

Web Database Programming

Created: 2011-01-21

Last update: 2014-01-14

Contents

Introduction	2
Use EasyDataSet as Data Source.....	2
Bind-data to single field	2
Data Query	2
Data Search	2
Data Navigation	2
Modify and Save Data	2
MySQL Database Credential.....	3
Bind Data to Html Table.....	3
Web Page Security	3
Field Editors	3
Data Repeater	3
Change Query Filters at Runtime.....	3
Create New Records.....	3
AddNewRecord	3
Execute AddNewRecord.....	4
Execute Update	5
Handle DataUpdated event	7
Test adding new records.....	11
CreateNewRecord	12
Execute CreateNewRecord	13
Test adding new records.....	18
Data Streaming	20
Use sorting	20
Specify batch size	21

Specify key type	21
Enable data batching	22
Batching performance	22
Fetch Data of One-to-Many Relation	23
Manually fetch detail records	23
Automatically fetch detail records.....	26
Change Data-binding.....	27
Feedbacks	34

Introduction

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Use EasyDataSet as Data Source

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Bind-data to single field

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Data Query

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Data Search

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Data Navigation

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Modify and Save Data

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

MySQL Database Credential

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

Bind Data to Html Table

See <http://www.limnor.com/support/webDatabaseProgramming1.pdf>

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

EasyDataSet provides two methods for adding new records to databases, `AddNewRecord` and `CreateNewRecord`. They can be used for different scenarios.

AddNewRecord

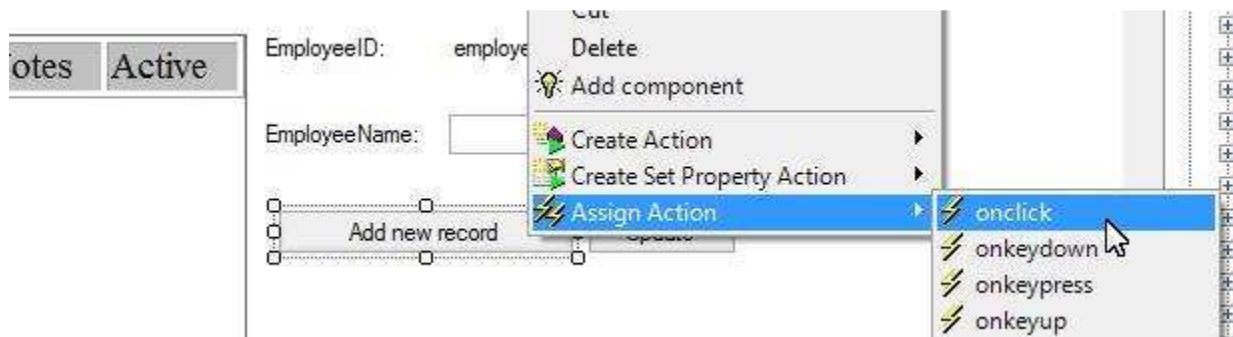
An `AddNewRecord` action adds a new record to the web page.

- The new record is added to the web page but not to the database. An `Update` action must be executed later, mostly after you let the user modify the record on the web page, to save the record to the database. If the web page is closed without executing an `Update` action then the new record is discarded.

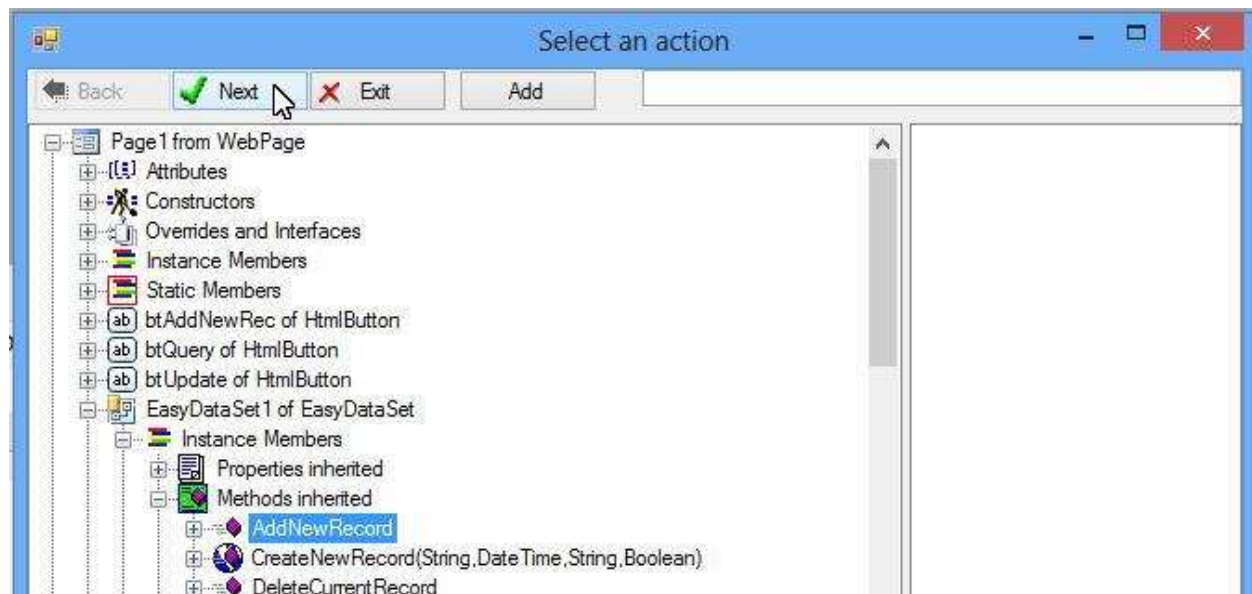
- If the record contains an identity field, also called an auto-number field, then the identity field is empty. You cannot use the new record as a primary record on the web page to create new secondary records.
- AddNewRecord can be executed many times to create many new records on the web page. One Update action will save all new records to the database.
- NewRowCount property indicates how many new records are there on the web page. On executing an Update action, NewRowCount becomes 0.

Execute AddNewRecord

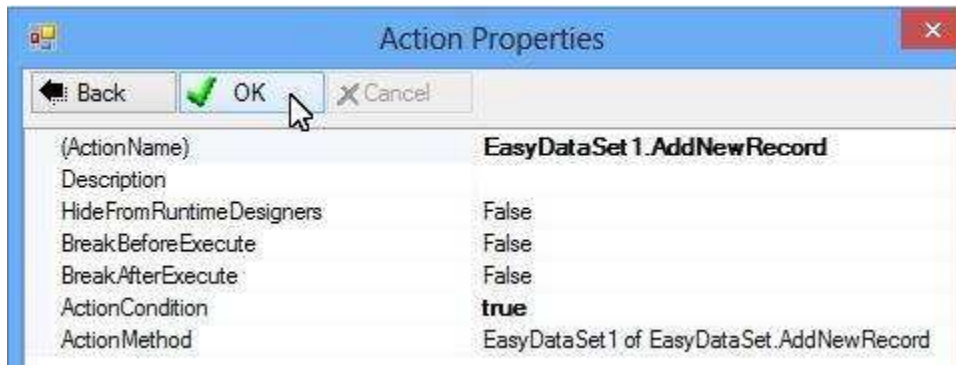
Let's assign an AddNewRecord action to a button:



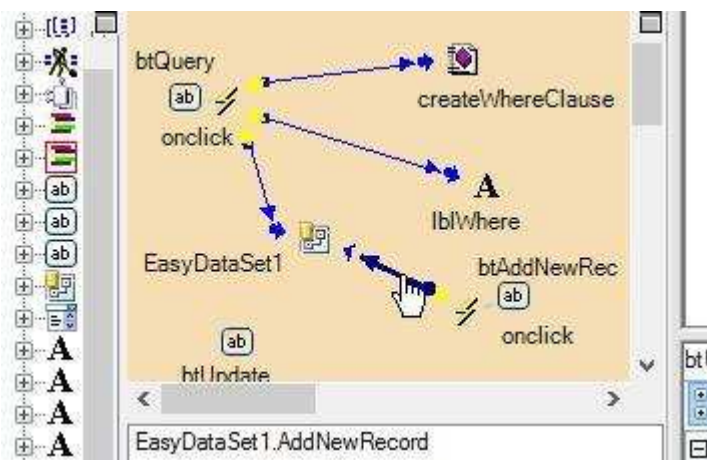
Select AddNewRecord method:



Click OK to create the action and assign it to the button:

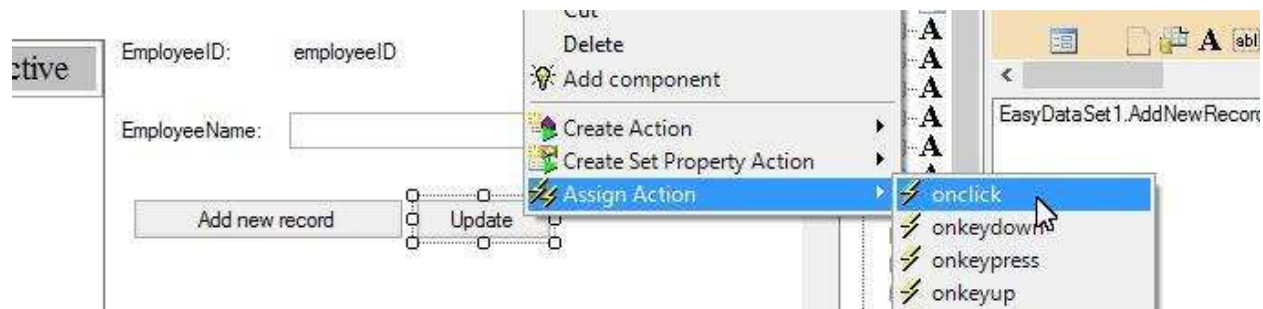


The action is created and assigned to the button:

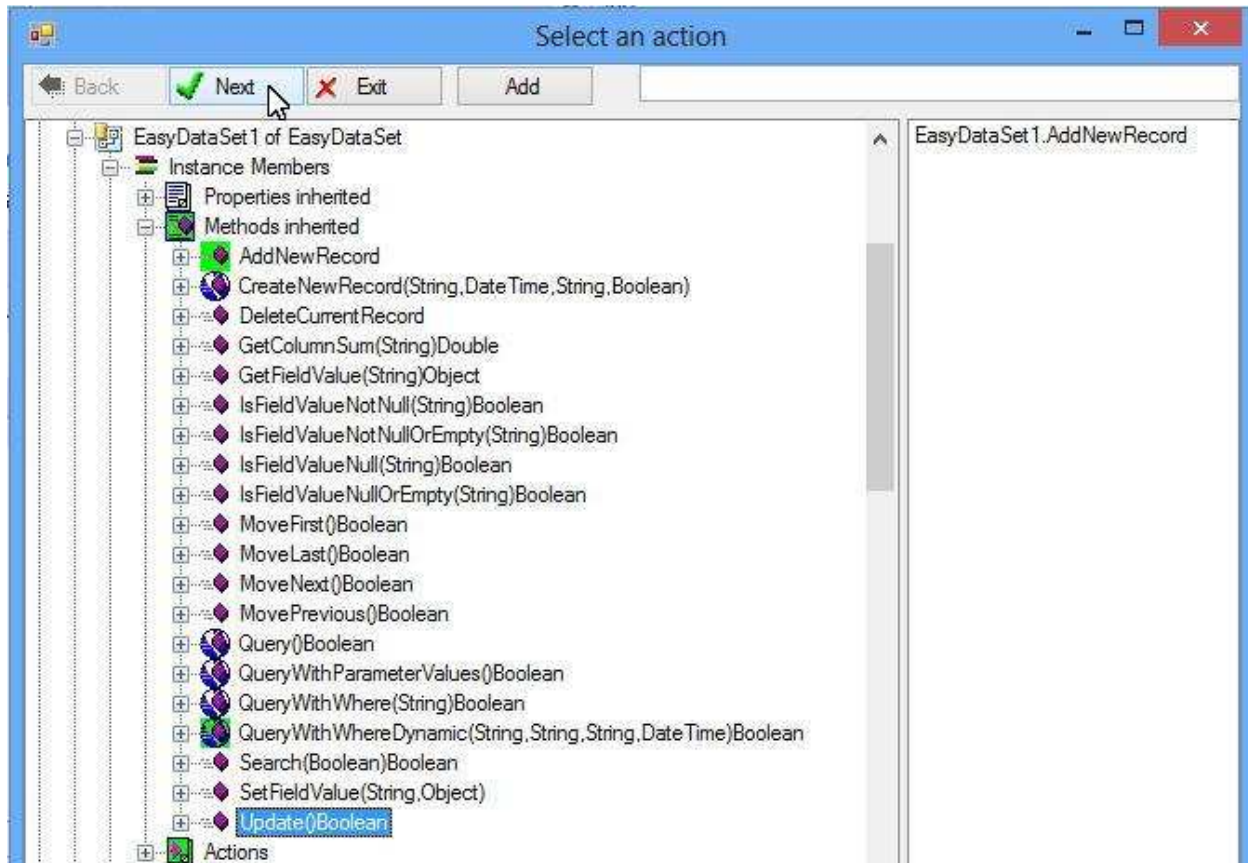


Execute Update

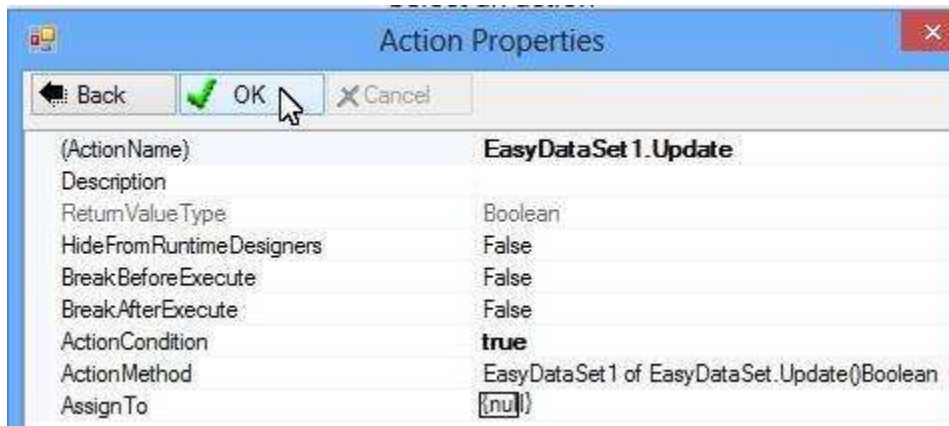
Let's assign an Update action to a button:



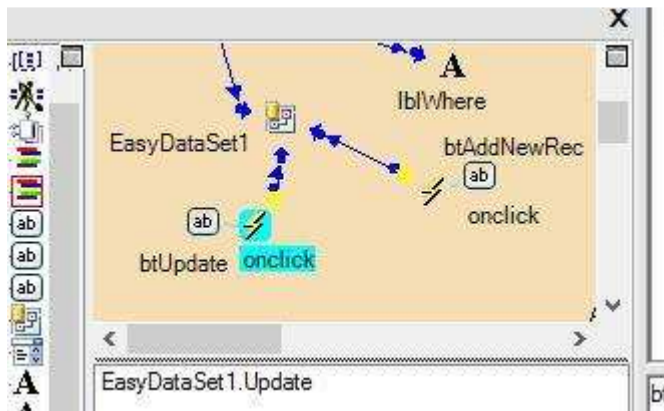
Select Update method:



Click OK to create the action:



The action is created and assigned to the button:



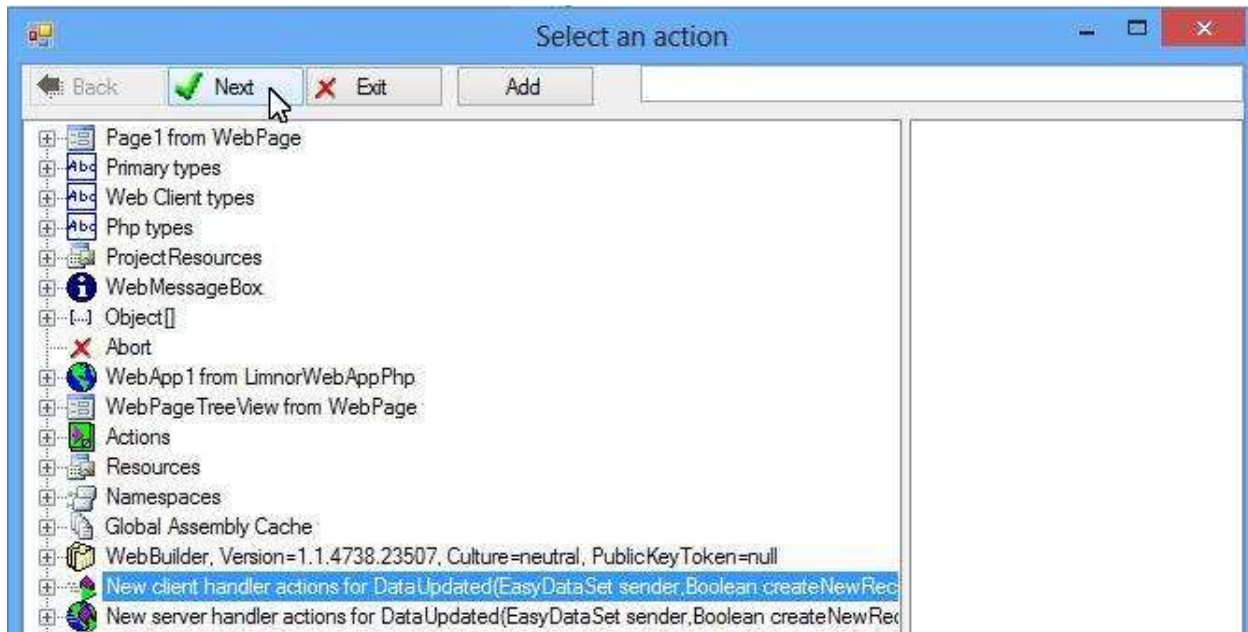
Handle DataUpdated event

DataUpdated event has a Boolean parameter, createNewRecord. If an Update action was executed then this parameter is False; if a CreateNewRecord action was executed then this parameter is True. If you want to handle DataUpdated event and you want to distinguish the two situations then you may use this parameter.

For a demonstration, we handle DataUpdated event by showing the value of parameter createNewRecord:



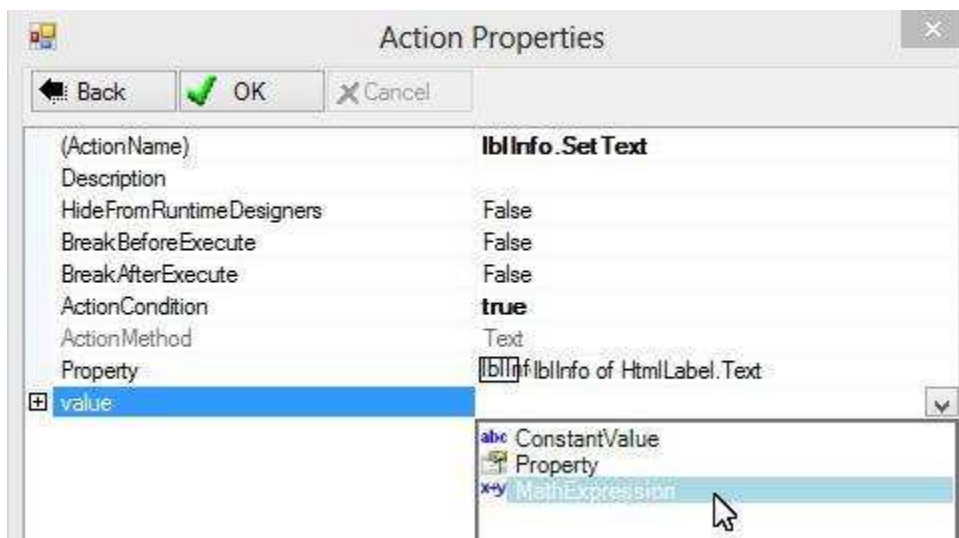
Because we want to use event parameters in the event handling, we need to create an event handler method:

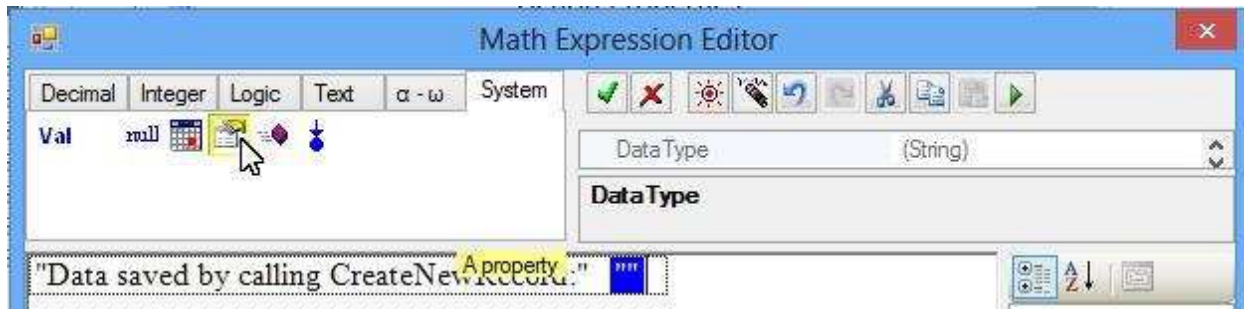
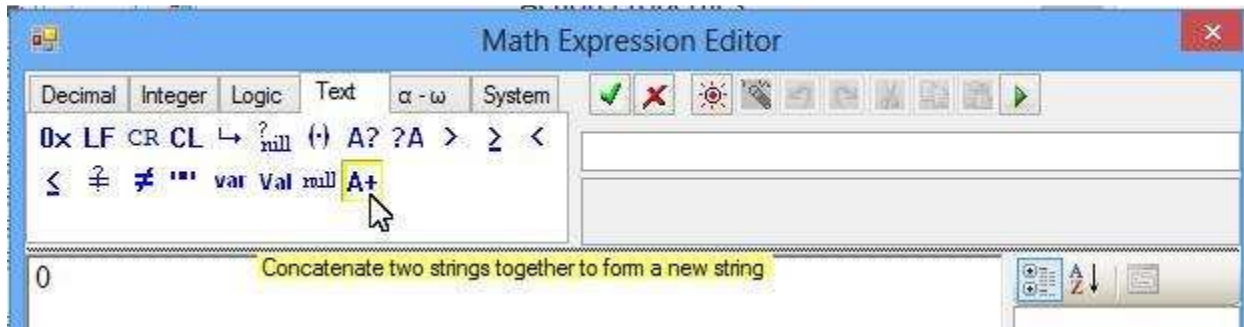


For this demo, we simply display event parameter “createNewRecord” in a label:

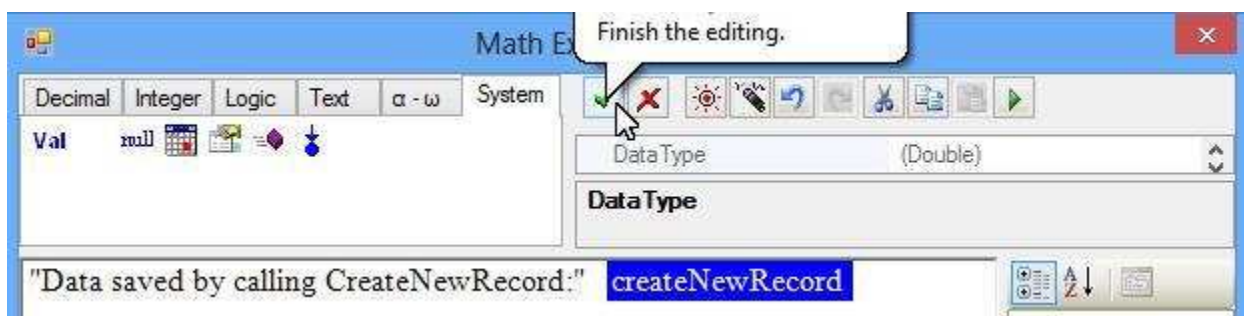


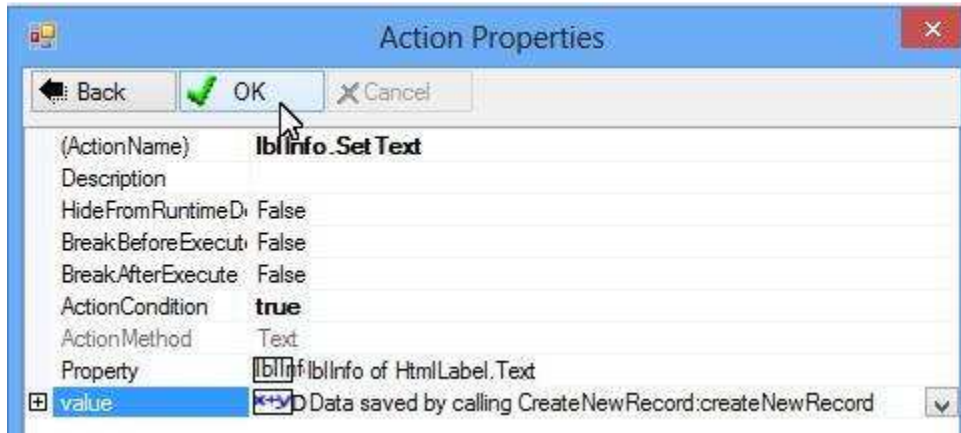
Create an expression for the message:



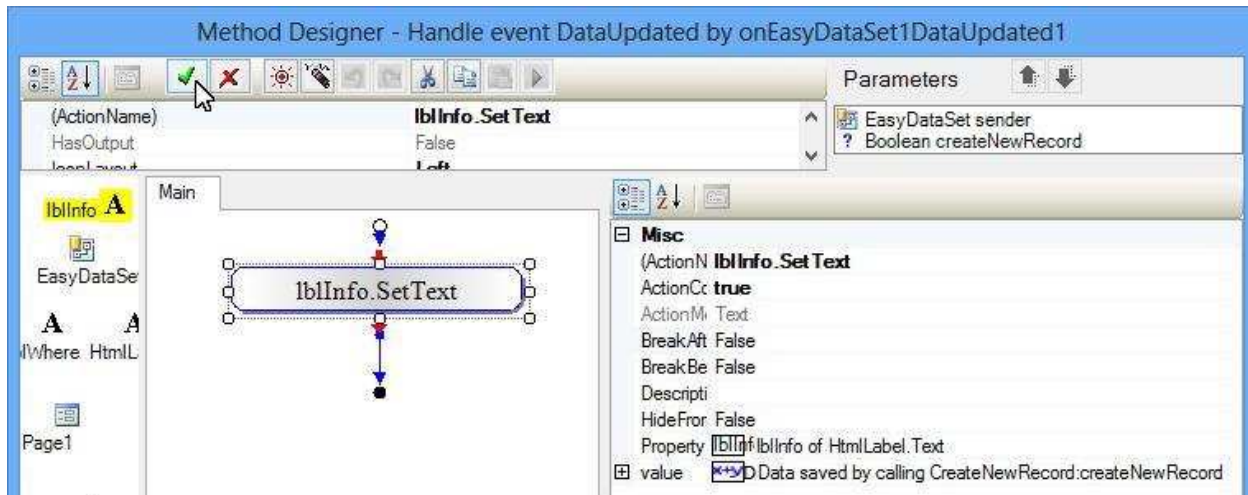


Select event parameter createNewRecord:

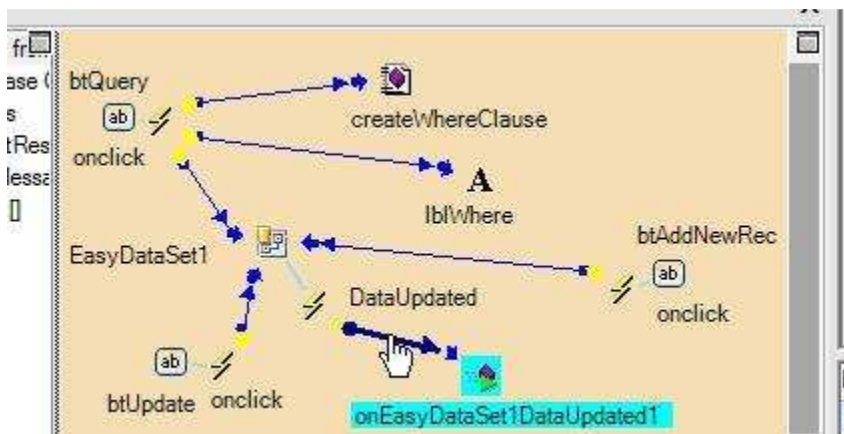




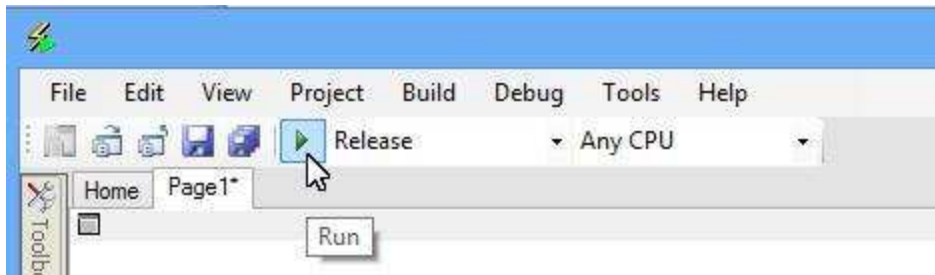
For this demo we just use this action:



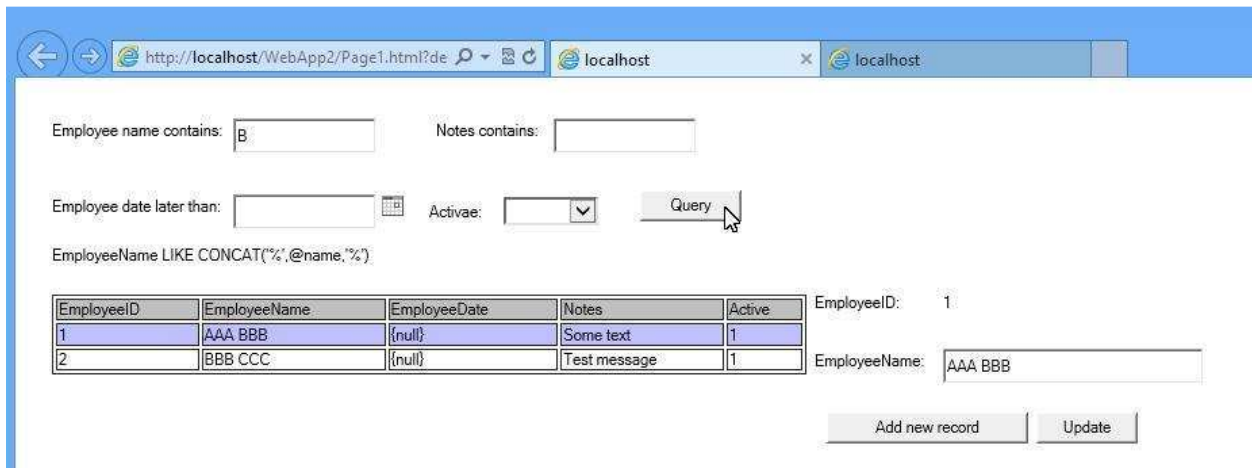
The event handler method for DataUpdated is created:



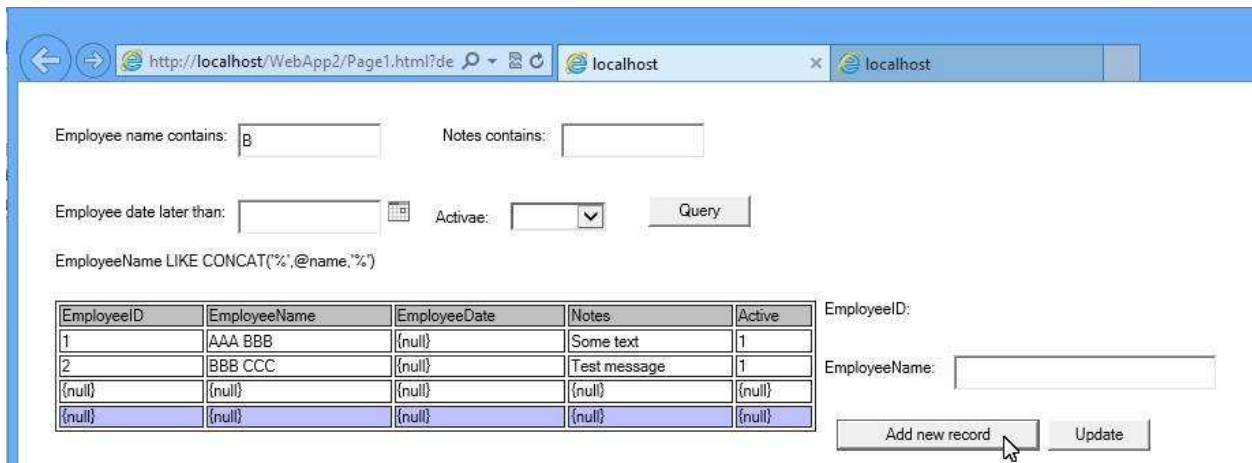
Test adding new records



The web page appears and we may do a search to get some records:



Click "Add new record" button twice to add two new records:



Note that EmployeeID is empty for new records (or an invalid value). Thus the new records cannot be used as primary records to add new secondary records using EmployeeID as foreign key.

