Web Database Programming

Created: 2011-01-21

Last update: 2014-01-14

Contents

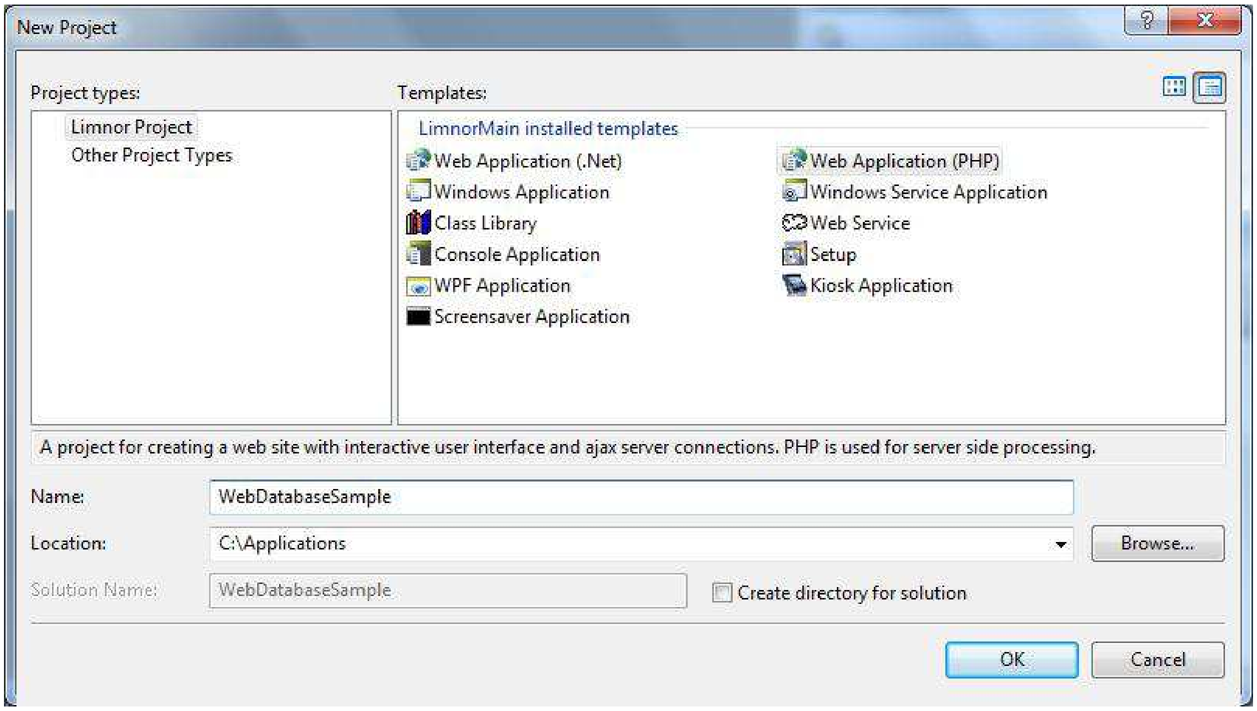
Introduction

Web pages can be designed as front tier for showing data from databases and entering data to be stored in databases.

Limnor Studio uses two-way data-binding to simplify web database programming. Ajax is used for retrieving data from databases and sending data to databases.

From visual programming point of view, using databases in a web application is very much the same as using databases in a standalone application.

We create a PHP web application to show the process:



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

Name: WebDatabaseSample
Location: C:\Applications
Solution Name: WebDatabaseSample
[ ] Create directory for solution

In this document we only use components which can be used in both PHP and .Net web applications. Therefore you may choose to select Web Application (.Net) and every operation described in this document is valid.

See http://www.limnor.com/support/WebApplicationDevelopment.pdf for an introduction on web application development.

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

Use EasyDataSet as Data Source

We use a Student table with following fields as our sample database.

StudentId, Firstname, Lastname, BirthDate, Grade

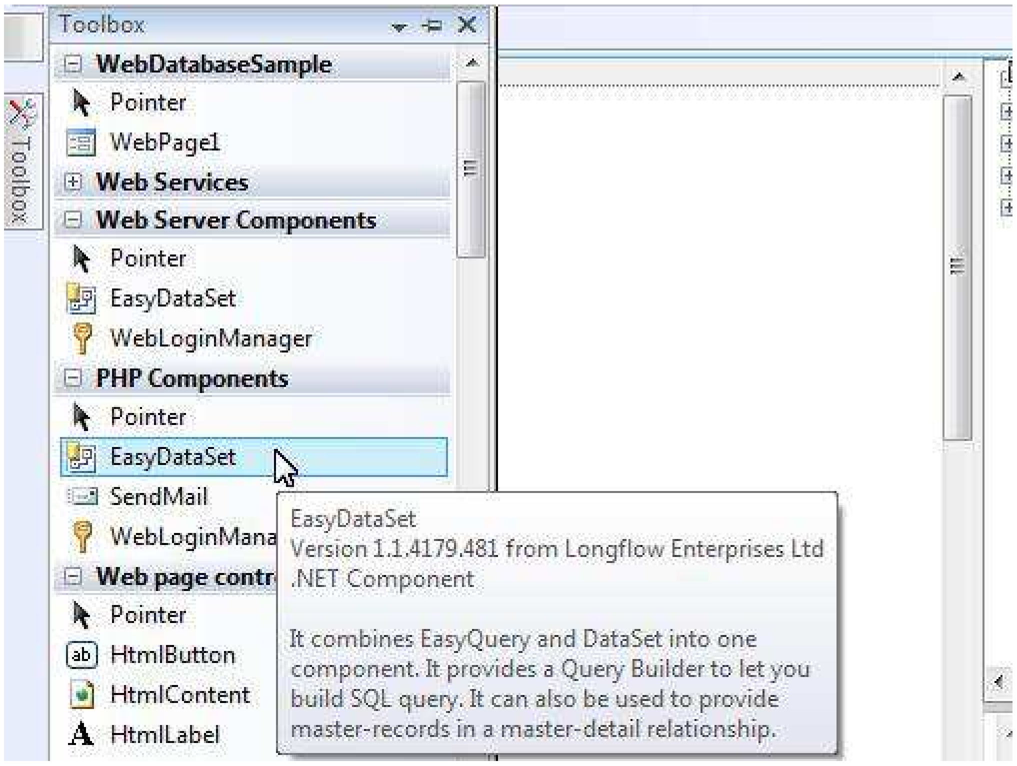
We used the following Sql script to create the table in a MySql database.

```
Create Table Student (
StudentId int not null auto_increment,
Firstname nvarchar (30),
Lastname nvarchar (30),
BirthDate datetime,
Grade int,
Primary Key (StudentId)
);
```

Or you may use some GUI tool to create and manage your MySql database. This is not in the scope of this document.

Add EasyDataSet to web page

We add an EasyDataSet to the web page to connect the MySql database and get data from the Student table:



EasyDataSet
Version 1.1.4179.481 from Longflow Enterprises Ltd
.NET Component

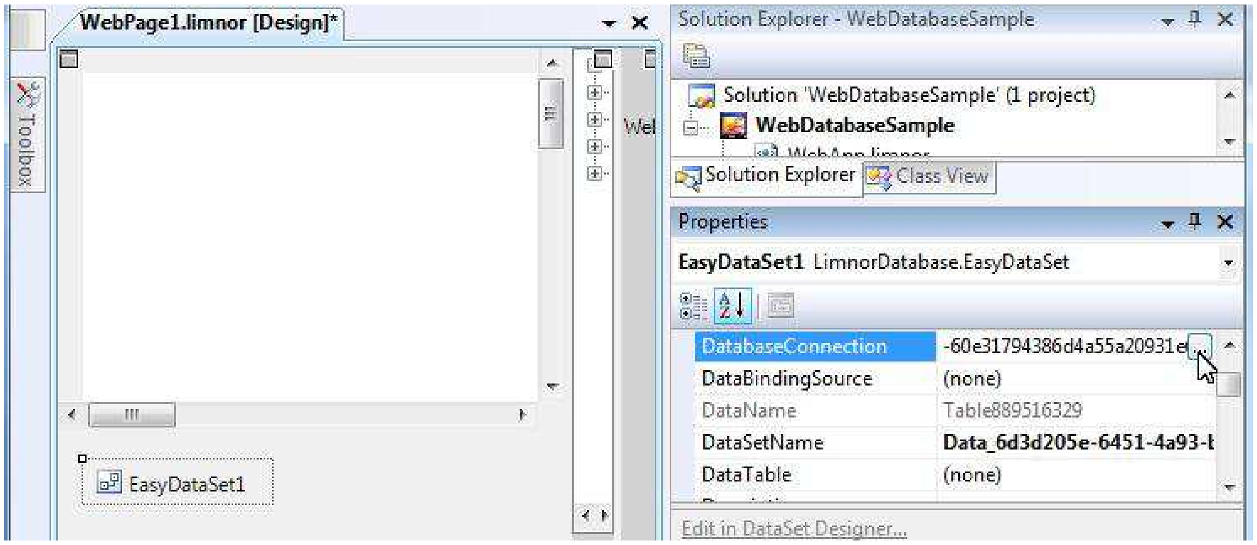
It combines EasyQuery and DataSet into one component. It provides a Query Builder to let you build SQL query. It can also be used to provide master-records in a master-detail relationship.

Note that the EasyDataSet component can be used in all kinds of applications: standalone application; PHP web applications; .Net web applications. The configuration of this component is also exactly the same for all kinds of applications.

If you already used EasyDataSet in standalone applications then you may skip this chapter.

Make Database Connection

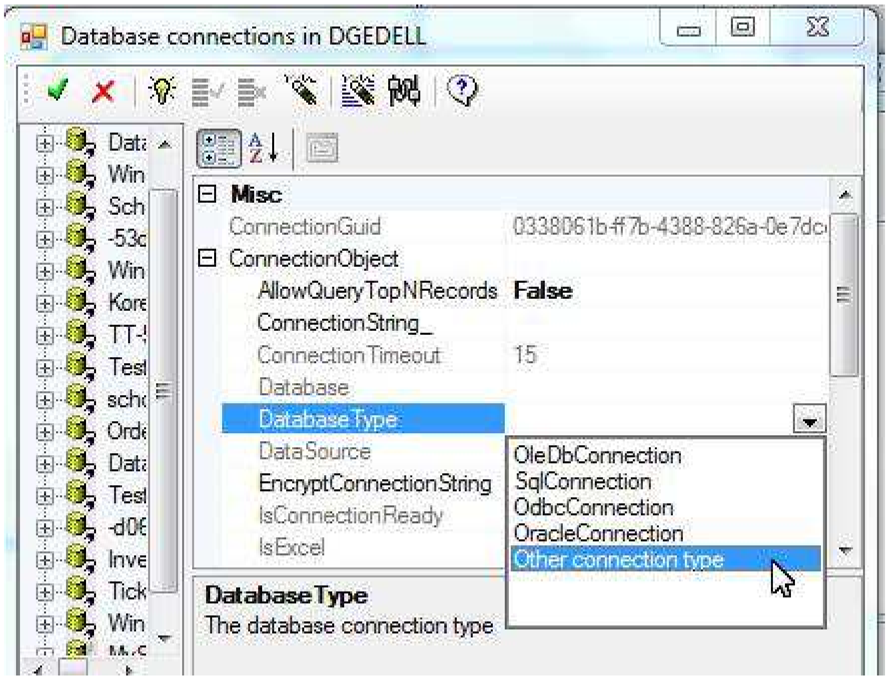
Set DatabaseConnection property of the EasyDataSet:


WebPage1.limnor [Design]*
Toolbox
EasyDataSet1
Solution Explorer - WebDatabaseSample
Solution 'WebDatabaseSample' (1 project)
WebDatabaseSample
Solution Explorer
Class View
Properties
Edit in DataSet Designer...

EasyDataSet1 LimnorDatabase.EasyDataSet

DatabaseConnection	-60e31794386d4a55a20931e
DataBindingSource	(none)
DataName	Table889516329
DataSetName	Data_6d3d205e-6451-4a93-t
DataTable	(none)

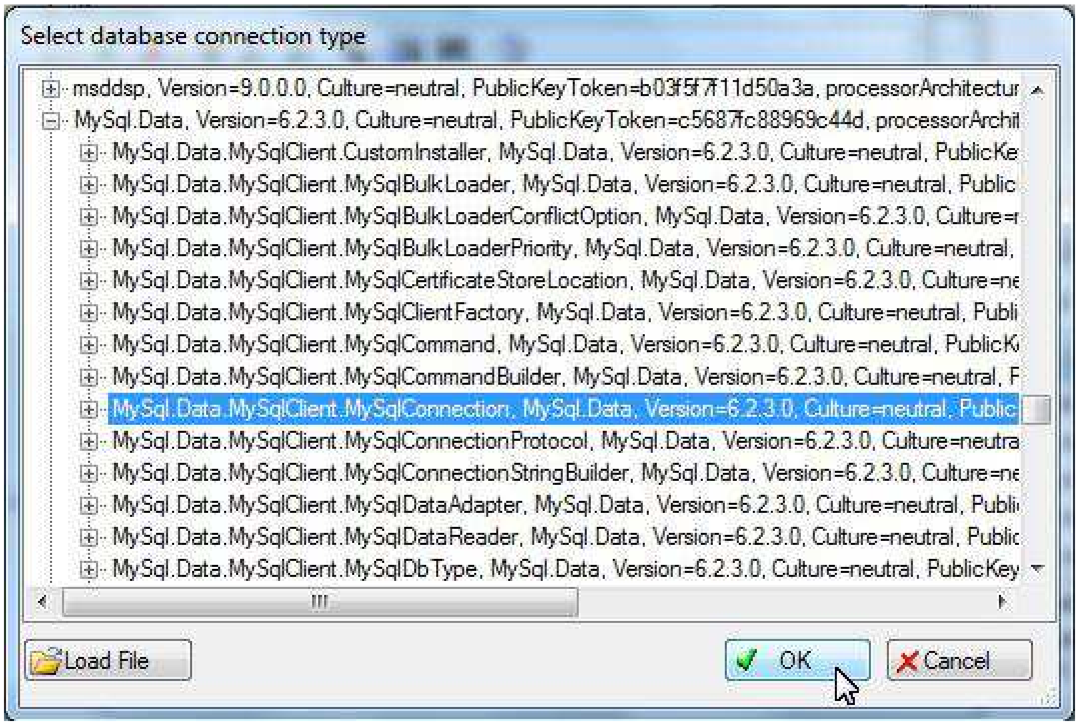
Choose “Other connection type” for the DatabaseType property:



ConnectionGuid	0338061b-ff7b-4388-826a-0e7dc
ConnectionObject	
AllowQueryTopNRecords	False
ConnectionString_	
ConnectionTimeout	15
Database	
DatabaseType	
DataSource	
EncryptConnectionString	
IsConnectionReady	
IsExcel	

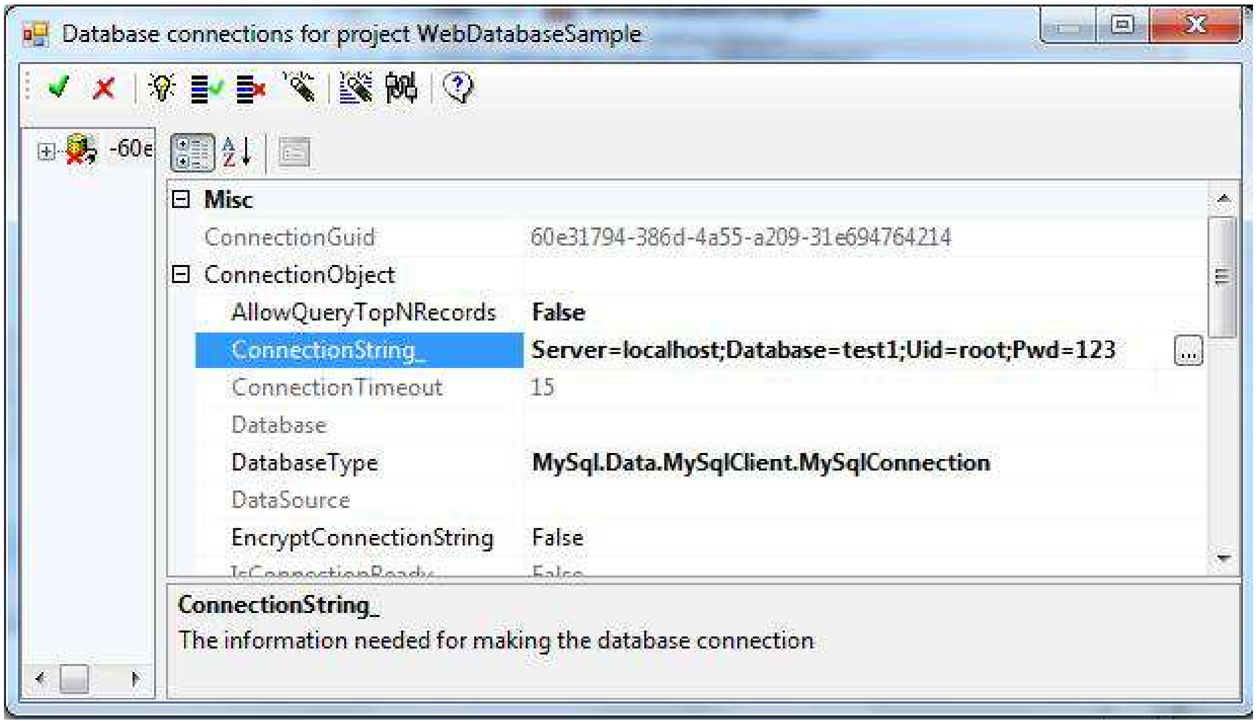
OleDbConnection
SqlConnection
OdbcConnection
OracleConnection
Other connection type

Choose “MySql.MySqlClient.MySqlConnection”. Click OK:


Select database connection type

```
⊞ msddsp, Version=9.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitectur
⊟ MySql.Data, Version=6.2.3.0, Culture=neutral, PublicKeyToken=c5687fc88969c44d, processorArchit
    ⊞ MySql.Data.MySqlClient.CustomInstaller, MySql.Data, Version=6.2.3.0, Culture=neutral, PublicKe
    ⊞ MySql.Data.MySqlClient.MySqlBulkLoader, MySql.Data, Version=6.2.3.0, Culture=neutral, Public
    ⊞ MySql.Data.MySqlClient.MySqlBulkLoaderConflictOption, MySql.Data, Version=6.2.3.0, Culture=r
    ⊞ MySql.Data.MySqlClient.MySqlBulkLoaderPriority, MySql.Data, Version=6.2.3.0, Culture=neutral,
    ⊞ MySql.Data.MySqlClient.MySqlCertificateStoreLocation, MySql.Data, Version=6.2.3.0, Culture=ne
    ⊞ MySql.Data.MySqlClient.MySqlClientFactory, MySql.Data, Version=6.2.3.0, Culture=neutral, Publi
    ⊞ MySql.Data.MySqlClient.MySqlCommand, MySql.Data, Version=6.2.3.0, Culture=neutral, PublicK
    ⊞ MySql.Data.MySqlClient.MySqlCommandBuilder, MySql.Data, Version=6.2.3.0, Culture=neutral, F
    ⊞ MySql.Data.MySqlClient.MySqlConnection, MySql.Data, Version=6.2.3.0, Culture=neutral, Public
    ⊞ MySql.Data.MySqlClient.MySqlConnectionProtocol, MySql.Data, Version=6.2.3.0, Culture=neutra
    ⊞ MySql.Data.MySqlClient.MySqlConnectionStringBuilder, MySql.Data, Version=6.2.3.0, Culture=ne
    ⊞ MySql.Data.MySqlClient.MySqlDataAdapter, MySql.Data, Version=6.2.3.0, Culture=neutral, Publi
    ⊞ MySql.Data.MySqlClient.MySqlDataReader, MySql.Data, Version=6.2.3.0, Culture=neutral, Public
    ⊞ MySql.Data.MySqlClient.MySqlDbType, MySql.Data, Version=6.2.3.0, Culture=neutral, PublicKey
```

