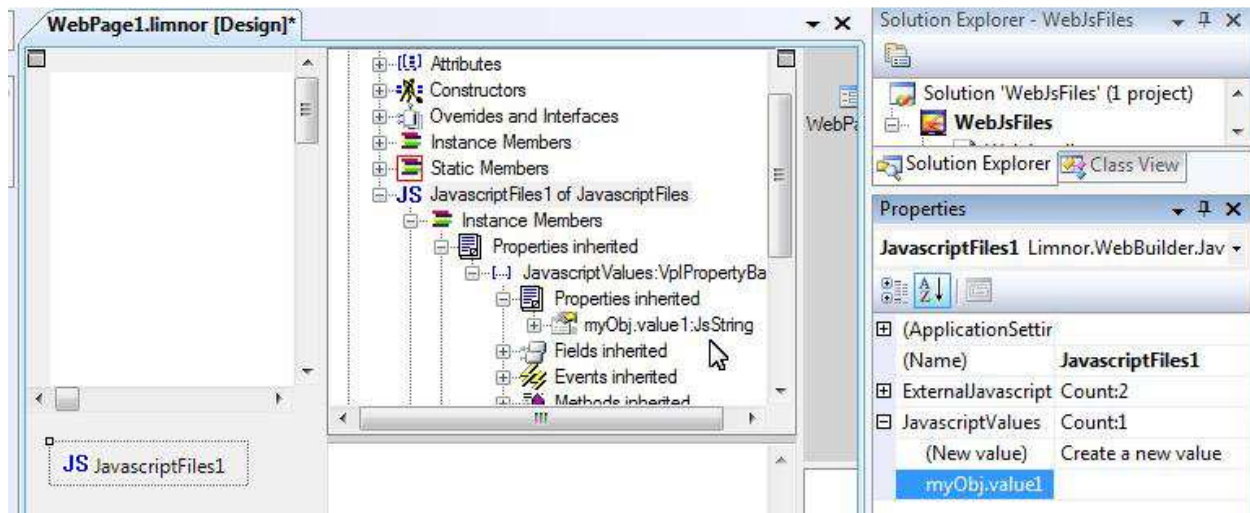
To make a JavaScript value available for visual programming, click button “...” for (New value):



For example, suppose we want to make myObj.value1 available to the visual codeless programming, we may create a value named after it:

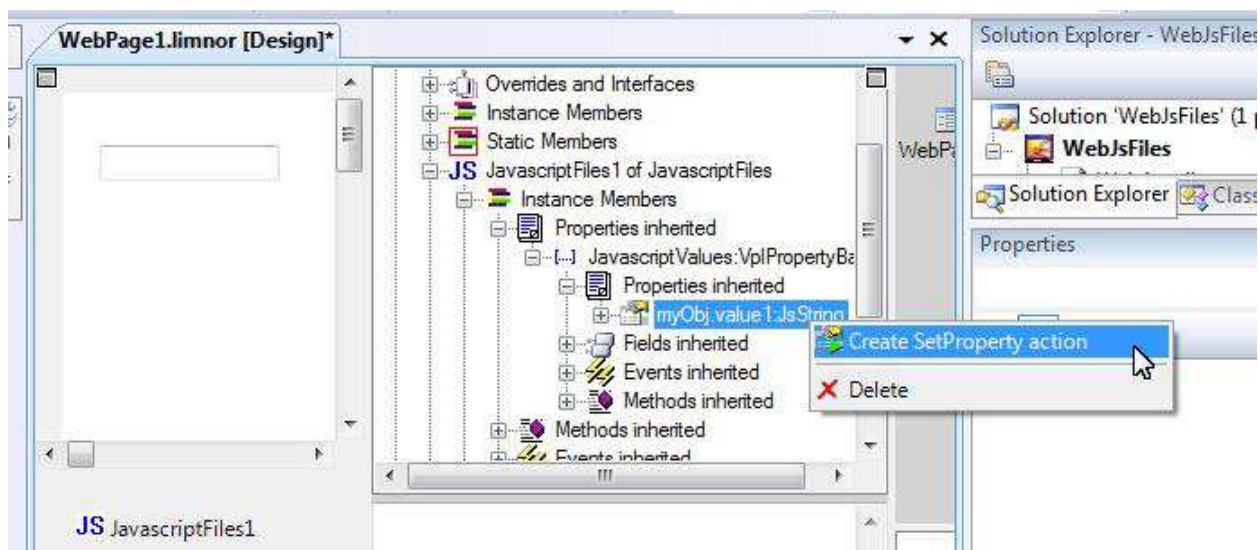


The value will be available to the visual and codeless programming:



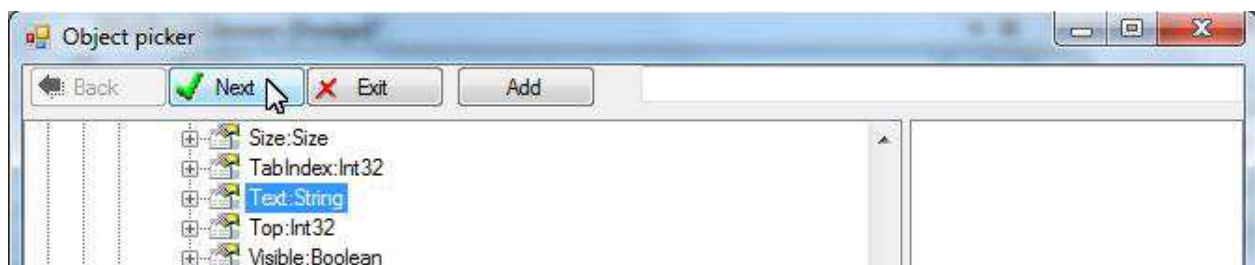
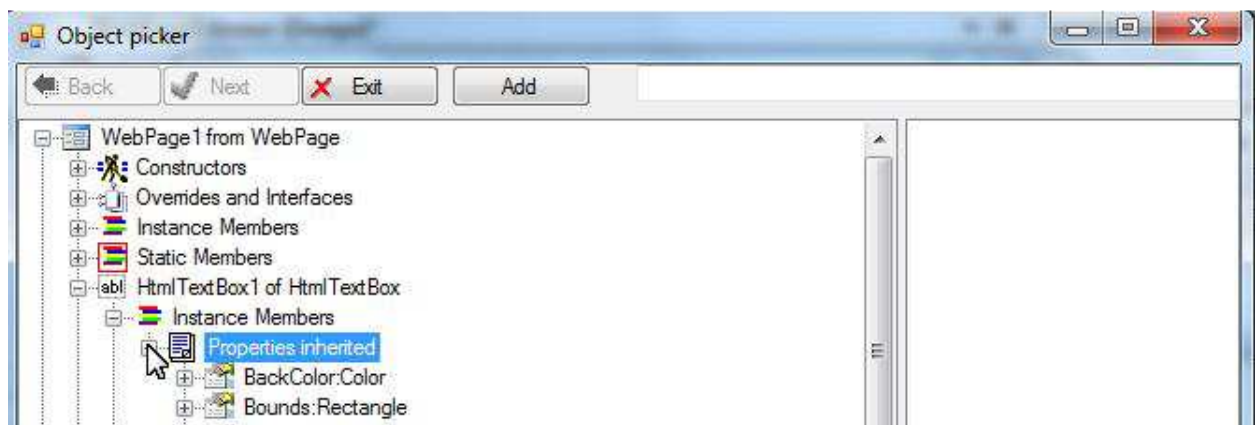
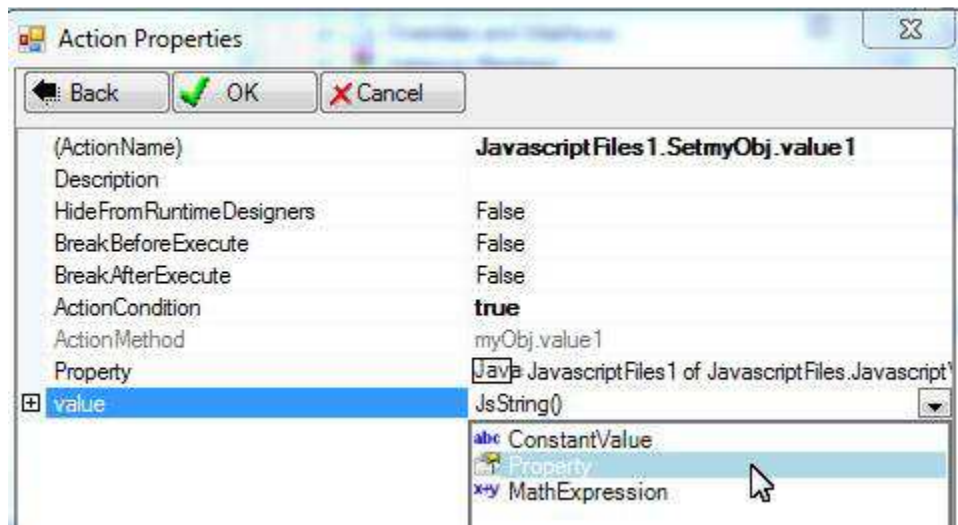
If we set value in the Properties window then the value will be set on loading the web page. Leave the value blank if we do not want to initialize the value.

Like using other properties, we may create actions to set the value at runtime. Right-click “myObj.value1” under JavascriptFiles1; choose “Create SetProperty action”:

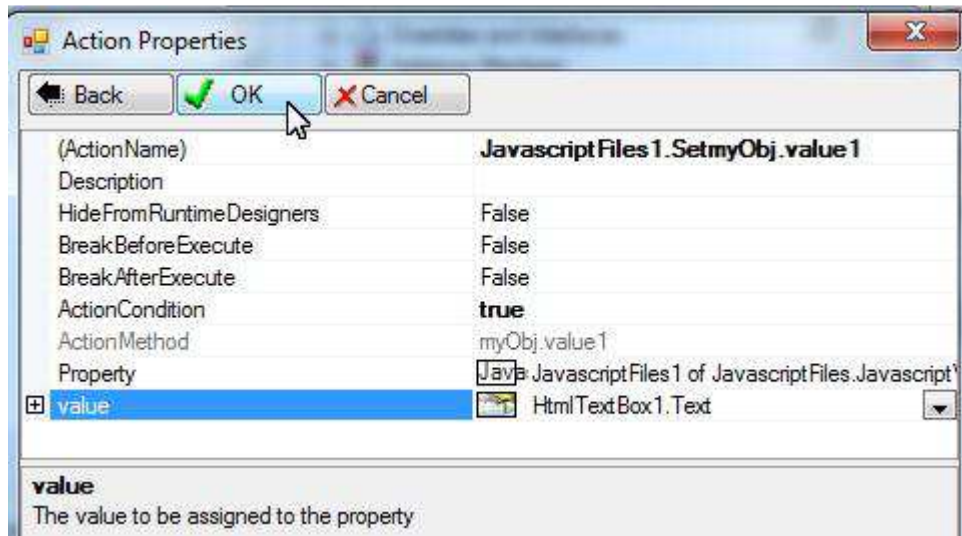


The “value” property of the action is the value we want to assign to myObj.value1. For this sample, we choose Property to select the Text property of a text box:

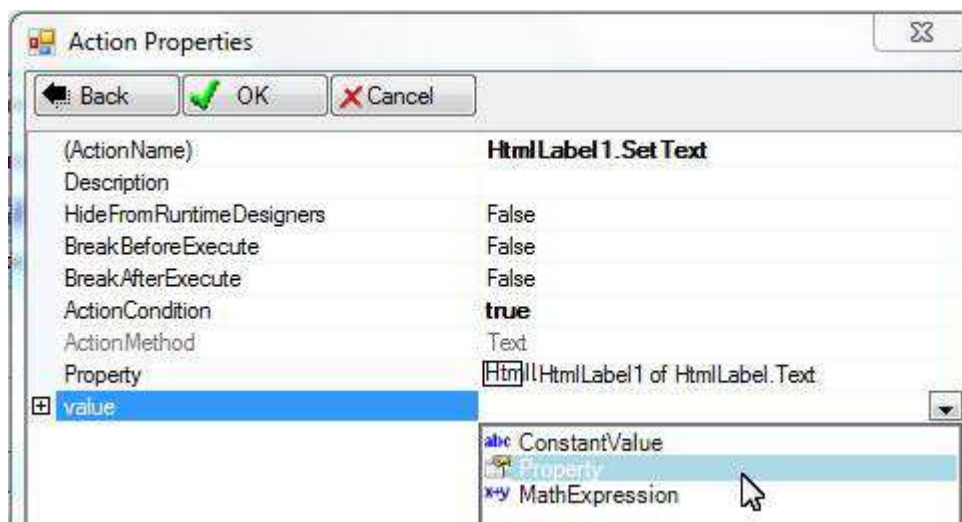
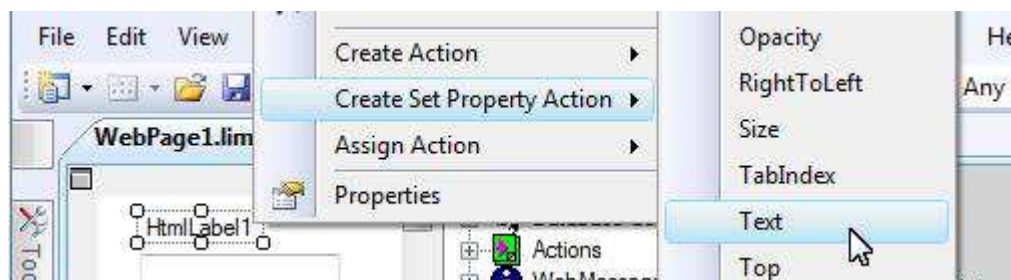


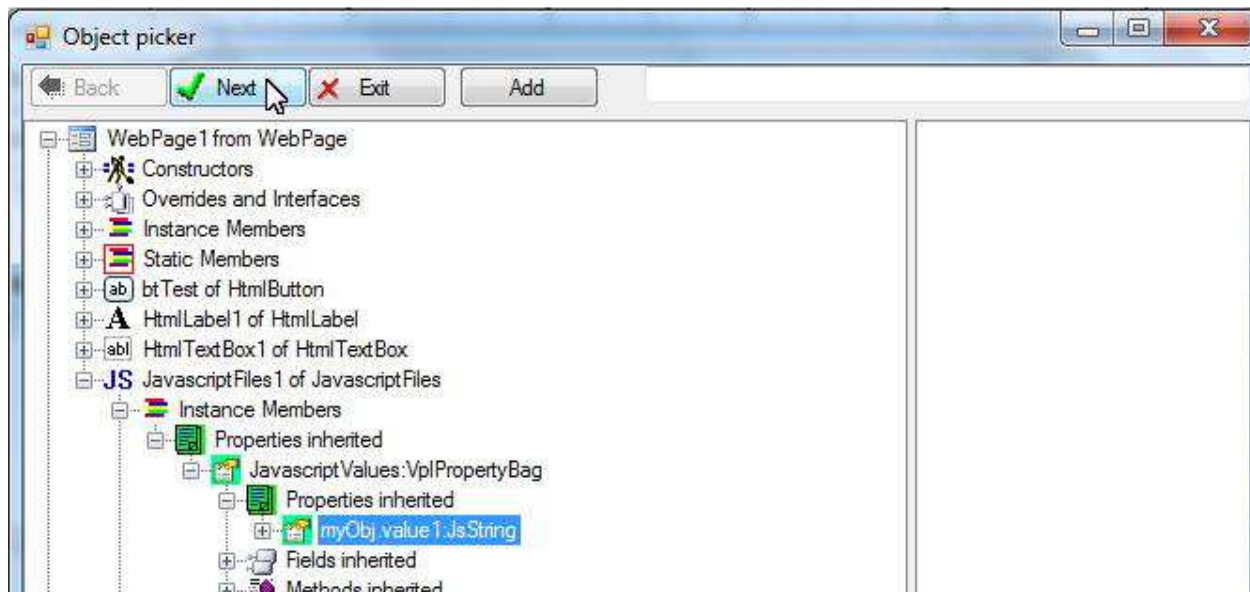


This is such an action:

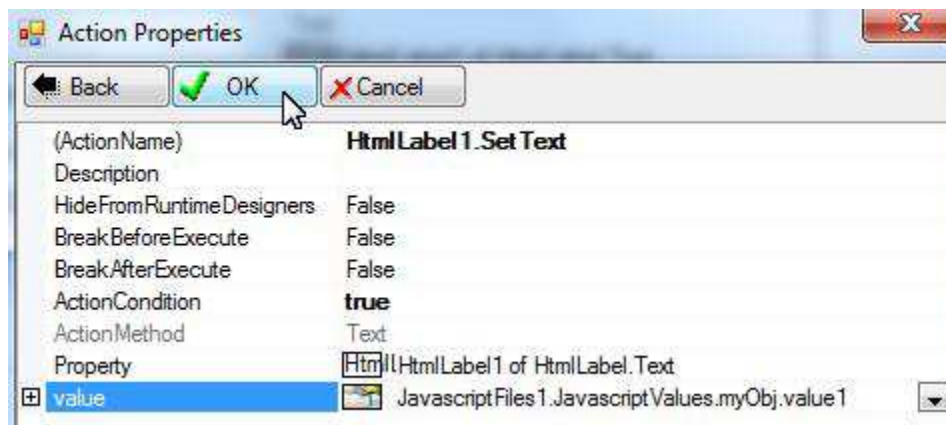


The above action is a sample of writing to the value. To show the reading of the value, we may create an action to assign the value to a Label.





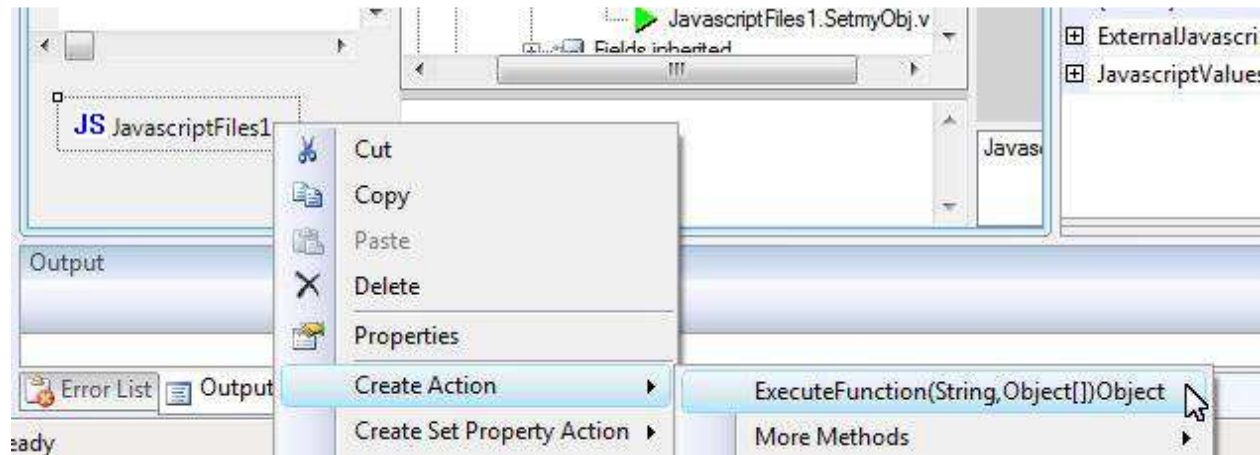
This is an action reading the JavaScript value:



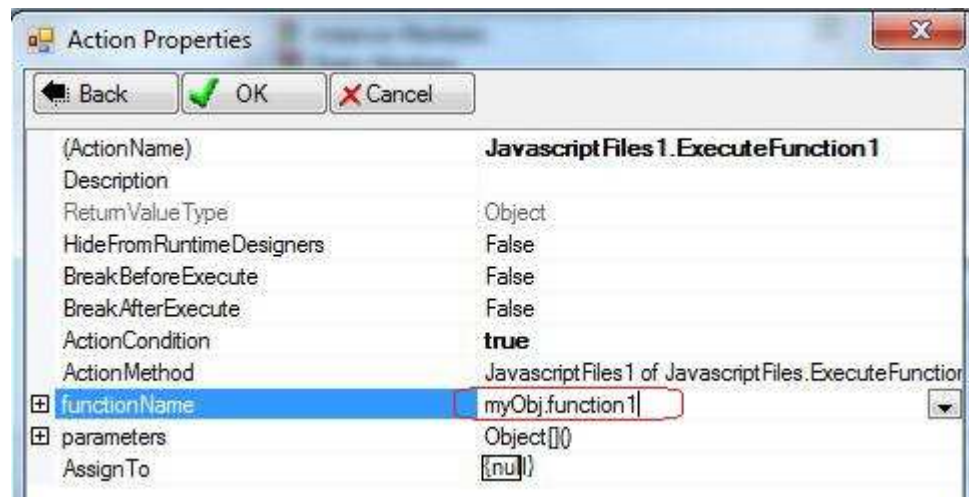
## Execute JavaScript Functions

JavascriptFiles component has an “ExecuteFunction” method which can be used to create actions to execute JavaScript function. In our sample JavaScript file, a function named myObj.function1 is defined. Let’s create an action to execute it.