Modify contents for records, including new records, and click Update button to save modifications to the database.

EmployeeID	EmployeeName	EmployeeDate	Notes	Active
1	AAA BBB	{null}	Some text	1
2	BBB CCC	{null}	Test message	1
{null}	Test Add New A	{null}	added by AddNewRecord	2
{null}	Test Add New B	{null}	added by AddNewRecord	3

The two new records are saved in the database. Note that the event parameter “createNewRecord” is “false” because the database action is “Update”, not “CreateNewRecord”:

EmployeeID	EmployeeName	EmployeeDate	Notes	Active
1	AAA BBB	{null}	Some text	1
2	BBB CCC	{null}	Test message	1
3	Test Add New A	{null}	added by AddNewRecord	2
4	Test Add New B	{null}	added by AddNewRecord	3

Data saved by calling CreateNewRecord false

Note that new EmployeeID values appear for the new records. These values are generated by the database engine on the web server, because EmployeeID is an auto-number field.

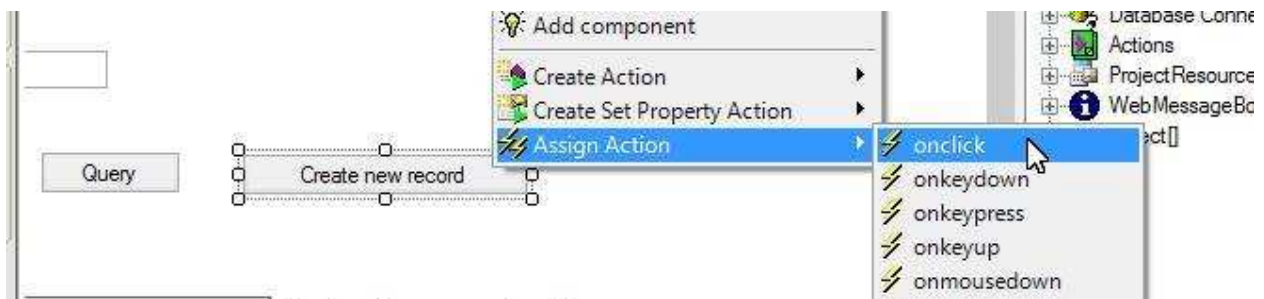
CreateNewRecord

A CreateNewRecord action creates a new record in the database and brings the new record to the web page.

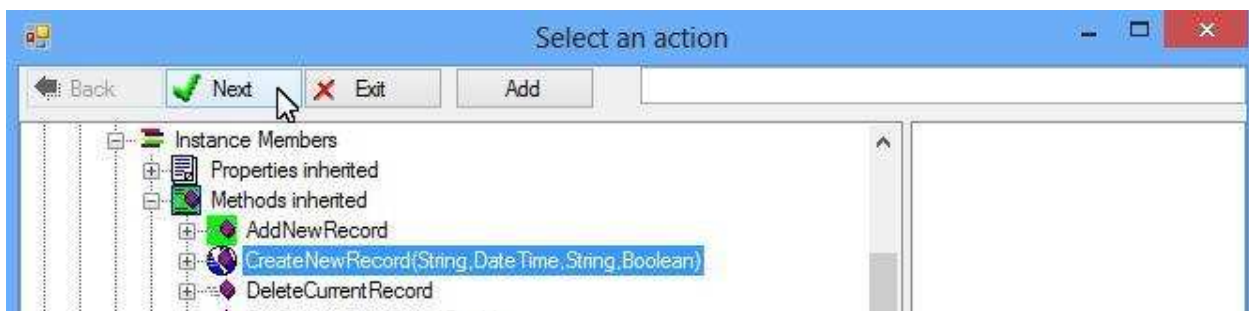
- The new record is added to both the database and the web page.
- If the record contains an identity field then the value for the identity field is available in the record on the web page. The value of the identity field is also available via a property of EasyDataSet named LastInsertID.
- A CreateNewRecord action must provide field values meeting table requirements, for example provide values for those fields not allowing null; provide unique values for fields forming unique indexes.

Execute CreateNewRecord

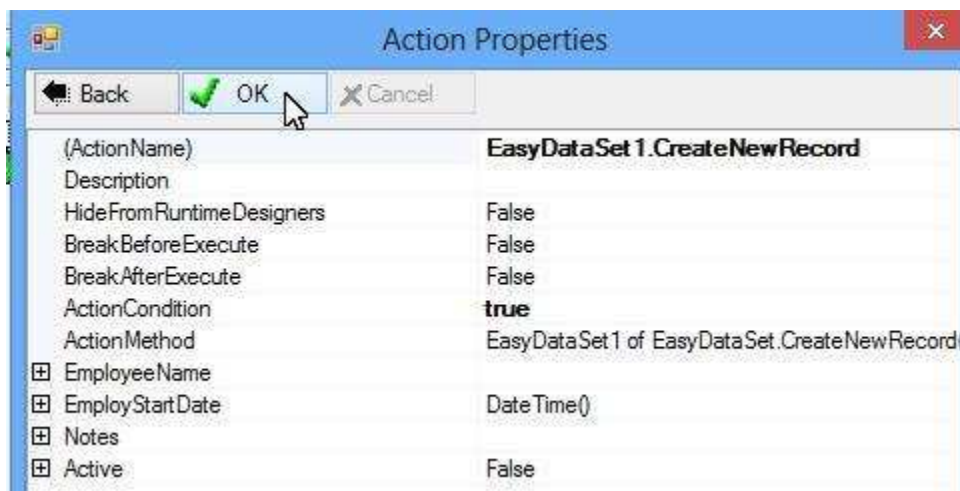
Let's assign a CreateNewRecord action to a button:



Select CreateNewRecord method:



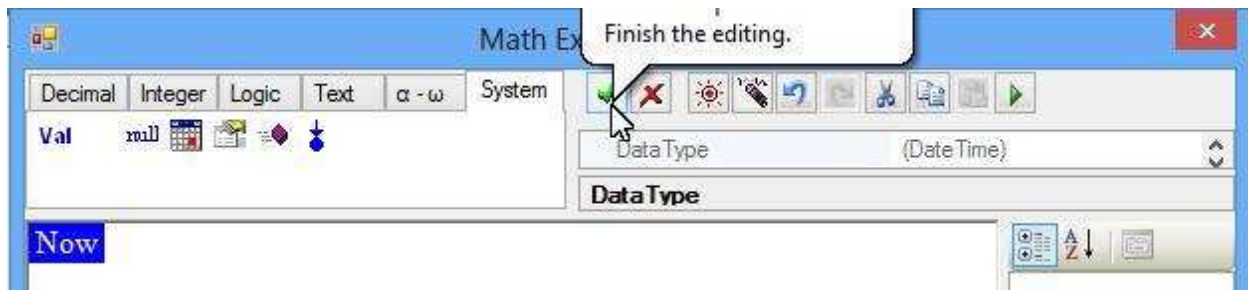
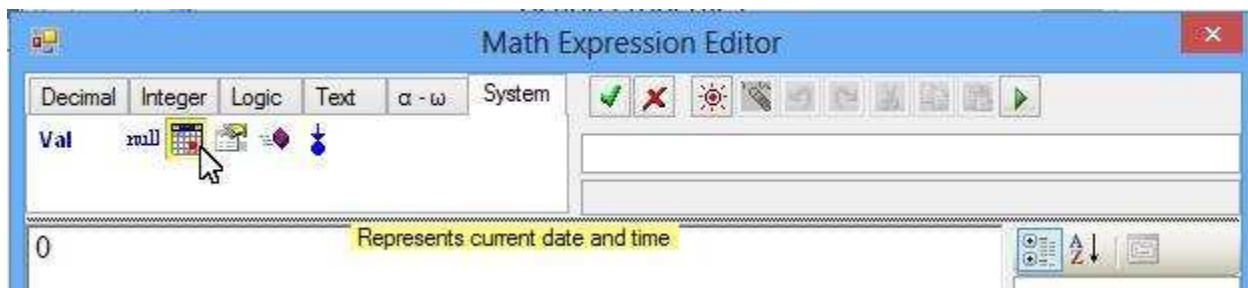
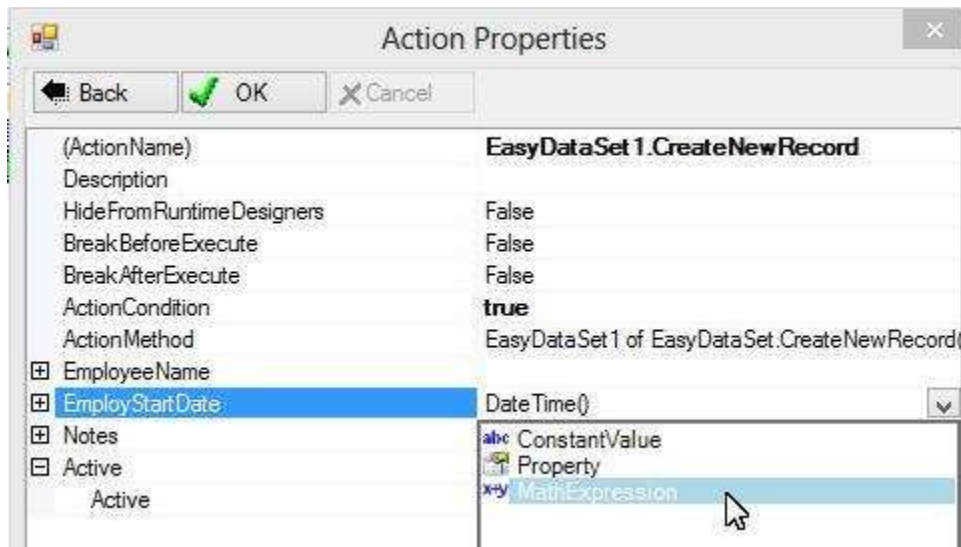
All writable fields become properties of the action, allowing specifying values for the new record:



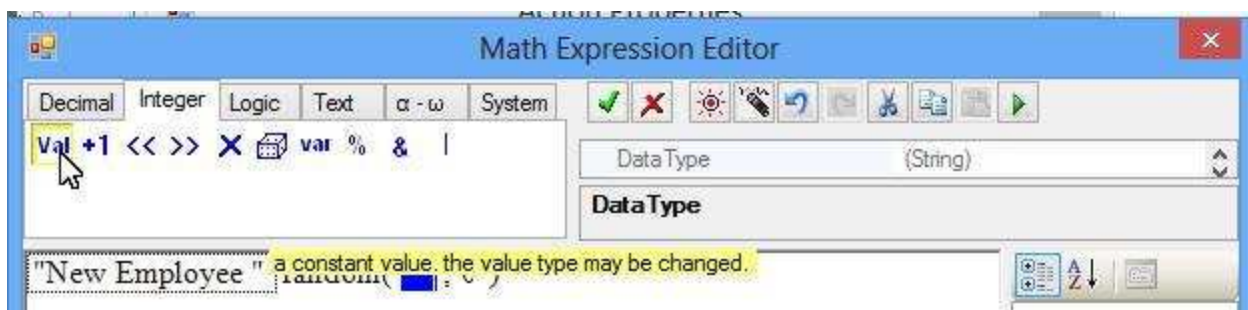
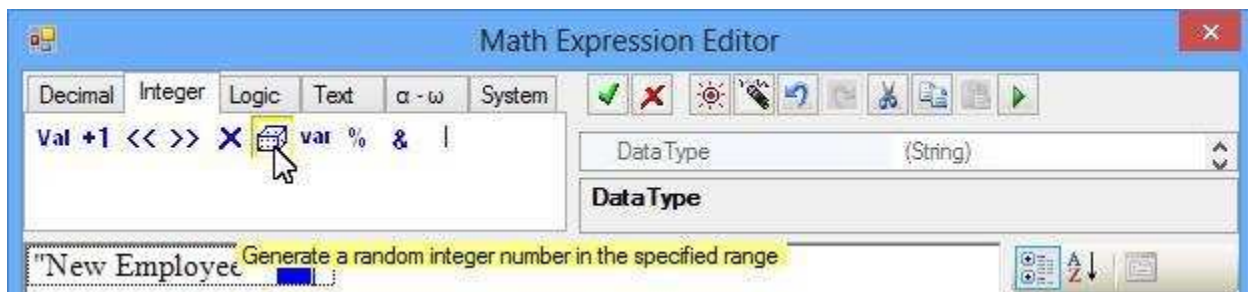
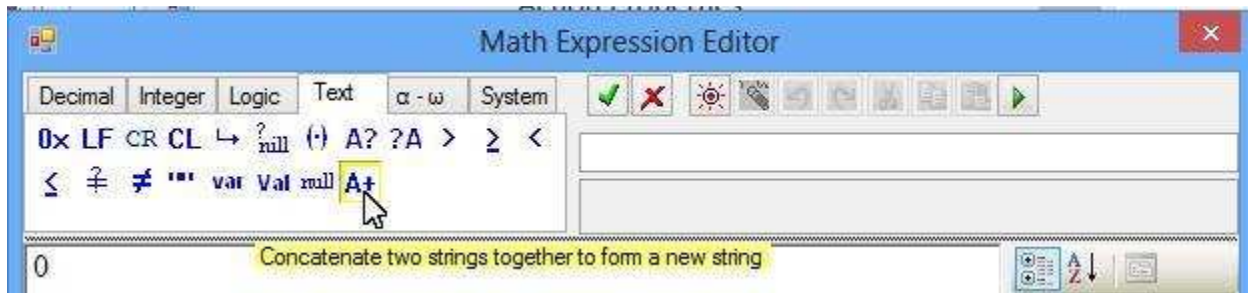
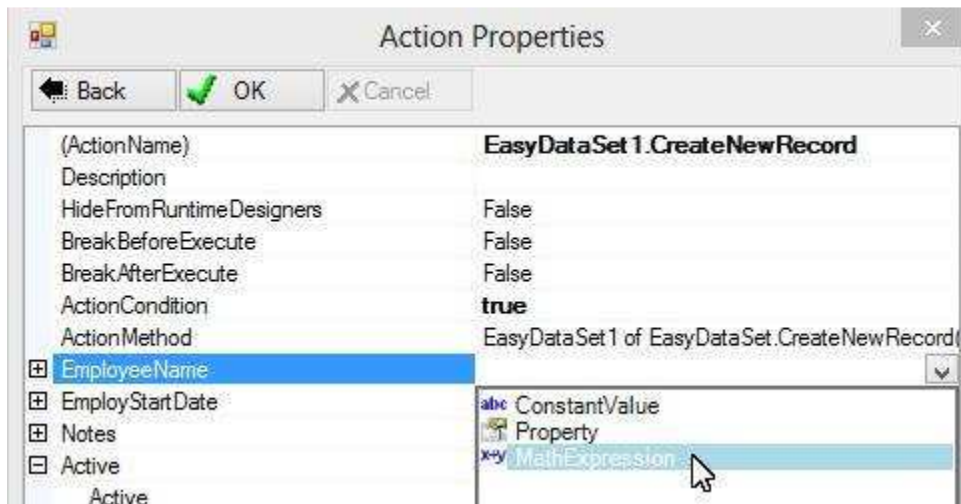
A CreateNewRecord action allows you to specify values for fields. You may leave all fields blank if the table-definition for the record allows it. If a field does not allow null value then you must specify a value for the field. If some fields form a unique index then you need to specify values for those fields to ensure uniqueness.