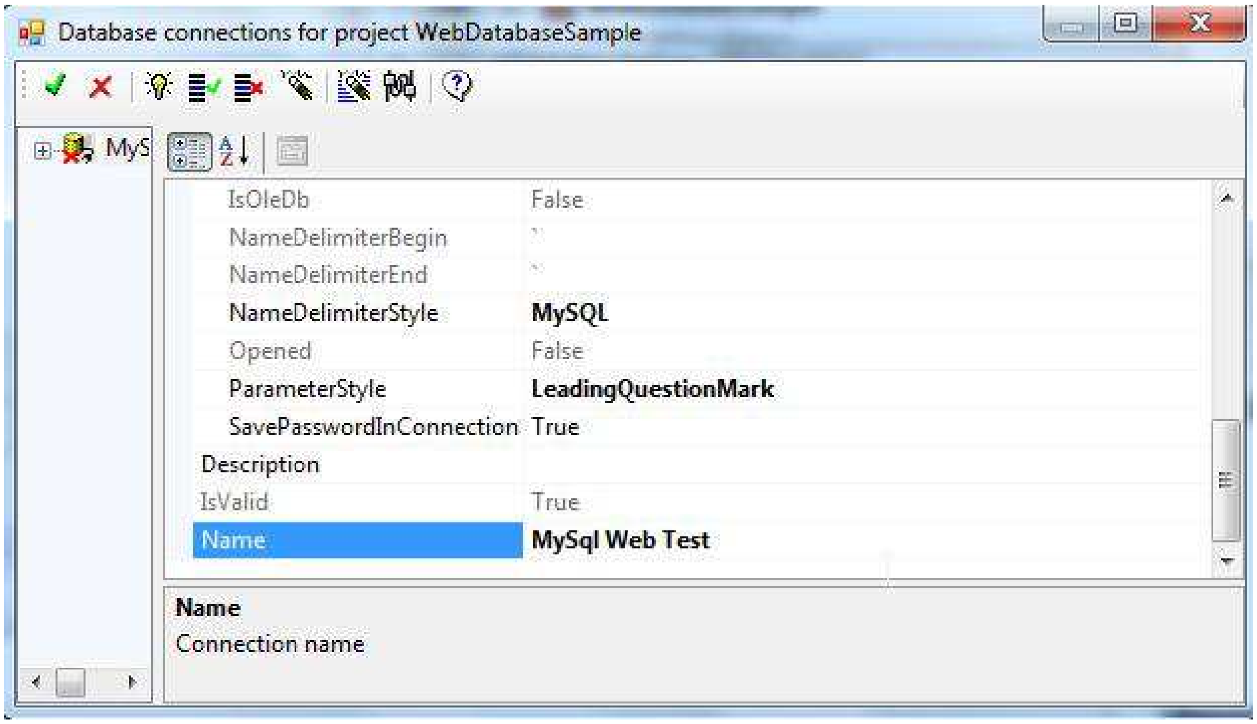
Set the connection string to connect to the MySql database:


Database connections for project WebDatabaseSample

Misc	
ConnectionGuid	60e31794-386d-4a55-a209-31e694764214
ConnectionObject	
AllowQueryTopNRecords	**False**
ConnectionString_	**Server=localhost;Database=test1;Uid=root;Pwd=123**
ConnectionTimeout	15
Database	
DatabaseType	**MySql.Data.MySqlClient.MySqlConnection**
DataSource	
EncryptConnectionString	False

ConnectionString_
The information needed for making the database connection

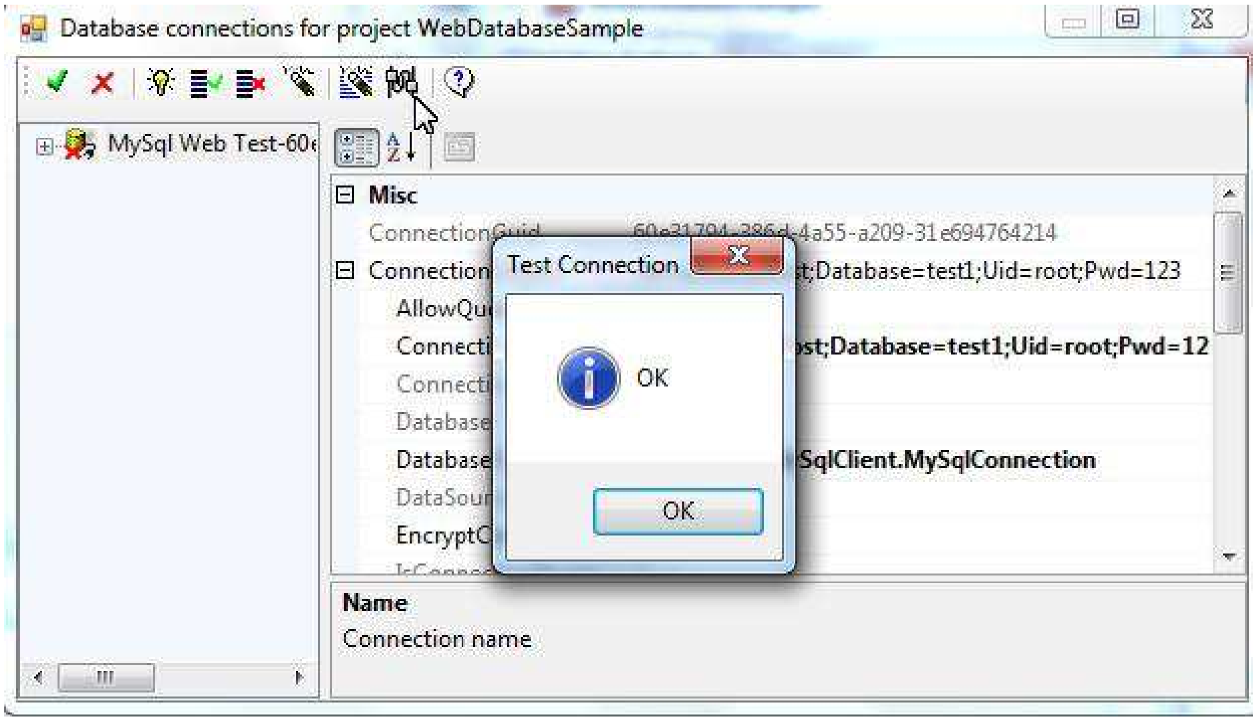
Give the connection a name:



IsOleDb	False
NameDelimiterBegin	`
NameDelimiterEnd	`
NameDelimiterStyle	**MySQL**
Opened	False
ParameterStyle	**LeadingQuestionMark**
SavePasswordInConnection	True
Description	
IsValid	True
Name	**MySql Web Test**

Name
Connection name

Click to test the connection:



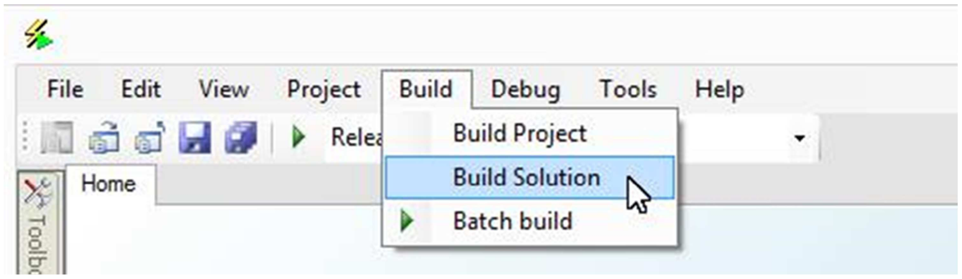
Misc	
ConnectionGuid	60e31794-386d-4a55-a209-31e694764214
Connection	;Database=test1;Uid=root;Pwd=123
AllowQu	
Connecti	**ost;Database=test1;Uid=root;Pwd=12**
Connecti	
Database	
Database	**SqlClient.MySqlConnection**
DataSour	
EncryptC	

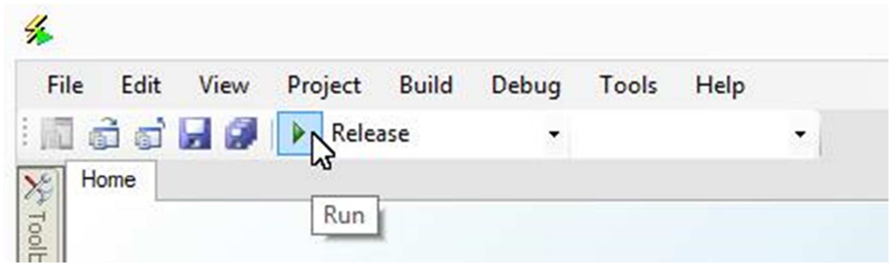
OK

Name
Connection name

Note that we are connecting the database using ADO.NET driver. **This is for design time only**. At runtime for PHP web applications, ADO.NET driver is not used because it may not be available in a Linux machine. PHP library for MySql database will be used. All this is done behind the scene. You, as a developer, develop in the same way for web applications and for standalone applications. Limnor Studio compiler will generate PHP code for you.

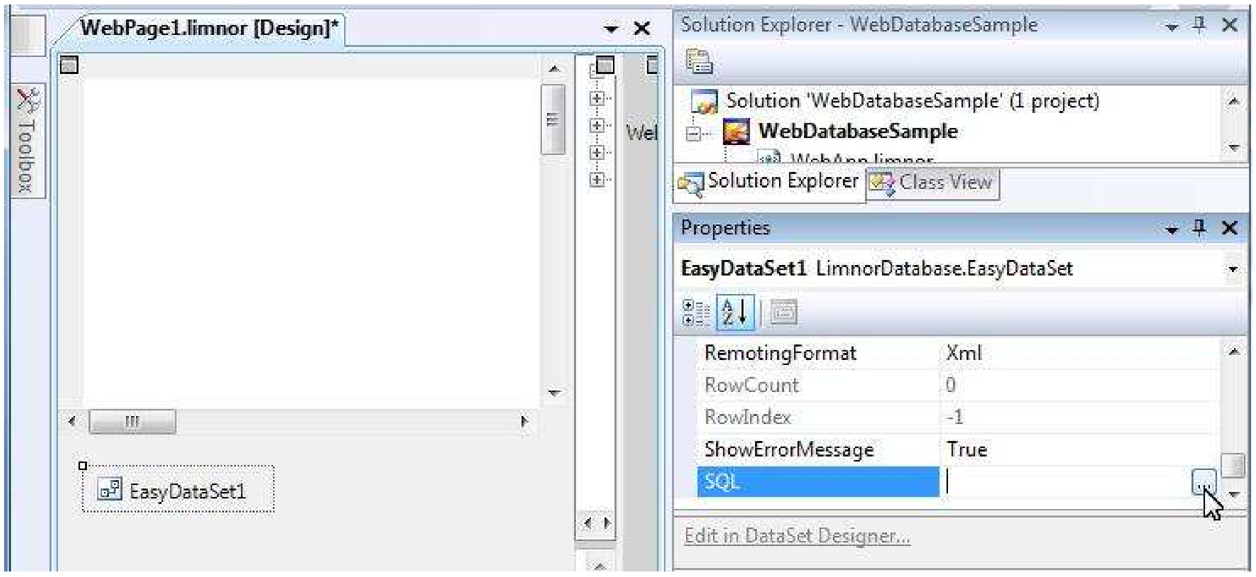
Note that after setting a database connection, the solution must be re-built before you can click the green arrow button to test the web pages.





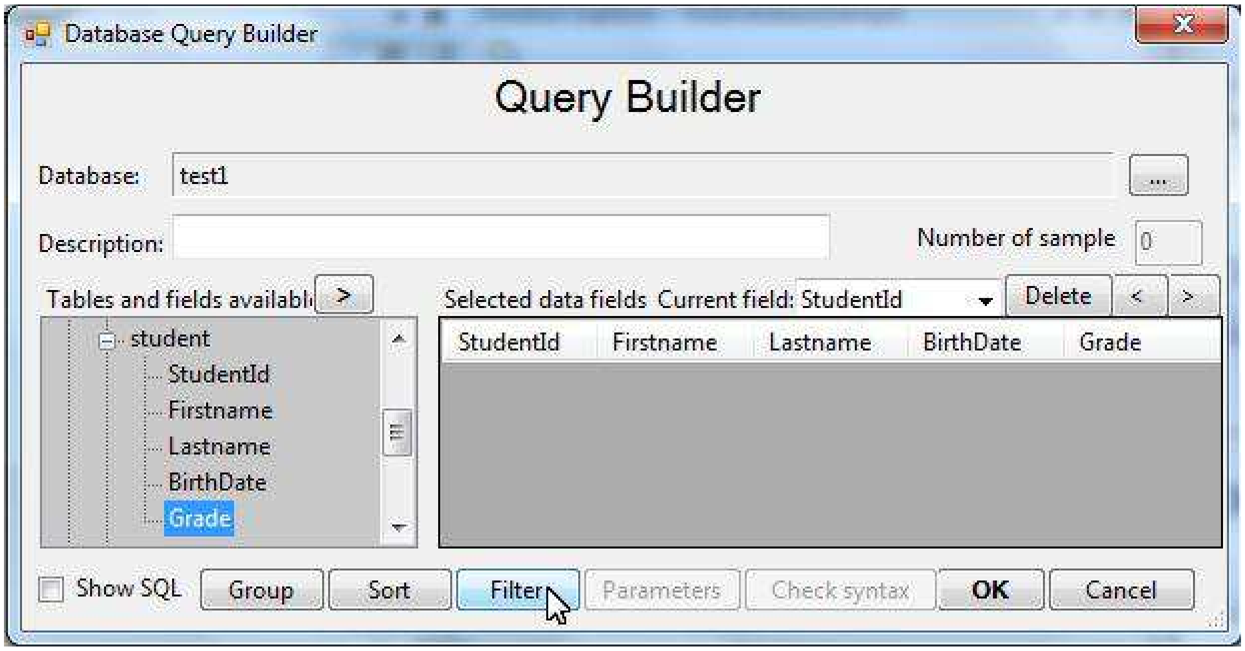
Use Query Builder to get data

Set the SQL property of the EasyDataSet to fetch Student data:


WebPage1.limnor [Design]*
Toolbox
EasyDataSet1
Solution Explorer - WebDatabaseSample
Solution 'WebDatabaseSample' (1 project)
WebDatabaseSample
Solution Explorer
Class View
Properties
EasyDataSet1 LimnorDatabase.EasyDataSet
Edit in DataSet Designer...

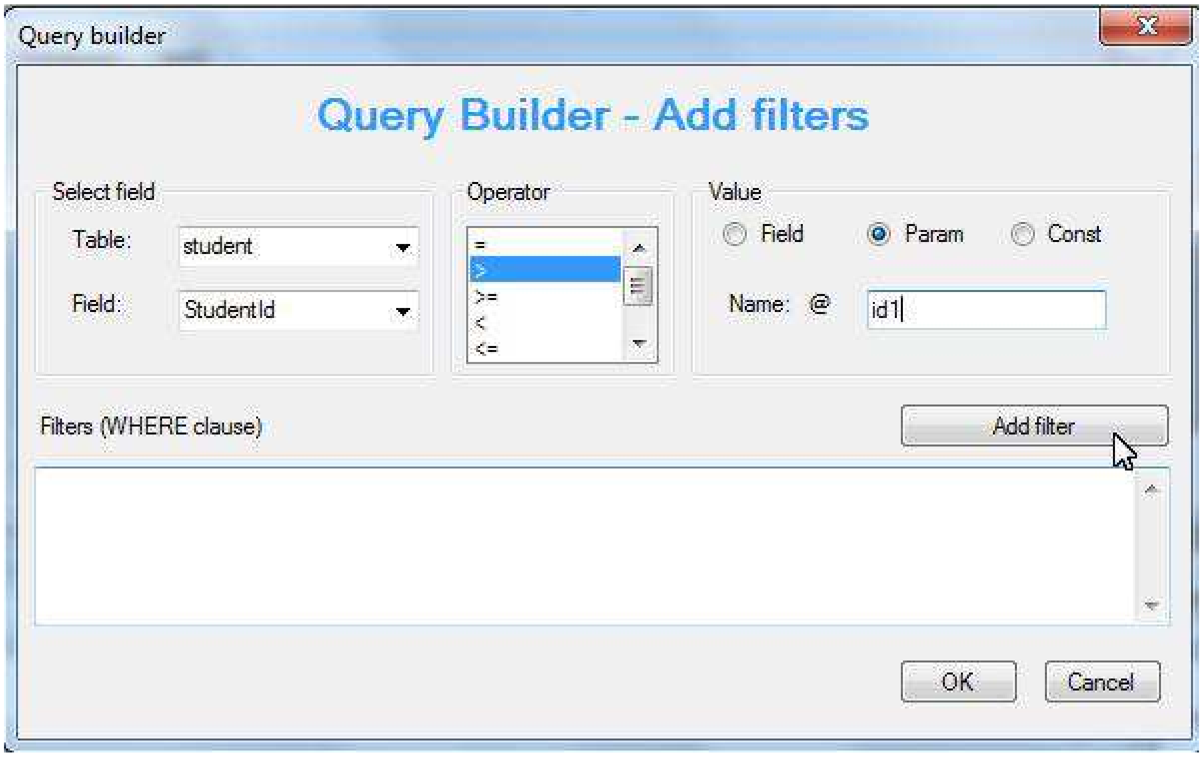
RemotingFormat	Xml
RowCount	0
RowIndex	-1
ShowErrorMessage	True
SQL	

Add all the fields we want to display in the web page to the query. Click Filter button to add filters for limiting the range of data:

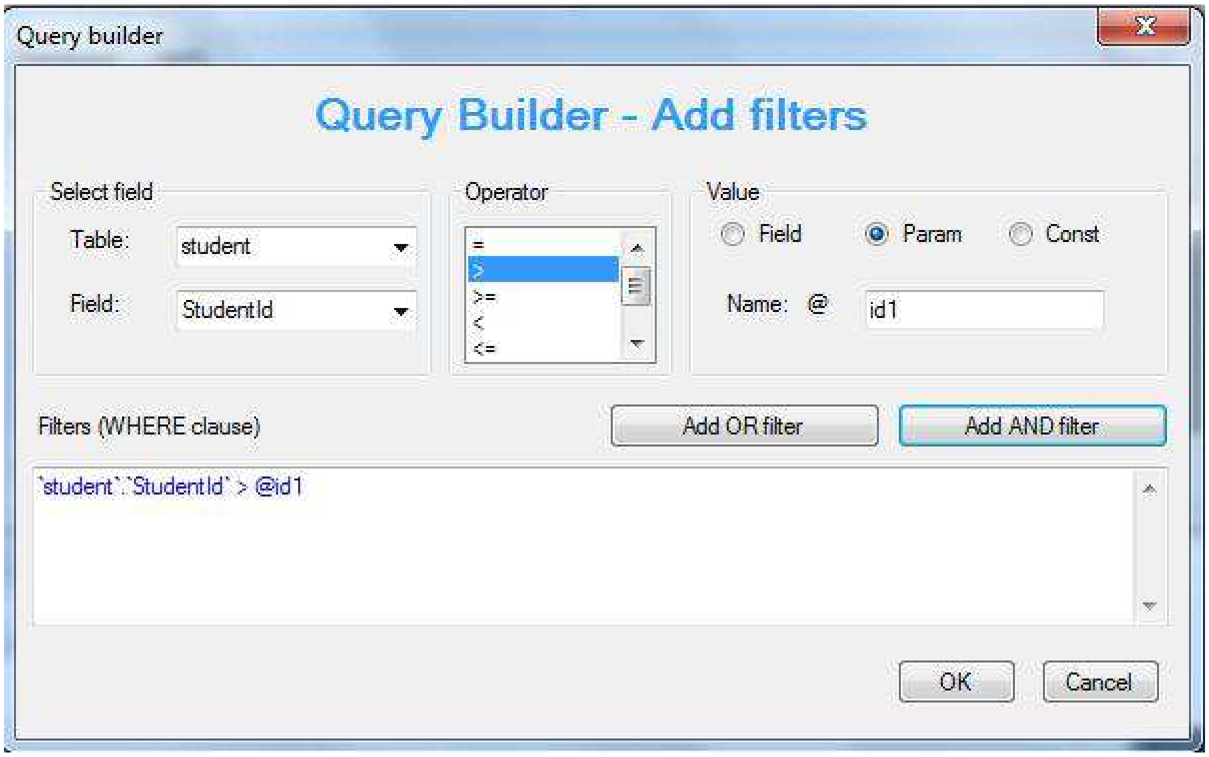


For this sample, we simply use a range of StudentId as the filter. We will let the web user to enter the range from web browser. So, we will use parameters in the filter.