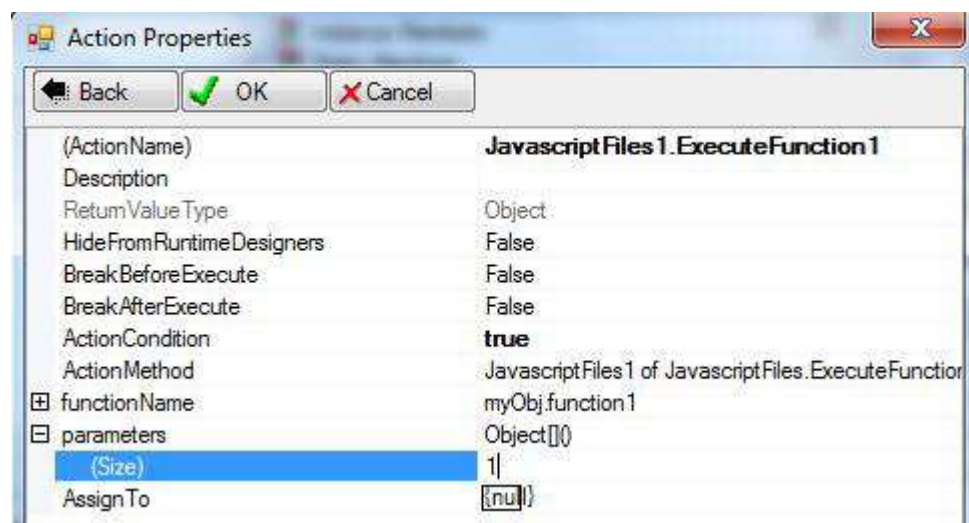




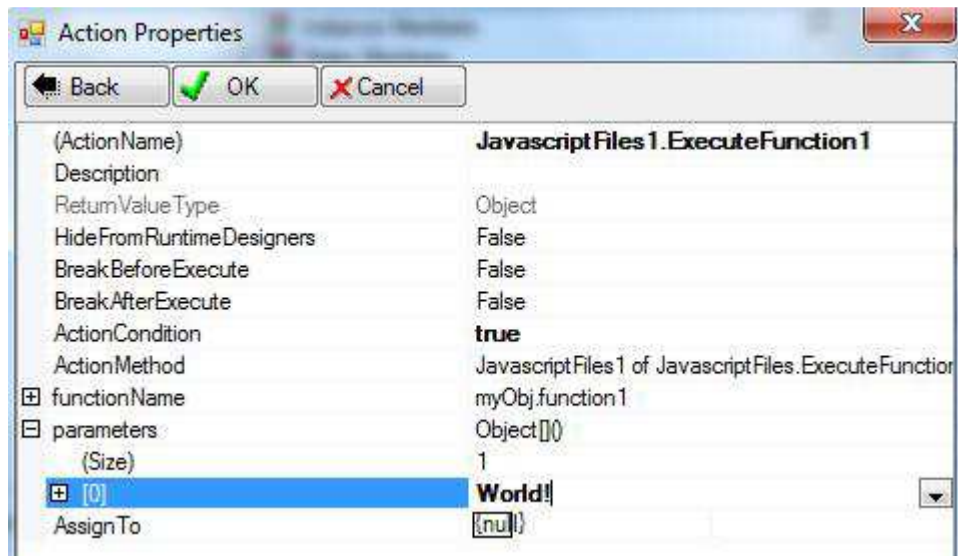
Specify the function name for the action. In this sample, it is myObj.function1:



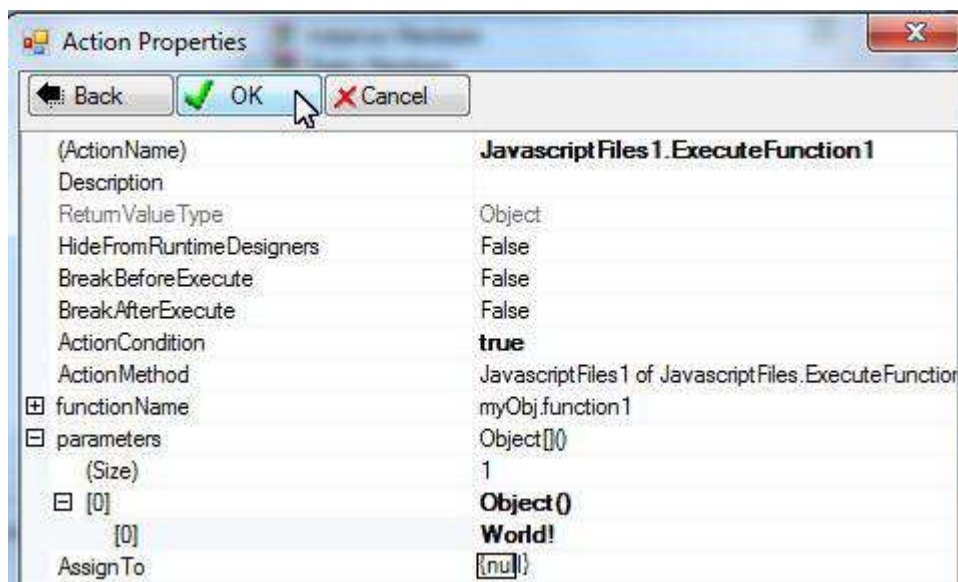
This function needs a parameter. Set the "(Size)" of the "parameters" of the action to the number of the parameters we want to supply. In this case, we may set it to 1:



On setting the size, we may set the value for each parameter.

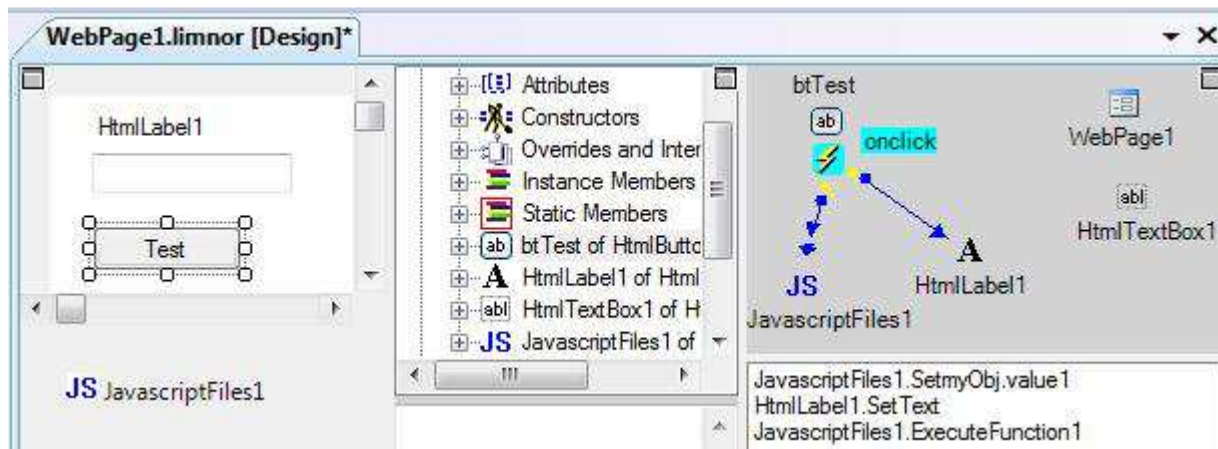


If the function returns a value then we may set “AssignTo” of the action to pass the value to where we want. In this case the function does not return a value. We may click OK to finish creating this action:



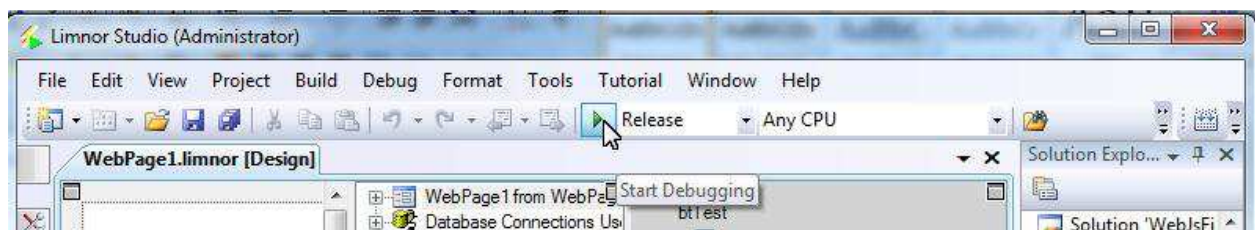
## Test

To test the above actions, we may add a button and assign the actions to the button.

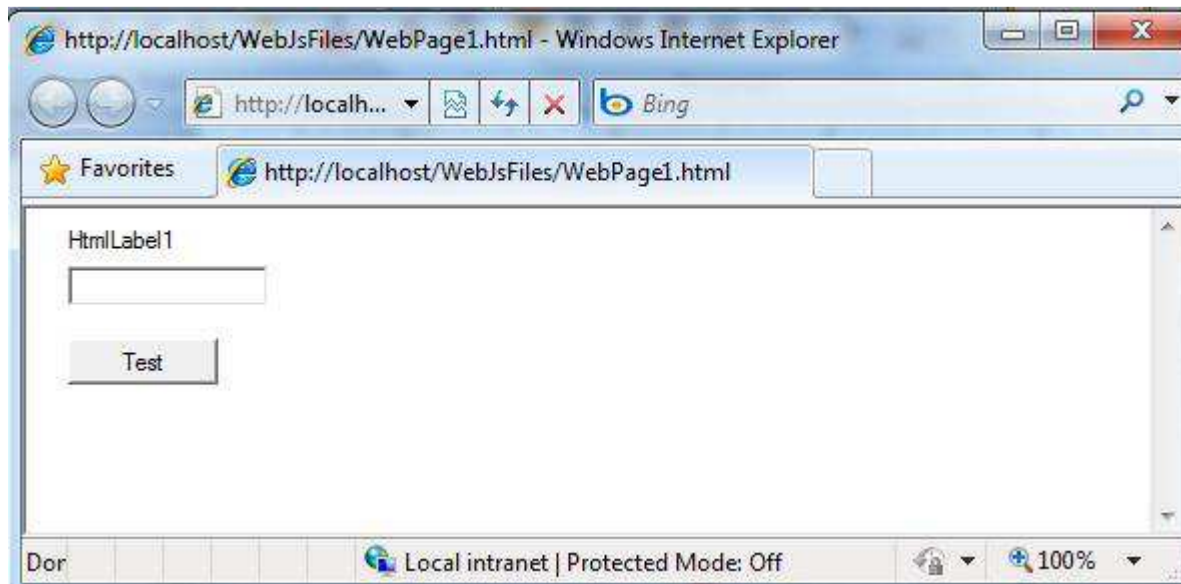


Note the action execute sequence. "SetmyObj.value1" is executed first because the next 2 actions will use the value, myObj.value1.