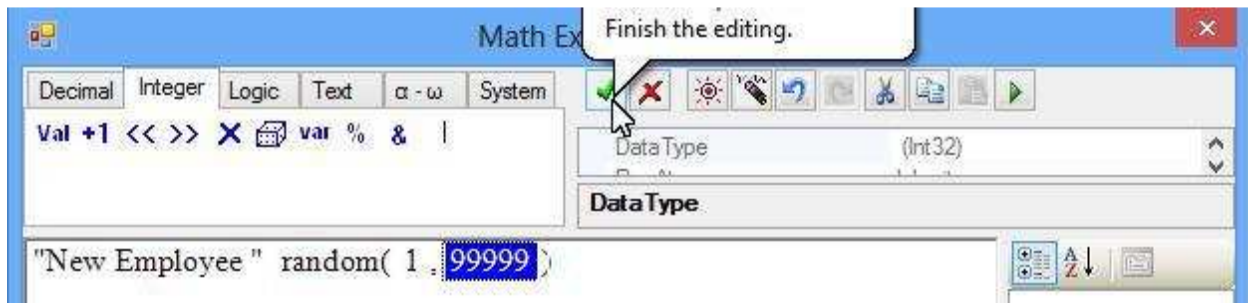
As a simple demo, we provide following field values: Active=True; Notes=Created by CreateNewRecord

Set `EmployStartDate` to current time:

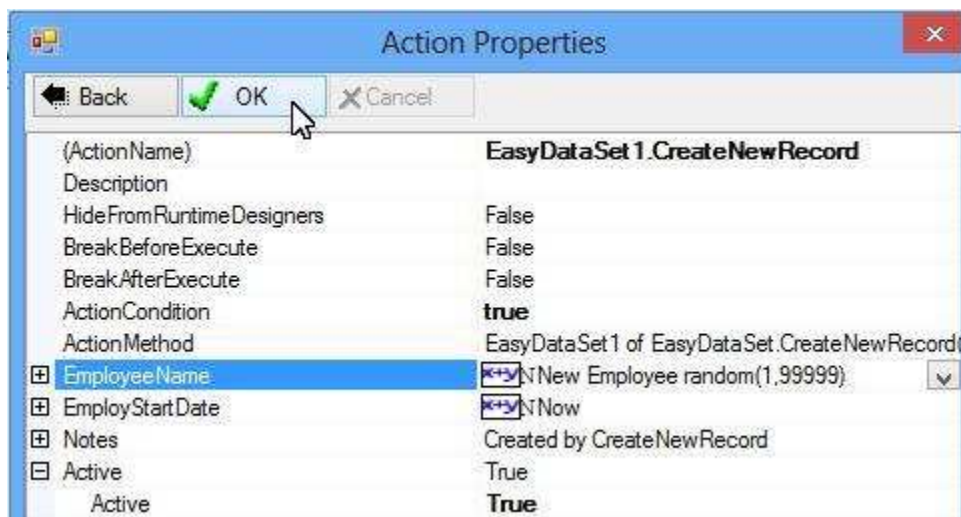


Form `EmployeeName` by an expression using random number:

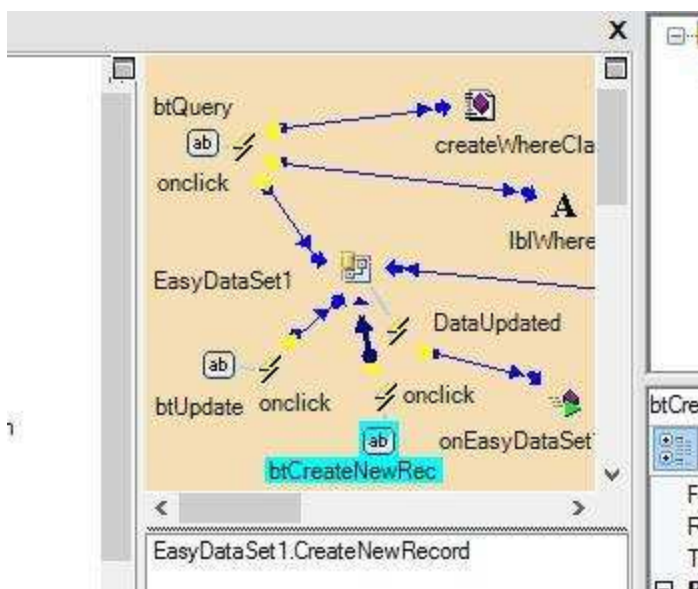




Click OK to create the action:

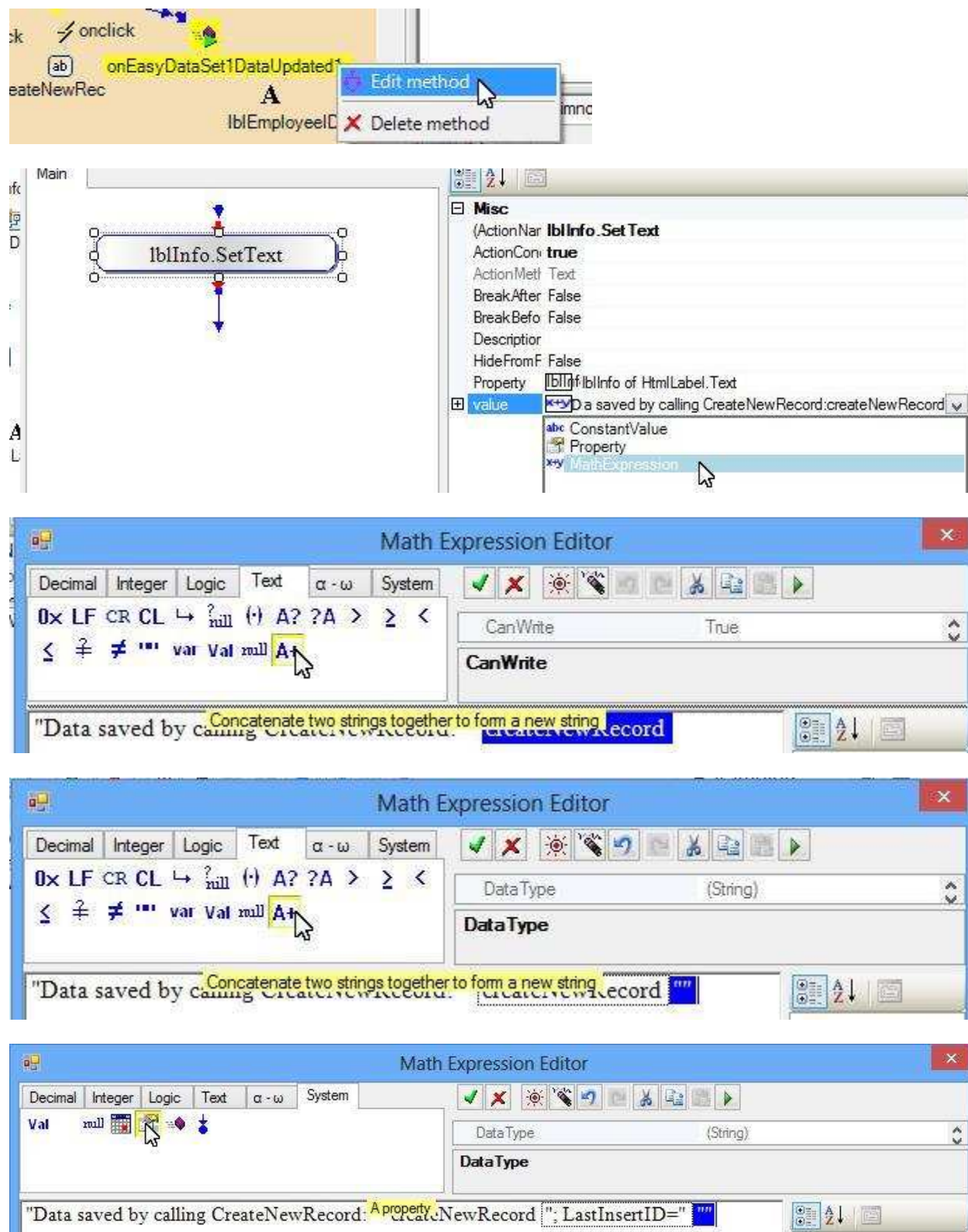


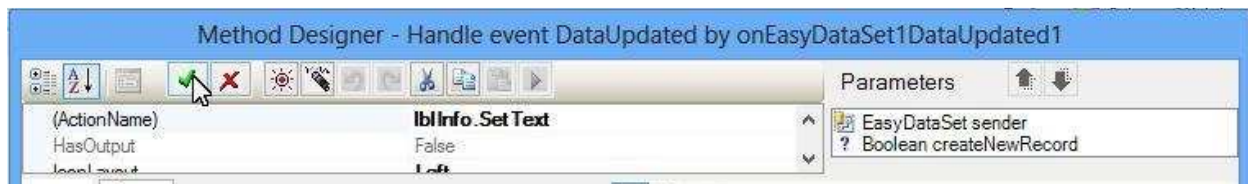
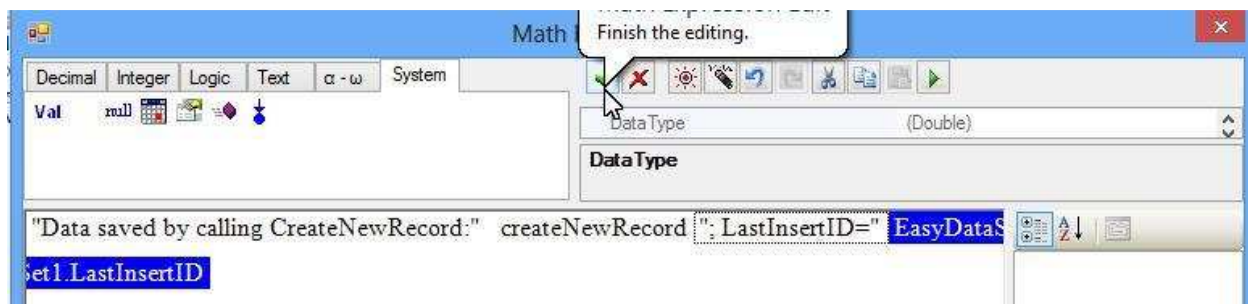
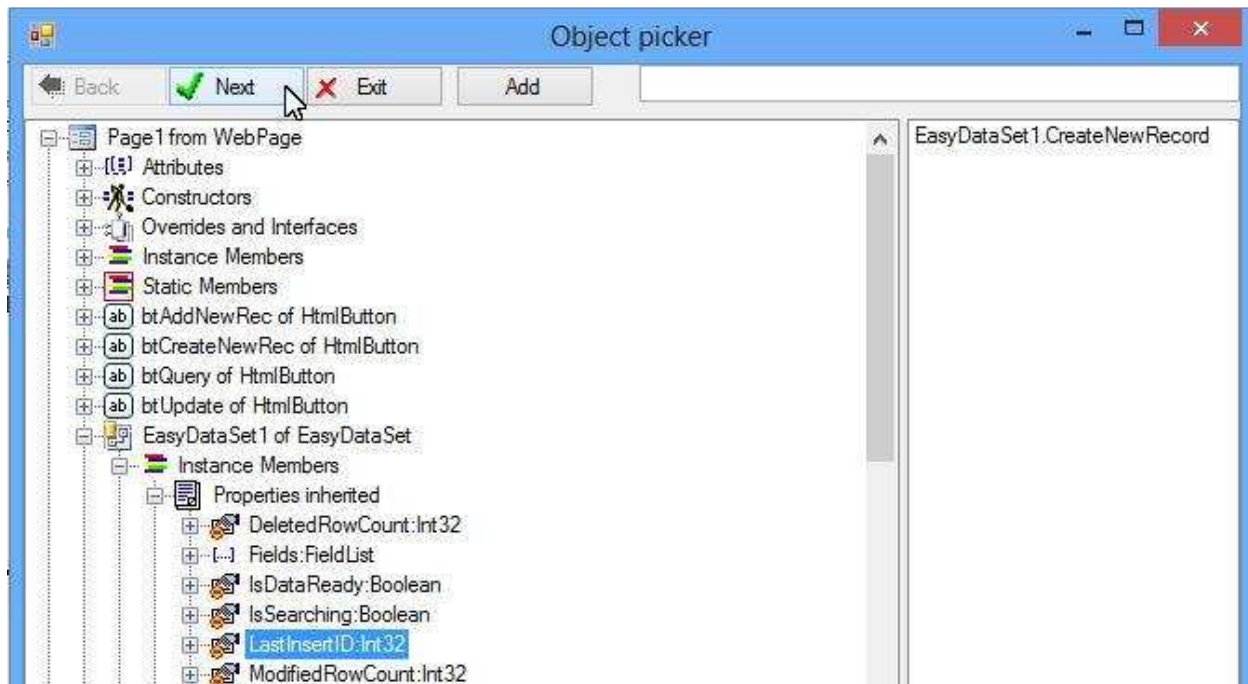
The action is created and assigned to the button:



Handle DataUpdated event

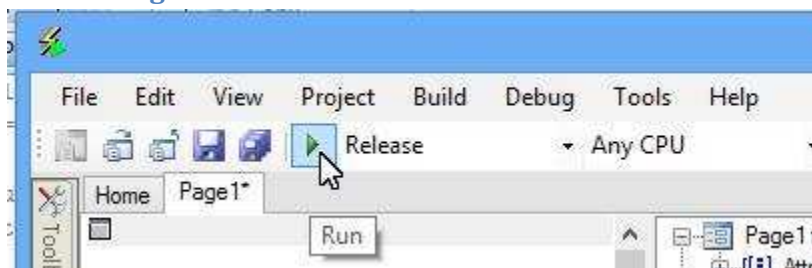
We have created an event handler method previously. For a demonstration, we modify the event handler method to add message showing LastInsertID property of the EasyDataSet:





Note that property LastInsertID is not an event parameter. It is also available outside event handler methods.