Select the StudentId, choose >, choose Param, and give a parameter name id1:

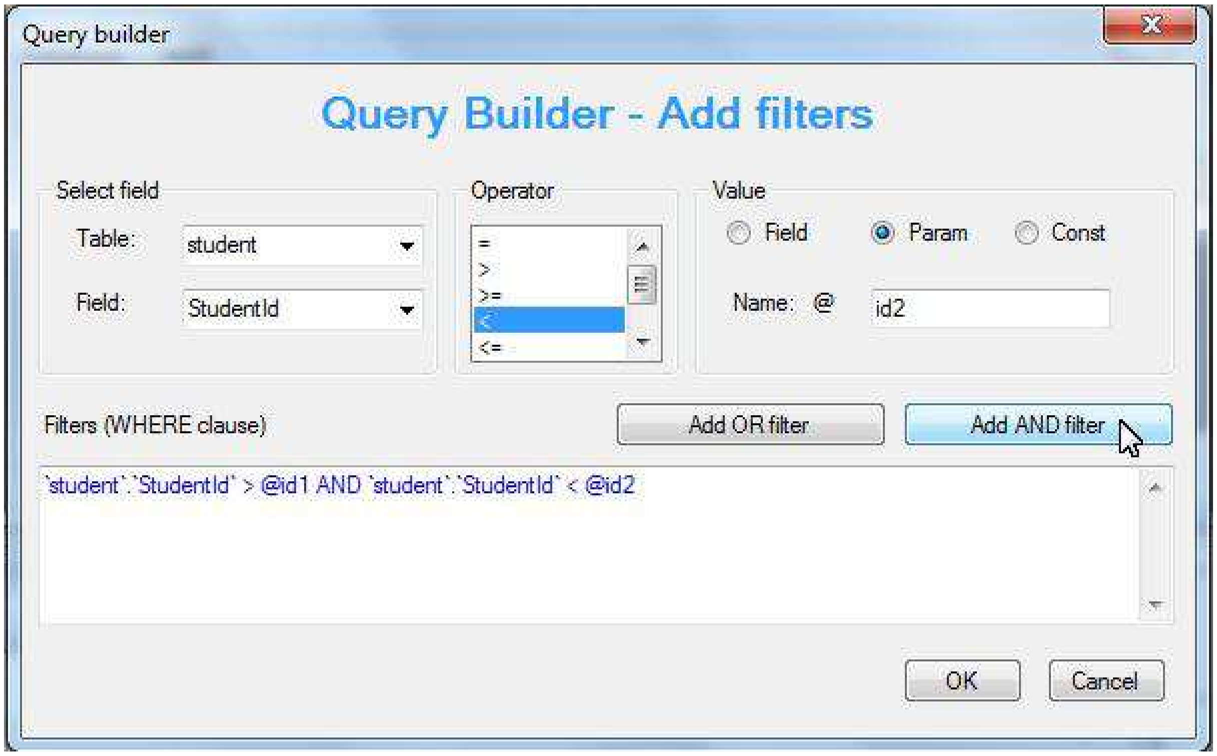


Click Add filter. The filter appears:



```
`student`.`StudentId` > @id1
```

Select <; give another parameter name id2. Click "Add AND filter":



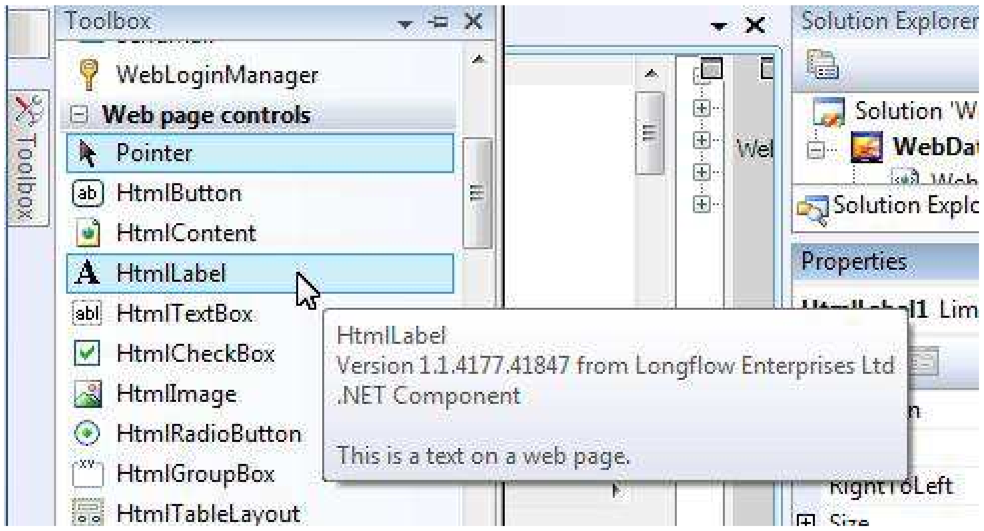
```
`student`.`StudentId` > @id1 AND `student`.`StudentId` < @id2
```

For this sample, that is all we want. Click OK buttons.

Bind-data to single field

To allow the user to do data entry we may use a text box. For displaying data only we may use a label.

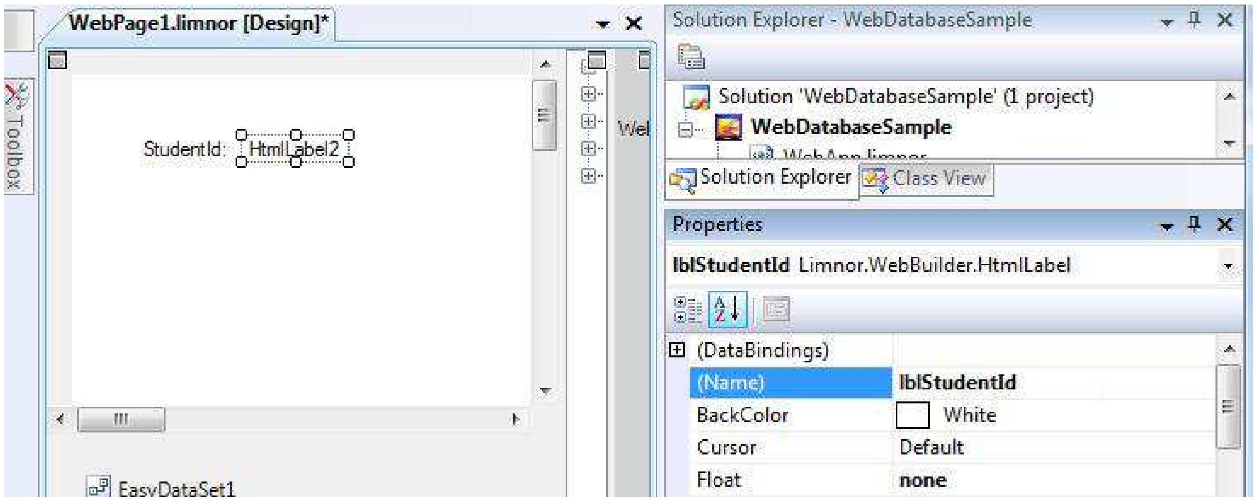
We use a label to display StudentId:



HtmlLabel
Version 1.1.4177.41847 from Longflow Enterprises Ltd
.NET Component

This is a text on a web page.

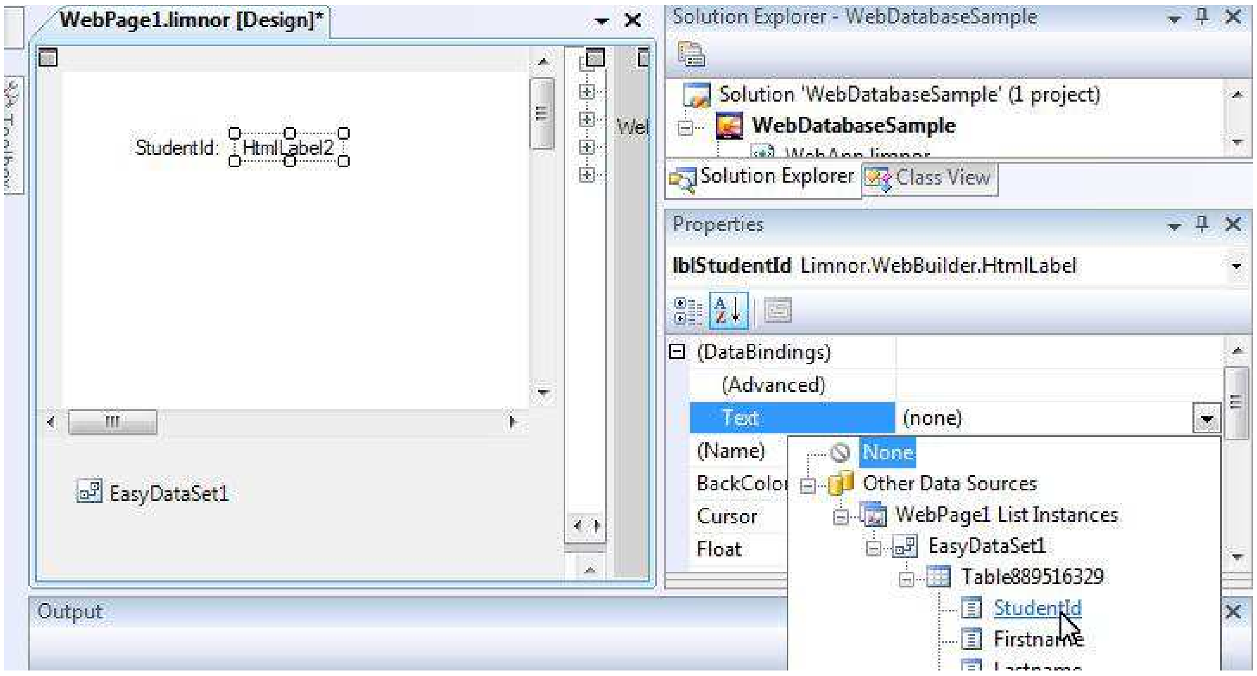
Rename it to lblStudentId:


WebPage1.limnor [Design]*
StudentId:
HtmlLabel2
EasyDataSet1
Solution Explorer - WebDatabaseSample
Solution 'WebDatabaseSample' (1 project)
WebDatabaseSample
Solution Explorer
Class View
Properties

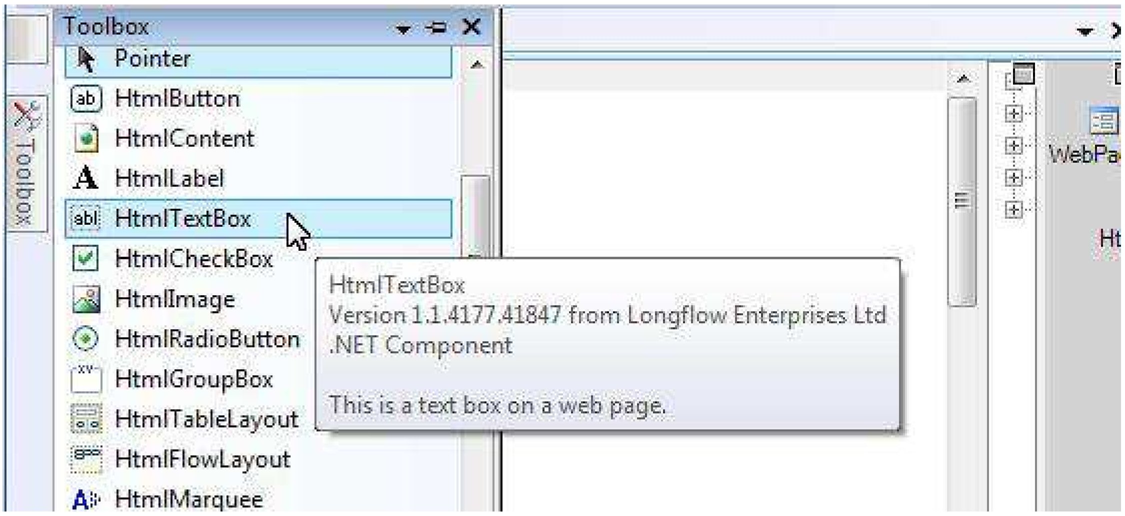
lblStudentId Limnor.WebBuilder.HtmlLabel

(DataBindings)	
(Name)	lblStudentId
BackColor	White
Cursor	Default
Float	none

Set its Databindings property to bind its Text to StudentId:



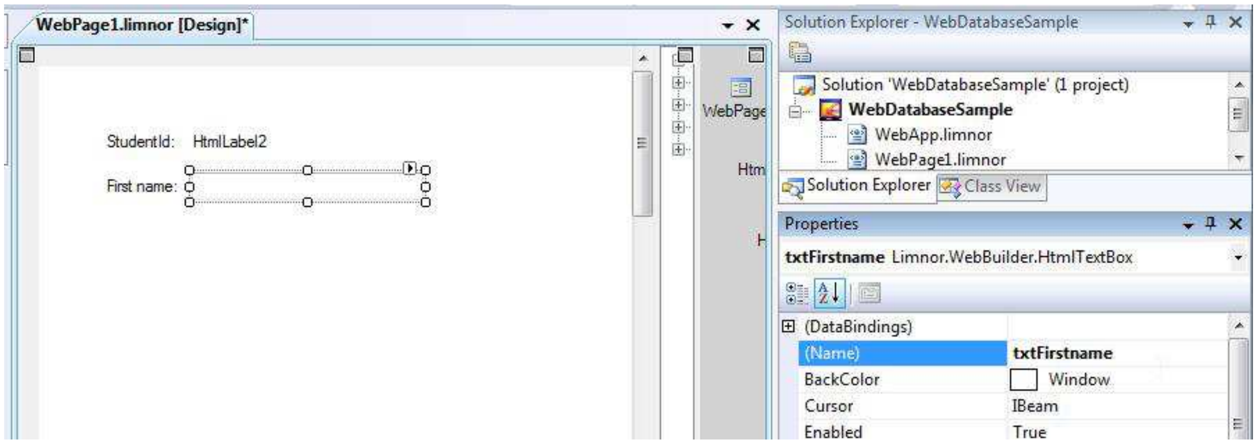
Use a text box to show and edit first name:



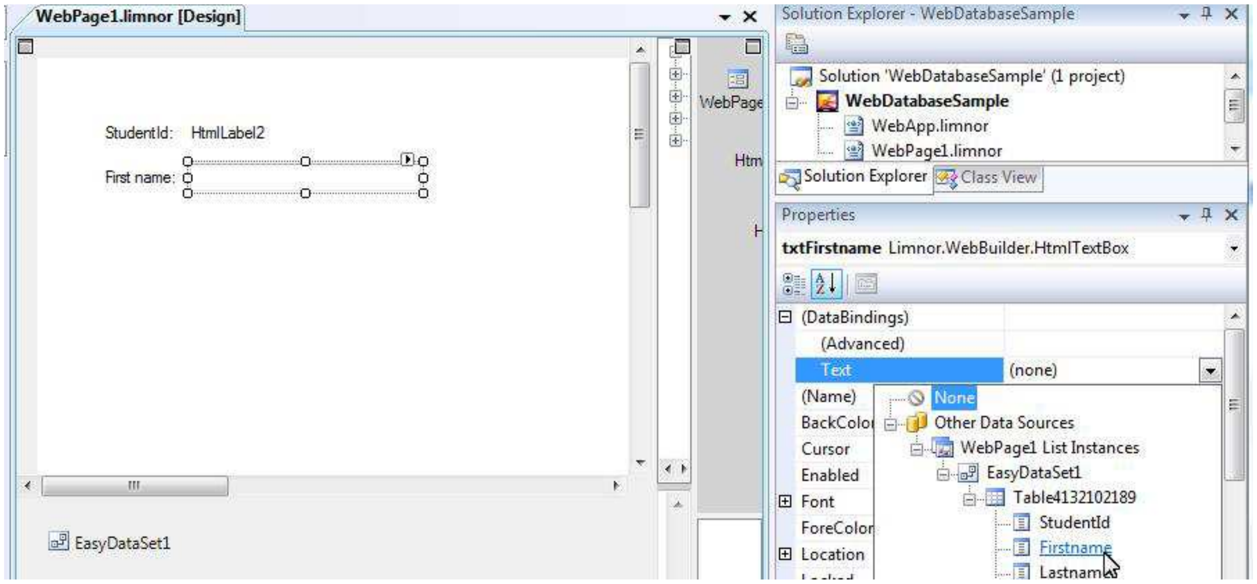
HtmlTextBox
Version 1.1.4177.41847 from Longflow Enterprises Ltd
.NET Component

This is a text box on a web page.

