Web Dialogue and Child Page

Create date: March 3, 2012

Last modified: March 3, 2012

Contents

Introduction 2
Parent Page Programming 2
Methods 2
ShowChildDialog 2
ShowChildWindow 4
ShowPopupWindow 4
CloseChildPage 4
HideChildPage 5
Properties 5
dialogResult 5
Child Page Programming 5
Methods 5
HidePage 5
ClosePage 5
ConfirmDialog 5
close 5
Properties 5
dialogResult 5
CloseDialogPrompt 6
CancelDialogPrompt 6
Events 6
onClosingWindow 6
Inter-Page Communications 6
Data-transfer via client properties 6
Publish data via client properties 6

Launch dialogue box 10
Access published data via default instance 12
Handle canceling of dialogue box 16
Test 19
Action execution via client methods 23
Publish functions via client methods 23
Execute client method from other web pages 26
Test 28
Trigger events across web pages 30
Publish functions via client events 31
Trigger events via default page instance 34
Test 38
Feedback 40

Introduction

In your web applications you may open child web pages from a parent web page. A child web page can be opened as a dialogue box or as a child window, or as a pop up window. A dialogue box blocks the user access to the parent web page until the dialogue box is closed. The actions following the action, which launches a dialogue box, will also be blocked until the dialogue box is closed. Direct communications between the child windows, dialogue boxes, popup windows, and the parent web page can be made via creating properties and methods and using default web page instances.

Parent Page Programming

Methods

ShowChildDialog

This method opens a child web page as a dialogue box. The access of the current web page by the users is blocked until the dialogue box is closed. The actions following the action executing this method are also suspended until the dialogue box is closed.

This method needs following parameters.

- webpage – it indicates the child web page to be opened
- center – it is a Boolean indicating whether the child page will be displayed in the center of this page.
- top – it is an integer indicating the top location of the child page, in pixels, if “center” is false; it is ignored if “center” is true.
- left – it is an integer indicating the left location of the child page, in pixels, if “center” is false; it is ignored if “center” is true.
- width – it is an integer indicating the width of the child page, in pixels.
- height – it is an integer indicating the height of the child page, in pixels.
- resizable – it is a Boolean indicating whether the child page can be re-sized by the user.
- borderStyle – it indicates the style of the border of the dialogue box. It can be one of the following values
    - none -- No border.
    - Dotted -- The border is a series of dots.
    - Dashed -- The border is a series of short line segments.
    - Solid -- The border is a single line segment.
    - Double -- The border is two solid lines. The sum of the two lines and the space between them equals the value of 'border-width'.
    - Groove -- The border looks as though it were carved into the canvas.
    - Ridge -- The opposite of 'groove': the border looks as though it were coming out of the canvas.
    - Inset -- In the separated borders model, the border makes the entire box look as though it were embedded in the canvas. In the collapsing border model, drawn the same as 'ridge'.
    - outset -- In the separated borders model, the border makes the entire box look as though it were coming out of the canvas. In the collapsing border model, drawn the same as 'groove'.
- borderWidthStyle, -- it indicates the style of the width of the dialogue box border. It can be one of the following values
    - inherit
    - thin
    - medium
    - thick
    - useNumber -- the “borderWidth” property is used.
- borderWidth – it is an integer indicating the width of the dialogue box border, in pixels.
- borderColor – it indicates the color of the dialogue box border
- iconPath – it is a string representing the file path to an image to be used as the dialogue box icon. The image will be displayed on the left side of the dialogue box title bar.
- title – it is a string representing the title of the dialogue box. It will be displayed on the dialogue box title bar.

ShowChildWindow

This method opens a child web page as a child window. It does not block the access to this page. It does not block the execution of the other actions.

It uses the same parameters as ShowChildDialog.

ShowPopupWindow

This method shows a web page in a popup window. Note that the web browser's popup-blocker may block popup window.

This method needs following parameters.

- string pageAddress -- Specifies the URL of the page to open. If no URL is specified, a new window with about:blank is opened
- string windowName -- Specifies the target attribute or the name of the window. The following values are supported:
  - _blank - URL is loaded into a new window. This is default
  - _parent - URL is loaded into the parent frame
  - _self - URL replaces the current page
  - _top - URL replaces any framesets that may be loaded
  - name - The name of the window
- bool showStatusBar -- Whether or not to add a status bar.
- bool showToolbar -- Whether or not to display the browser toolbar.
- bool showAddressBar -- Whether or not to display the address field.
- bool showMenubar -- Whether or not to display the menu bar
- bool showScrollbar -- Whether or not to display scroll bars.
- bool showTitlebar -- Whether or not to display the title bar. Ignored unless the calling application is an HTML Application or a trusted dialog box.
- bool resizable -- Whether or not the window is resizable.
- int left -- The left position of the window. IE only
- int top -- The top position of the window. IE only
- int width -- The width of the window. Min. value is 100
- int height -- The height of the window. Min. value is 100
- bool replaceHistoryEntry -- Specifies whether the URL creates a new entry or replaces the current entry in the history list. The following values are supported:
  - true - URL replaces the current document in the history list
  - false - URL creates a new entry in the history list

CloseChildPage

It closes the specified child page which was opened via a ShowChildWindow action, removing the child page from the memory. A subsequent ShowChildWindow action will create a new child page.

Parameters of this method:

- webpage – it indicates the child web page to be closed

HideChildPage

It hides the specified child page which was opened via a ShowChildWindow action. The child page is kept in the memory. A subsequent ShowChildWindow action will make the child page visible again.

Parameters of this method:

- webpage – it indicates the child web page to be hidden

Properties

dialogResult

It is a string indicating how a dialogue box is closed. It can be 'ok' or 'cancel'. 'ok' indicates that the user wants to accept and use the data on the dialogue box. 'cancel' indicates that the user wants to discard and not use the data on the dialogue box.

Child Page Programming

Methods

HidePage

It hides the current page which was opened via a ShowChildWindow action by another page (parent page). This page is kept in the memory. A subsequent ShowChildWindow action by its parent page will make this page visible again.

ClosePage

It closes the current page which was opened via a ShowChildWindow or a ShowChildDialog action by another page (parent page), removing this page from the memory. A subsequent ShowChildWindow action by the parent page will create a new page. The dialogResult is set to 'cancel'.

ConfirmDialog

It closes the current page which was opened via a ShowChildDialog action by another page (parent page), removing this page from the memory. A subsequent ShowChildWindow action by the parent page will create a new page. The dialogResult is set to 'ok'.

close

It closes the current window without using prompt and not setting the dialogResult property.

Properties

dialogResult

It is a string indicating how a dialogue box is to be closed. It can be 'ok' or 'cancel'. 'ok' indicates that the user wants to accept and use the data on the dialogue box. 'cancel' indicates that the user wants to

discard and not use the data on the dialogue box. If this property is set to an empty string while handling event `onClosingWindow` then the dialogue box will not be closed.

CloseDialogPrompt

Gets and sets a prompt asking whether the user wants to close this window. This prompt is displayed when this page is being closed. This prompt is used only if this page is opened as a dialogue box, that is, it is opened via a ShowChildDialog action of another page, the parant page. The parent page's dialogResult property will be 'ok' if the user confirms the prompt.

CancelDialogPrompt

Gets and sets a prompt asking whether the user wants to cancel a dialogue box or close this window. This prompt is displayed when this page is being closed. This prompt is used only if this page is opened as a child page, that is, it is opened via a ShowChildWindow or ShowChildDialog action of another page, the parent page. The parent page's dialogResult property will be 'cancel' if the user confirms the prompt.