Test adding new records



The web page appears and we may do a search to get some records:

Employee name contains: Notes contains:

Employee date later than: Active:

EmployeeName LIKE CONCAT('%',@name,'%')

EmployeeID	EmployeeName	EmployeeDate	Notes	Active
1	AAA BBB	{null}	Some text	1
2	BBB CCC	{null}	Test message	1
33	Test Add New B	{null}	added by AddNewRecord	3

EmployeeID: EmployeeName:

Click "Create new record" to create a new record:

Employee name contains: Notes contains:

Employee date later than: Active:

EmployeeName LIKE CONCAT('%',@name,'%')

EmployeeID	EmployeeName	EmployeeDate	Notes	Active
1	AAA BBB	{null}	Some text	1
2	BBB CCC	{null}	Test message	1
33	Test Add New B	{null}	added by AddNewRecord	3
34	New Employee 26972	2012-12-22 10:13:25	Created by CreateNewRecord	0

EmployeeID: EmployeeName:

Data saved by calling CreateNewRecord:true;
LastInsertID=34

The new record becomes the current record.

Note that the event parameter, createNewRecord, is True; LastInsertID is 34. The data-bound controls, a table and a label, show EmployeeID as 34. This new record is ready to serve as a primary record for creating new secondary records using EmployeeID as foreign key.

Click "Create new record" again, another new record is created:

Employee name contains: Notes contains:

Employee date later than: Active:

EmployeeName LIKE CONCAT('%:@name,%')

EmployeeID	EmployeeName	EmployeeDate	Notes	Active
1	AAA BBB	{null}	Some text	1
2	BBB CCC	{null}	Test message	1
33	Test Add New B	{null}	added by AddNewRecord	3
34	New Employee 26972	2012-12-22 10:13:25	Created by CreateNewRecord	0
35	New Employee 84866	2012-12-22 10:19:03	Created by CreateNewRecord	0

EmployeeID: EmployeeName:

Data saved by calling CreateNewRecord:true;
LastInsertID=35

Data Streaming

Your web page may provide large amount of data to your visitors. For example, if you allow your visitors to search your databases, the searches may return large amount of records. Transferring large amount of data from a web server to a web page over the internet may take a long time, and keep your visitors waiting.

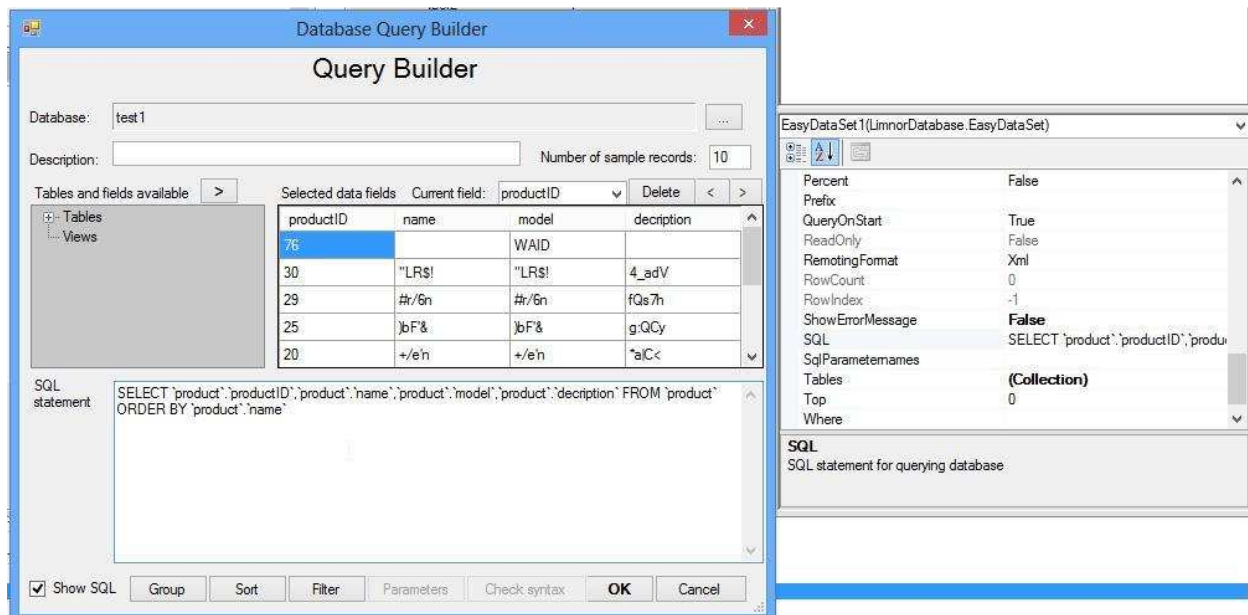
Limnor Studio gives you an option to transfer data form your database batch by batch. Thus, no matter how large amount the data are, your web page remains responsive. While the visitor is playing with the web page, new data keep coming from the web server.

Data batching is enabled through the DataBatching property of an EasyDataSet.

Note that at this time this feature is available for PHP web application only. If you want this feature in ASP.NET web applications then please send your vote to support@limnor.com. Thanks!

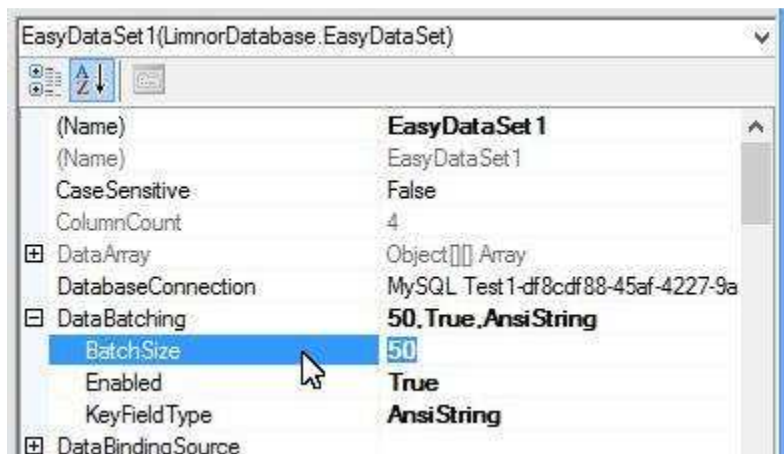
Use sorting

We must have an ORDER BY in the EasyDataSet to be able to enable data batching:



Specify batch size

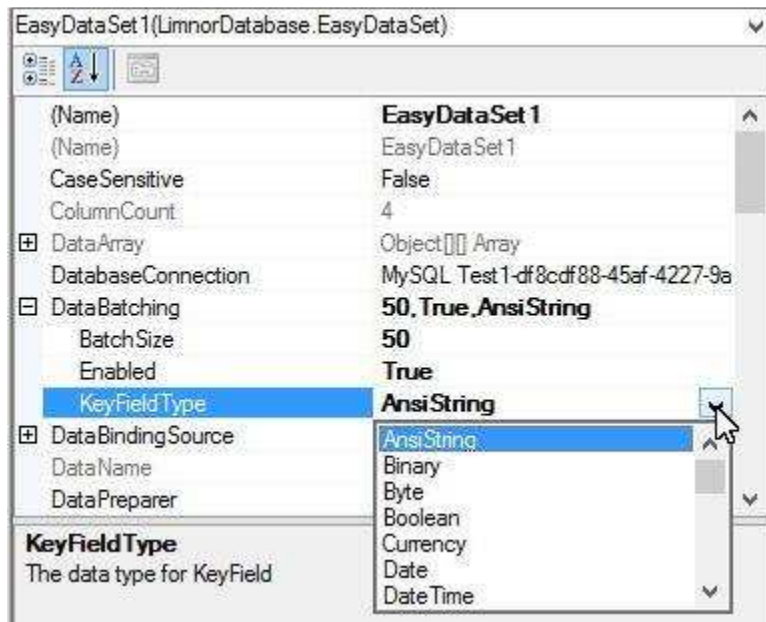
BatchSize property indicates how many records will be transferred from the web server in each batch.



The larger the batch size is the longer it takes to transfer one batch of data. But you also want to give the visitor enough data to look at. You need to make a trade-off in choosing the BatchSize.

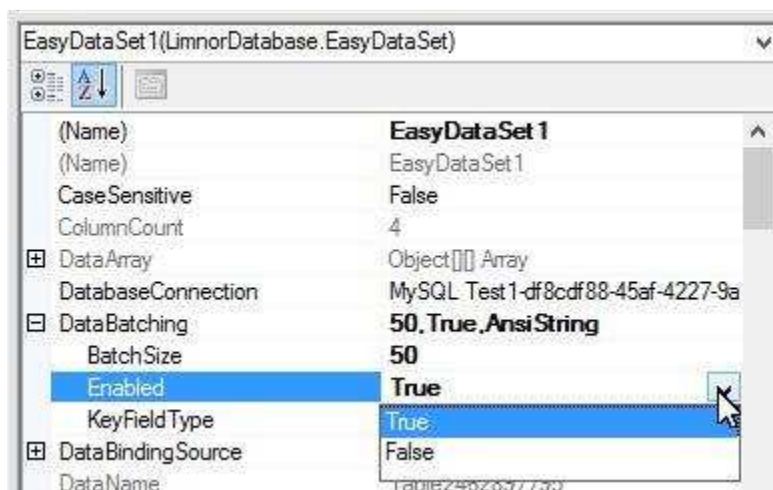
Specify key type

KeyFieldType indicates the data type for the first field in ORDER BY clause.



Enable data batching

Set Enabled property to True to enable data batching.



Batching performance

When you choose ORDER BY clause, one requirement is that there should not be too many duplicated values for the first field in the ORDER BY clause. For example, it is not a good idea to use a Boolean field as the first field of ORDER BY clause. A Boolean field only has two possible values: True and False. If your data have 100,000 records, there could be 50,000 or more records have duplicated values for the first field of ORDER BY.

If you have to use a field of highly duplicated values in ORDER BY then one way to get around the problem is to combine more than one field to form the first field of ORDER BY.

For example, suppose you have a field named Active and it is a Boolean field or some other type but the data are quite duplicated in your table, and you have to use it as its first ORDER BY field. Suppose your data include a ProductID field which is an auto-number. Then you may combine the two fields to form the first field of ORDER BY as below for MySQL database:

```
SELECT * from product ORDER BY CONCAT(CAST(CAST(Active AS SIGNED) AS CHAR), CAST(ProductID AS CHAR));
```

Fetch Data of One-to-Many Relation

One-to-many relationship is the most important concept of relational databases. In such a relationship, one record in a master table may be linked to many records in another table or the same table. For example, Order and OrderItem tables form a one-to-many relationship:

Master table: Order (OrderID, ...)

Detail table: OrderItem (OrderItemID, OrderID, ...)

In Limnor Studio User Forum web site, thread and replies form a one-to-many relationship:

Master table: thread (ThreadID, ...)

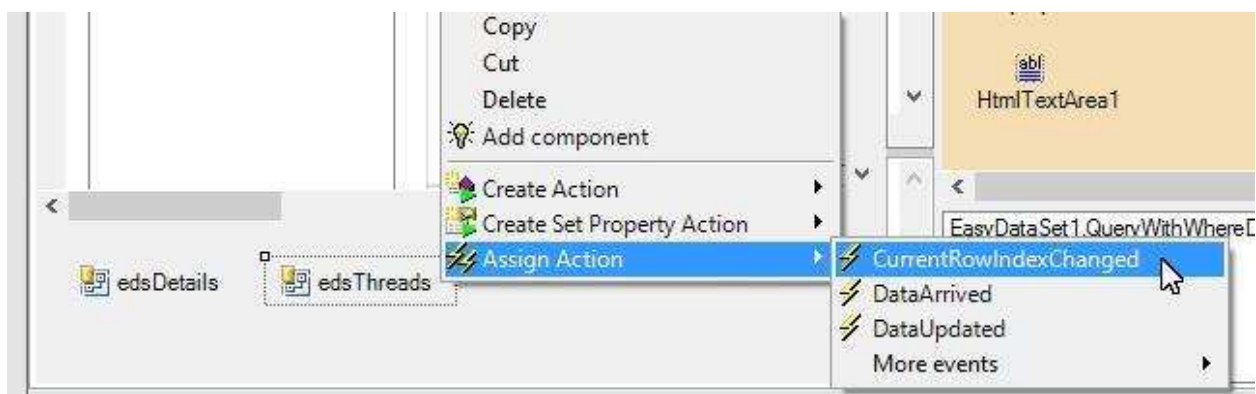
Detail table: threadmessage (ThreadMessageID, ThreadID, ...)

In a web page, when a visitor selects a master record, you may want to automatically display details records linked to the selected master records. In this chapter, we show two ways of doing it: manually and automatically.

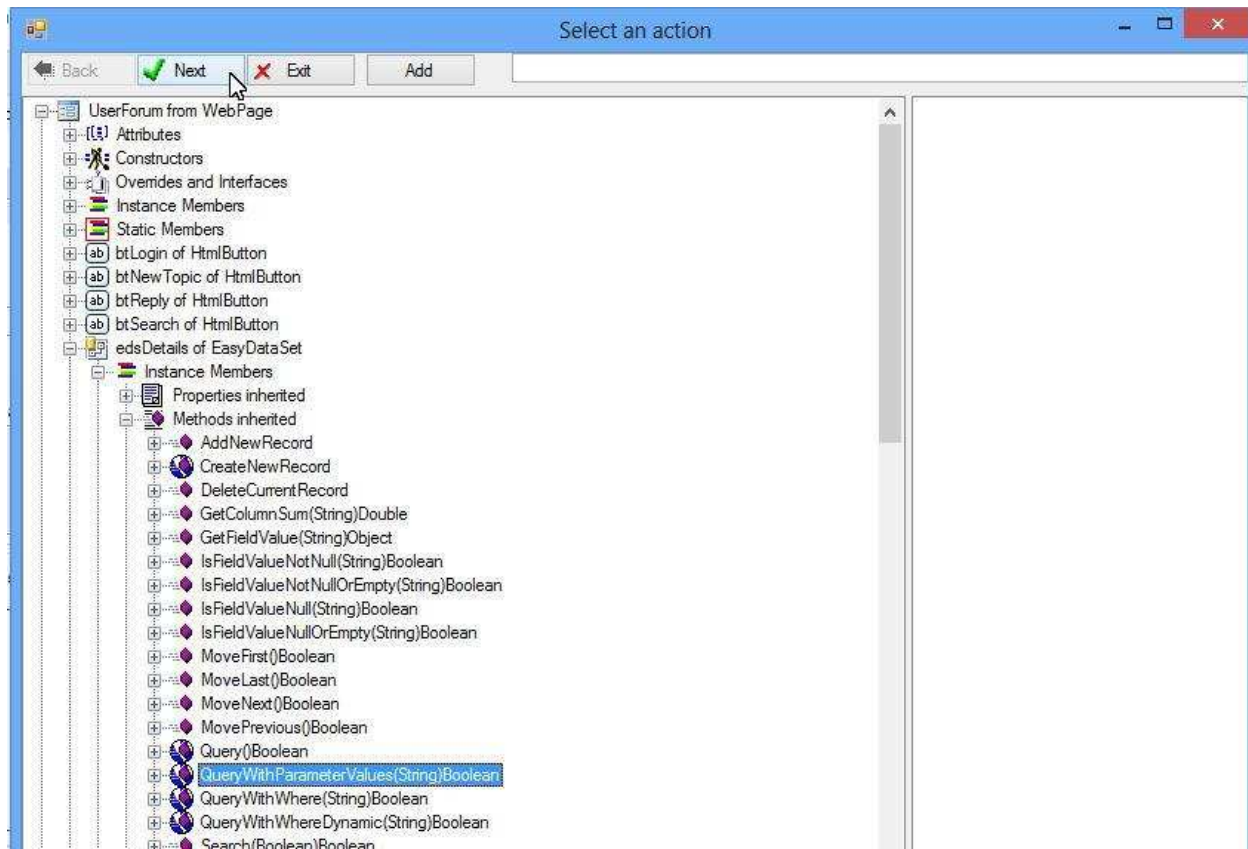
Manually fetch detail records

Manually fetch detail records gives you maximum flexibility, and it is not hard to do.