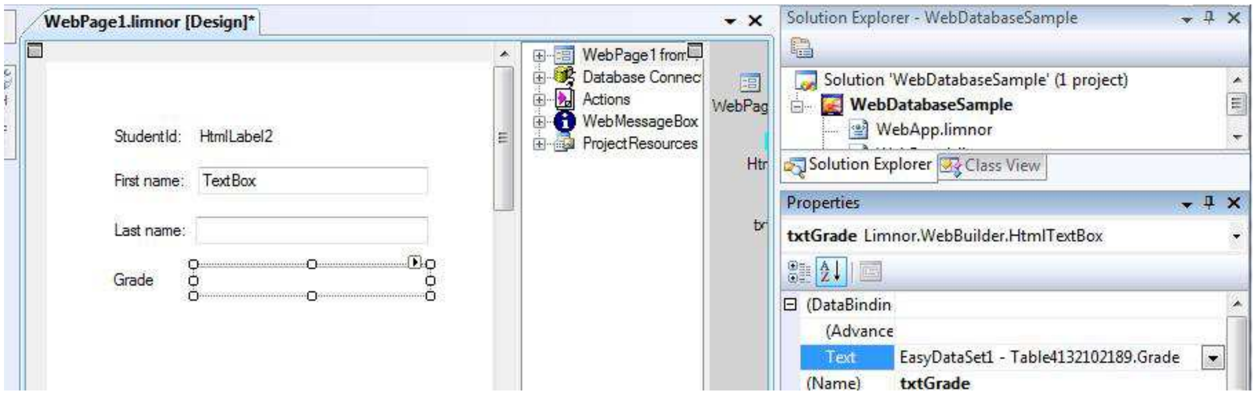
Rename it to txtFirstname:



Bind its Text to the Firstname field:



In the same way we may create data-bindings for other fields:

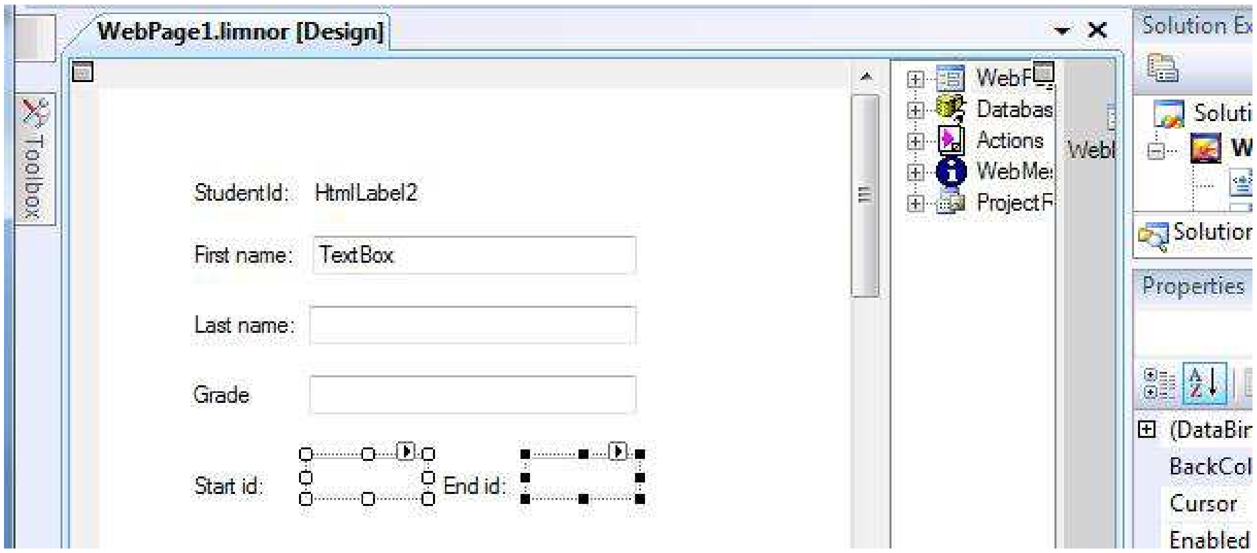


To change data-binding at runtime, see chapter “Change Data-binding”.

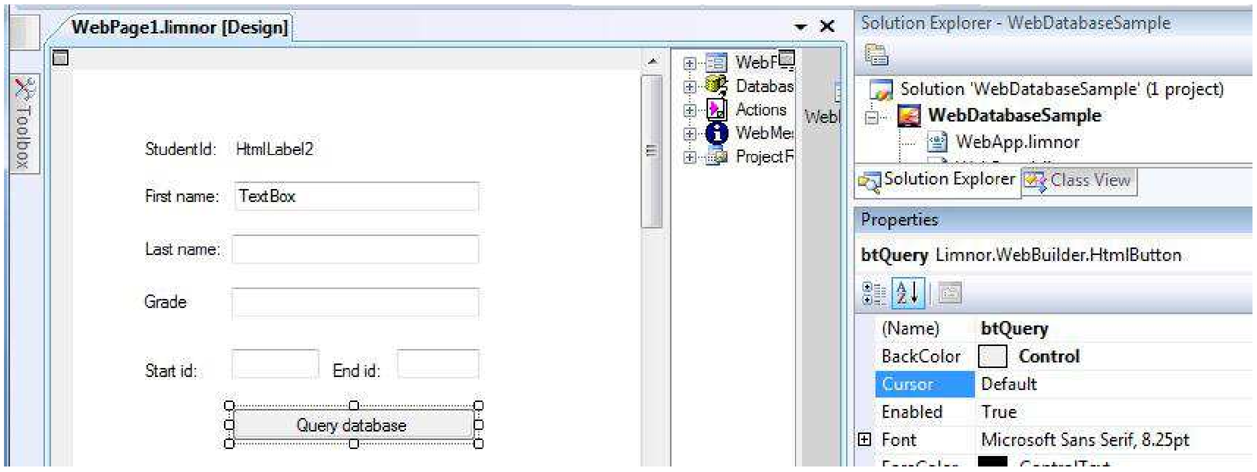
Data Query

To get data from the database, we need to execute a query action. Since we use parameters in filters, we need to build a UI to let the user to enter values for the parameters.

We use two text boxes for entering id1 and id2:



We use a button to trig a query:

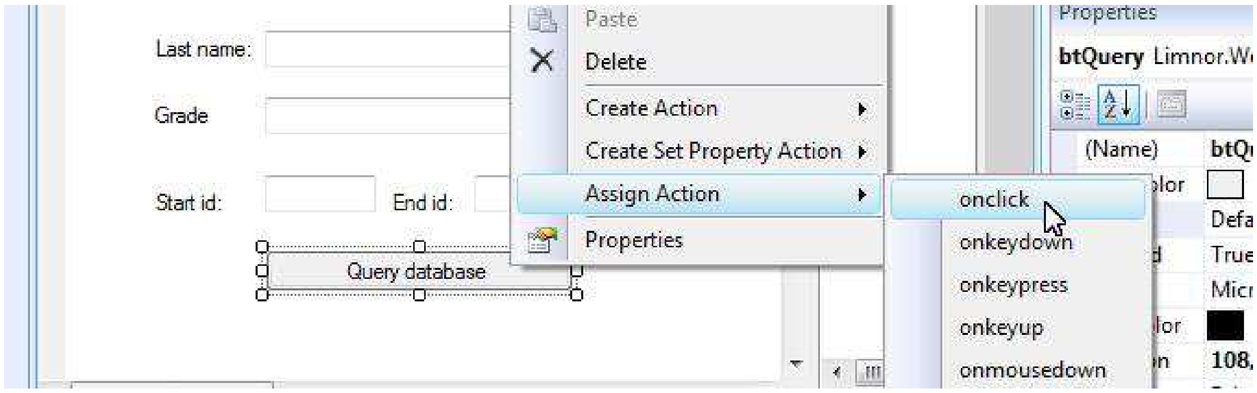


btQuery Limnor.WebBuilder.HtmlButton

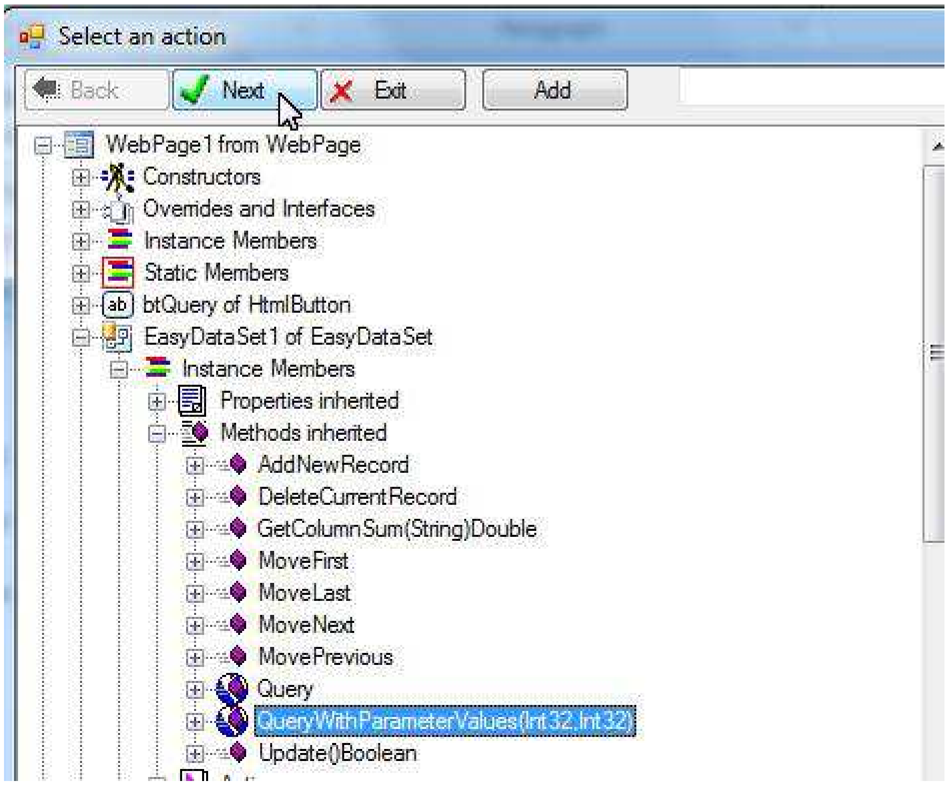
(Name)	btQuery
BackColor	Control
Cursor	Default
Enabled	True
Font	Microsoft Sans Serif, 8.25pt

When the user clicks the button we want to execute a query.

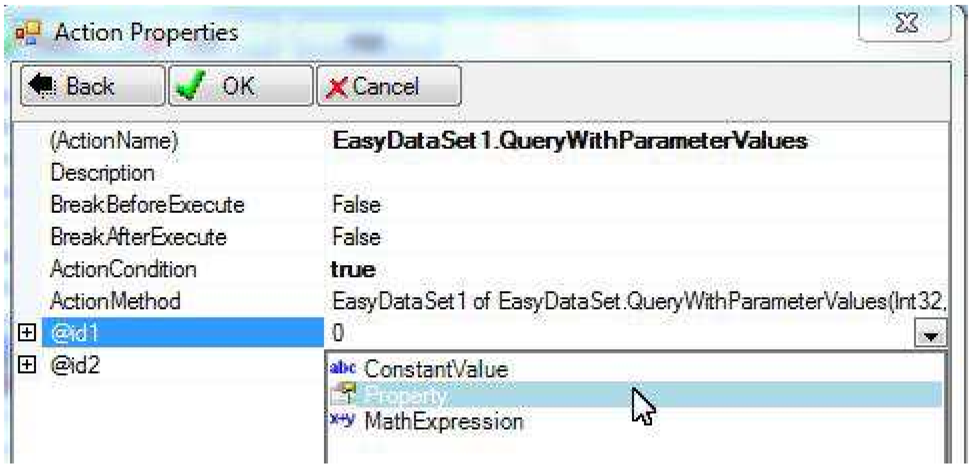
Right-click the button; choose “Assign Action”; choose “onclick” event:



Select QueryWithParameterValues of the EasyDataSet; click Next:

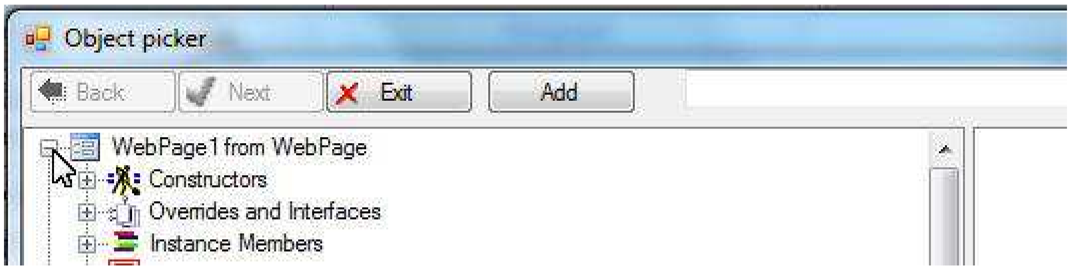


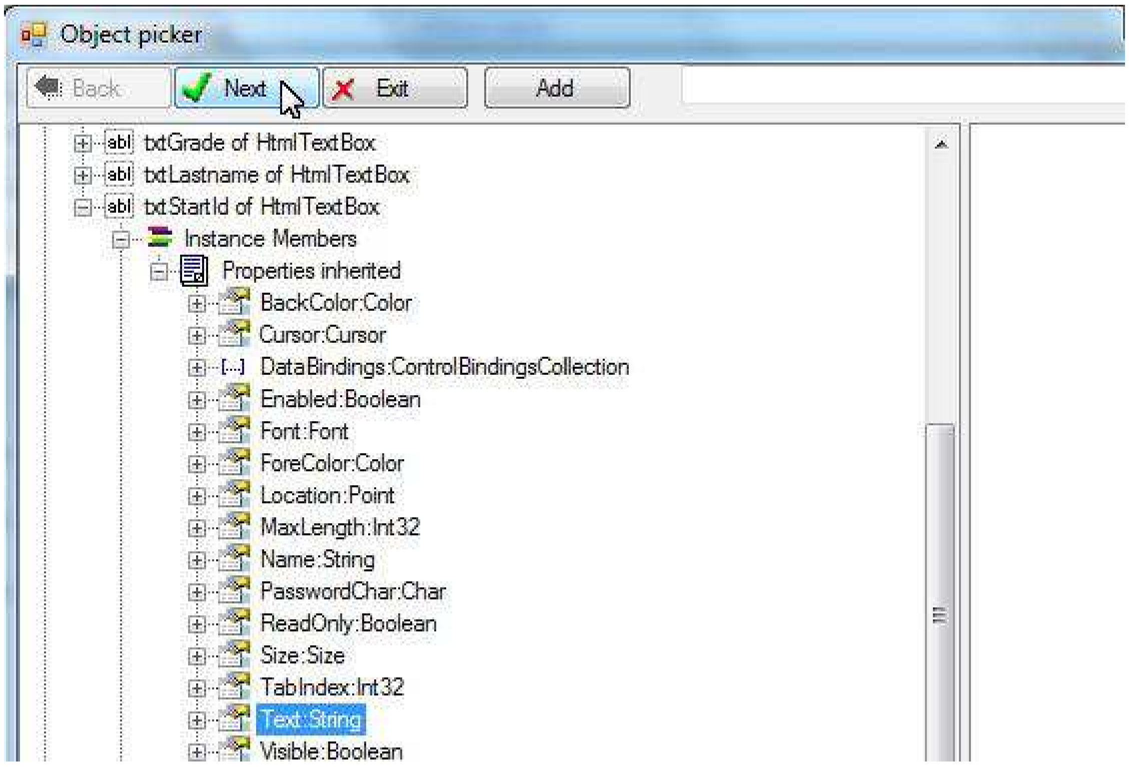
Select Property for @id1:



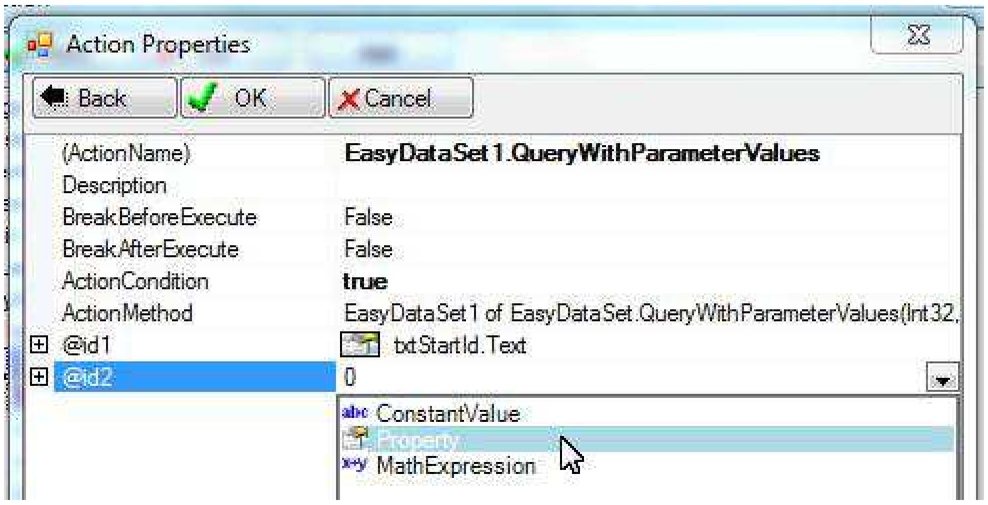
(ActionName)	EasyDataSet1.QueryWithParameterValues
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyDataSet1 of EasyDataSet.QueryWithParameterValues(Int32,
@id1	0
@id2	

Select the Text property of the text box for starting id:

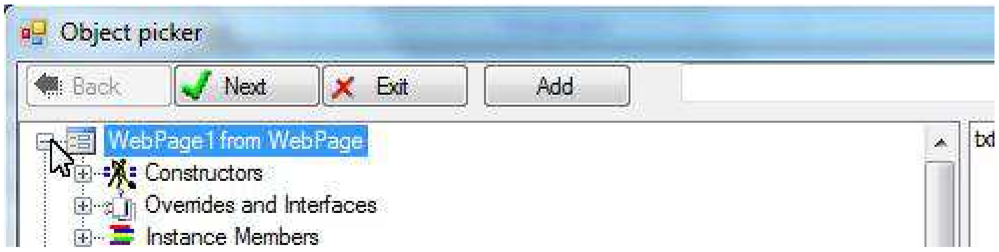


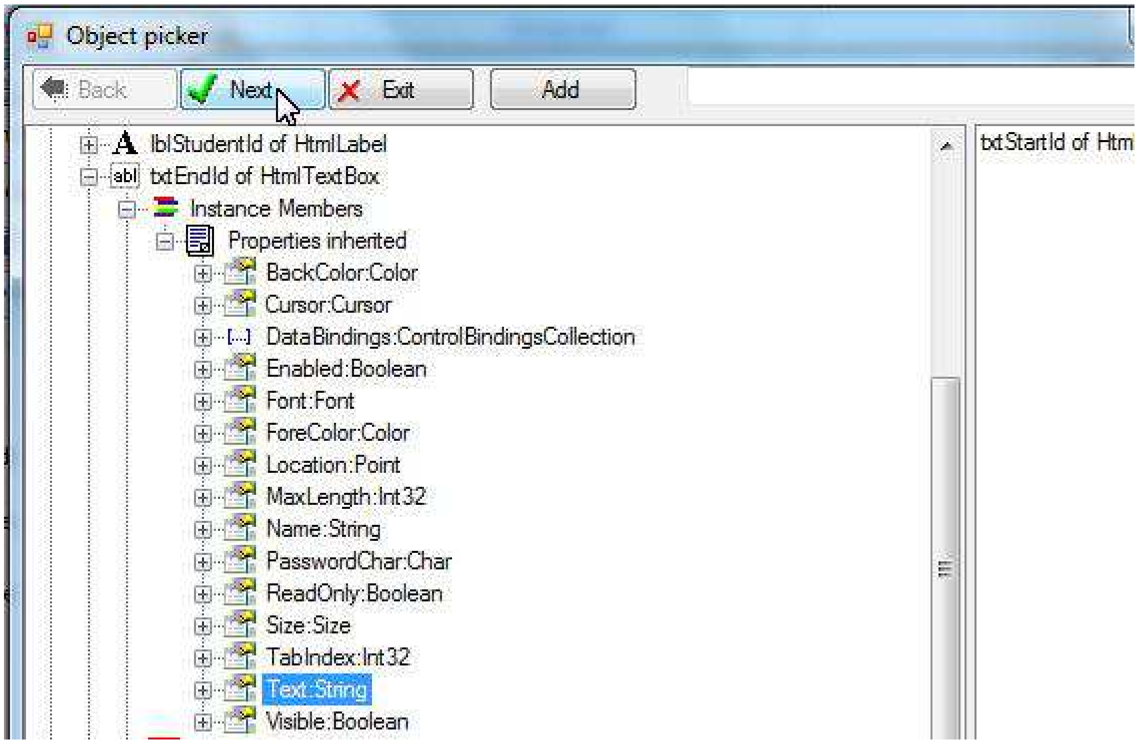


For @id2, select the Text property of the text box for end id:

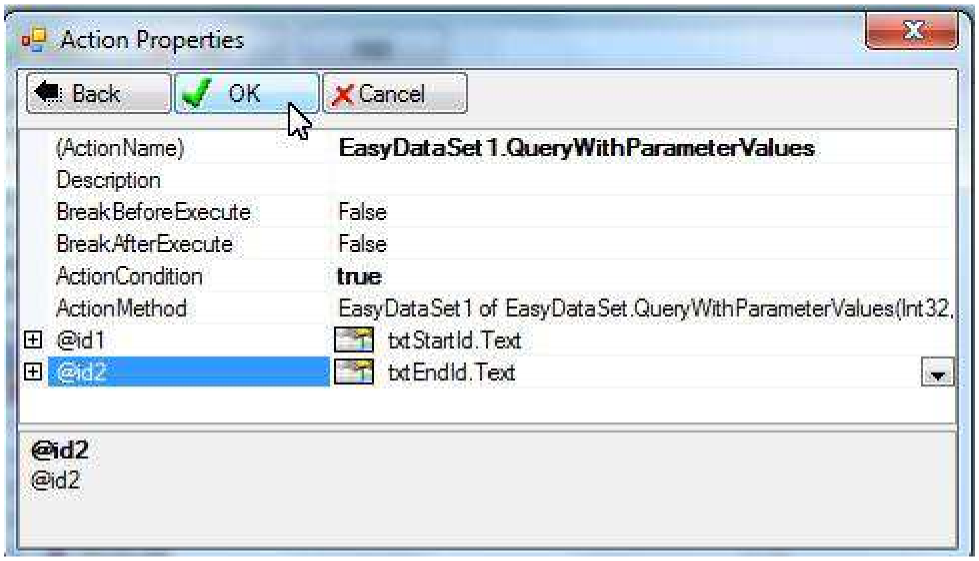


(ActionName)	**EasyDataSet1.QueryWithParameterValues**
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	EasyDataSet1 of EasyDataSet.QueryWithParameterValues(Int32,
@id1	txtStartId.Text
@id2	0





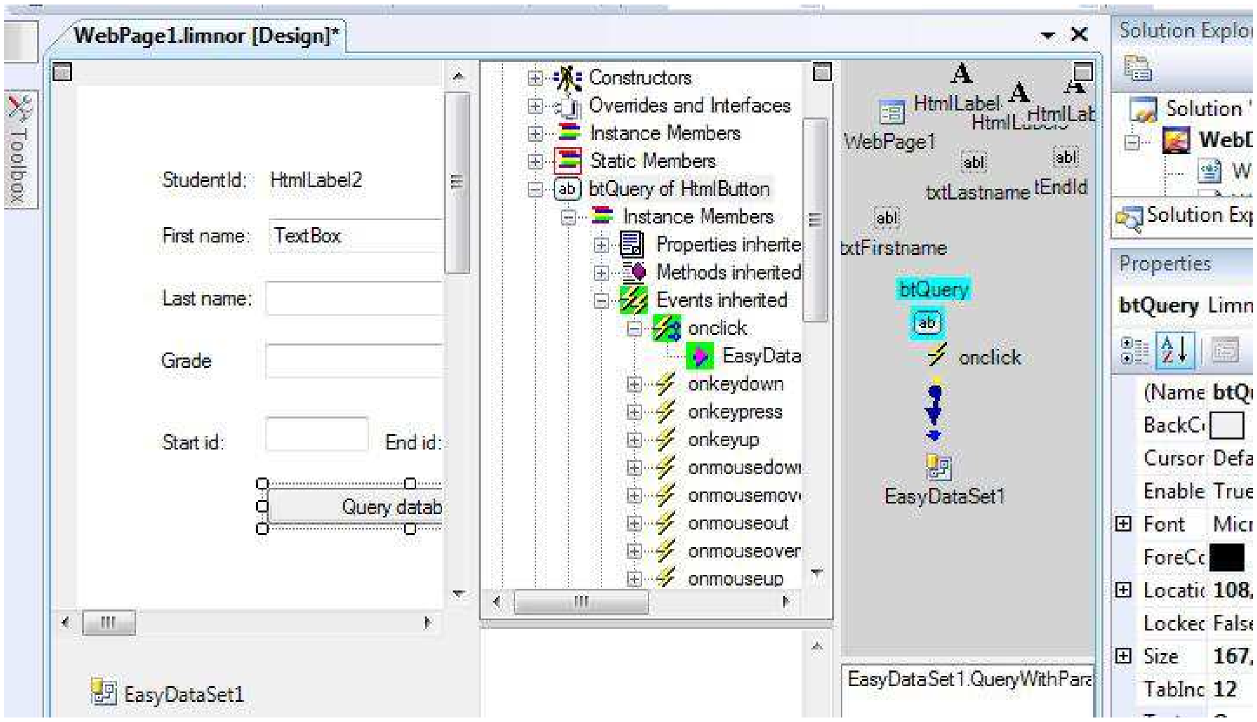
Click OK to finish creating the query action and assigning it to the button:



(ActionName)	EasyDataSet1.QueryWithParameterValues
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyDataSet1 of EasyDataSet.QueryWithParameterValues(Int32,
@id1	txtStartId.Text
@id2	txtEndId.Text

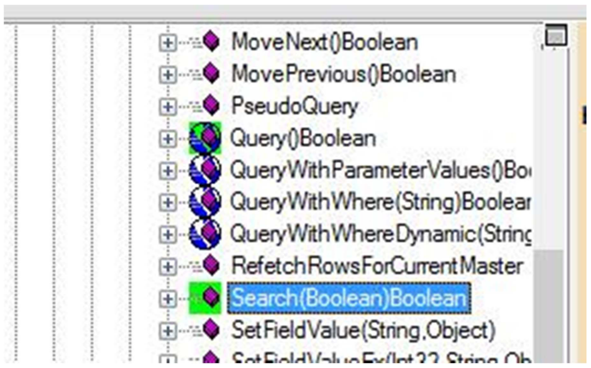
@id2
@id2

We are done creating an action and assigning it to the button. You can see it in the Event Map and Object Explorer.

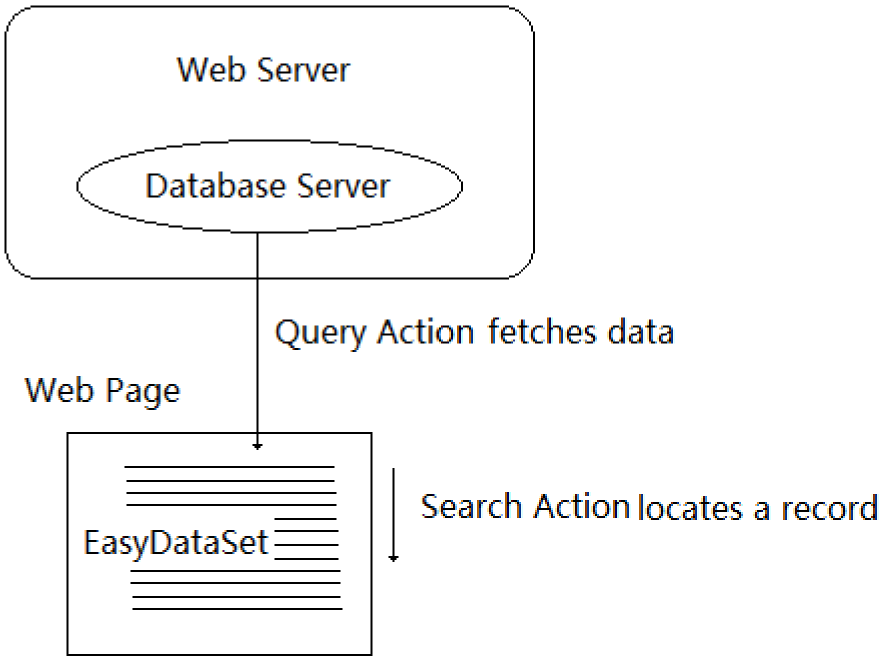


Data Search

EasyDataSet has a method Search for searching data fetched from database:


MoveNext()Boolean
MovePrevious()Boolean
PseudoQuery
Query()Boolean
QueryWithParameterValues()Bo
QueryWithWhere(String)Boolear
QueryWithWhereDynamic(Strin
RefetchRowsForCurrentMaster
Search(Boolean)Boolean
SetFieldValue(String,Object)

There are huge differences between a Search action and a Query action. A Query action searches data in the database and downloads found data to web page. A Search action searches among the data downloaded from the database; the found record becomes the current record.

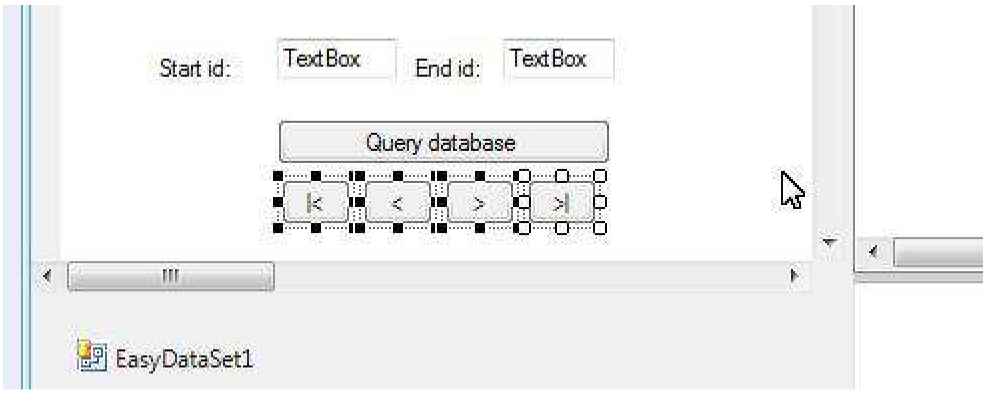


To do a Search, a Query action must be executed first to fetch data from the database server.

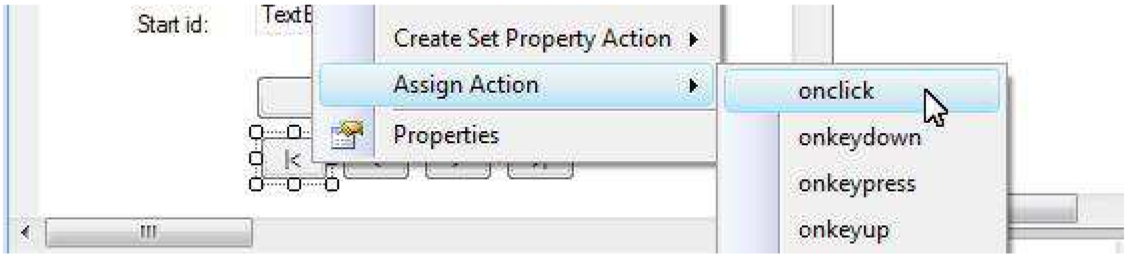
Data Navigation

On fetching a range of student records, we may use “go next”, “go previous”, “go first” and “go last” actions to navigate through the records.

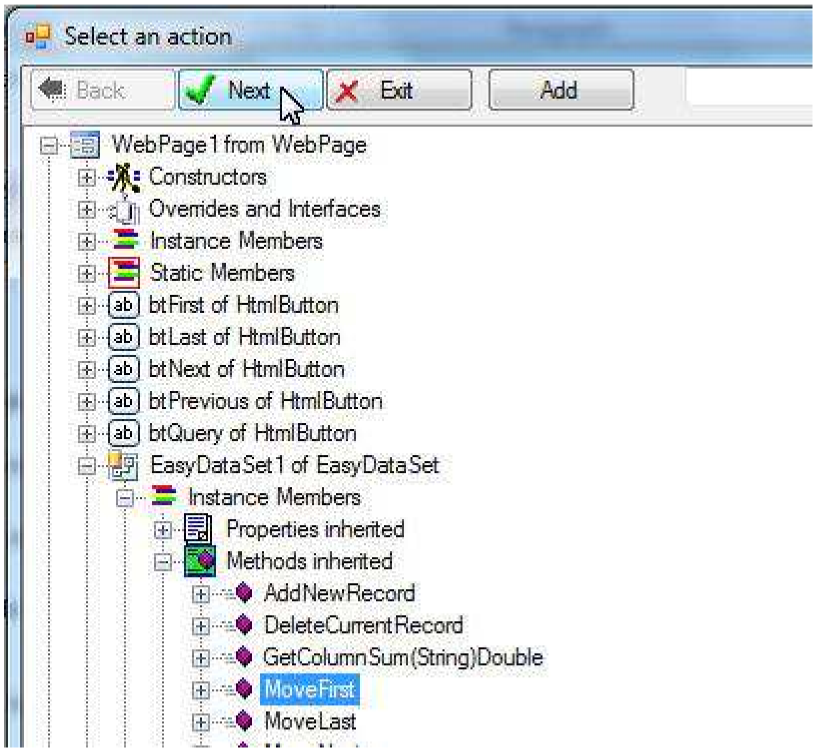
We add 4 buttons for this purpose:



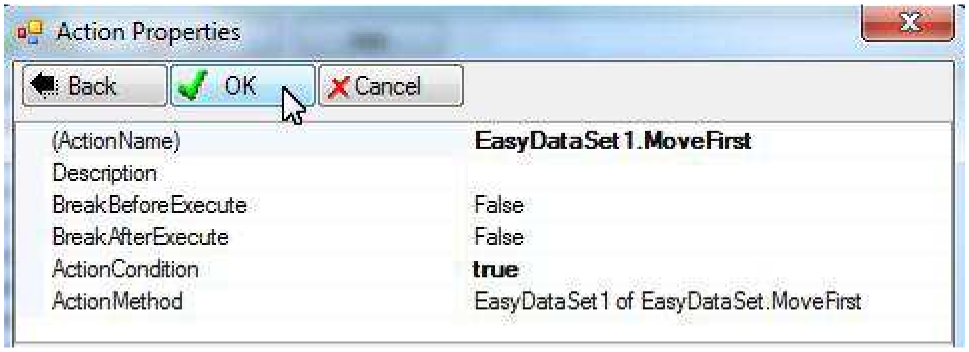
Let these buttons trig actions. Right-click button "|<"; choose "Assign Action"; choose "onclick" event:



Choose "MoveFirst" from the EasyDataSet; click Next:

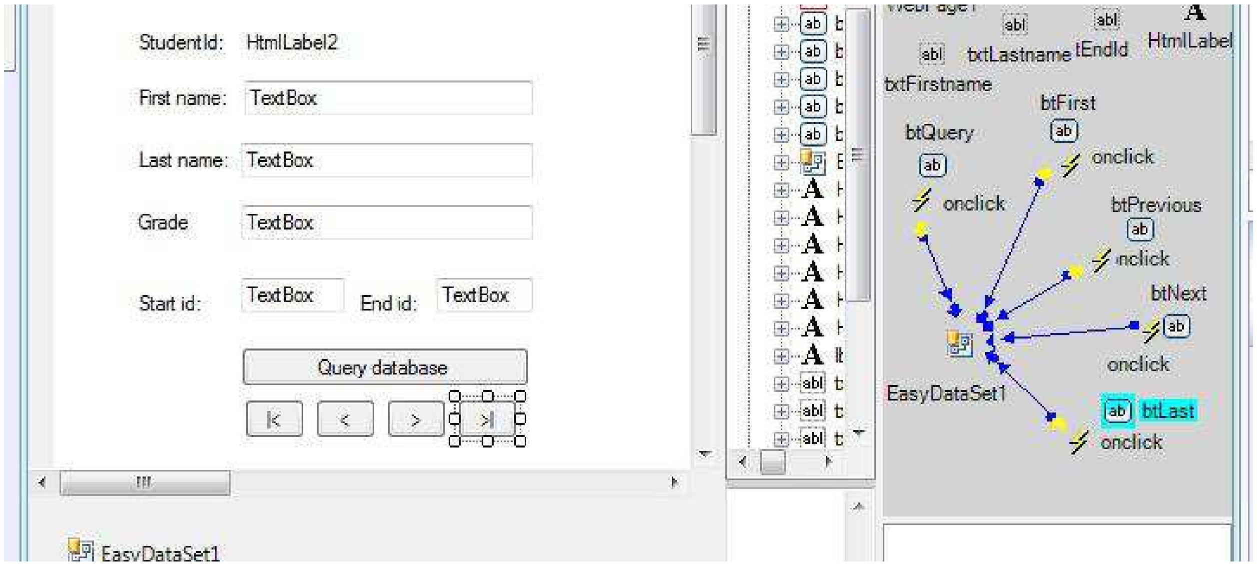


Click OK to finish creating the action and assigning the action to the event:



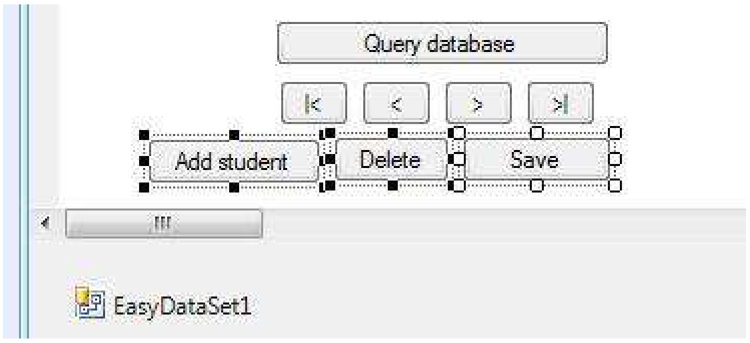
(ActionName)	EasyDataSet1.MoveFirst
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyDataSet1 of EasyDataSet.MoveFirst

Do the same for the other 3 buttons and use “MovePrevious”, “MoveNext” and “MoveLast” respectively.