Click Run button to run the web application:

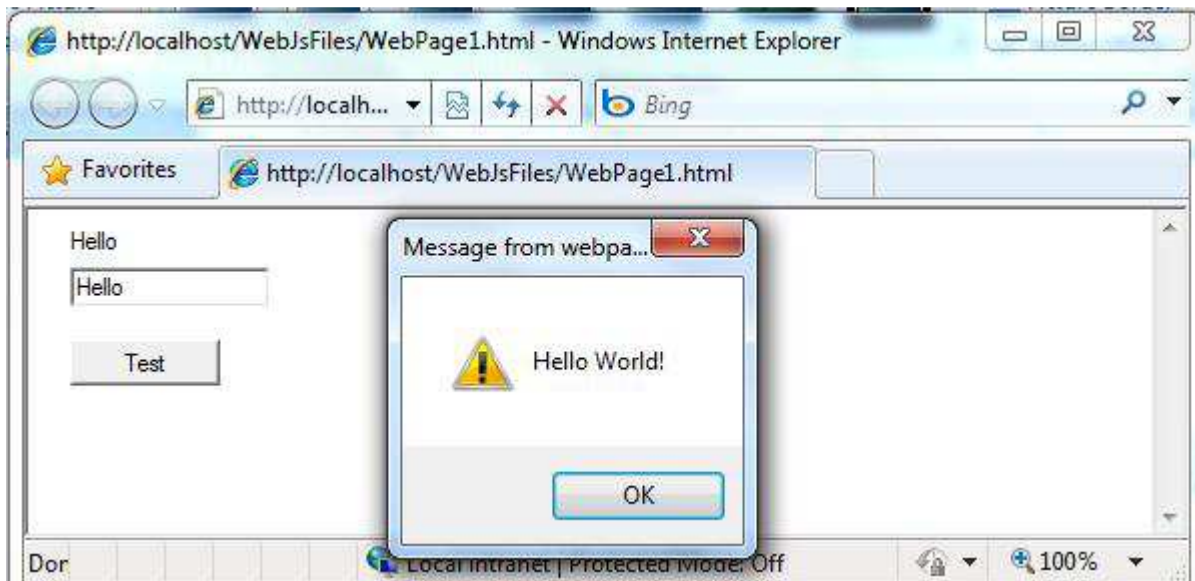


The web page appears:



Type some text in the text box and click the button:

A message box appears:



This is the result of executing function `myObj.function1`, which is the last action. “Hello “ is the text we pass into the function from the text box. “World!” is from the value `myObj.value1`. It is set to “World!” by the first action.

The Label also shows “Hello”. It is the result of executing the second action which reads the JavaScript value, `myObj.value1`.

## Example: Use CKEditor

CKEditor is a JavaScript library for editing HTML. It brings rich editing features to your products and web sites, providing powerful tools to your users with ease. As an example of using JavaScript libraries in web applications via Limnor Studio, we will show using CKEditor in a sample web page. For more information on CKEditor, see <http://cksource.com/ckeditor>

We are not experts in using CKEditor. The purpose of the example is not teaching how to use CKEditor. The purpose of the example is to show how JavaScript libraries can be used.

For more samples of using JavaScript libraries see Webcam sample and Google Maps sample:

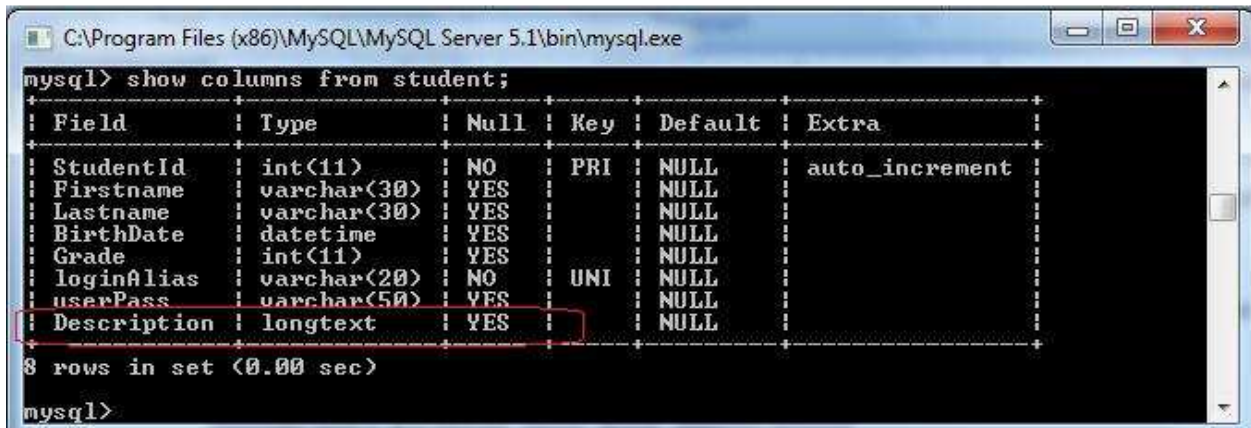
<http://www.limnor.com/support/WebCamInWebPage.pdf>

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

## Sample scenario

Suppose we have a database table for “student” records.

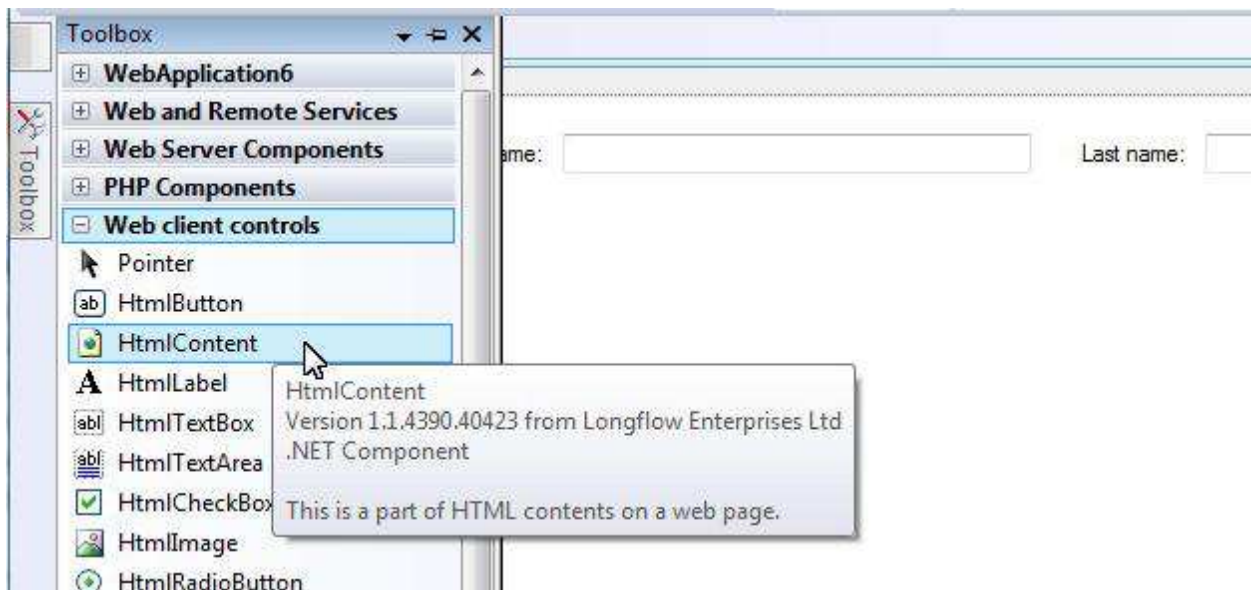




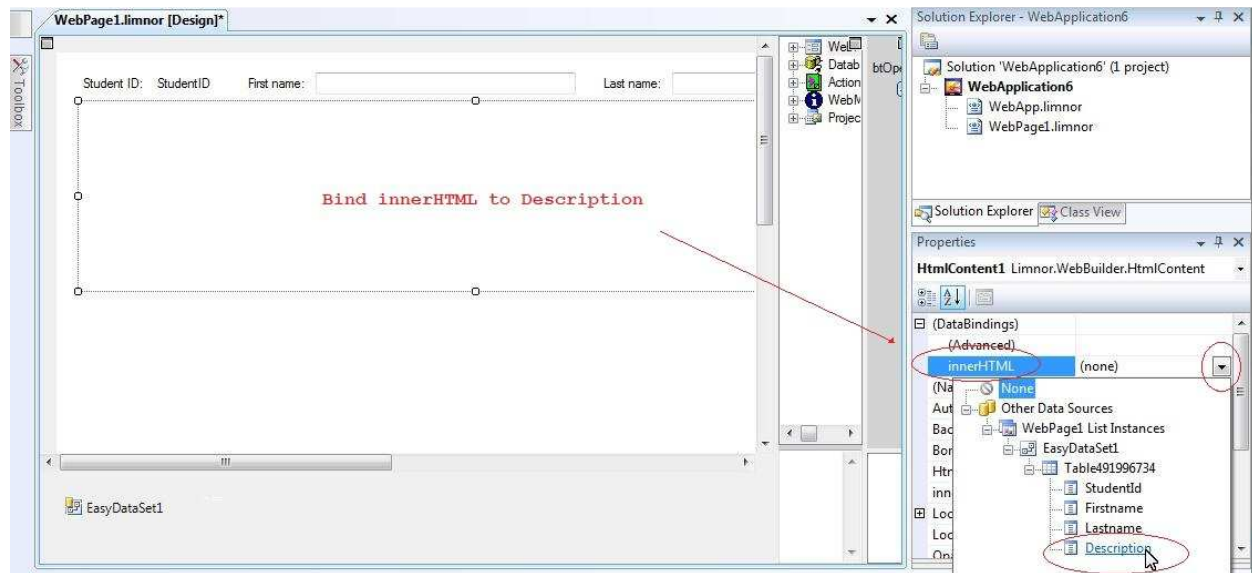
It has a “Description” field for showing HTML contents. We may use an HtmlContent control binding to the “Description” field for displaying and editing the HTML contents. Suppose we want to use CKEditor to let the web page visitors edit the HTML contents of the HtmlContent control. We will use a button to start the CKEditor and use another button to close the CKEditor, based on the documentation of the CKEditor.