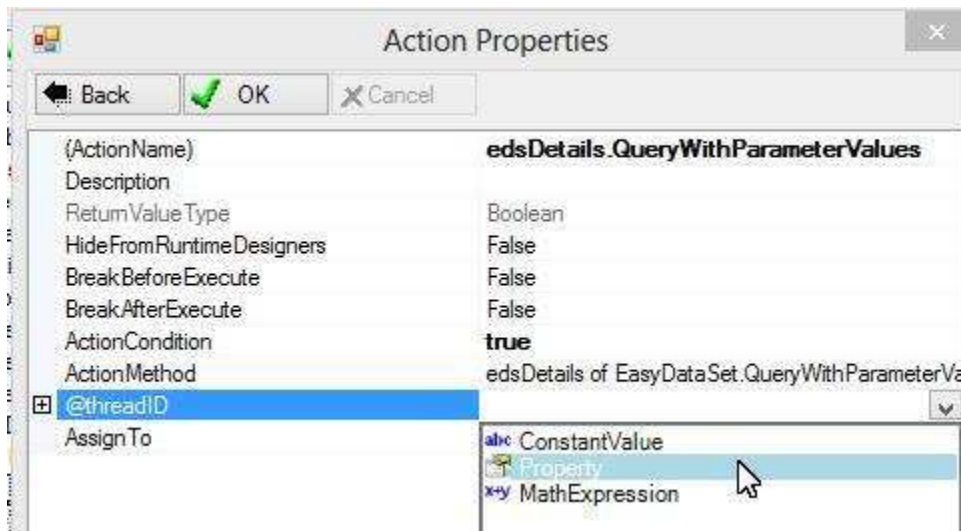
Use two EasyDataSet instances, one for the master records and one for the detail records. Handle event `CurrentRowIndexChanged` of the master EasyDataSet to execute a query action on the detail EasyDataSet. In this sample, edsThreads is the EasyDataSet for master records:

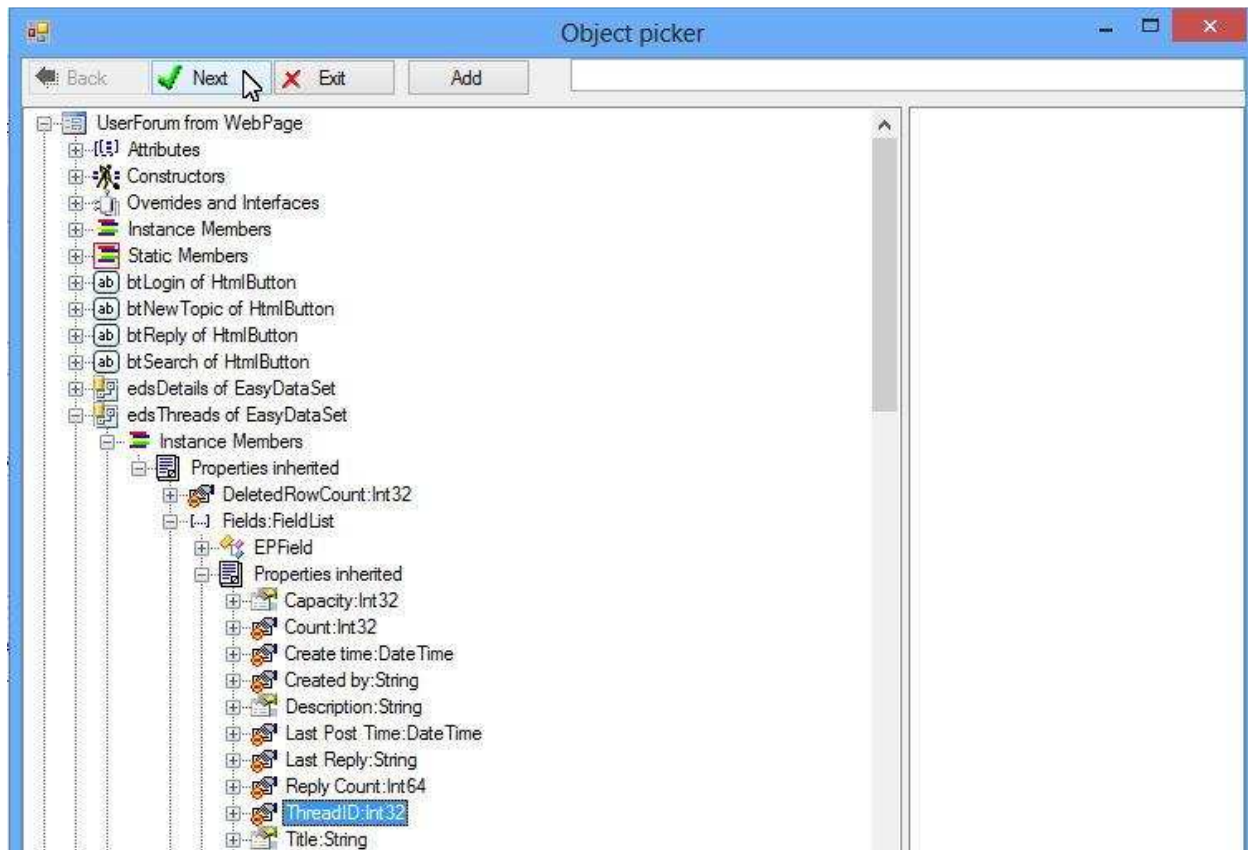


We may select a query method of the EasyDataSet for detail records, `QueryWithParameterValues`, `QueryWithWhere`, and `QueryWithWhereDynamic`, depending how we want to pass the key values of the master record to the action. In our sample, we use a parameter, `@threadID`, in the detail query, so, we choose `QueryWithParameterValues`:

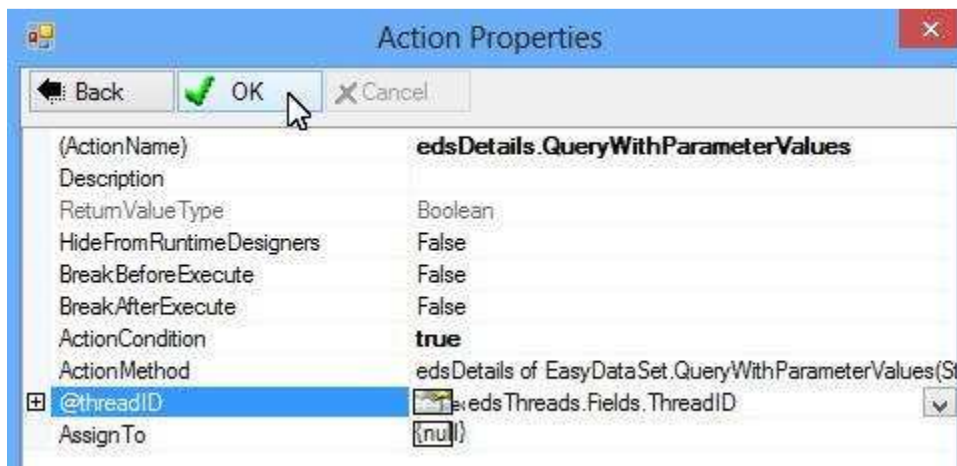


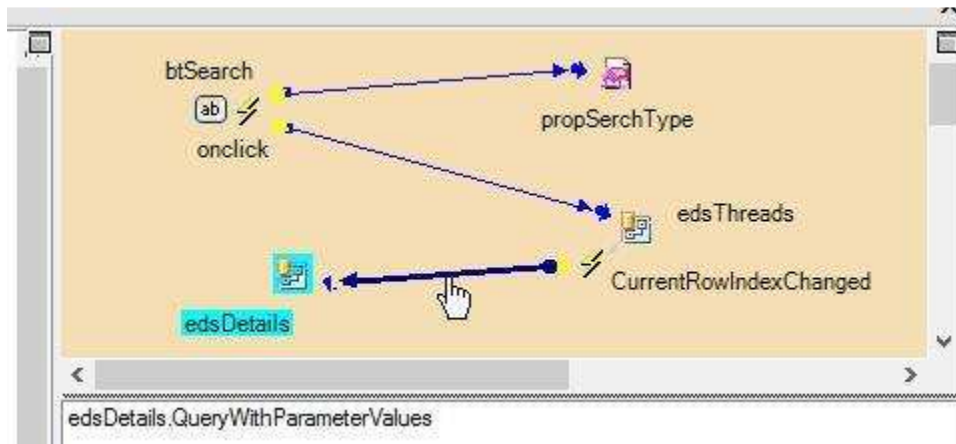
Pass the key value from the master record to the action:





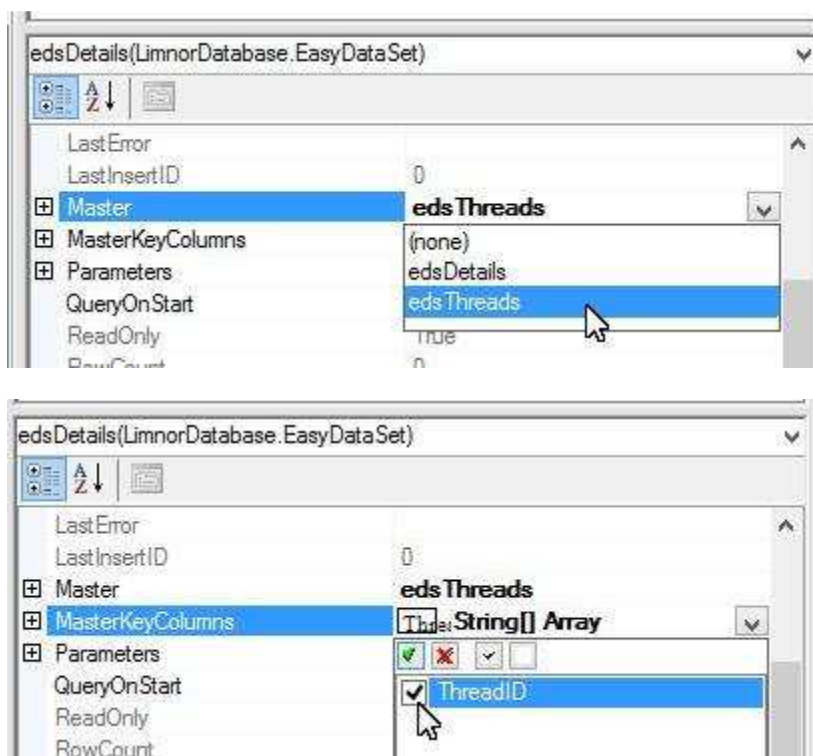
Click OK to create the action and assign it to the event:





Automatically fetch detail records

By setting properties, Master and MasterKeyColumns, of the details EasyDataSet, we do not need to create a query action and we do not need to handle event CurrentRowIndexChanged. It will automatically fetch detail records whenever a master record is selected.



You can see that the above programming is much simpler than manually creating action and handling CurrentRowIndexChanged event. But it has a limitation: it does not allow parameters in the details query.

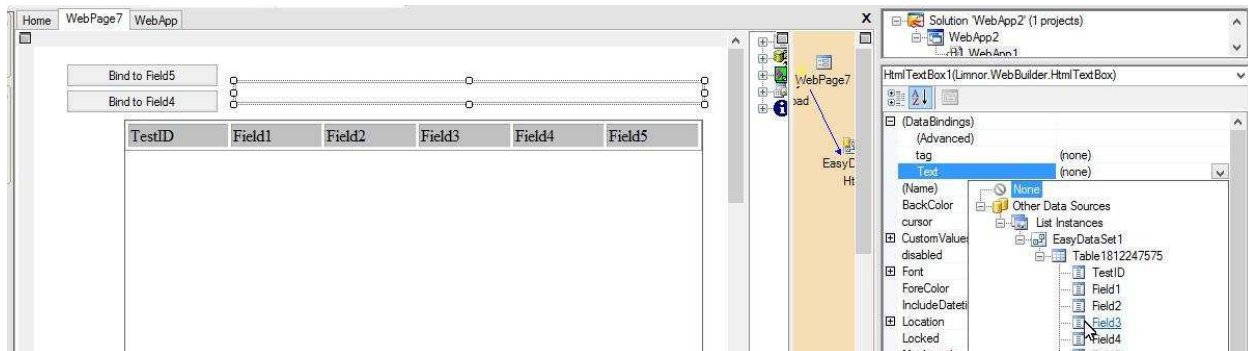
It has one good feature currently not available via manual detail fetching: record caching. Suppose the visitor selects master record 1, and 100 details records are fetched; the visitor selects master record 2

and 50 details records are fetched. In manual detail fetching, the 100 details records are discarded, if the visitor selects back master record 1, the web page will go to web server again to query the database and download that 100 details records again. For automatic detail fetching, that 100 details records are not discarded when another master record is selected, when the user selects back master record 1, the 100 details records are immediately displayed because the web page does not need to go to the web server to query the database and download the records.

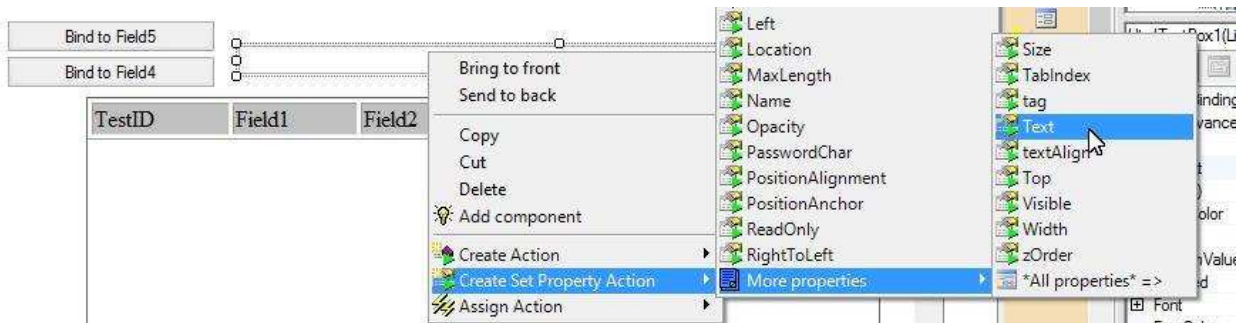
Change Data-binding

In chapter “Bind-data to single field”, we see how data-binding is created at design time. We may change data-binding at runtime by executing “set property” actions. Let’s use an example to show the programming process.