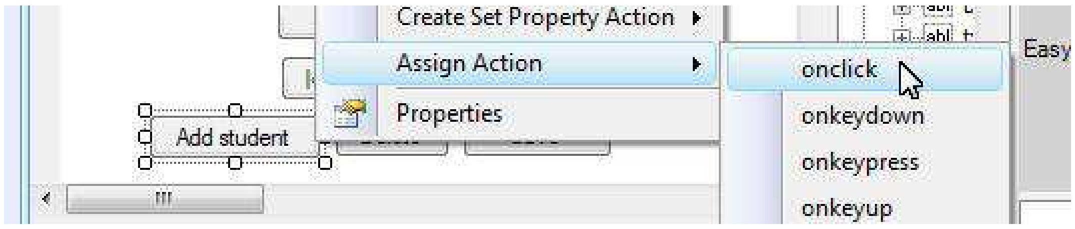


Modify and Save Data

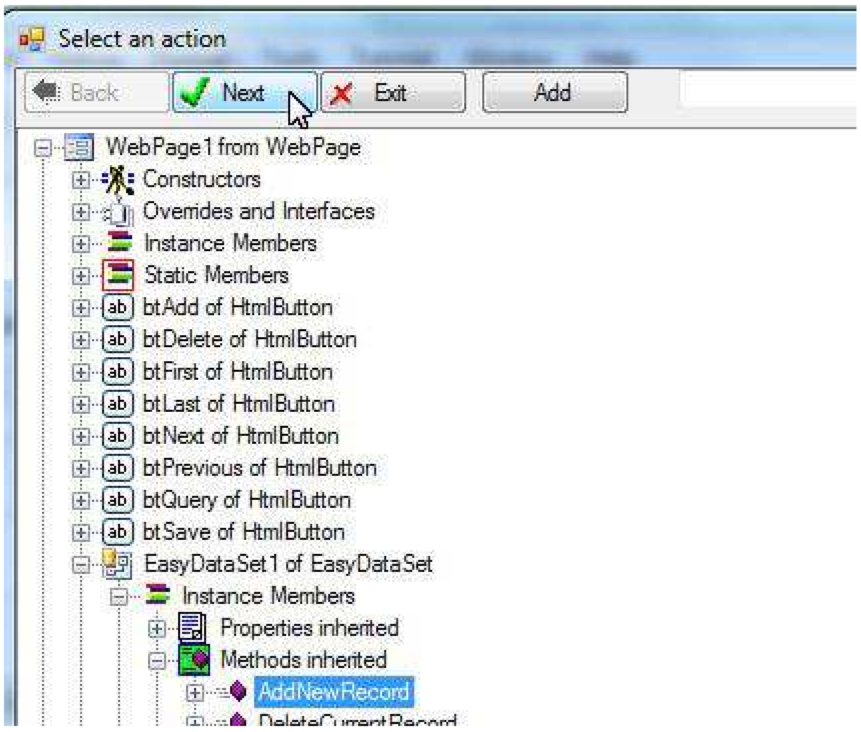
We may use a button for adding new record and a button for removing the current record.



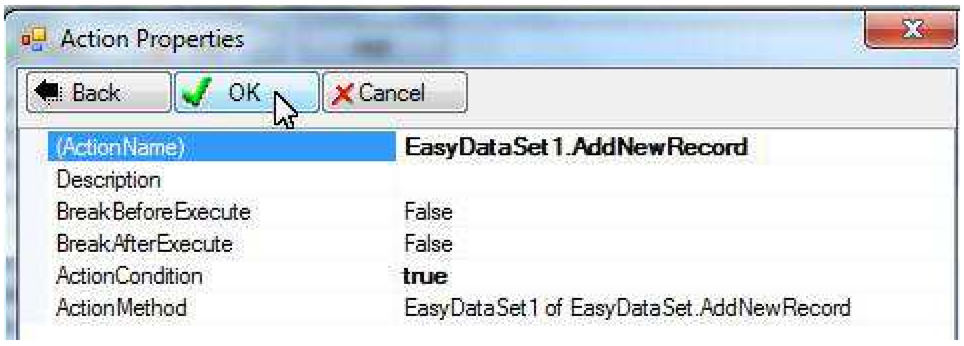
Right-click “Add student” button; choose “Assign Action”; choose “onclick” event:



Choose "AddNewRecord" and click Next:

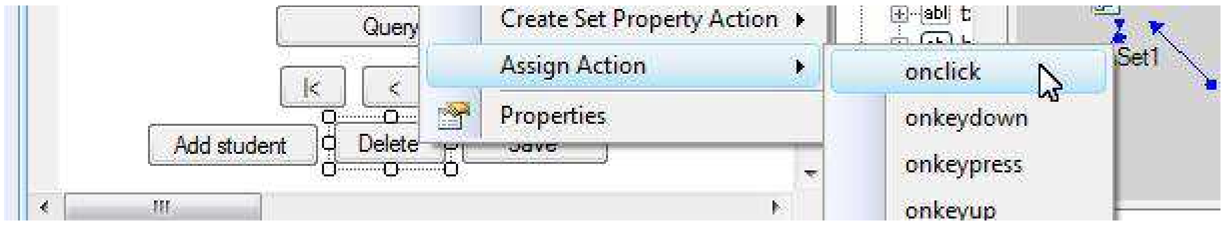


Click OK to finish creating the action and assigning it to the event:

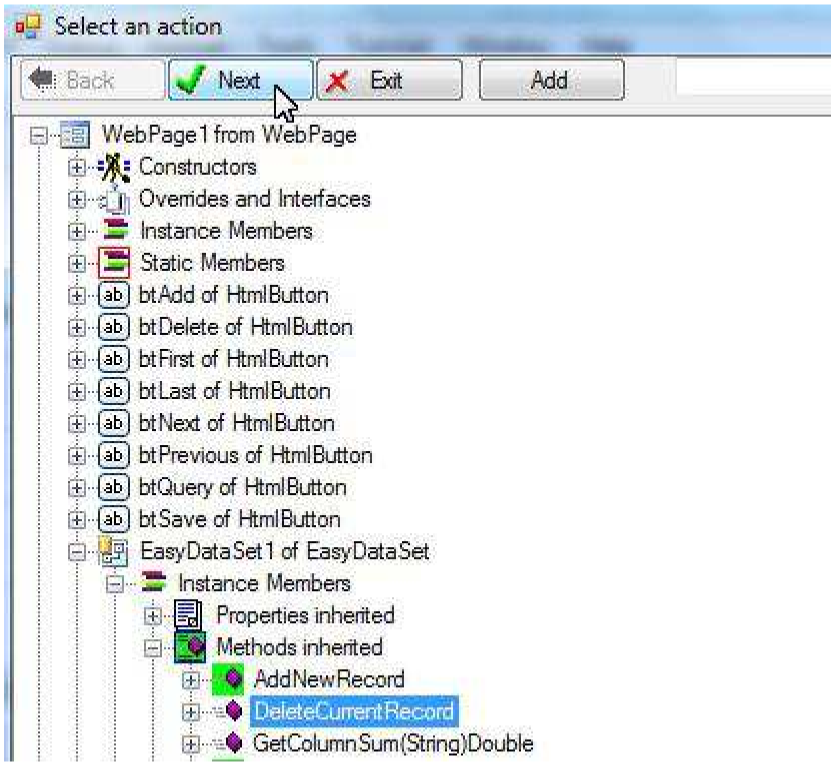


(ActionName)	**EasyDataSet1.AddNewRecord**
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	EasyDataSet1 of EasyDataSet.AddNewRecord

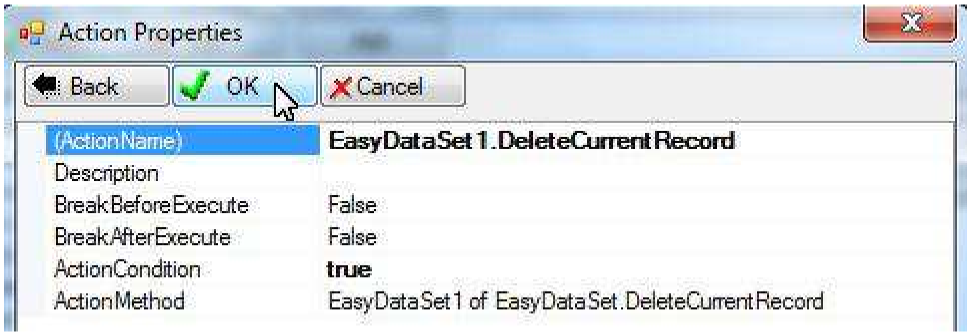
Right-click button "Delete"; choose "Assign Action"; choose "onclick" event:



Choose “DeleteCurrentRecord”; click Next:



Click OK to finish creating the action and assigning it to the event:



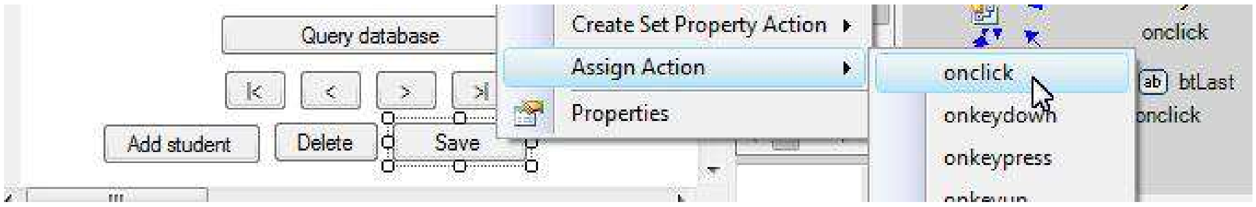
(ActionName)	EasyDataSet1.DeleteCurrentRecord
Description	
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyDataSet1 of EasyDataSet.DeleteCurrentRecord

One click of the “Add student” button will add one new student record on the page. Click the “Delete” button will remove the current record from the page. Typing in the text boxes will modify the student data on the page.

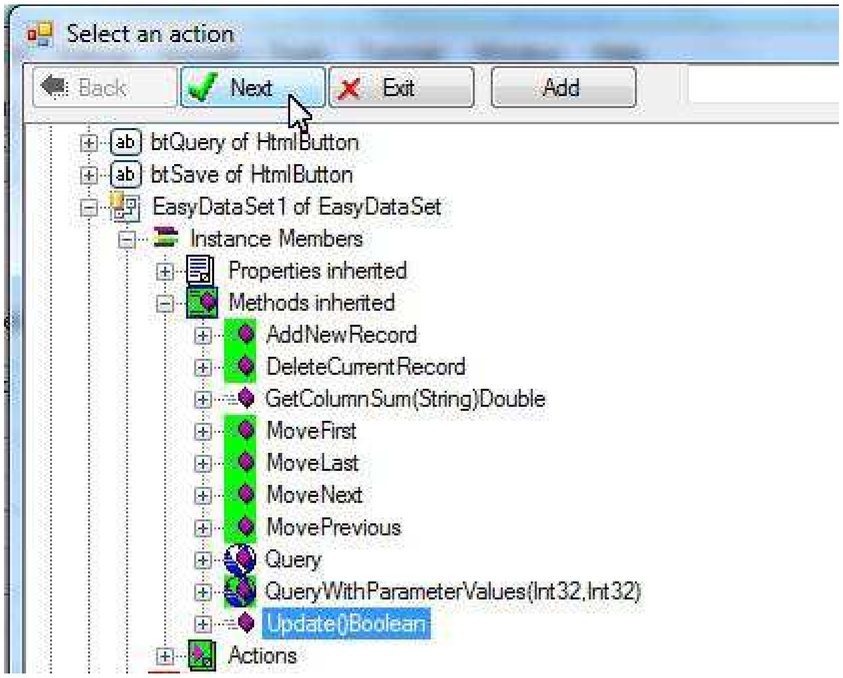
Note that all the modifications are kept only in the web page. Nothing is changed in the database.

To save all the modifications kept in the page to the database, we need to execute an Update action. We use a “Save” button to trig such an action.

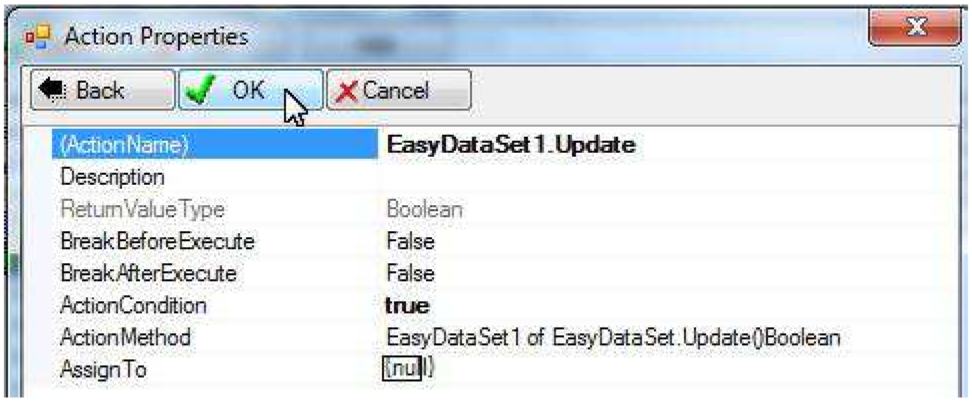
Right-click the “Save” button; choose “Assign Action”; choose “onclick” event:



Choose “Update”; click Next:

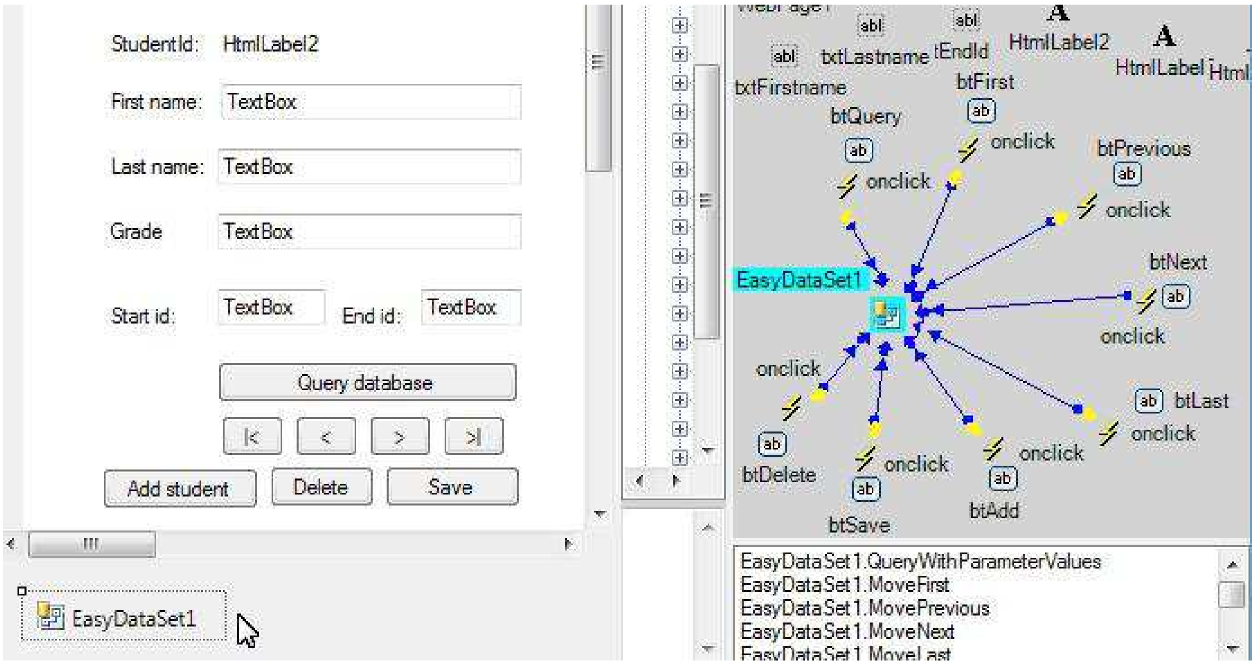


Click OK to finish creating the action and assigning it to the event:



(ActionName)	EasyDataSet1.Update
Description	
ReturnValueType	Boolean
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	EasyDataSet1 of EasyDataSet.Update()Boolean
AssignTo	(null)

We have all the buttons triggering actions executed by the EasyDataSet:

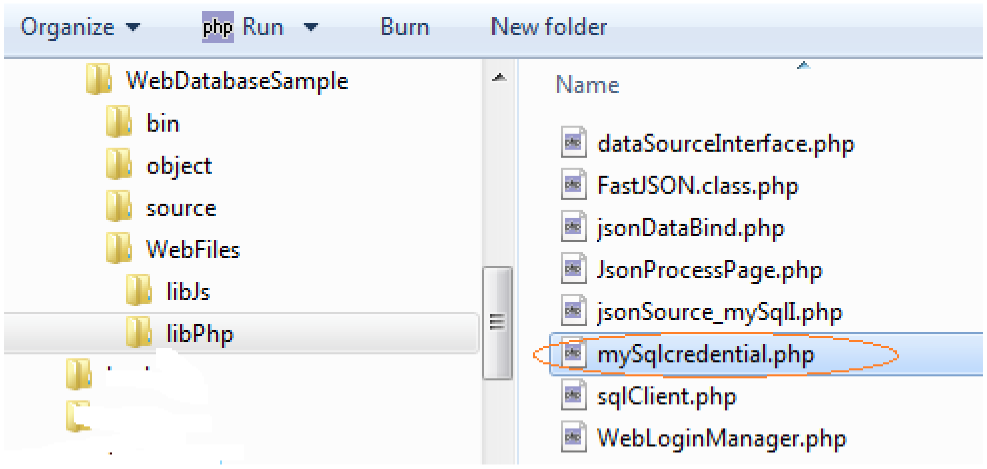


EasyDataSet1.QueryWithParameterValues
EasyDataSet1.MoveFirst
EasyDataSet1.MovePrevious
EasyDataSet1.MoveNext
EasyDataSet1.MoveLast

MySql Database Credential

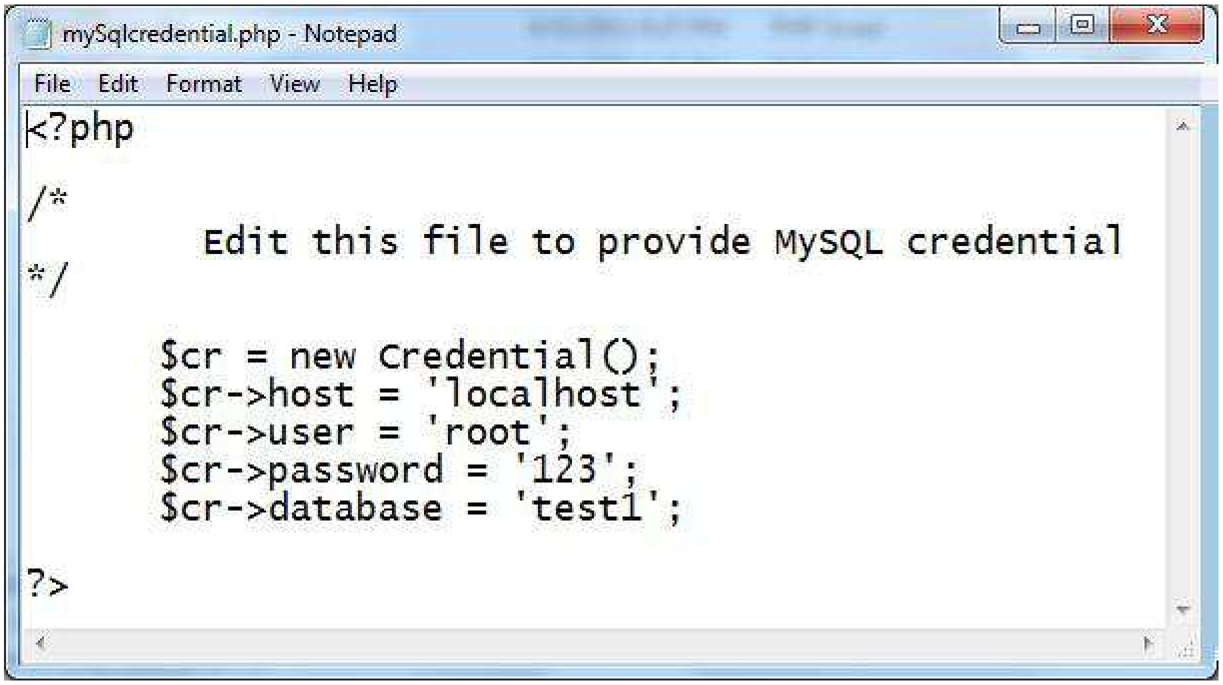
Compile the project. All web files are generated under WebFiles folder.