### Data entry form

We design a simple data entry form for entering student records. An EasyDataSet is used to connect to a database containing a “student” table. An HtmlLabel is bound to “StudentId”; two HtmlTextBox controls are bound to “Firstname” and “Lastname” respectively. An HtmlContent is bound to “Description”:

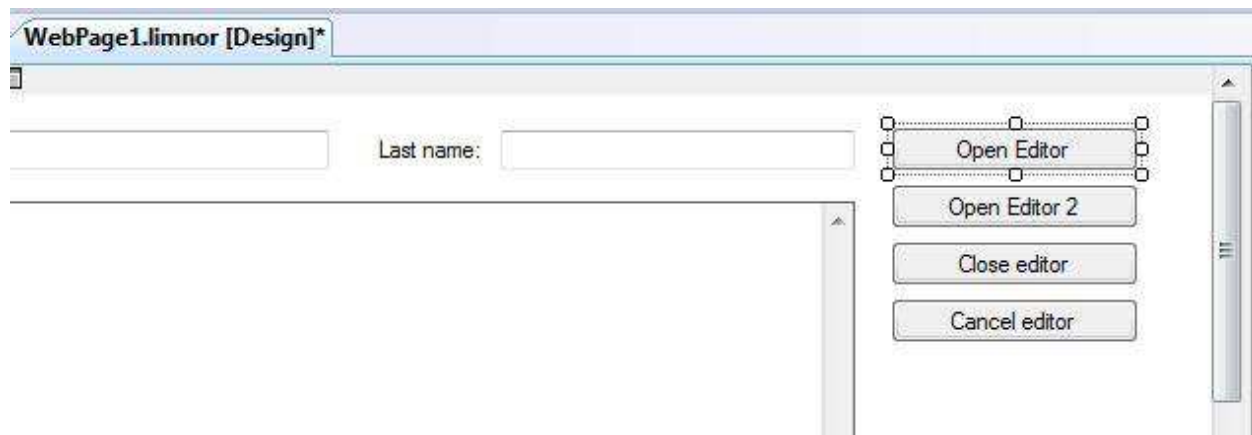






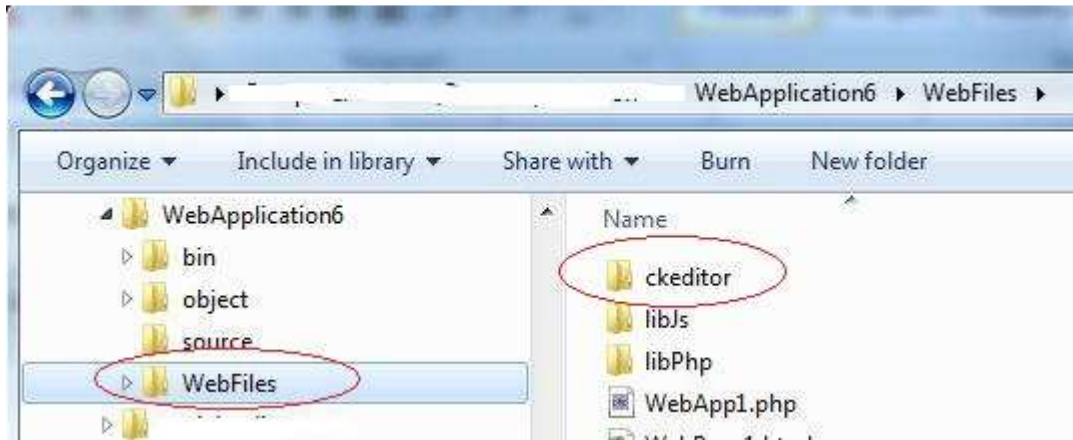
For details of using EasyDataSet to connect to database and do data-binding, see <http://www.limnor.com/support/webDatabaseProgramming.pdf>

We use buttons to open and close CKEditor:



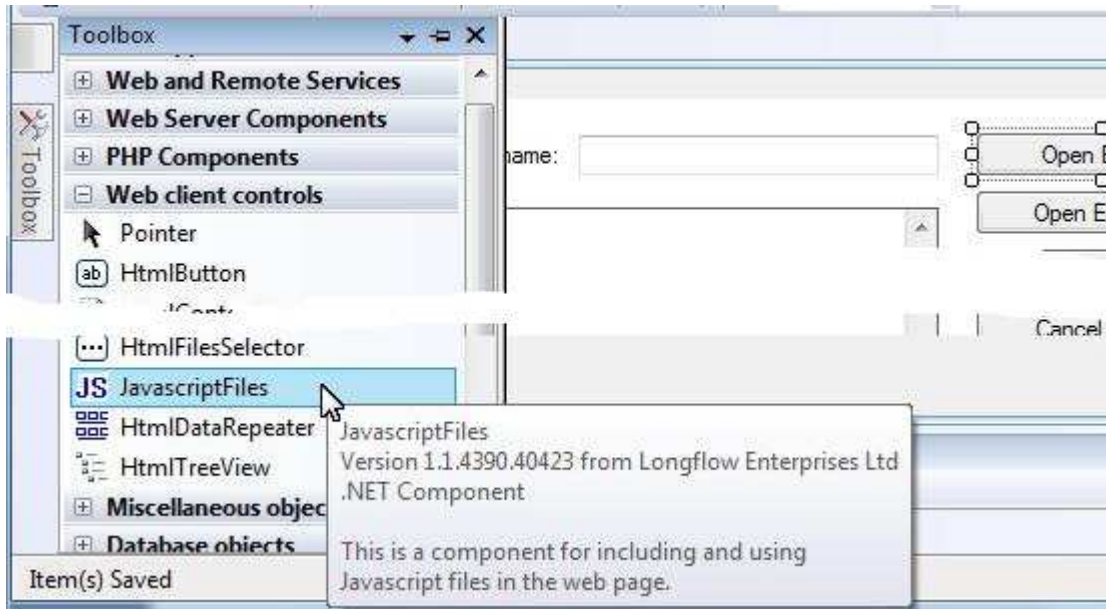
## Integrate CKEditor into web application

According to the documentation of the CKEditor, the files of the CKEditor should be copied to a folder under the web root. In a web project, the "WebFiles" folder under the project folder represents the web root. So, we need to copy the files of the CKEditor to the "WebFiles" folder:

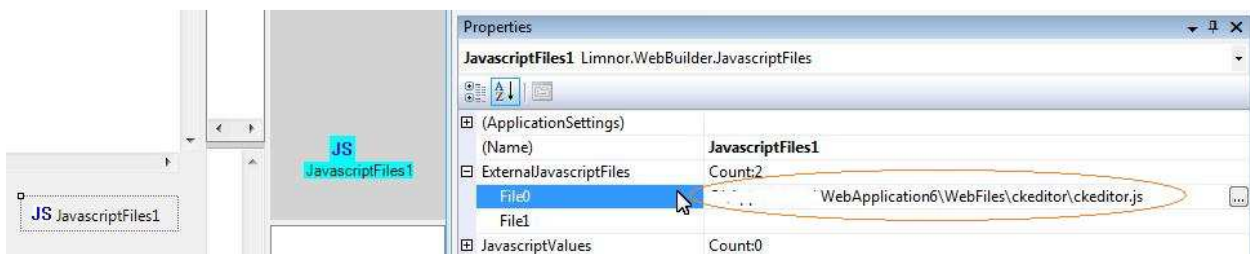


## Use JavaScriptFiles component

To use JavaScript libraries, add a JavaScriptFiles component to the page:

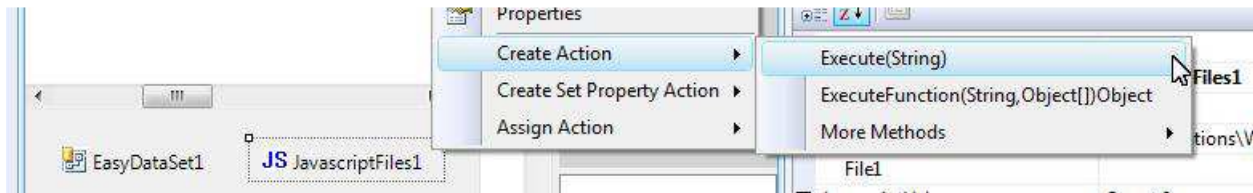


Include CKEditor file, ckeditor.js, in the JavaScriptFiles component:



## Execute CKEditor functions

Both Execute and ExecuteFunction can be used to execute JavaScript functions:



Execute is used to execute JavaScript code. If you like to write your own JavaScript code then you may create an Execute action and provide your own JavaScript code.

One ExecuteFunction action is used to execute one JavaScript function. You provide function name and parameters to the action. Limnor Studio compiler will generate the JavaScript code for the action.

We will show examples of both Execute action and ExecuteFunction action.

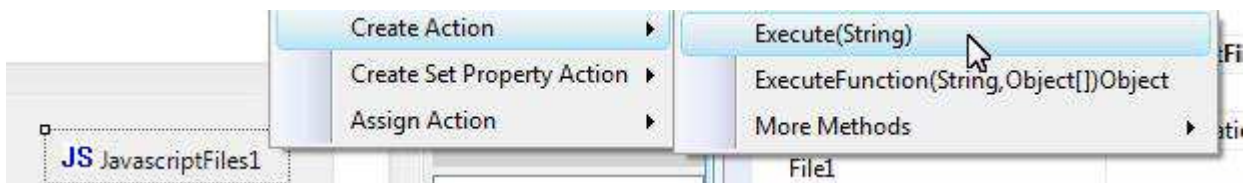
According to the CKEditor documentation, to start editing the HtmlContent, we need to execute following command

```
CKEDITOR.replace( {div element} );
```

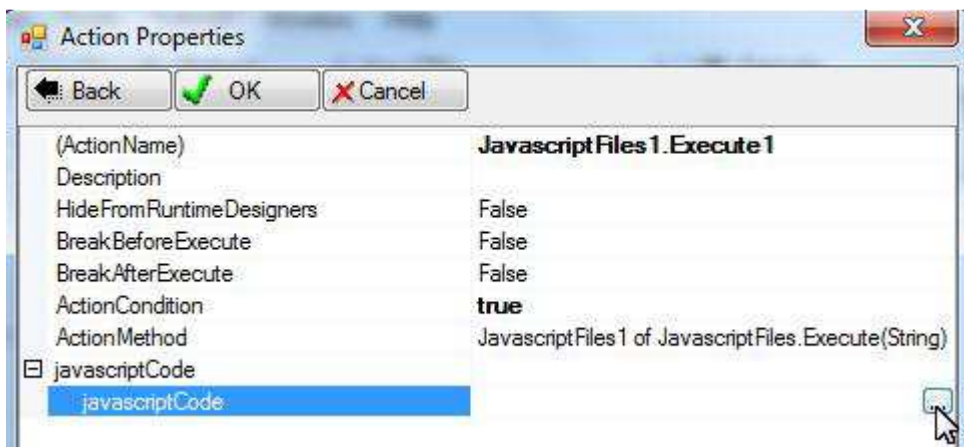
where {div element} can be an id of a div element or the div element itself. Let's use both Execute and ExecuteFunction to demonstrate of executing above command.