Events

onClosingWindow

It occurs when the page is going to be closed. It occurs only if the page is opened via a ShowChildWindow or ShowChildDialog action by another page. The dialogResult property indicates how the window will be closed. It can be ‘ok’ or ‘cancel’. In handling this event, if dialogResult is set to an empty string then the page will not be closed.

Inter-Page Communications

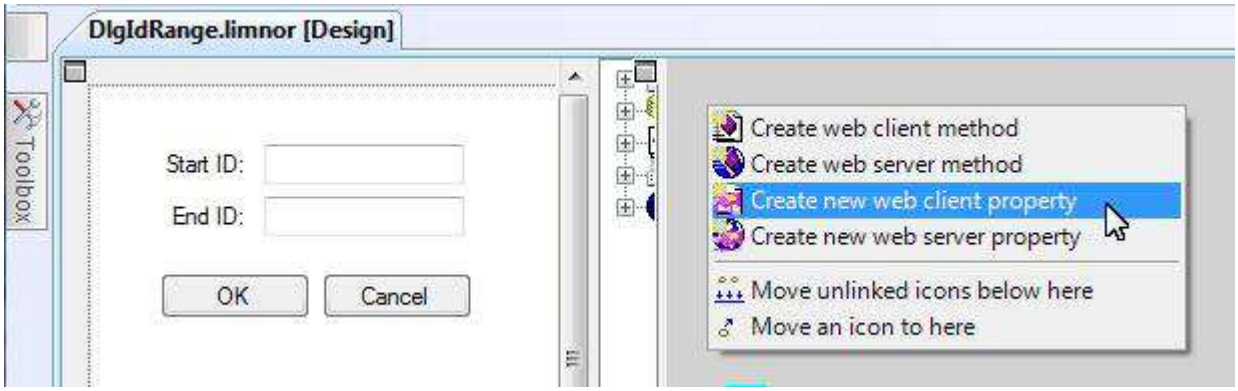
Data-transfer via client properties

Suppose we want to use a dialogue box to collect user inputs and pass the data from the dialogue box to the parent window.

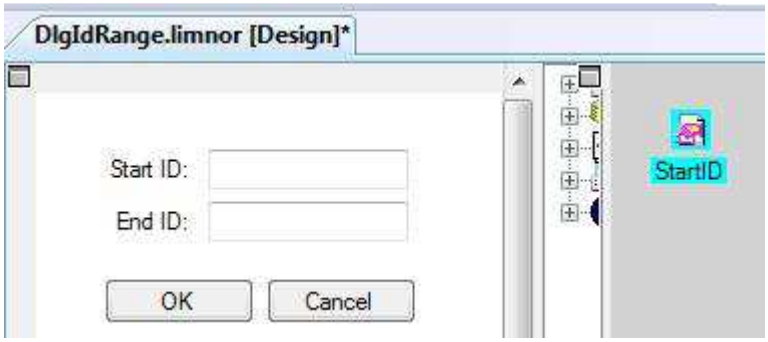
Suppose we want to let parent pages get data from the dialogue page.

Publish data via client properties

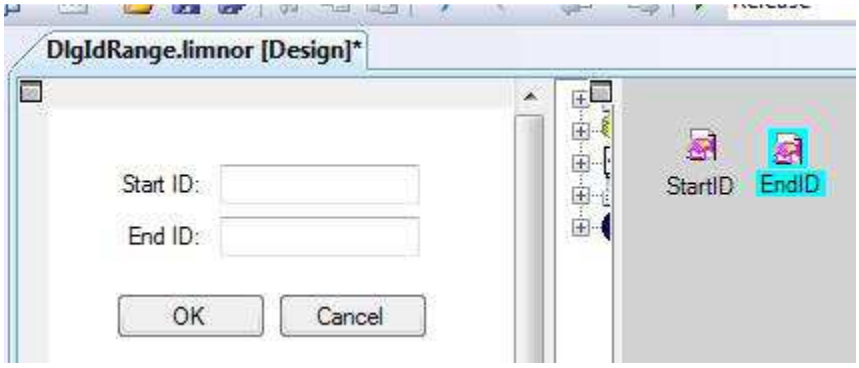
On the dialogue page, we may create a client property for each piece of data to be published by the dialogue page.



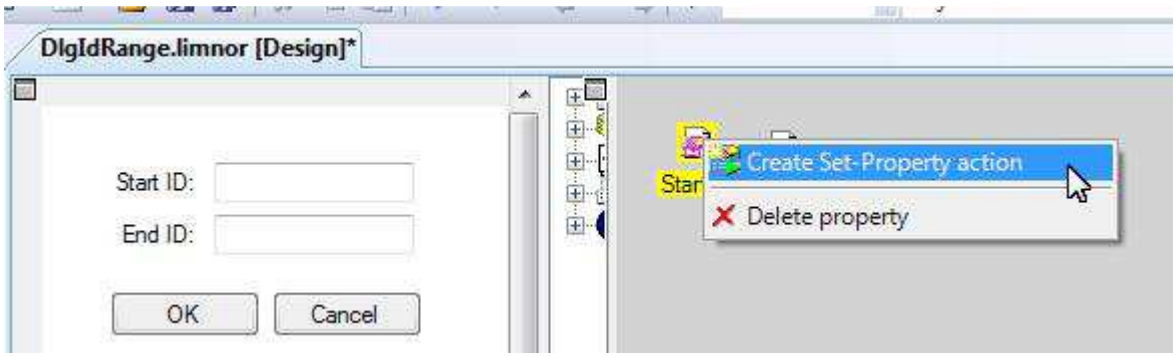
Let's name the new property StartID:

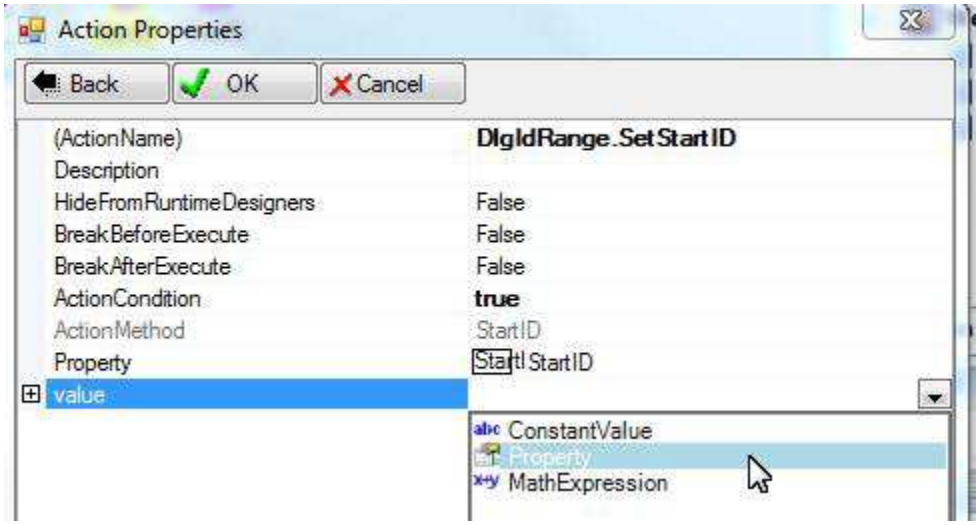


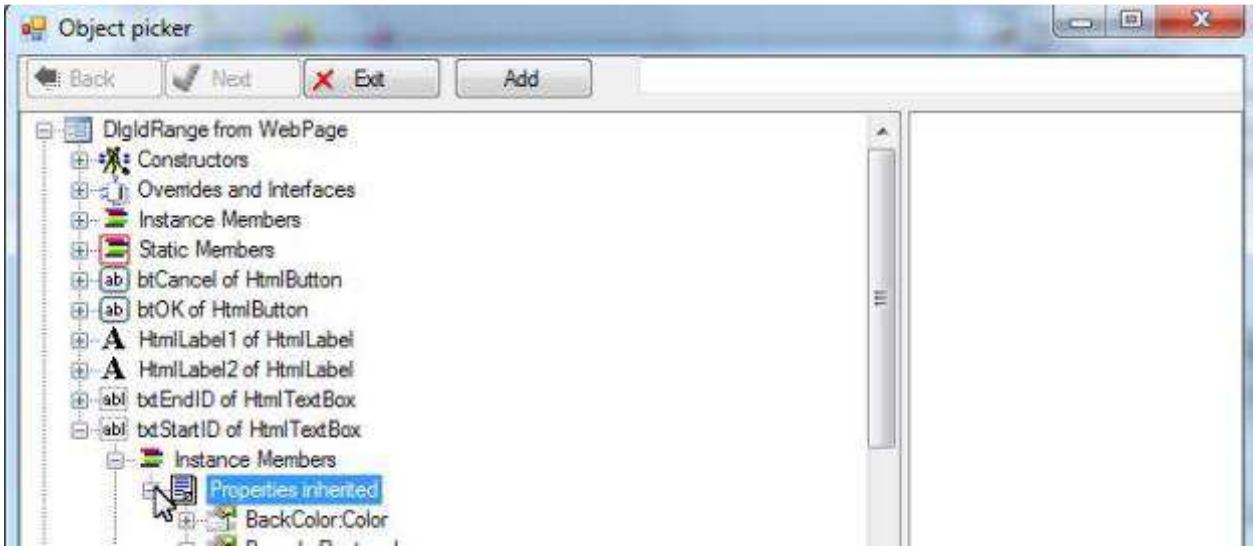
Also create a web client property named EndID:

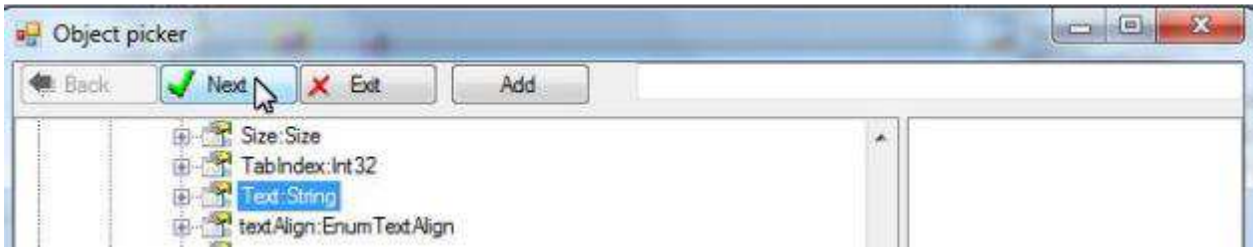


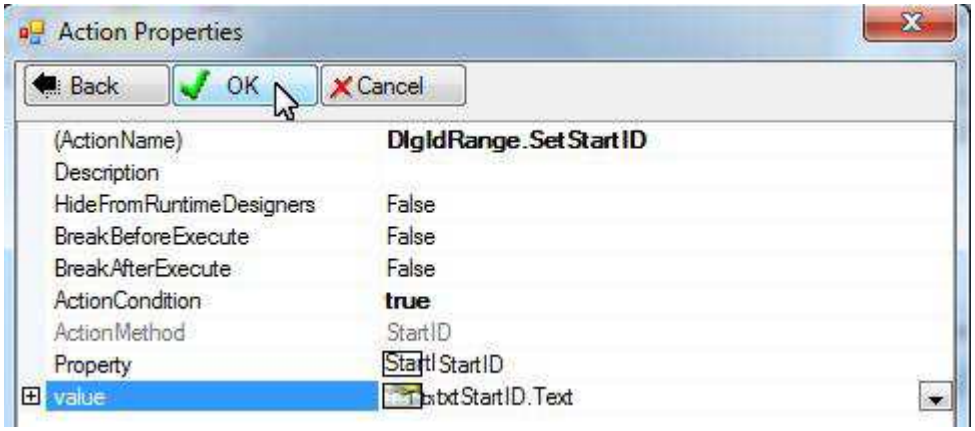
Create an action to set the StartID to the Text property of the first text box:



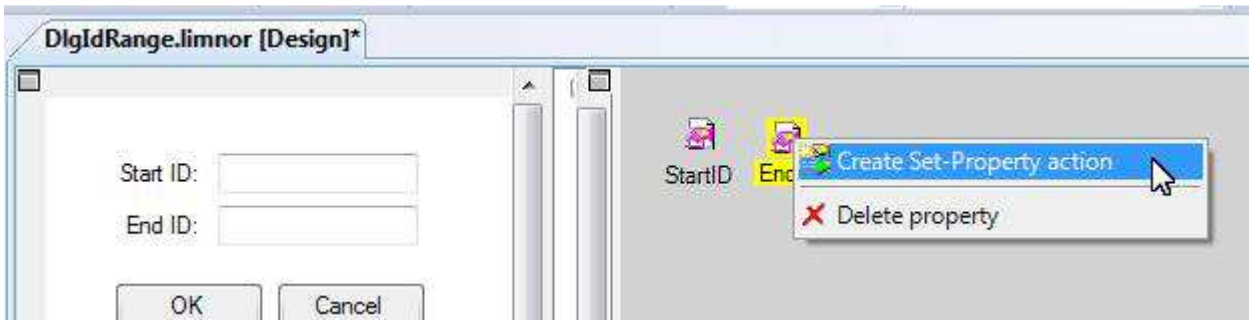

Action Properties
Back
OK
Cancel
(ActionName)
DlgIdRange.SetStartID
Description
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
StartID
Property
StartI StartID
value
ConstantValue
Property
MathExpression

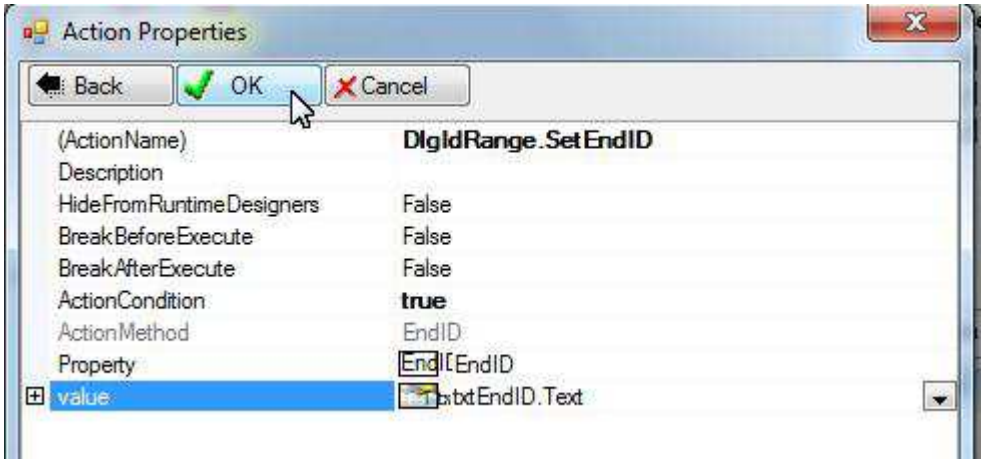

Object picker
Back
Next
Exit
Add
DlgIdRange from WebPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
btCancel of HtmlButton
btOK of HtmlButton
HtmlLabel1 of HtmlLabel
HtmlLabel2 of HtmlLabel
txtEndID of HtmlTextBox
txtStartID of HtmlTextBox
Instance Members
Properties inherited
BackColor:Color


Object picker
Back
Next
Exit
Add
Size:Size
TabIndex:Int32
Text:String
textAlign:EnumTextAlign