The credential for MySql database is in a file named mySqlcredential.php in the libPhp folder:



dataSourceInterface.php
FastJSON.class.php
jsonDataBind.php
JsonProcessPage.php
jsonSource_mySqlI.php
mySqlcredential.php
sqlClient.php
WebLoginManager.php

You may edit this page to set the credential for you database.

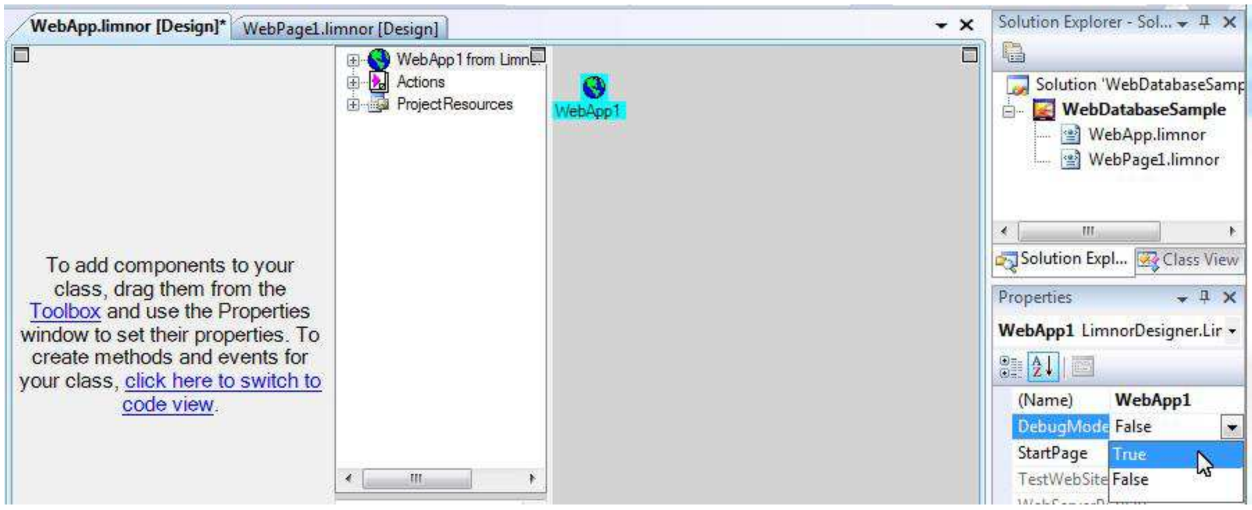


```
<?php

/*
        Edit this file to provide MySQL credential
*/

      $cr = new Credential();
      $cr->host = 'localhost';
      $cr->user = 'root';
      $cr->password = '123';
      $cr->database = 'test1';

?>
```

You edit the PHP code in the above file to provide MySql credential in a variable named $cr. The file will not be overwritten by the compiler if you modify it.

Test Data Entry

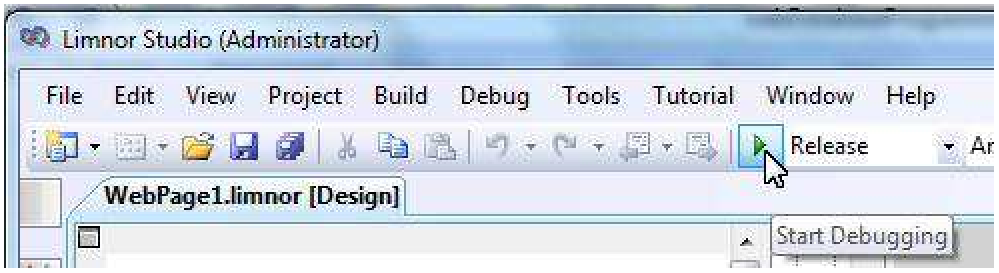
Let's turn on DebugMode so that we may watch client/server communications and watch for any errors:



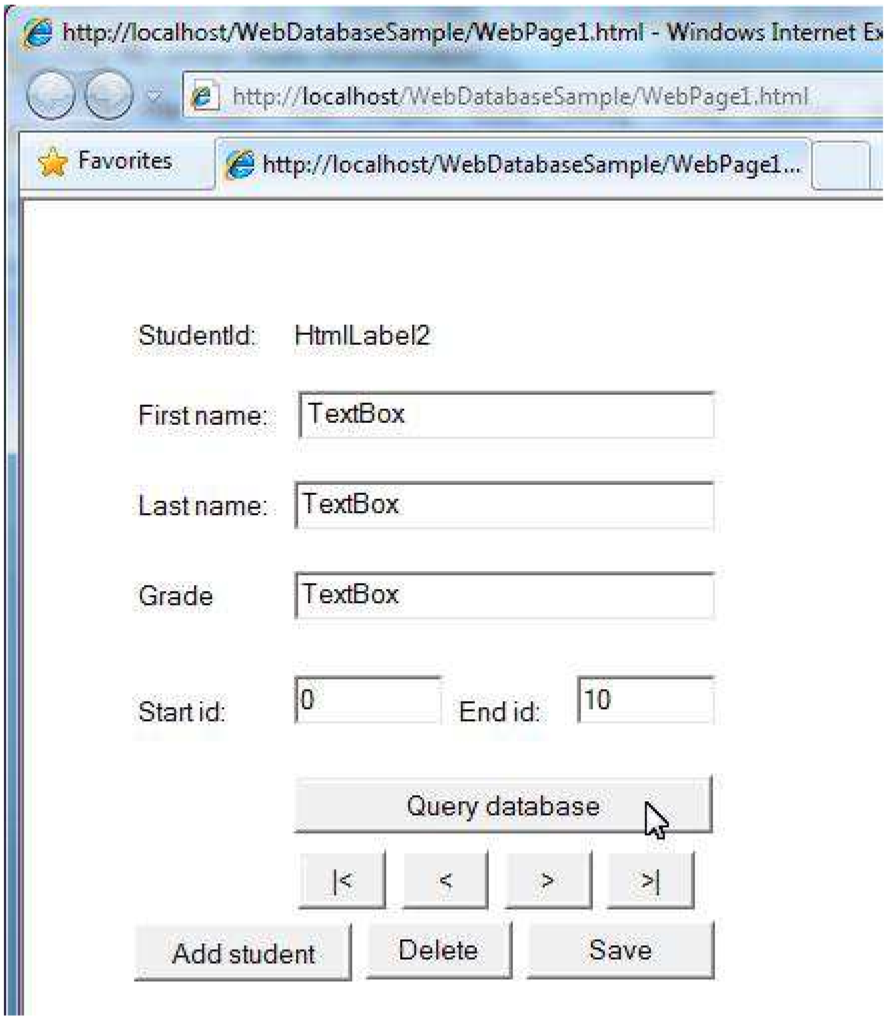
To add components to your class, drag them from the Toolbox and use the Properties window to set their properties. To create methods and events for your class, click here to switch to code view.

(Name)	WebApp1
DebugMode	False
StartPage	True
TestWebSite	False

Start debugging:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window Help
Release
WebPage1.limnor [Design]
Start Debugging

Enter id range and click "Query database" to get data:


http://localhost/WebDatabaseSample/WebPage1.html - Windows Internet Ex
http://localhost/WebDatabaseSample/WebPage1.html
Favorites
http://localhost/WebDatabaseSample/WebPage1...
Query database
|<
<
>
>|
Add student
Delete
Save

StudentId: HtmlLabel2

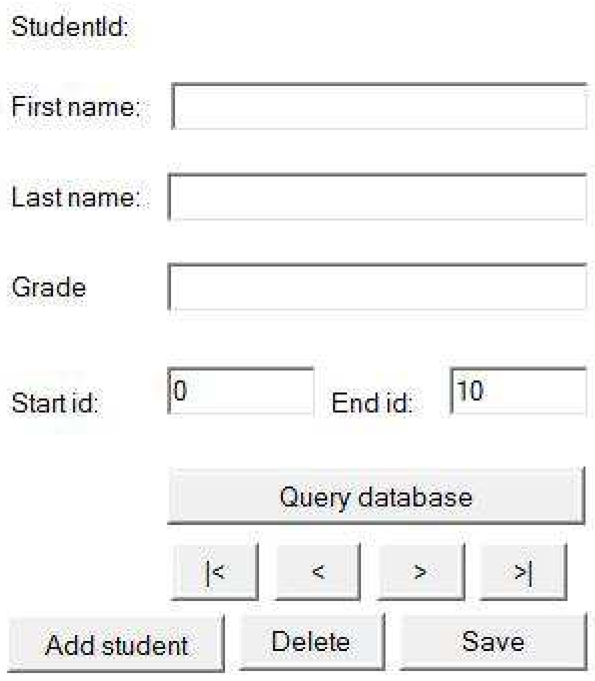
First name: TextBox

Last name: TextBox

Grade TextBox

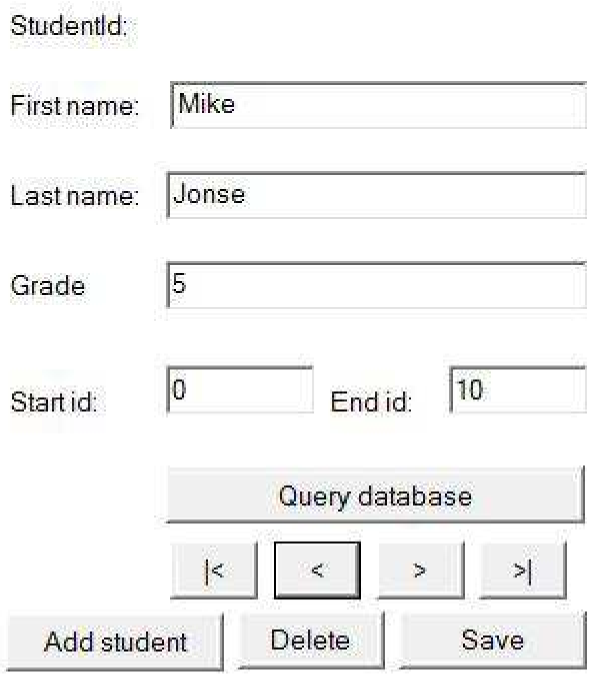
Start id: 0 End id: 10

We'll see empty data if no records are found in the id range:



StudentId:			
First name:			
Last name:			
Grade			
Start id:	0	End id:	10

Click “Add student” button. Enter some data in the text boxes.

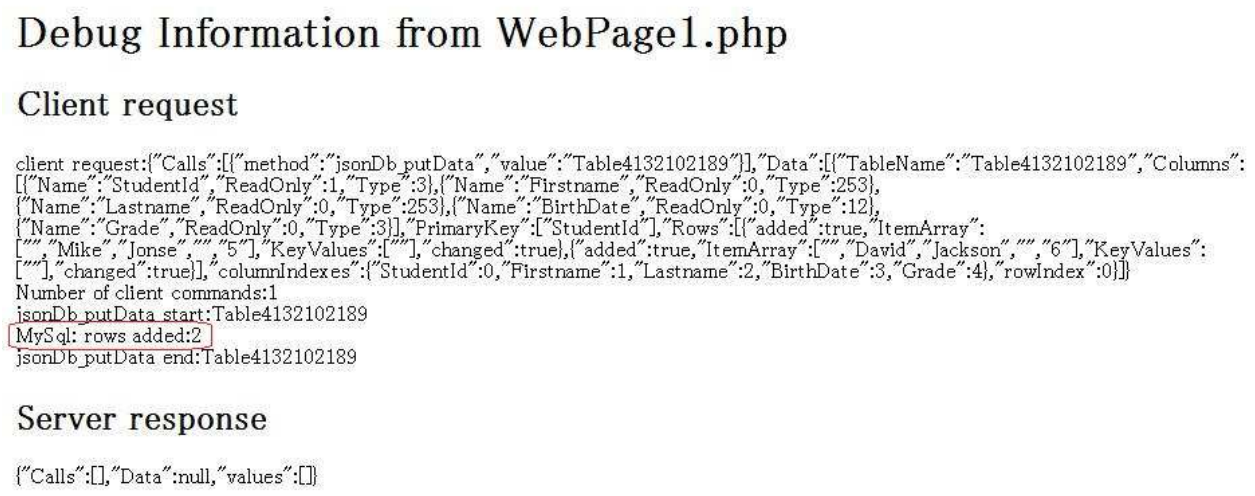


StudentId:			
First name:	Mike		
Last name:	Jonse		
Grade	5		
Start id:	0	End id:	10

You may click “Add student” many times to add many students. You may click the navigation buttons “|<”, “<”, “>” and “>|” to go through the students. You may also click “Delete” button to remove the student record currently displayed on the page.

After making all the changes, click “Save” button to save the data to the database.

From the debugging information, we can see that how many records are added; how many records are deleted and how many records are modified:



Debug Information from WebPage1.php

Client request

```
client request:{"Calls":[{"method":"jsonDb_putData","value":"Table4132102189"}],"Data":[{"TableName":"Table4132102189","Columns":
[{"Name":"StudentId","ReadOnly":1,"Type":3},{"Name":"Firstname","ReadOnly":0,"Type":253},
{"Name":"Lastname","ReadOnly":0,"Type":253},{"Name":"BirthDate","ReadOnly":0,"Type":12},
{"Name":"Grade","ReadOnly":0,"Type":3}],"PrimaryKey":["StudentId"],"Rows":[{"added":true,"ItemArray":
["","Mike","Jonse","","5"],"KeyValues":[""],"changed":true},{"added":true,"ItemArray":["","David","Jackson","","6"],"KeyValues":
[""],"changed":true}],"columnIndexes":{"StudentId":0,"Firstname":1,"Lastname":2,"BirthDate":3,"Grade":4},"rowIndex":0}]}
Number of client commands:1
jsonDb_putData start:Table4132102189
MySql: rows added:2
jsonDb_putData end:Table4132102189
```

Server response

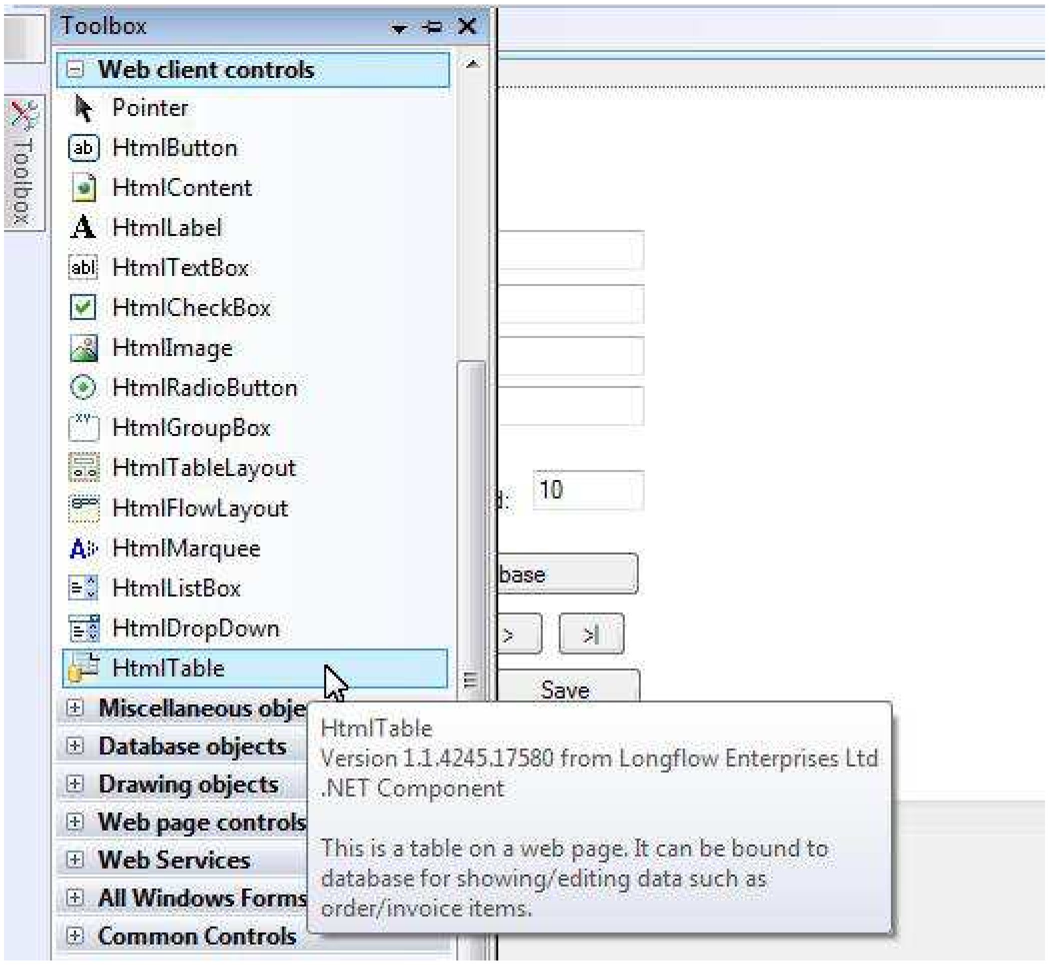
```
{"Calls":[],"Data":null,"values":[]}
```

We may close the web page and re-open it. Set the id range and click “Query database” button. Now we will see student records we just entered. It proves that the data we entered have been saved in the database.