## "Execute" action

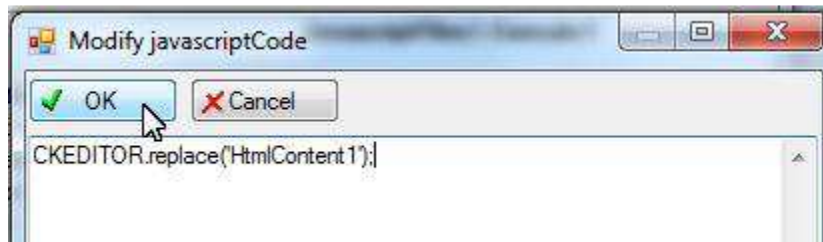
Create an Execute action:



Set javascriptCode parameter of the action:

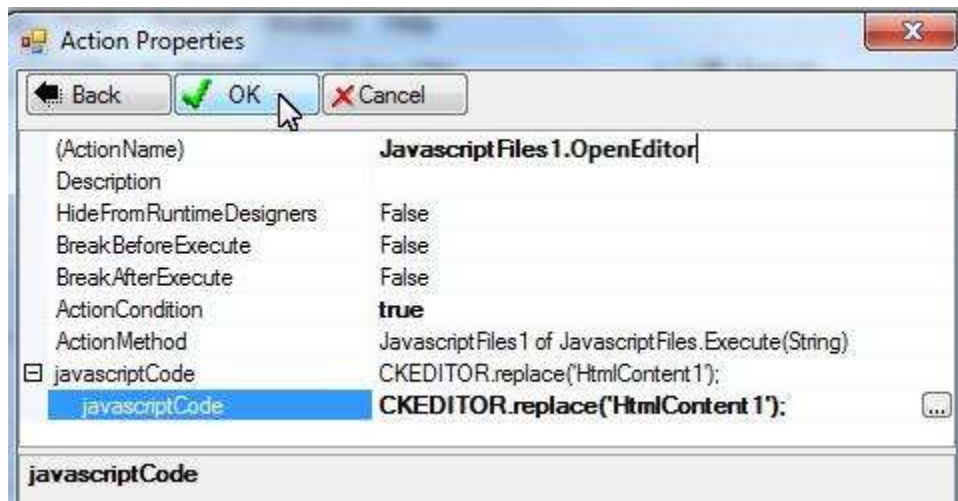


Enter the JavaScript code the CKEditor requires:



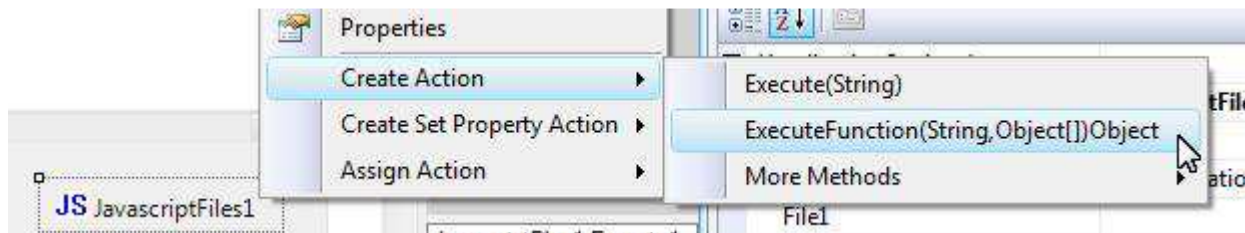
Where 'HtmlContent1' is the name of our HtmlContent control.

Change the action name and click OK to finish creating this action:

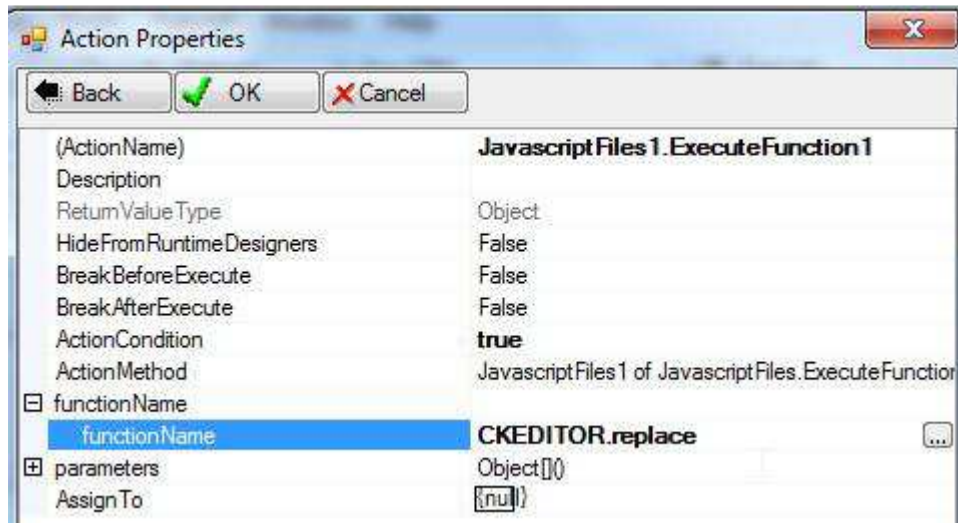


### “ExecuteFunction” action

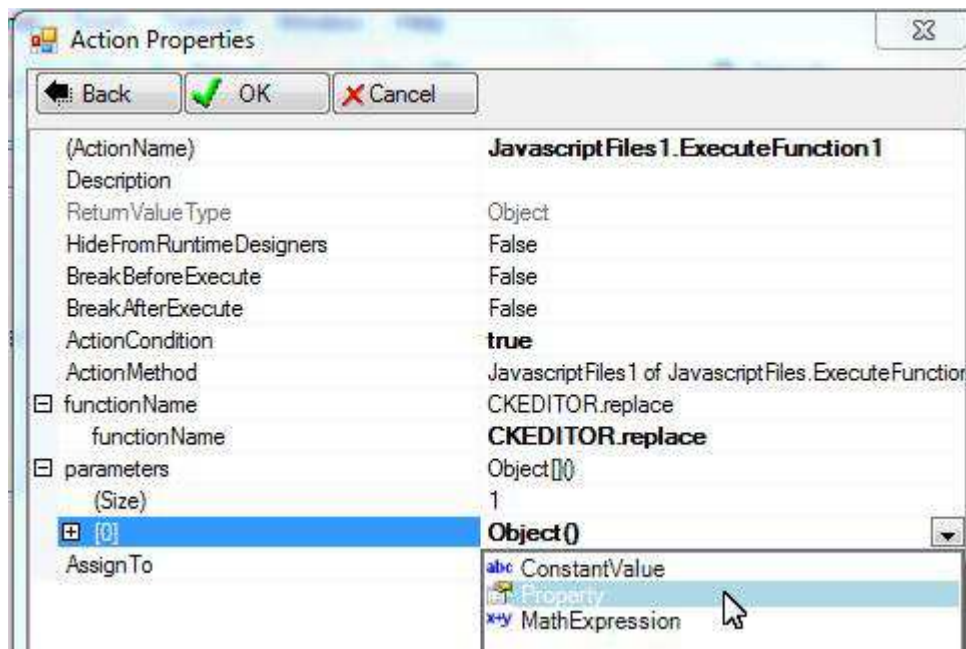
Create an ExecuteFunction action:



Enter CKEDITOR.replace for functionName:

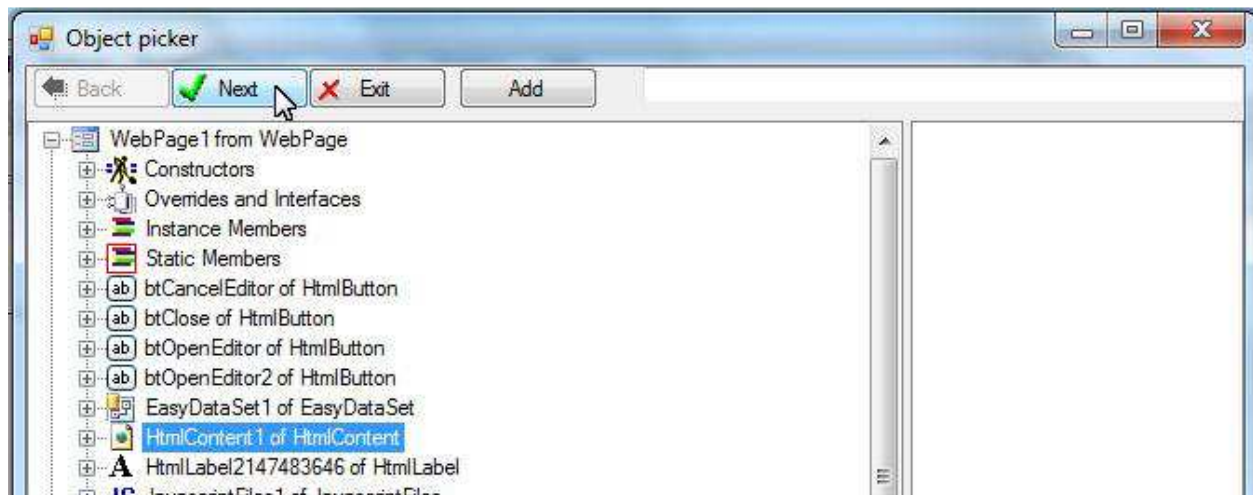


Set Size of the parameters to 1. Select Property for parameter 0:

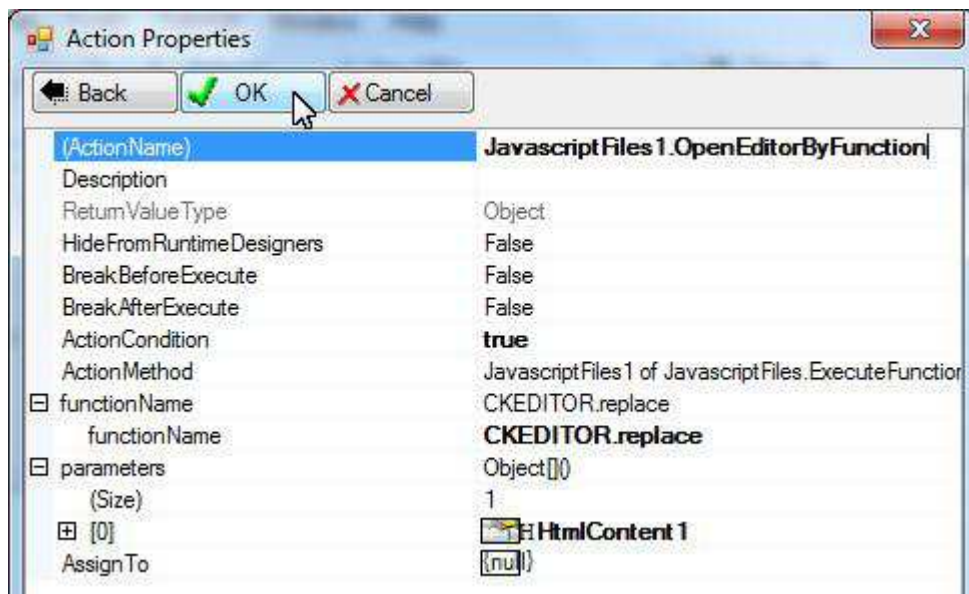


Select the HtmlContent for the parameter:





Change the action name and click OK:

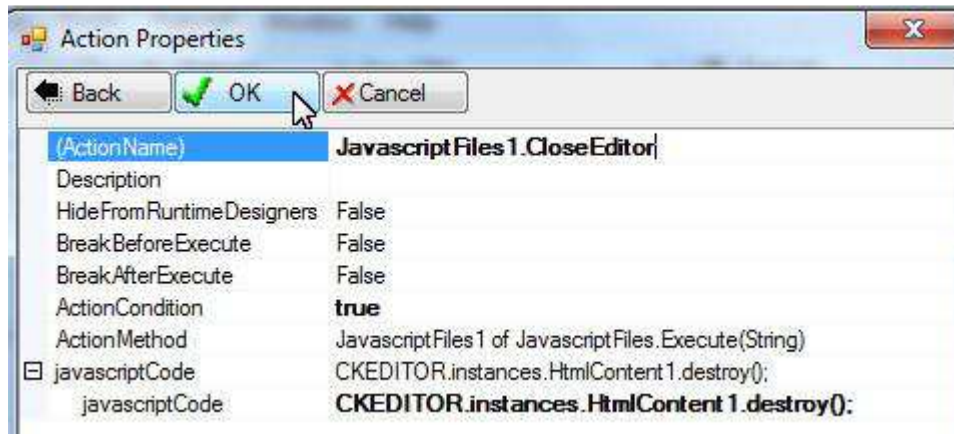


### Close editor

According to the CKEditor documentation, to close the editor and save the contents to the html element, the following command should be executed:

```
CKEDITOR.instances.HtmlContent1.destroy();
```