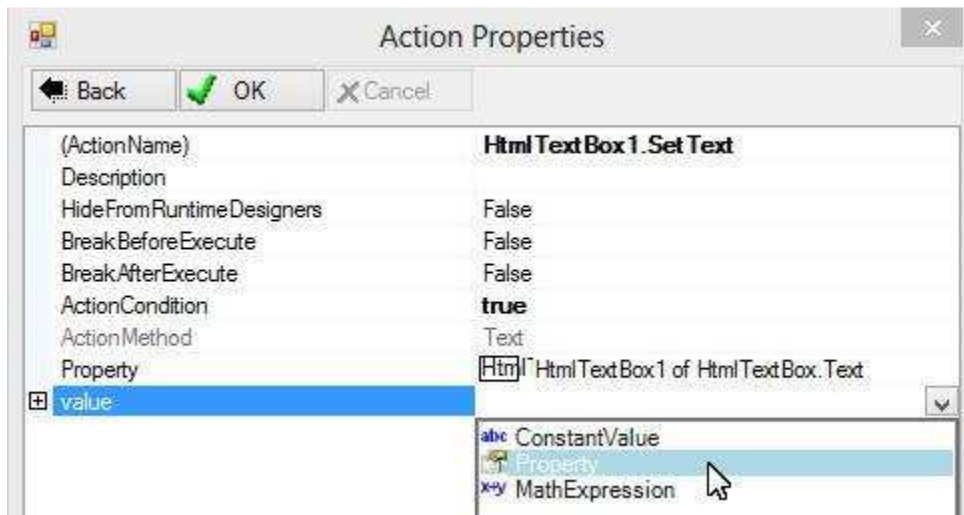
Create a web page to show some data. Add a text box and bind it to “Field3”:



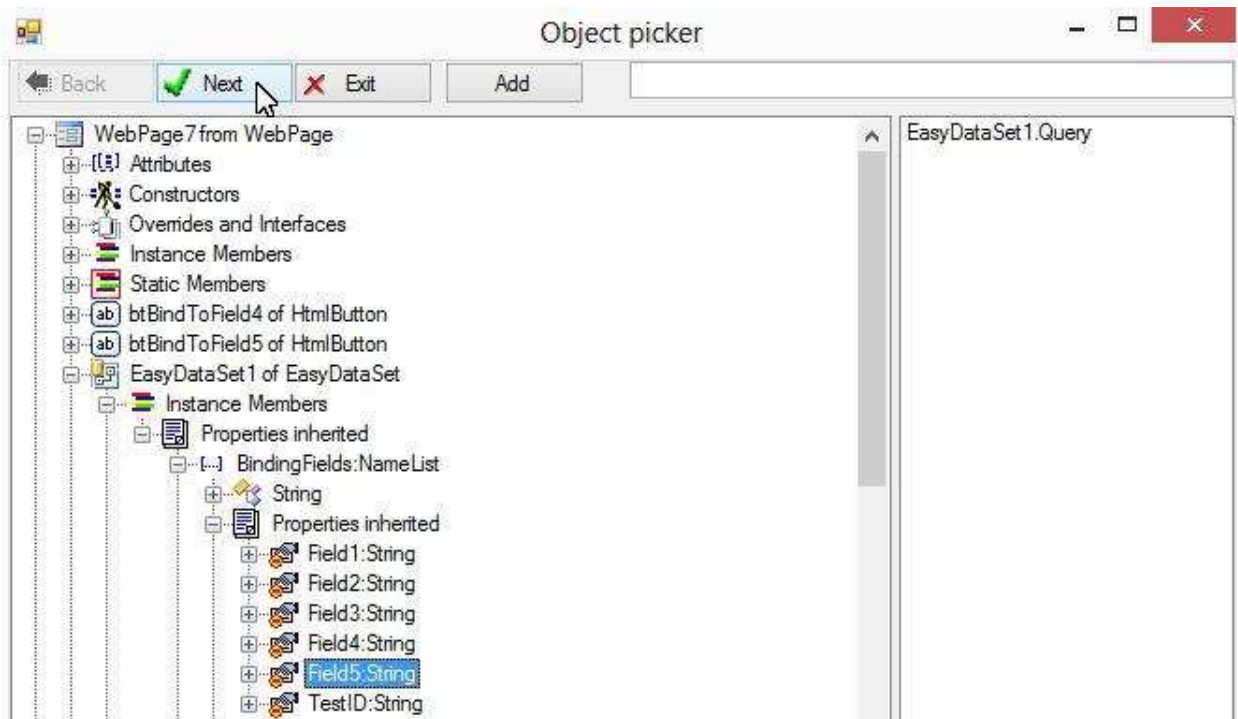
Create a “Set Property” action to bind the Text property of the text box to Field5:



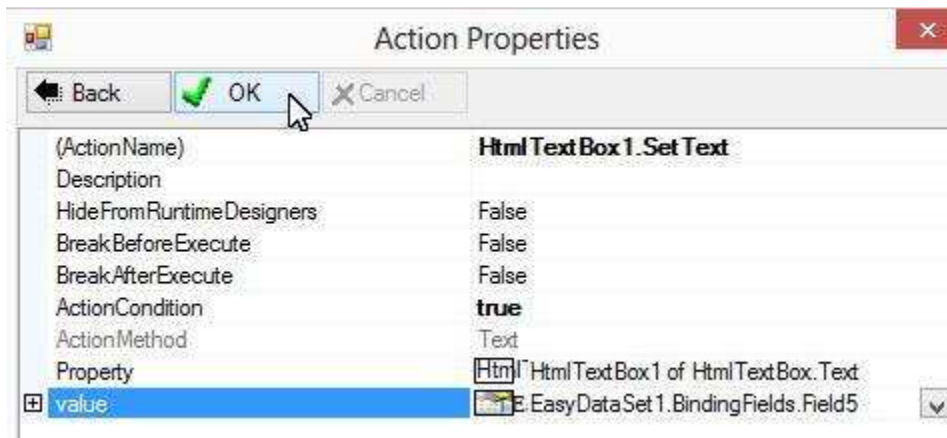
Select “Property”:



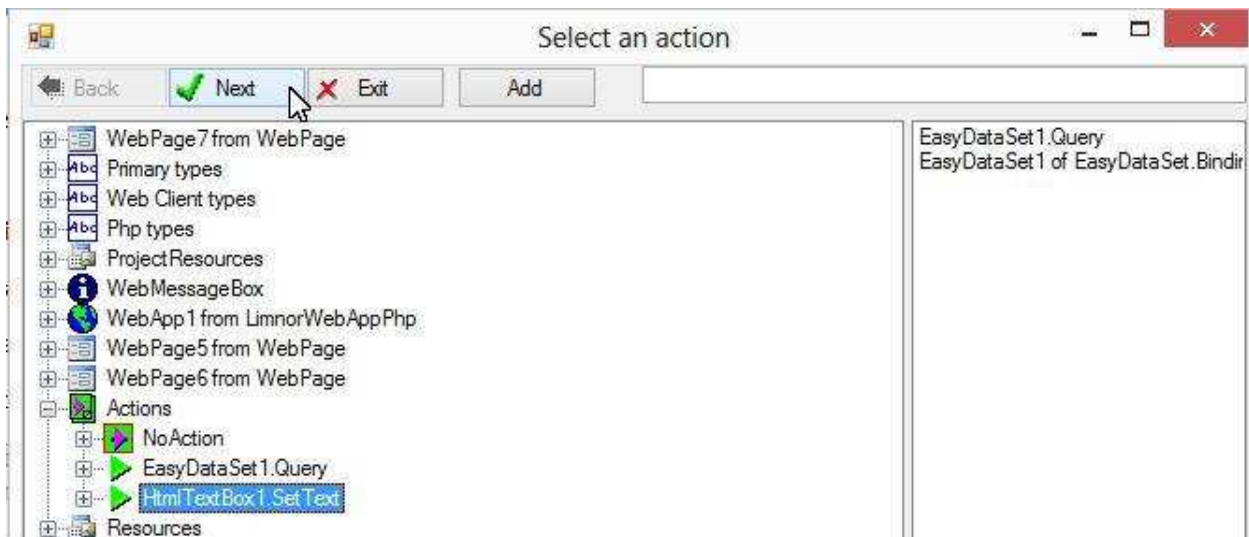
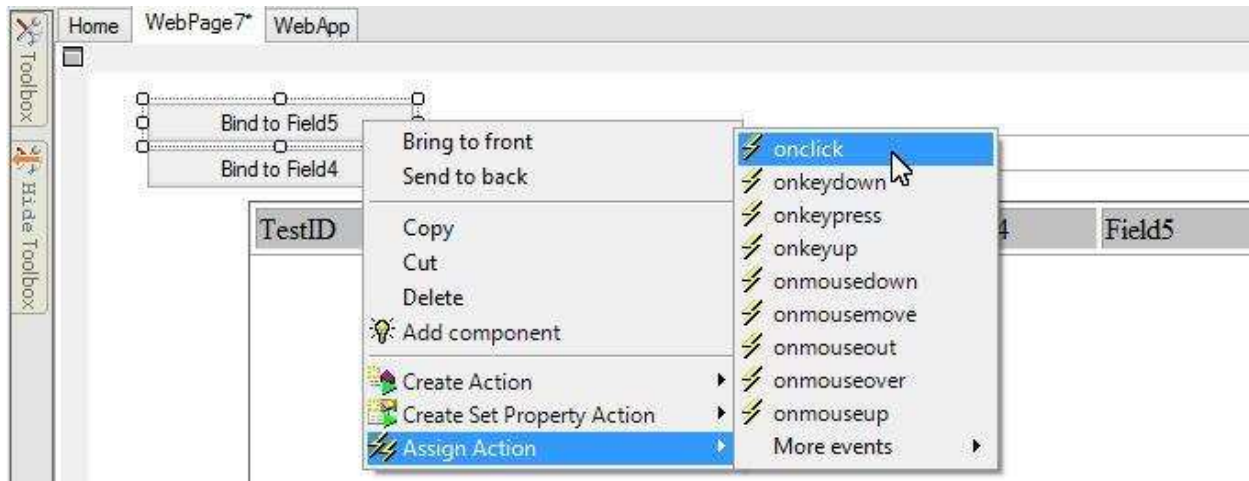
Select Field5 under BindingFields property of the EasyDataSet component:



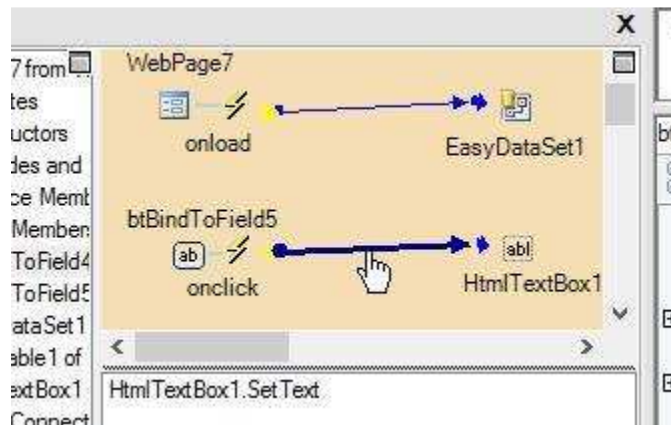
Click OK to finish creating this action:



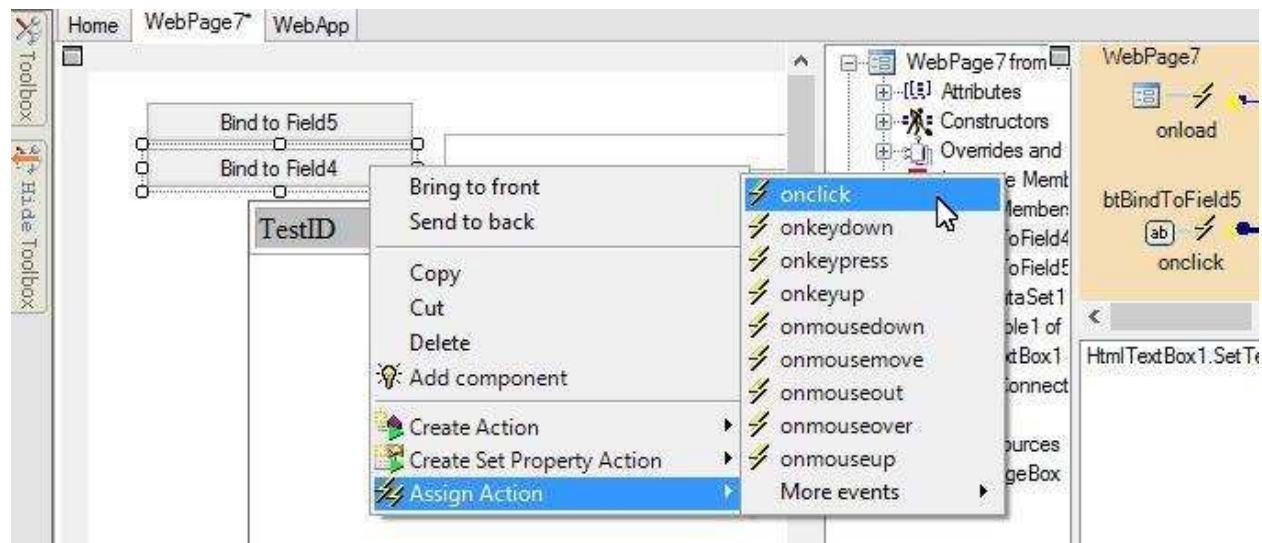
We may assign this action to a button:



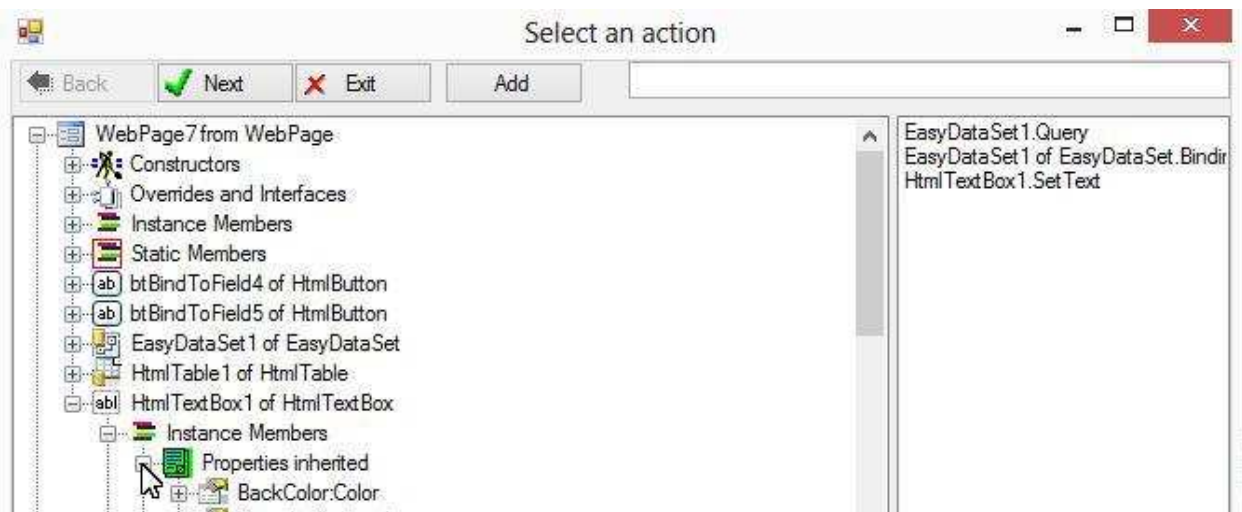
The action is assigned to the button:

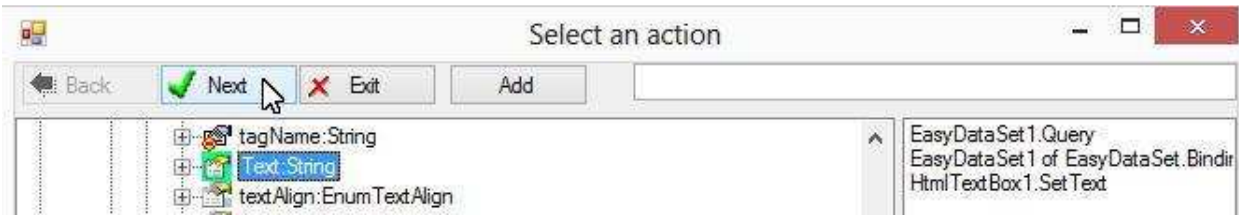


Let's use another button to change the data binding to another field:

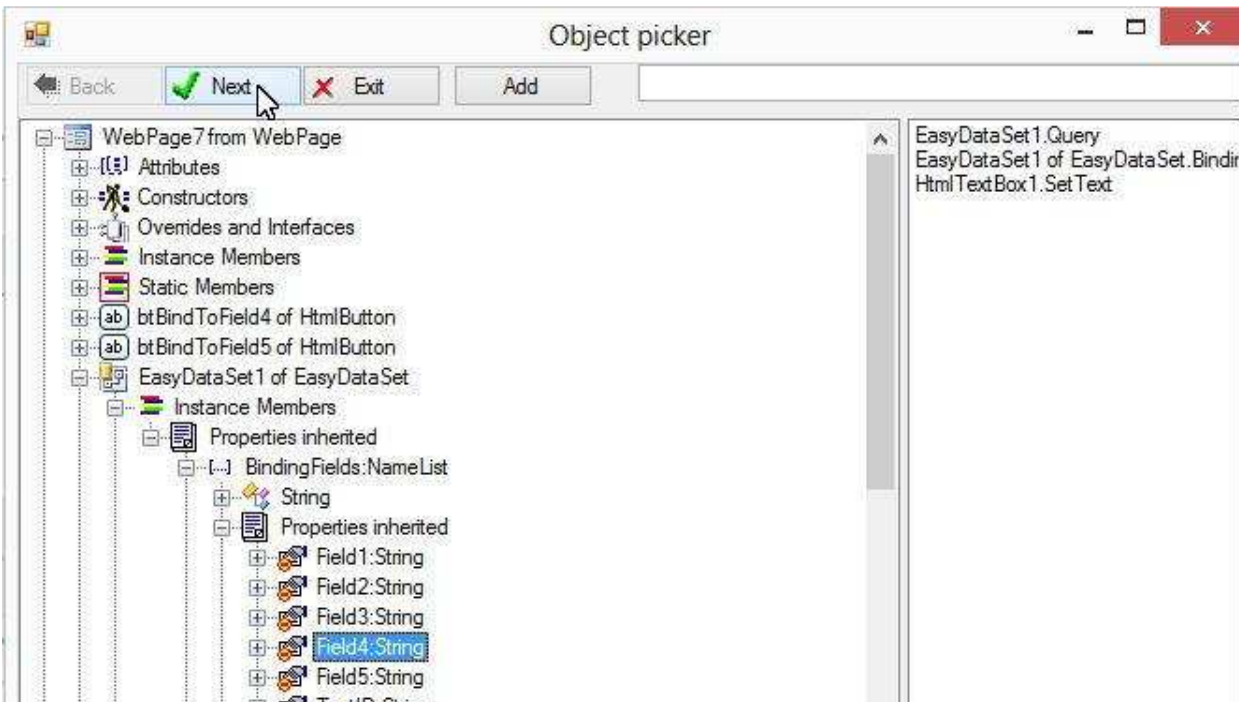
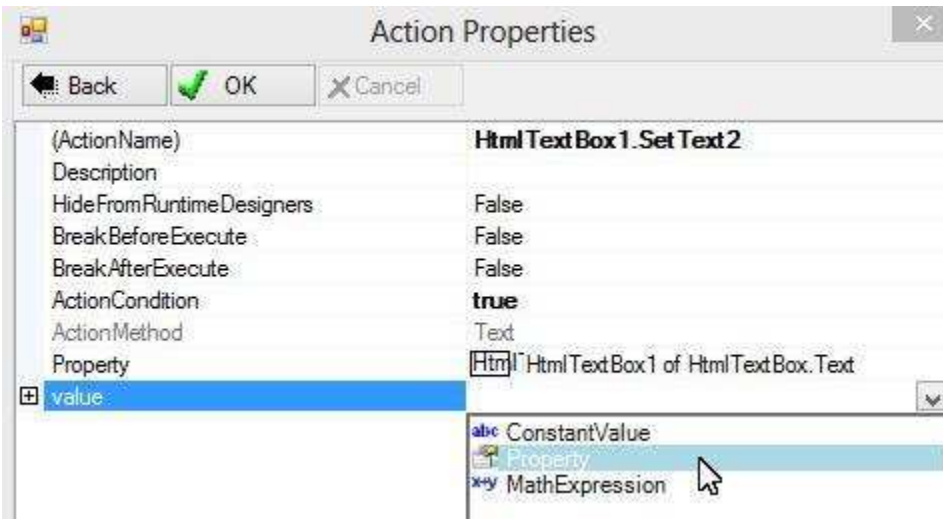


Select the Text property of the text box:

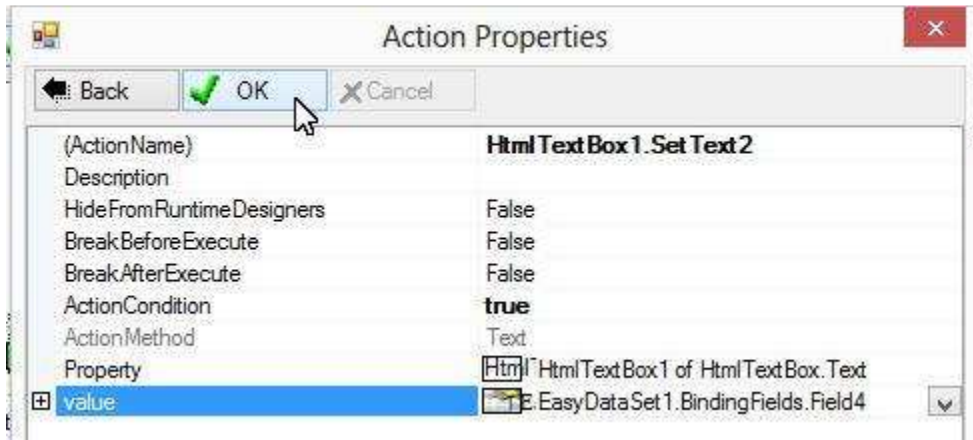




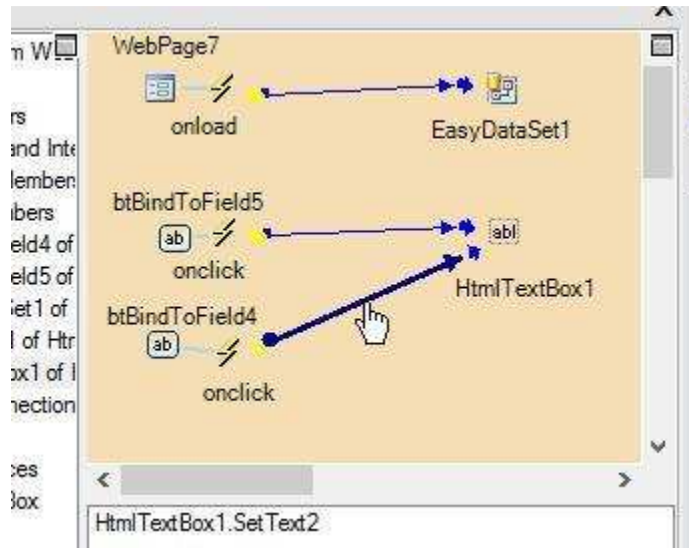
Select Field4 under BindingField of EasyDataSet1:



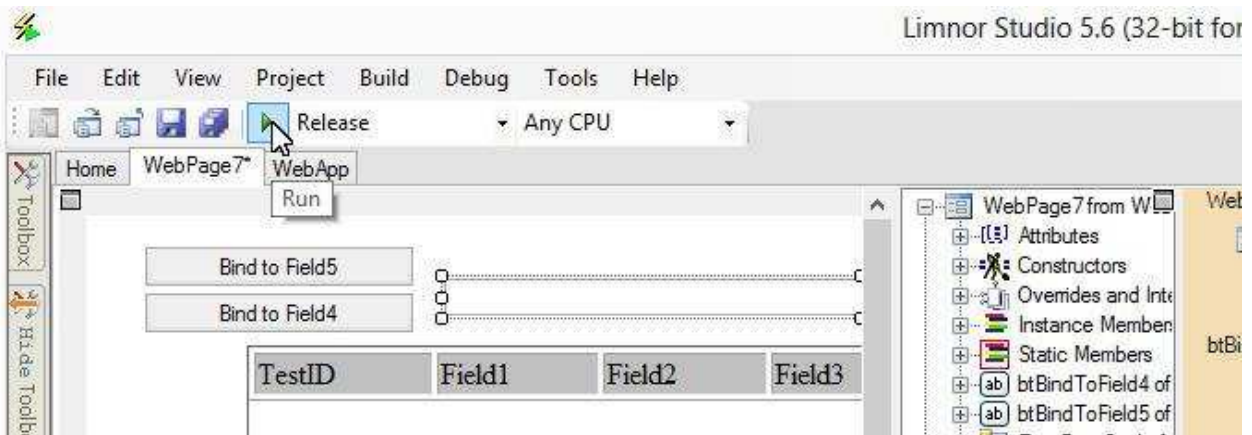
Click OK to finish creating this action:



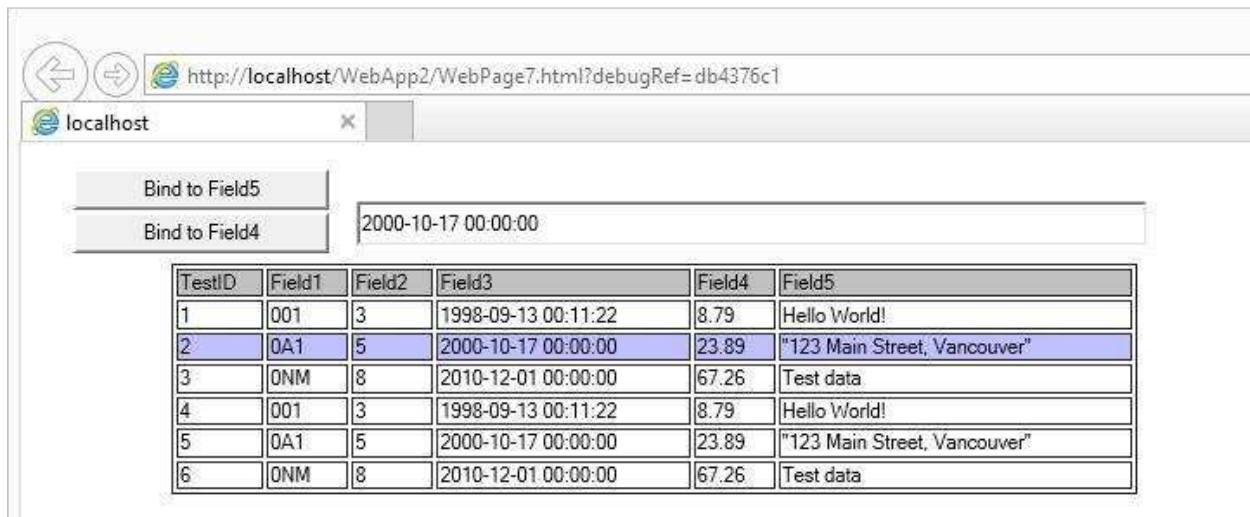
The action is created and assigned to the button:



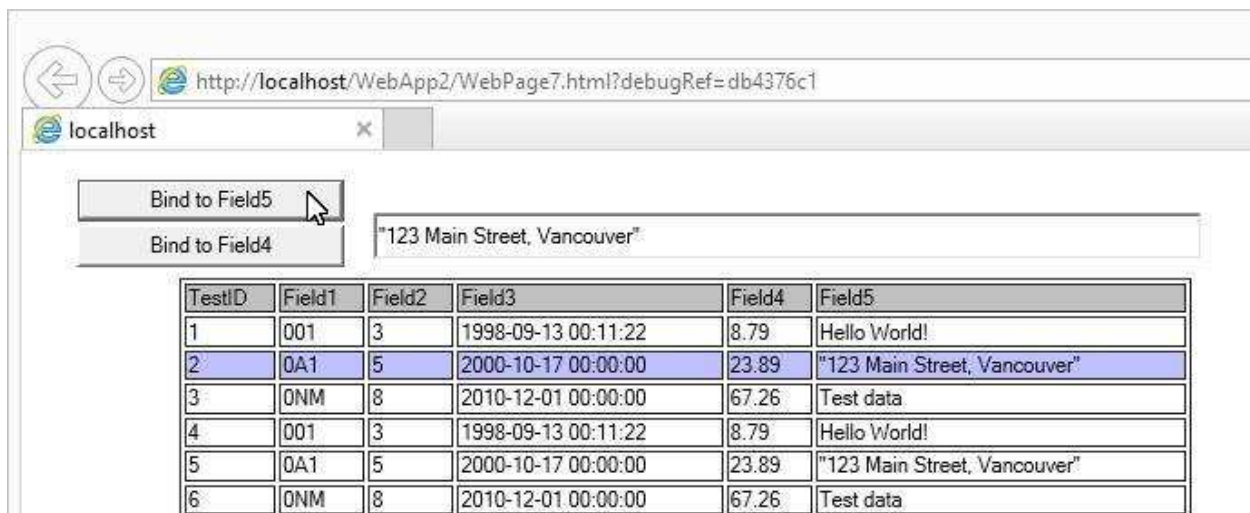
Run the web project to show the web page:



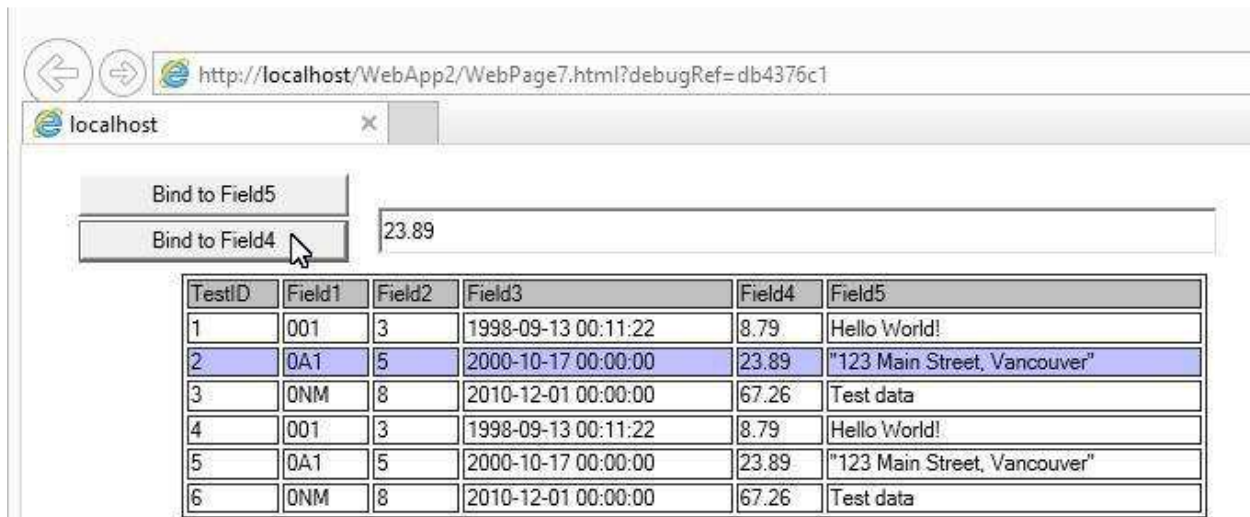
Initially the text box is bound to Field3:



Click button "Bind to Field5". The text switches the binding to Field5:



Click button "Bind to Field4". The text box switches the binding to Field4:



The screenshot shows a web browser window with the address bar displaying `http://localhost/WebApp2/WebPage7.html?debugRef=db4376c1`. Below the browser window, there are two buttons: "Bind to Field5" and "Bind to Field4". A mouse cursor is hovering over the "Bind to Field4" button. To the right of these buttons is a text input field containing the value "23.89". Below the buttons and input field is a table with 6 rows and 6 columns. The columns are labeled "TestID", "Field1", "Field2", "Field3", "Field4", and "Field5". The table contains the following data:

TestID	Field1	Field2	Field3	Field4	Field5
1	001	3	1998-09-13 00:11:22	8.79	Hello World!
2	0A1	5	2000-10-17 00:00:00	23.89	"123 Main Street, Vancouver"
3	0NM	8	2010-12-01 00:00:00	67.26	Test data
4	001	3	1998-09-13 00:11:22	8.79	Hello World!
5	0A1	5	2000-10-17 00:00:00	23.89	"123 Main Street, Vancouver"
6	0NM	8	2010-12-01 00:00:00	67.26	Test data

Feedbacks

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