Bind Data to Html Table

Records from a database can be bound to a HTML table for displaying and editing.

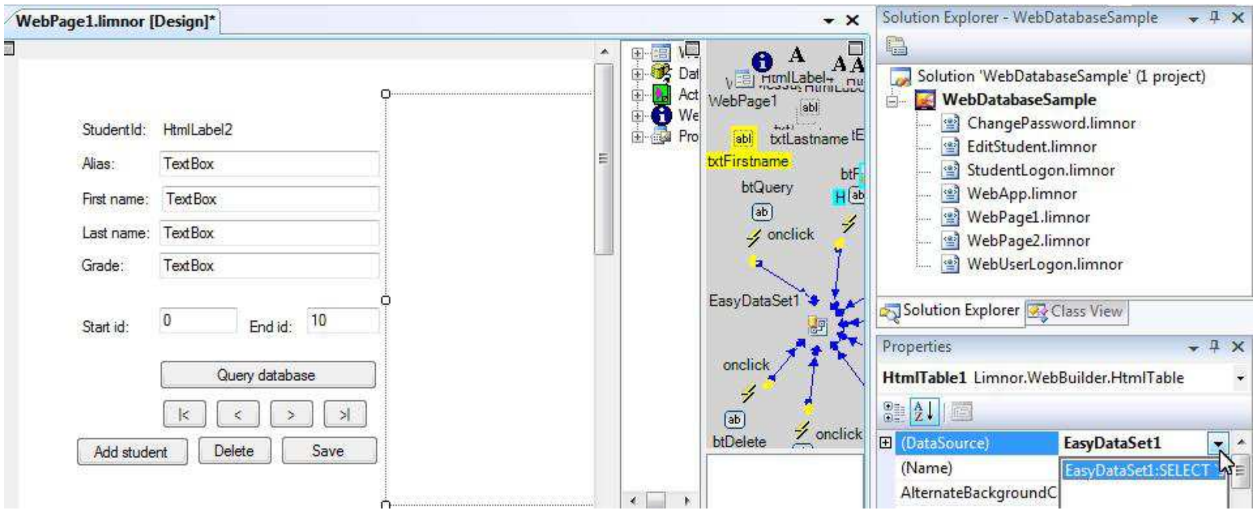
Drop an HtmlTable to the web page:



HtmlTable
Version 1.1.4245.17580 from Longflow Enterprises Ltd
.NET Component

This is a table on a web page. It can be bound to database for showing/editing data such as order/invoice items.

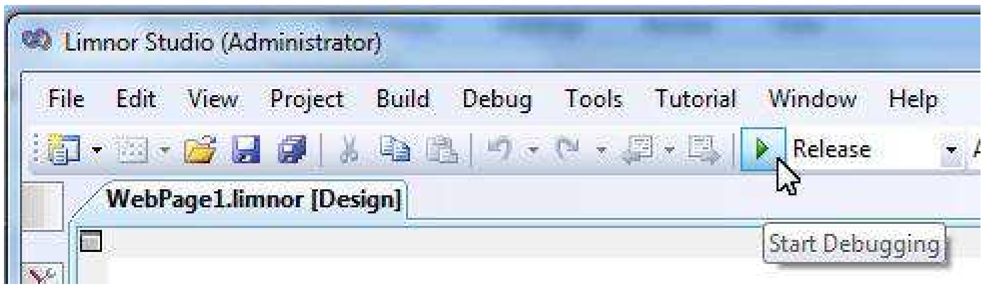
Set its DataSource property to EasyDataSet1:



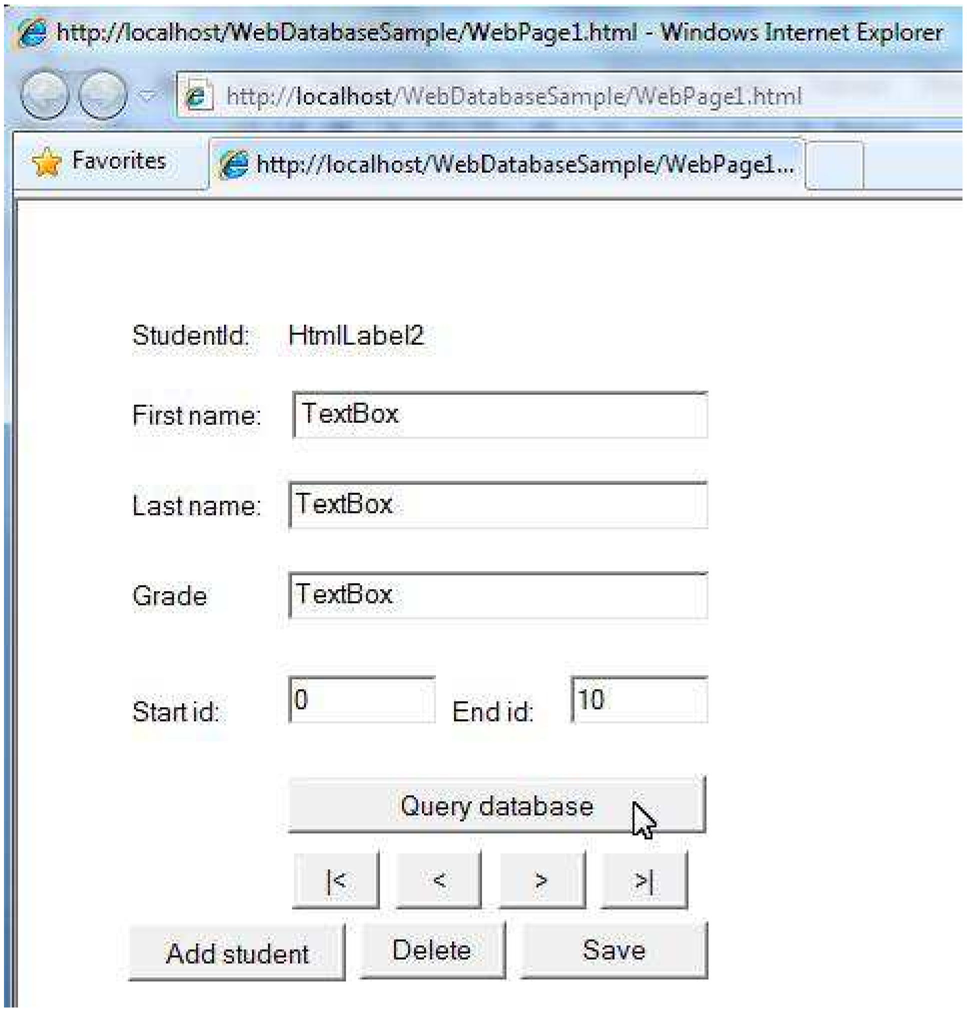
HtmlTable1 Limnor.WebBuilder.HtmlTable

(DataSource)	EasyDataSet1
(Name)	EasyDataSet1:SELECT
AlternateBackgroundC	

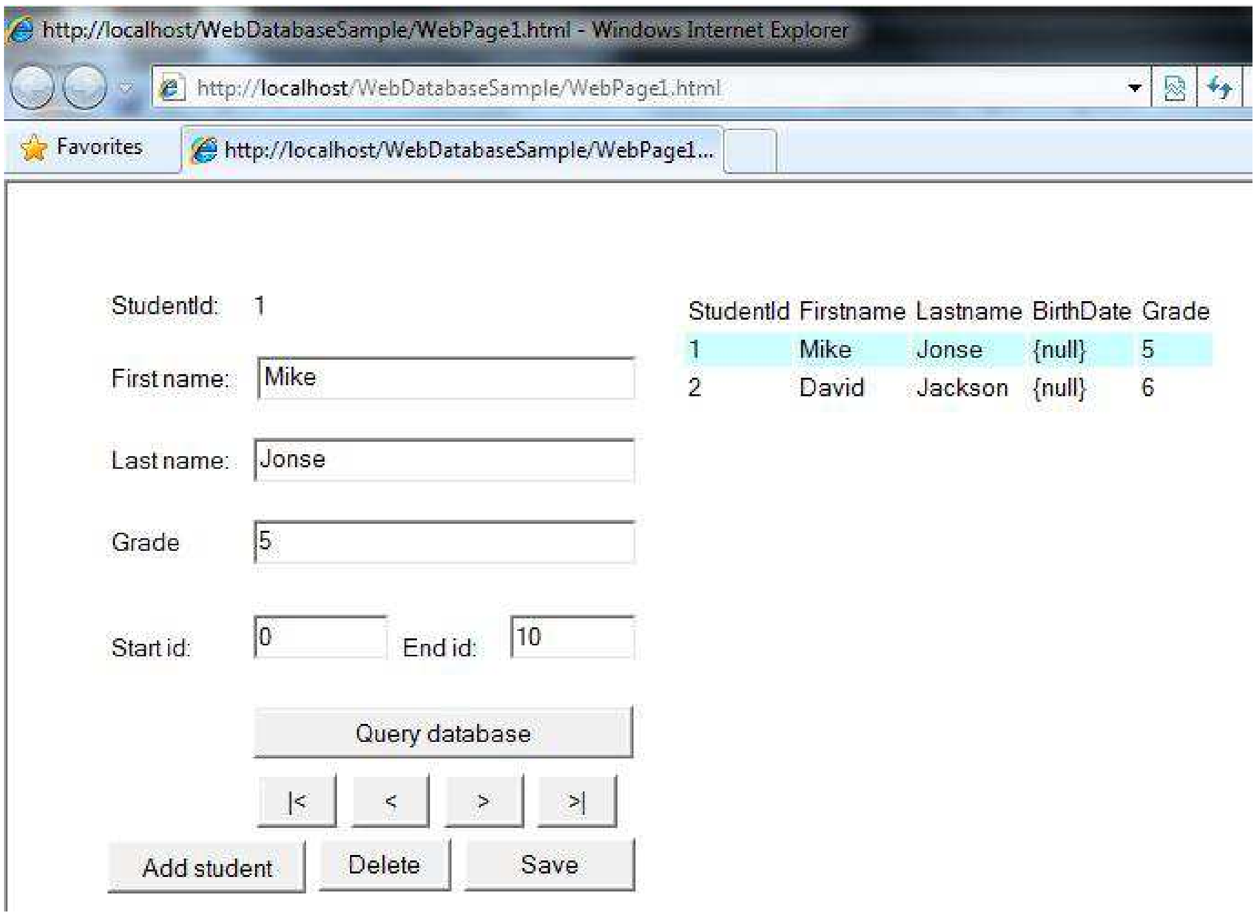
Run the project:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window Help
Release
WebPage1.limnor [Design]
Start Debugging

Query the database by clicking the “Query database” button:


http://localhost/WebDatabaseSample/WebPage1.html - Windows Internet Explorer
http://localhost/WebDatabaseSample/WebPage1.html
Favorites
http://localhost/WebDatabaseSample/WebPage1...
StudentId: HtmlLabel2
First name: TextBox
Last name: TextBox
Grade TextBox
Start id: 0 End id: 10
Query database
|< < > >|
Add student Delete Save

The data appears in the html elements:


http://localhost/WebDatabaseSample/WebPage1.html - Windows Internet Explorer
http://localhost/WebDatabaseSample/WebPage1.html
Favorites
http://localhost/WebDatabaseSample/WebPage1...

StudentId: 1

First name: Mike

Last name: Jonse

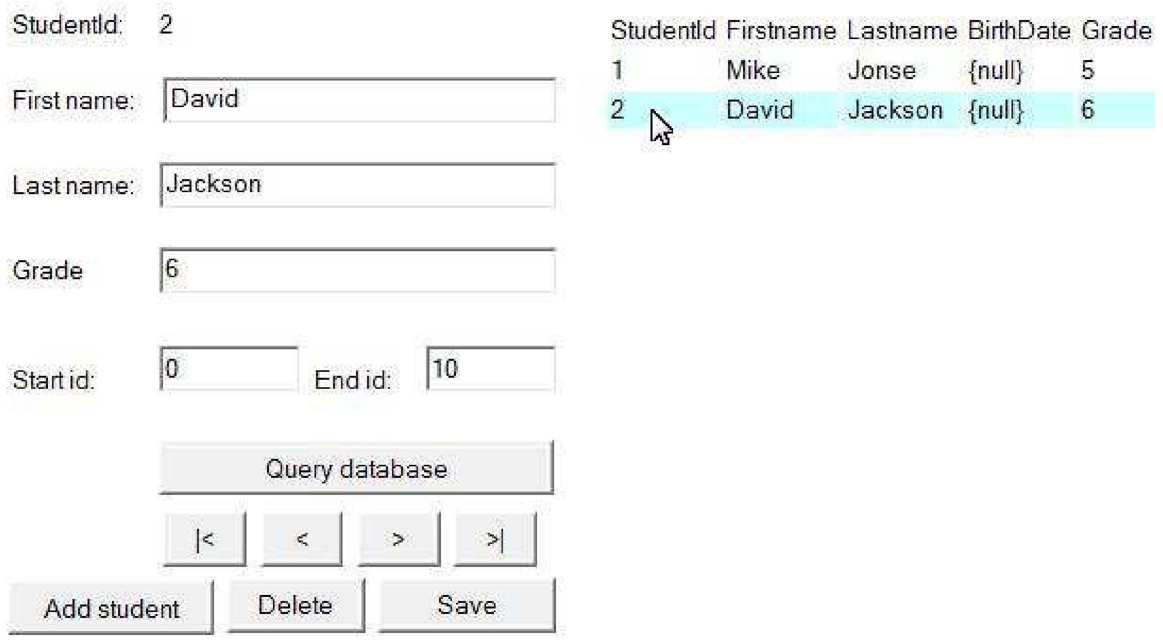
Grade 5

Start id: 0 End id: 10

Query database
|<
<
>
>|
Add student
Delete
Save

StudentId	Firstname	Lastname	BirthDate	Grade
1	Mike	Jonse	{null}	5
2	David	Jackson	{null}	6

The highlighted row of the html table indicates the current record of the data. The user may click a row in the html table to make it the current record. Let’s click the second row:



StudentId: 2

First name: David

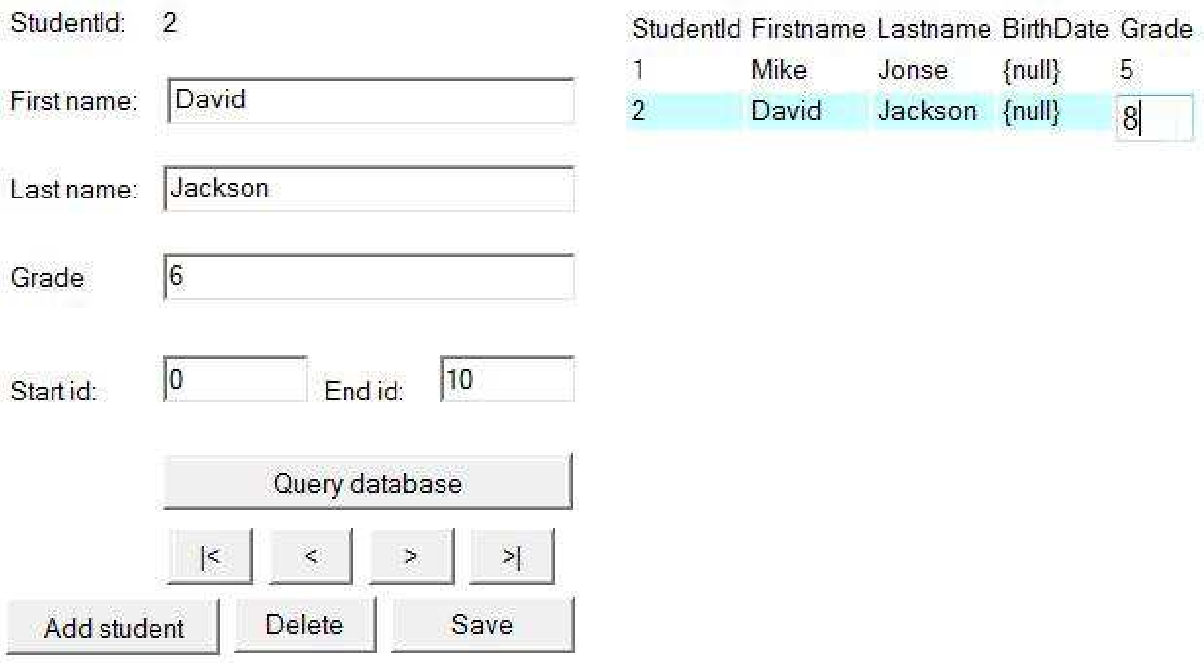
Last name: Jackson

Grade 6

Start id: 0 End id: 10

StudentId	Firstname	Lastname	BirthDate	Grade
1	Mike	Jonse	{null}	5
2	David	Jackson	{null}	6

The user may click a cell in the html table to edit the data in the cell. Let's click the Grade cell of the second row:



StudentId: 2

First name: David

Last name: Jackson

Grade 6

Start id: 0 End id: 10

StudentId	Firstname	Lastname	BirthDate	Grade
1	Mike	Jonse	{null}	5
2	David	Jackson	{null}	8

Web Page Security

See http://www.limnor.com/support/webDatabaseProgramming2.pdf

Field Editors

See http://www.limnor.com/support/webDatabaseProgramming3.pdf

Data Repeater

In chapter “Bind-data to single field”, we saw that a form can be designed with data-bound controls. Data from database are automatically displayed on the controls. The user may modify data on the controls and the data modifications can be saved back to database.

In such arrangement, one record is displayed on one web page.

Data Repeater allows you to design the form with data-bound controls, but the same design can be repeated on one web page. Thus many records can be displayed on one web page.

For details, see http://www.limnor.com/support/WebDataRepeater.pdf

Change Query Filters at Runtime

See http://www.limnor.com/support/webDatabaseProgramming4.pdf

Create New Records

See http://www.limnor.com/support/webDatabaseProgramming5.pdf

Data Streaming

See http://www.limnor.com/support/webDatabaseProgramming5.pdf

Fetch Data of One-to-Many Relation

See http://www.limnor.com/support/webDatabaseProgramming5.pdf

Change Data-binding

See http://www.limnor.com/support/webDatabaseProgramming5.pdf

Feedbacks

Please send your feedbacks to support@limnor.com, thanks!