We may create an "Execute" action to execute the above code:

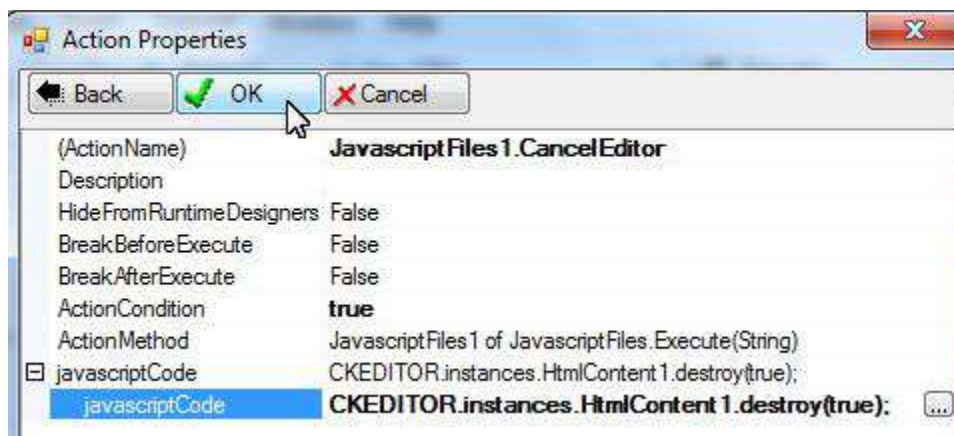


### Cancel editor

According to the CKEditor documentation, to cancel the editor and discard the modifications, the following command should be executed:

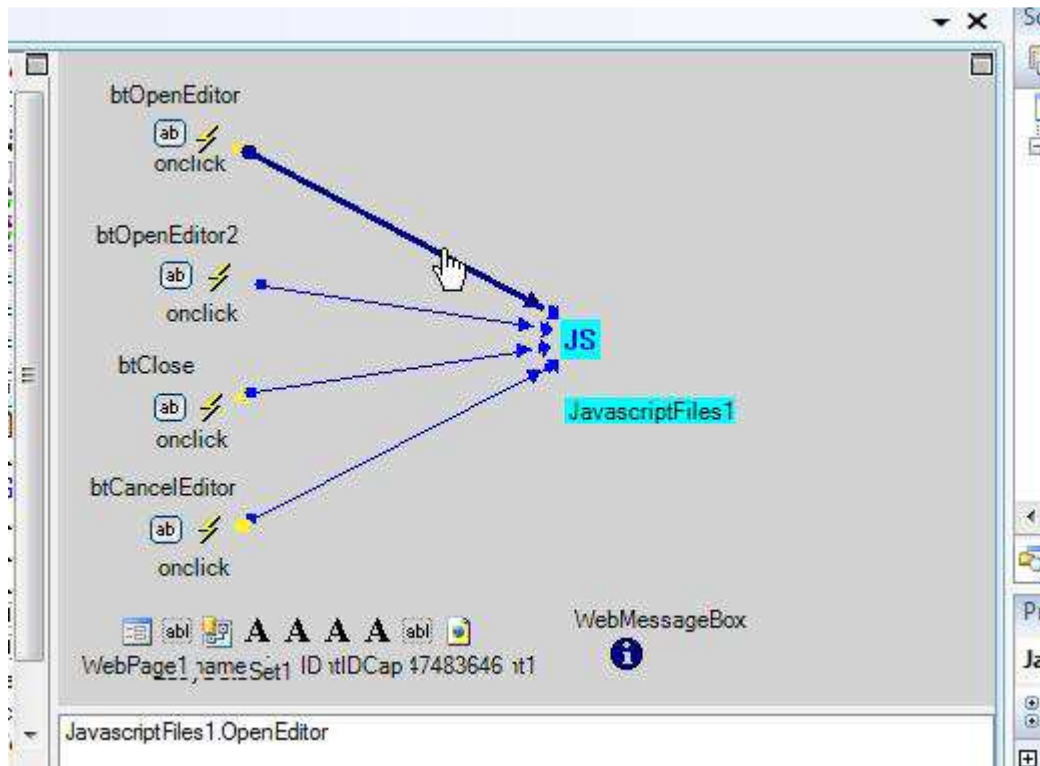
```
CKEDITOR.instances.HtmlContent1.destroy(true);
```

We may create an "Execute" action to execute the above code:



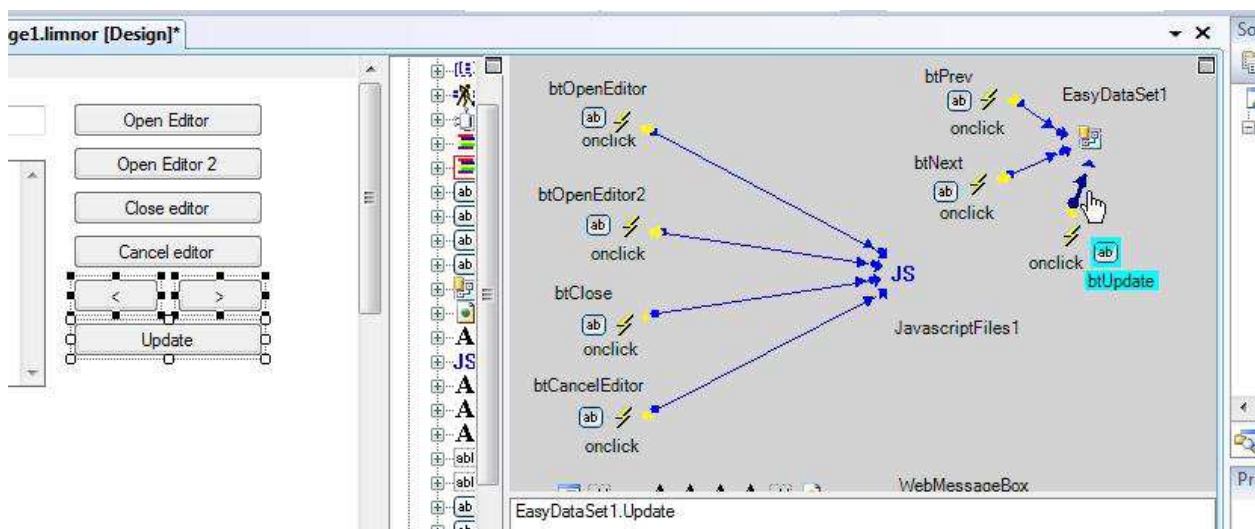
### Assign actions to buttons

Assign the above actions to buttons:

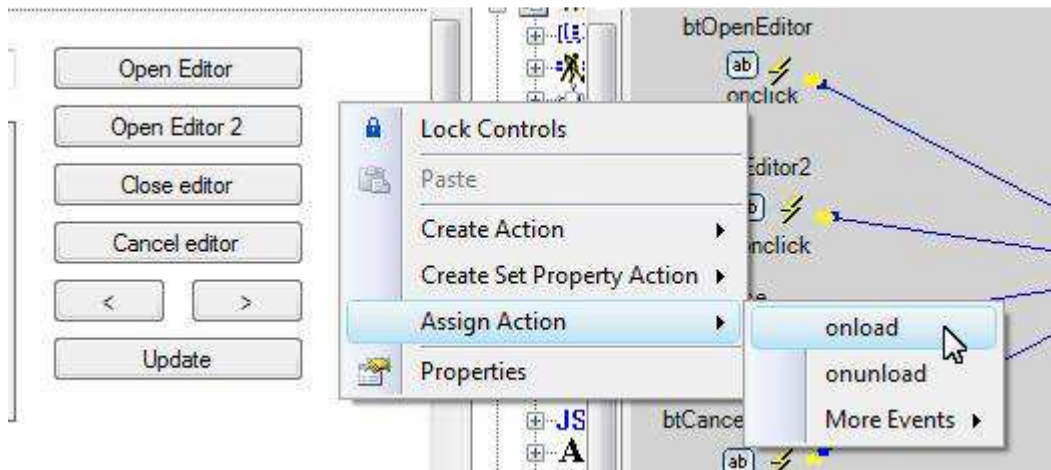


### Data loading, record navigation and update

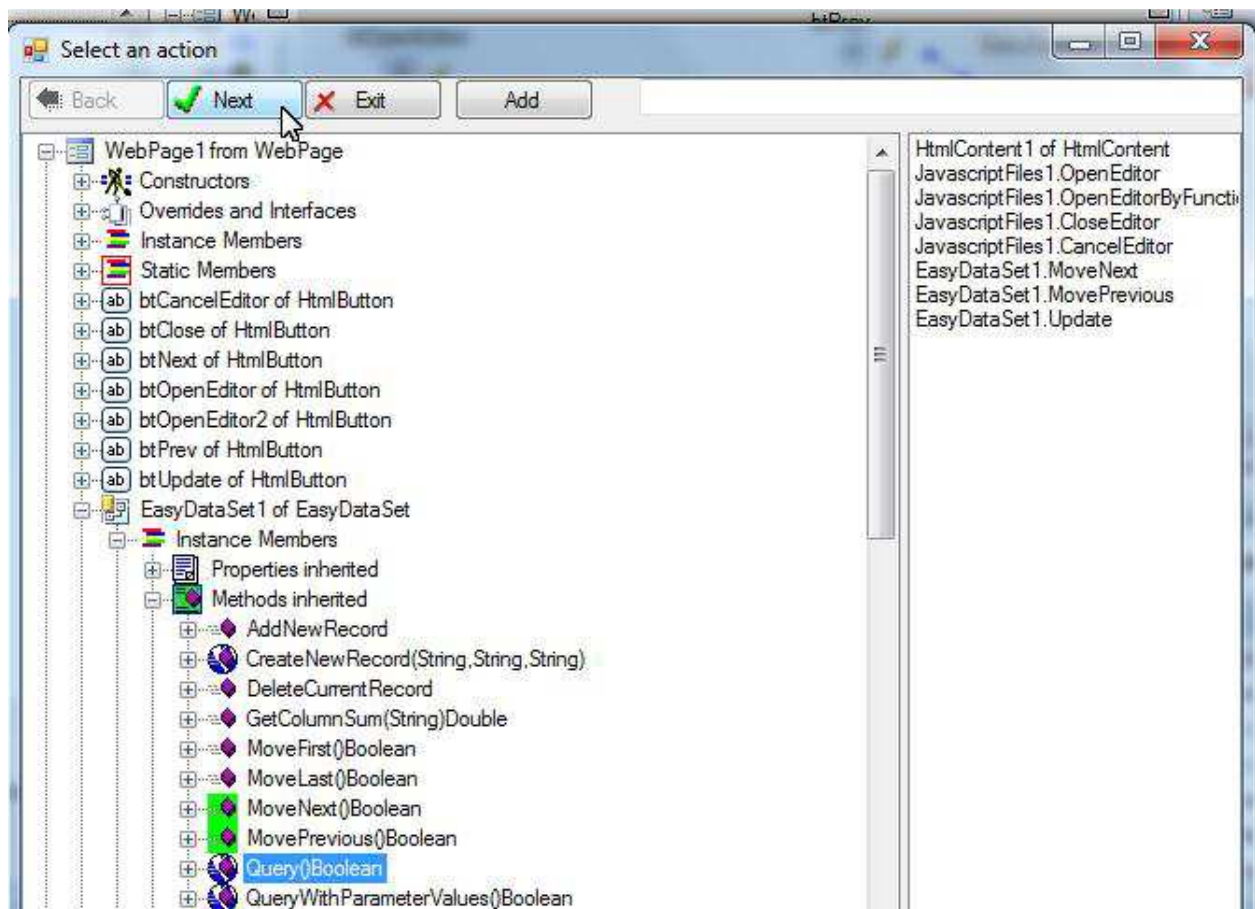
We use buttons to execute record navigations and database update:



We want to load data from the database when the web page is loaded. We may execute a Query action at the event of page onload. Right-click the page; choose "Assign Action"; choose "onload" event:

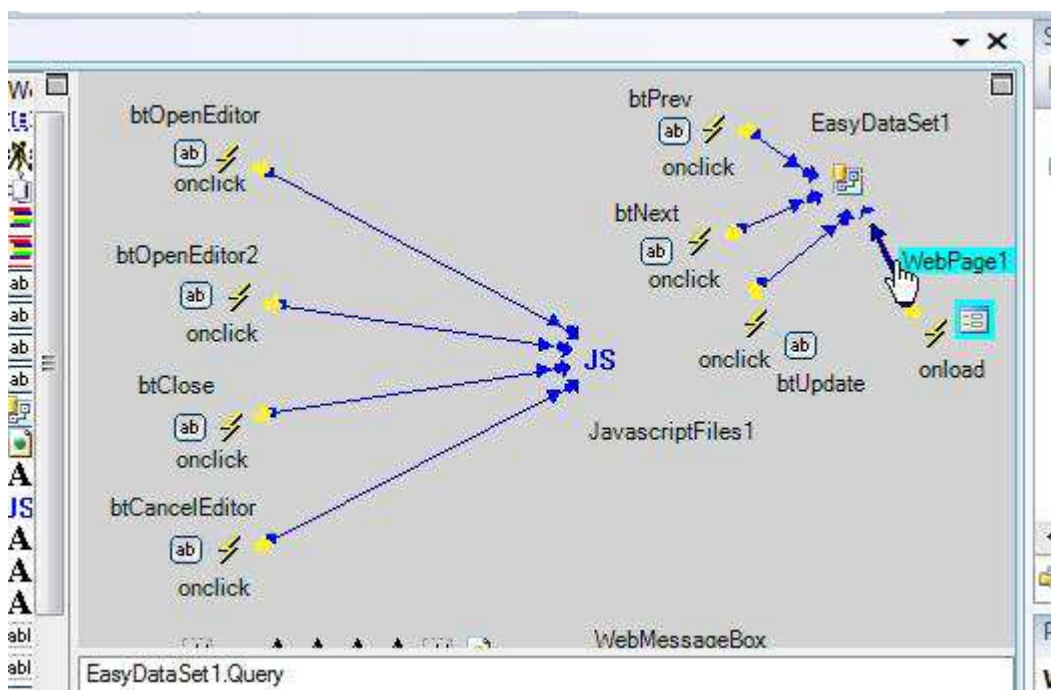
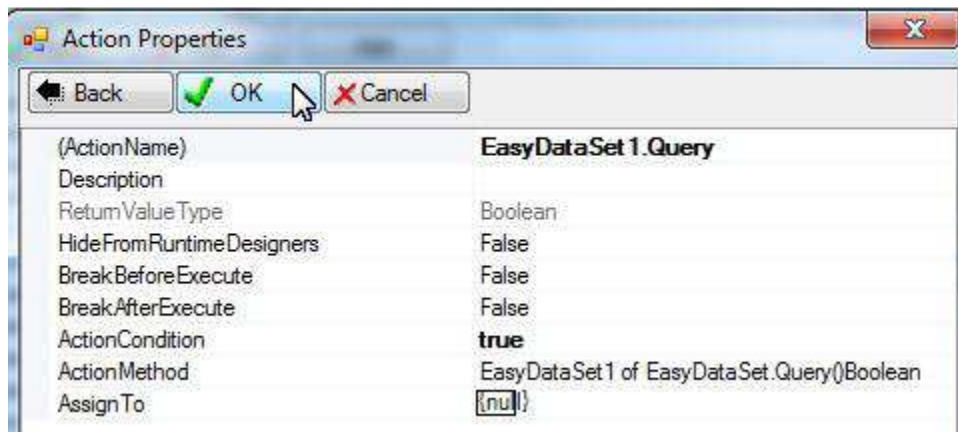


Select Query method of the EasyDataSet:



Click OK. A Query action is created and assigned to the page onload event:

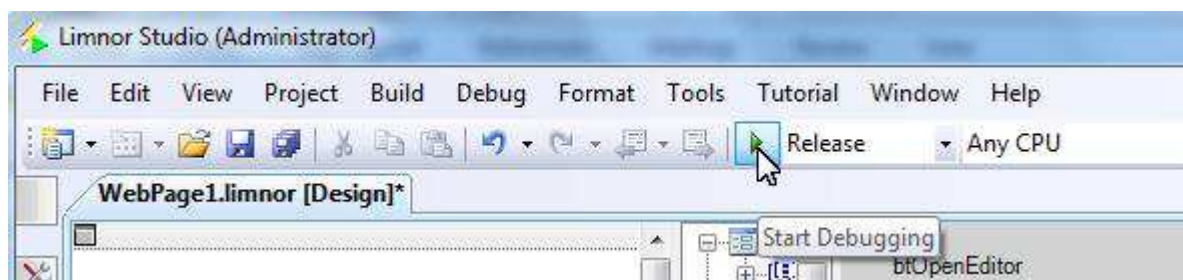






The above database programming has nothing to do with using JavaScript libraries. For details of database programming, see <http://www.limnor.com/support/webDatabaseProgramming.pdf>

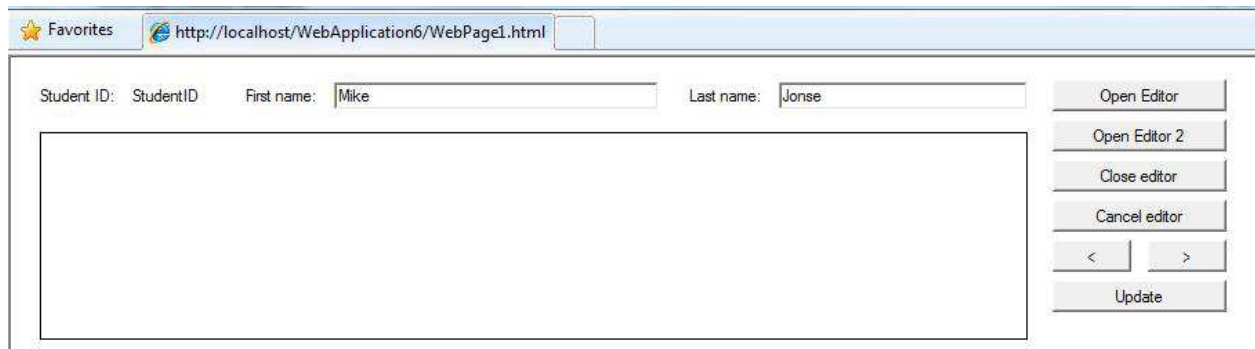
## Test

We may test this sample now.

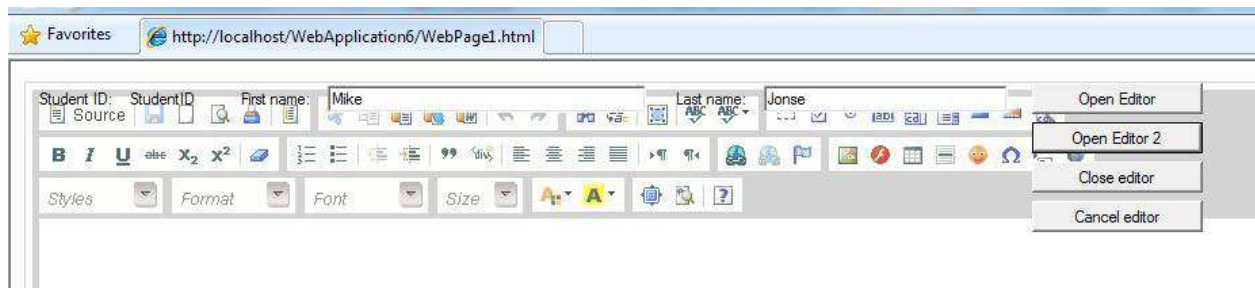




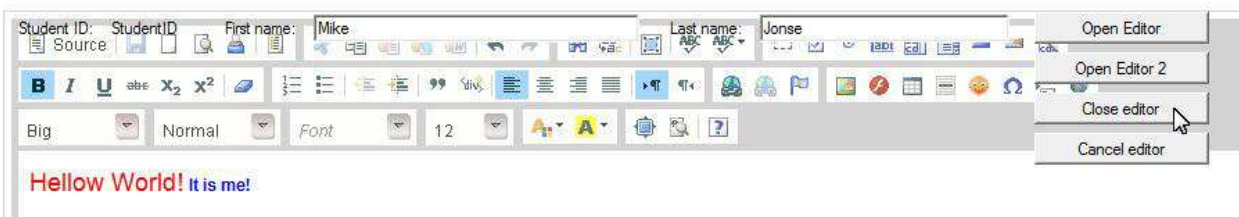
The web page appears. The first record appears. You may click  and  to navigate through the records.



You may modify the first name and last name directly in the text boxes. To modify the contents of the description, click “Open Editor” or “Open Editor 2”. The CKEditor appears.



Enter some contents in the CKEditor and click “Close editor”:



The contents appear in the HtmlContent control:



You may navigate to other records and do modifications. All the modifications are in the web page. Click Update button to send all the modifications to the web server and the database, and make the modifications visible by all the web visitors around the world.

As you may notice that the CKEditor is overlapping on other controls on the web page. We were not able to move the CKEditor to the right position using the JavaScript interface CKEditor provided. We have contacted the manufacturer for the problem but unable to get an answer. As we mentioned before we are not teaching CKEditor here. We are showing how JavaScript libraries can be used in developing web applications using Limnor Studio. The actual interface and functionality are provided by external JavaScript libraries.

Limnor Studio will be providing visual HTML editor to give you better control.

### Feedbacks

Please send your feedbacks to [support@limnor.com](mailto:support@limnor.com). Thanks!