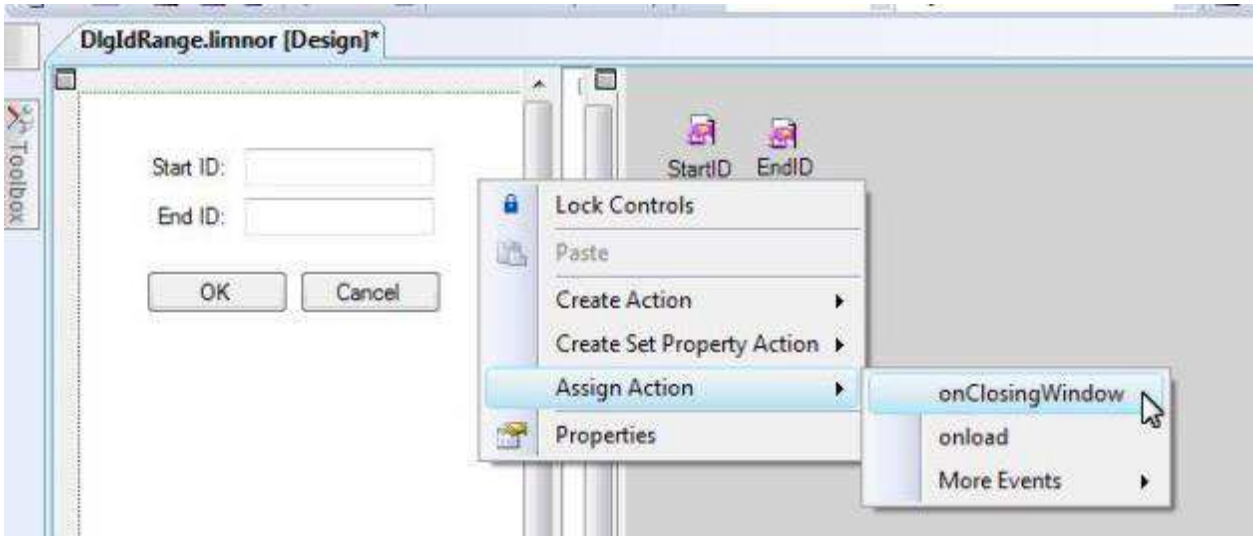


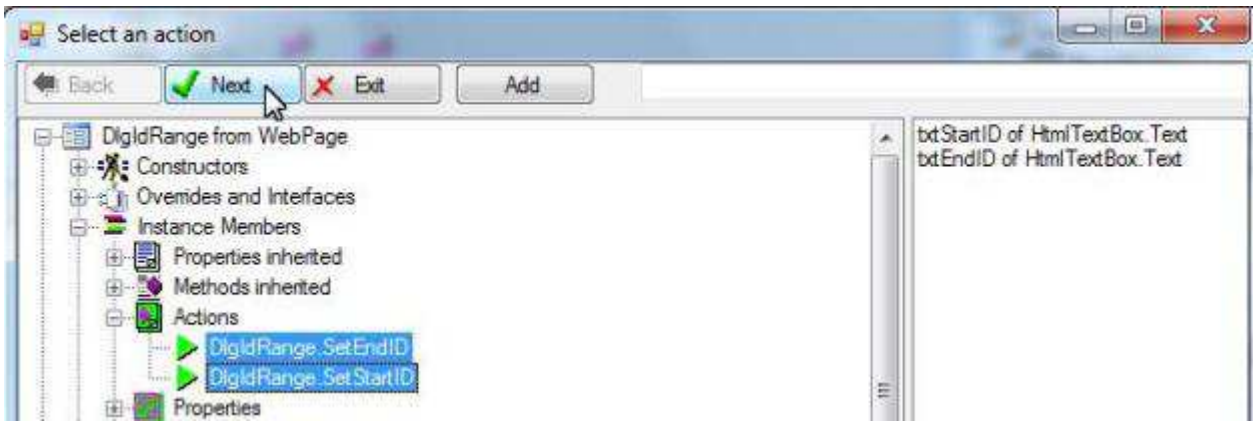
Create another action to set the EndID to the Text property of the second text box:

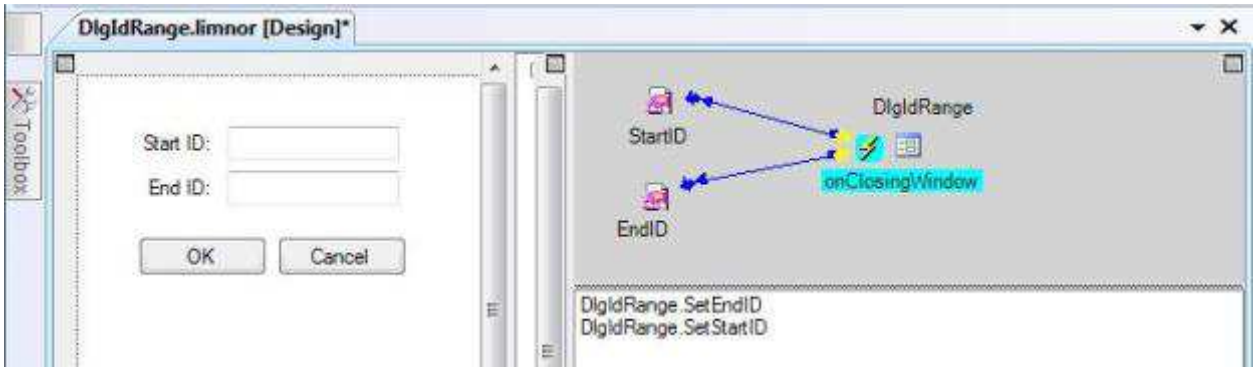




Assign these two actions to the onClosingWindow event:



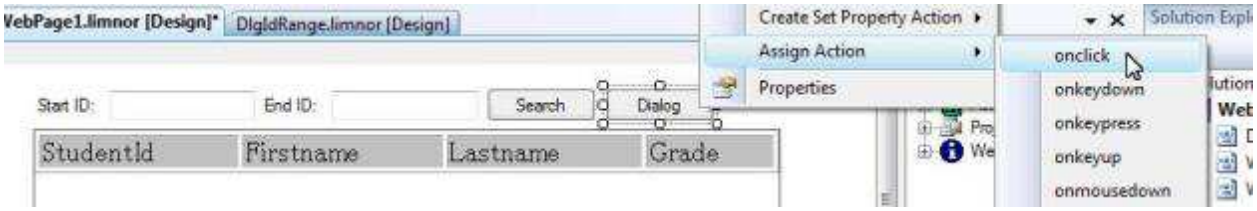


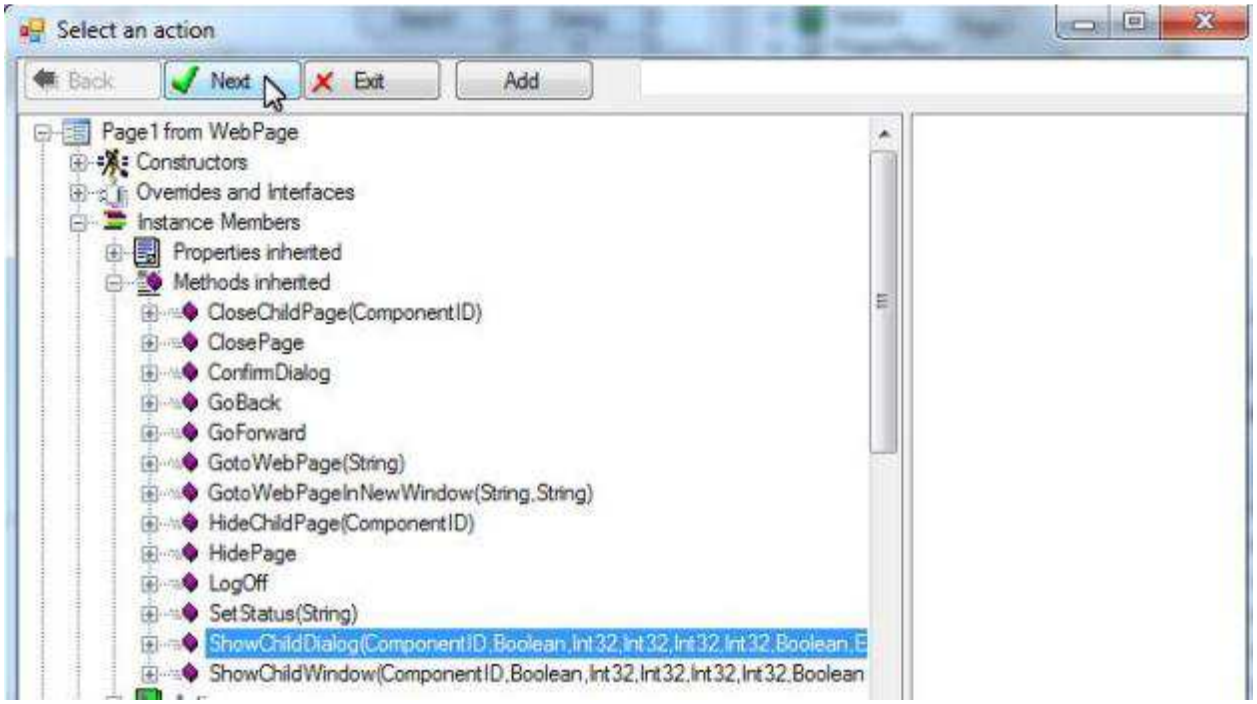


With this programming, any other web pages may act as a parent web page and get data from this web page.

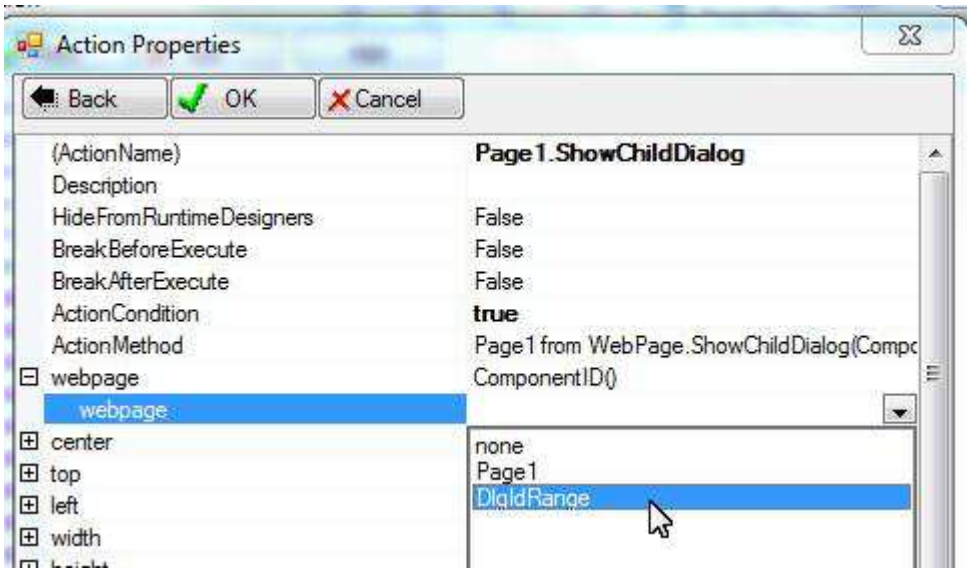
Launch dialogue box

Suppose we want to use the dialogue box to collect data. Add a button on page 1 to launch the dialogue box:

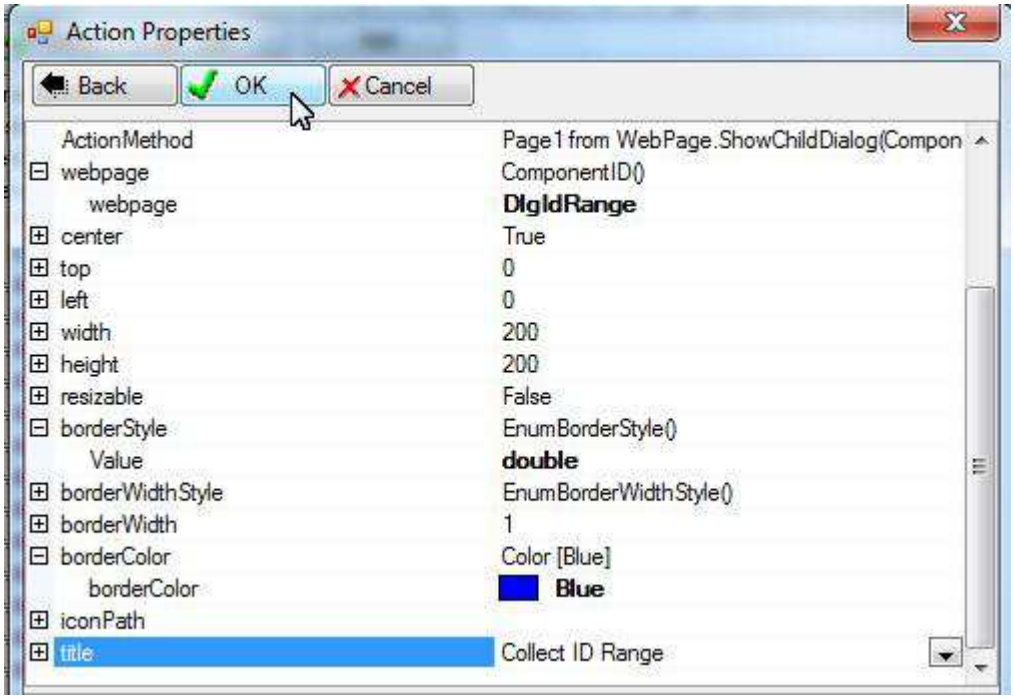




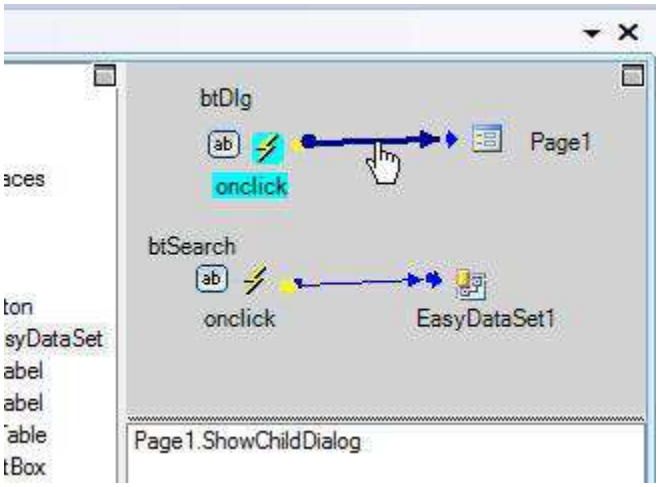
Specify the web page to be used as the dialogue box:



Specify other parameters for the dialogue box:



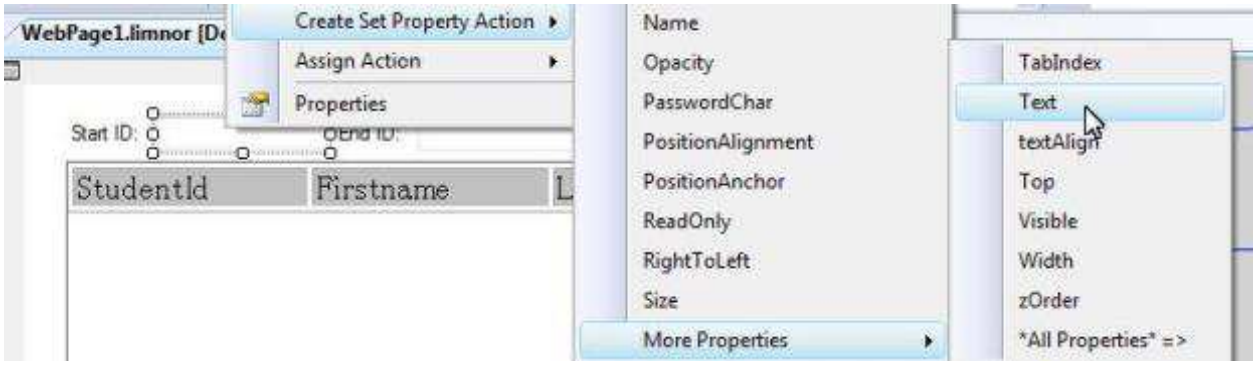
The dialogue launch action is created and assigned to the button:

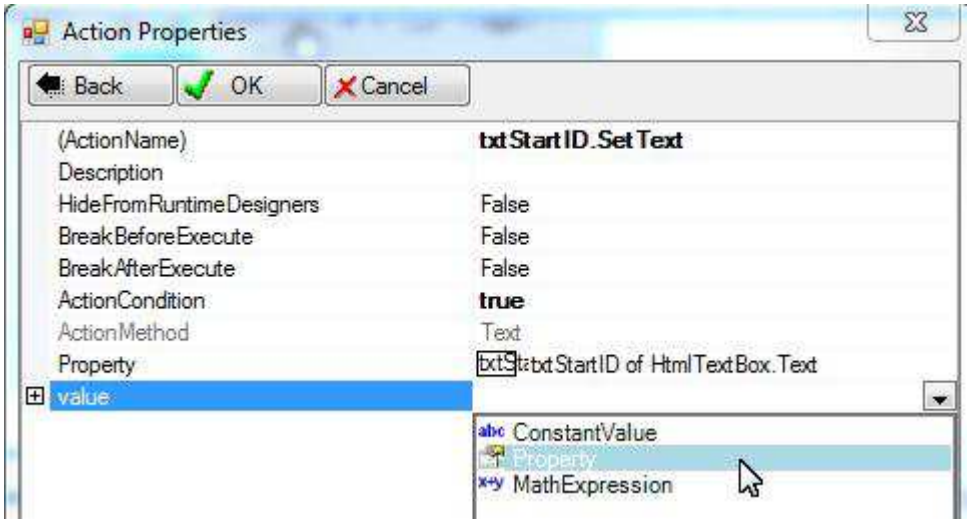


Access published data via default instance

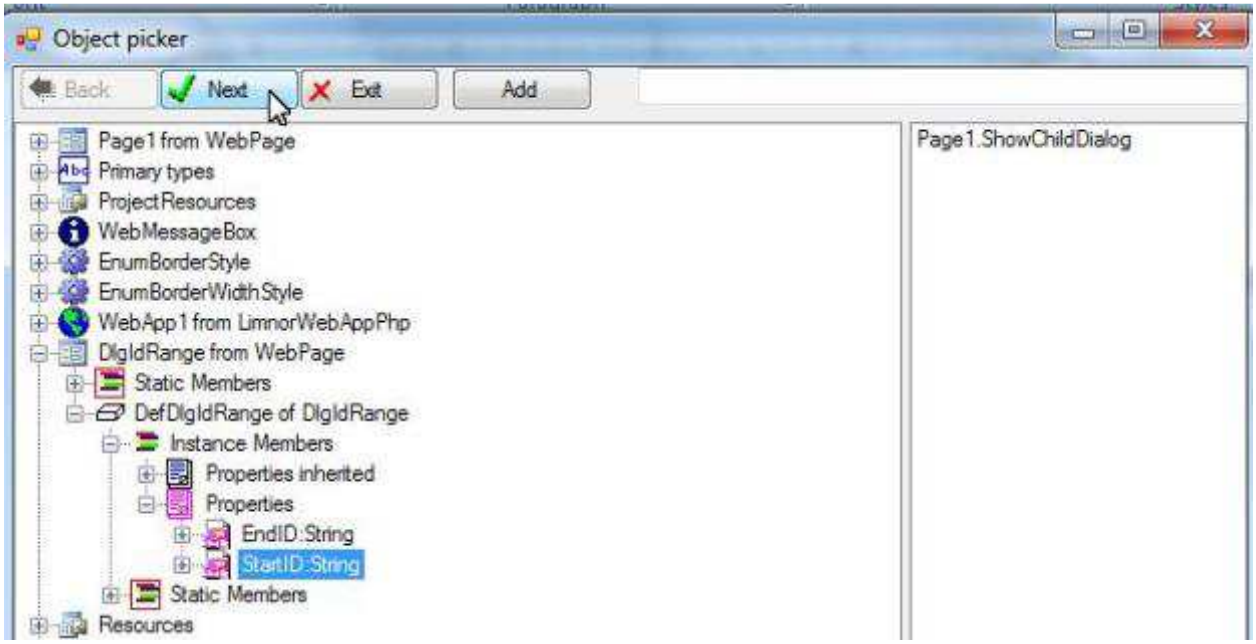
Suppose we want to get the data from the dialogue box into the text boxes on the parent page.

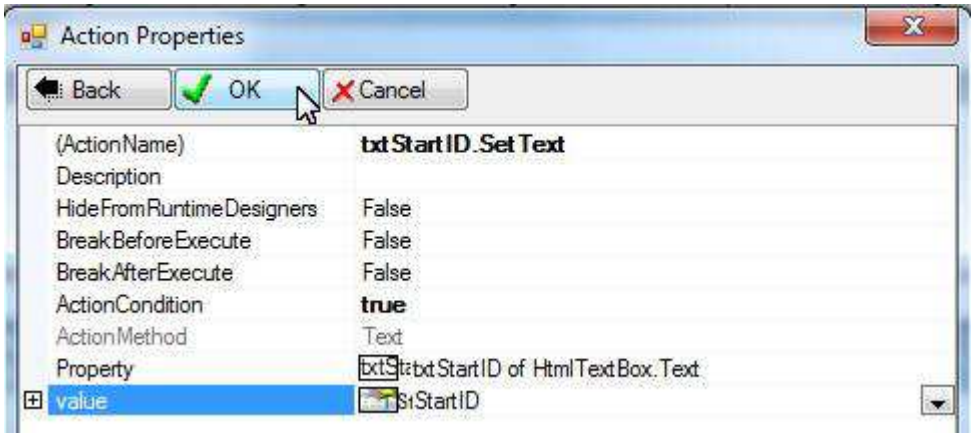
Create a set-property action for the first text box:





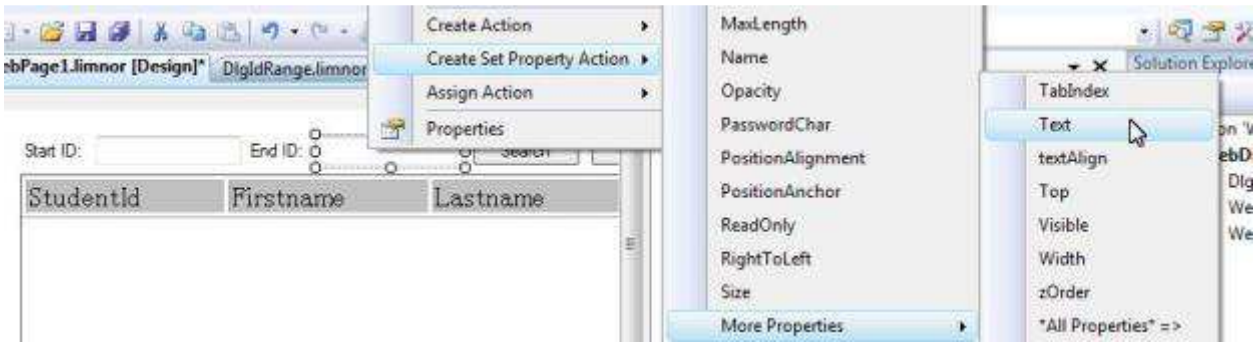
Select StartID property from the default instance of the dialogue page:

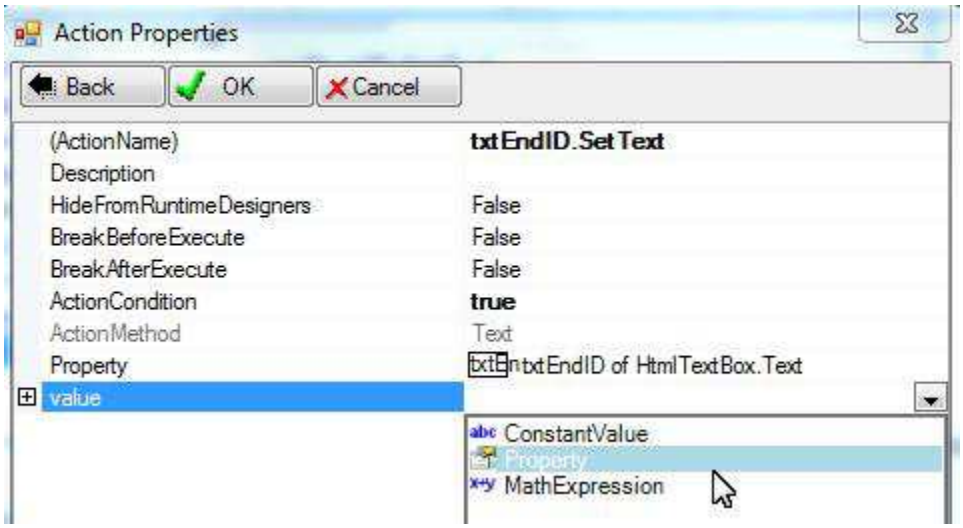




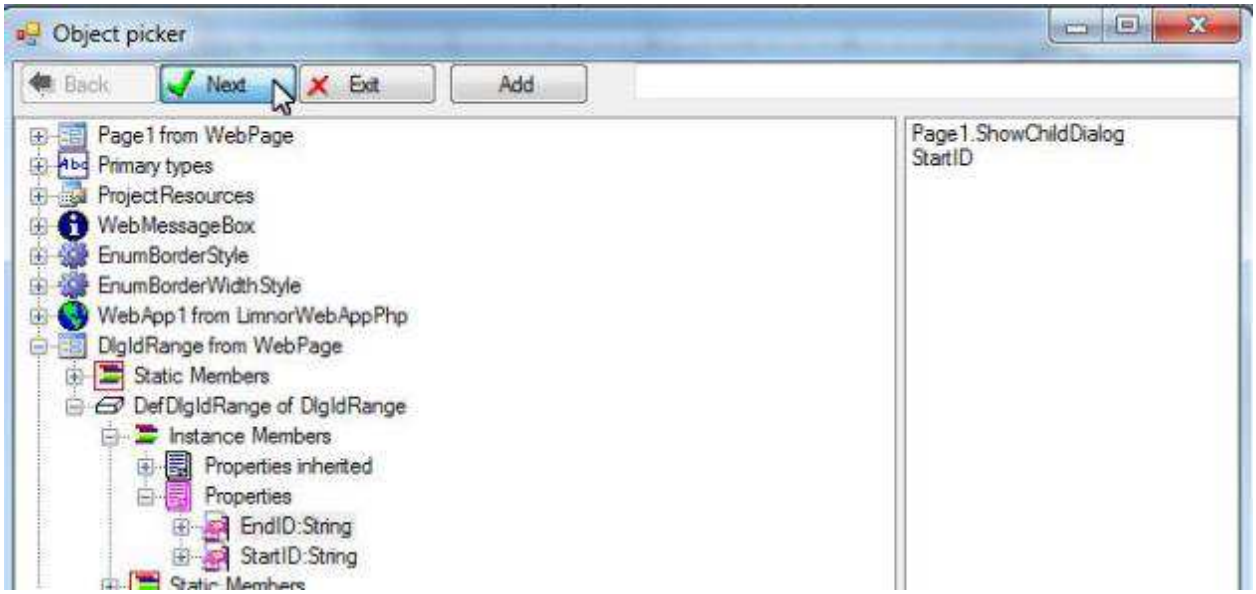
Note that for this action, “Property” and “value” belong to two different web pages.

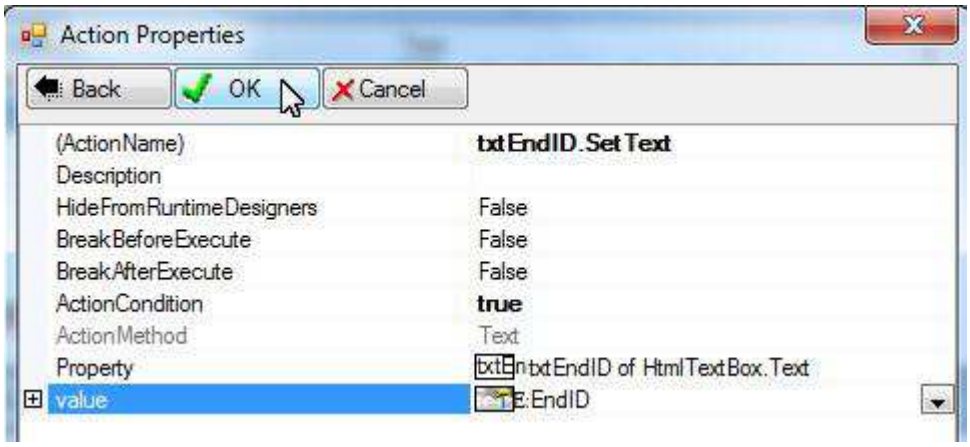
We may also create an action to get EndID property of the dialogue box:





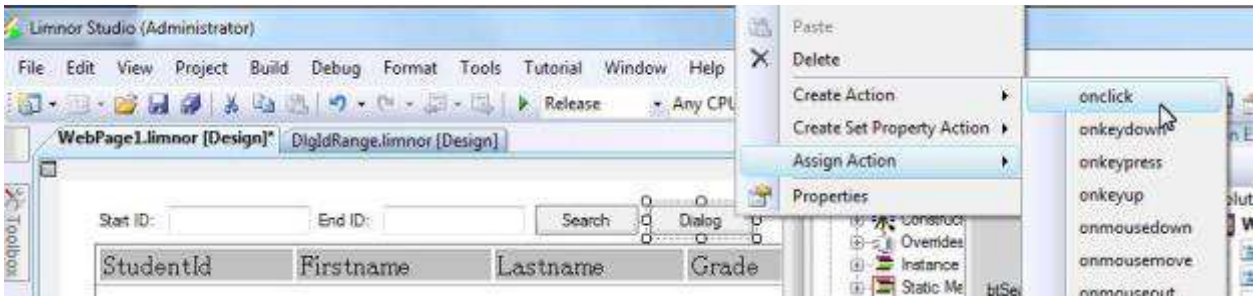
Select EndID property from the default instance of the dialogue page:

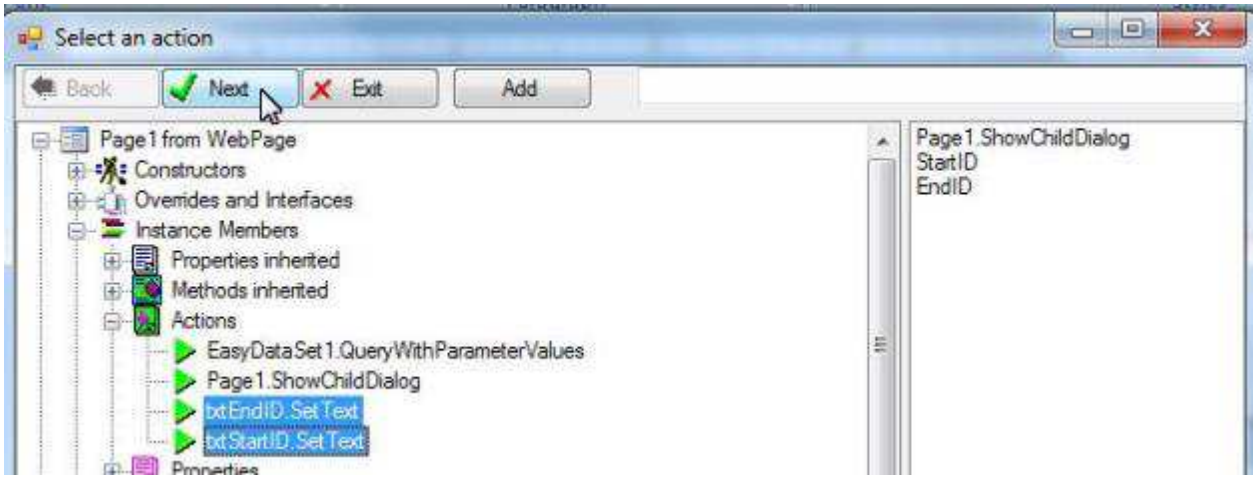


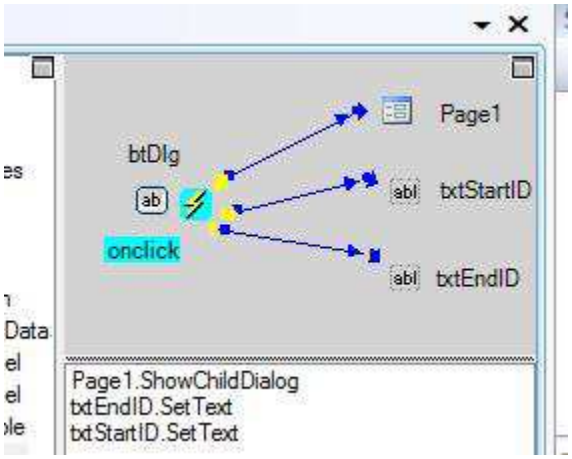


Note that for this action, “Property” and “value” belong to two different web pages.

Assign the above two actions to the button:



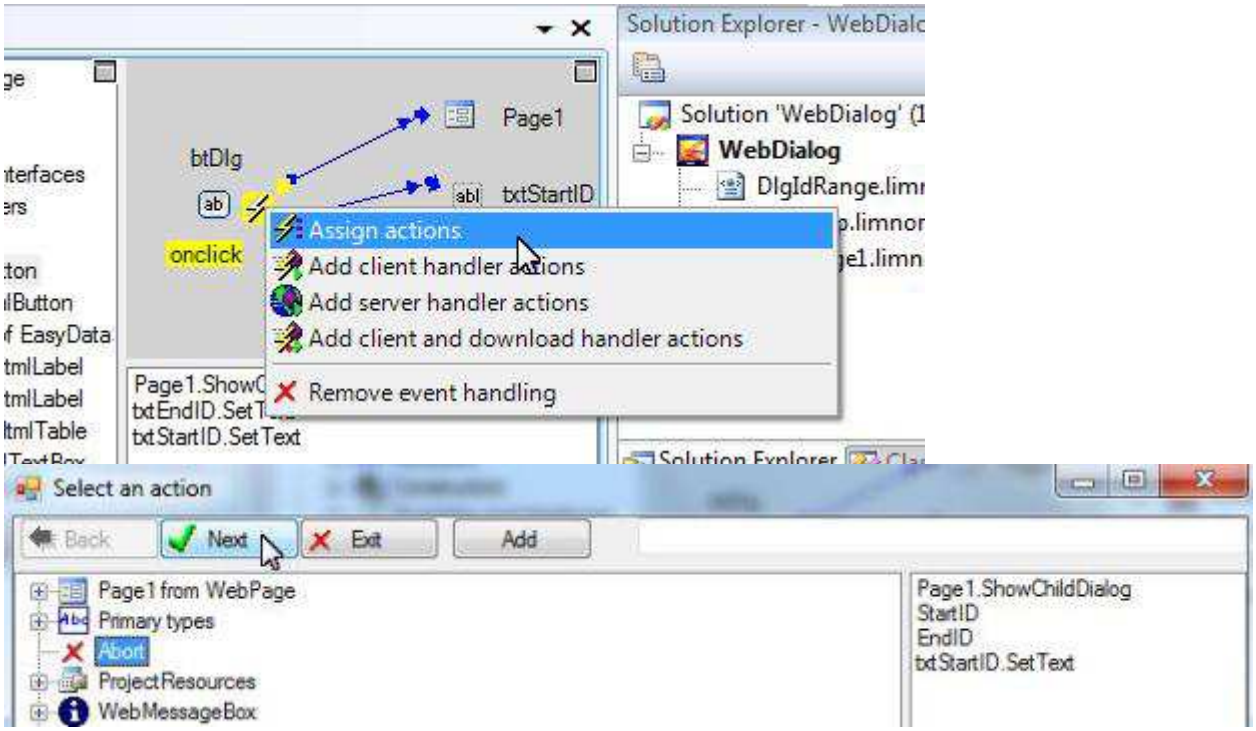




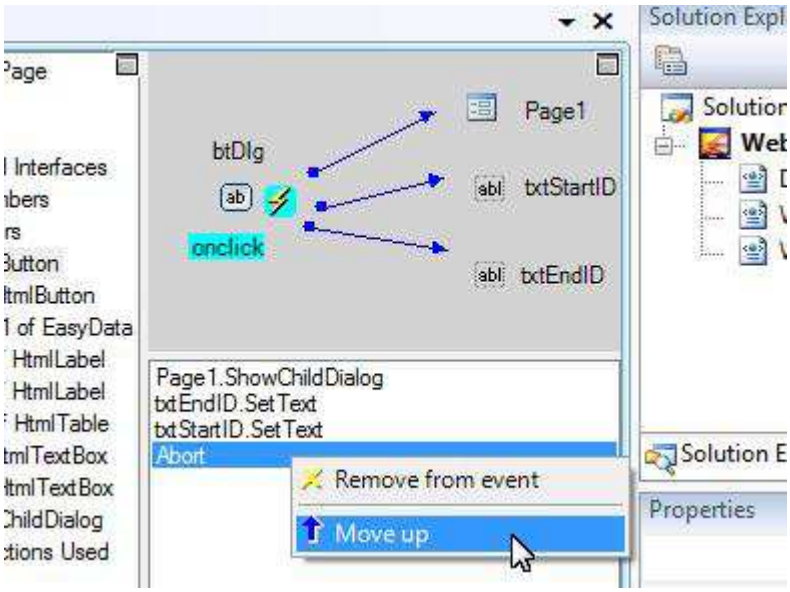
Note that the first action launches the dialogue box and the two “SetText” actions are blocked until the dialogue box is closed.

Handle canceling of dialogue box

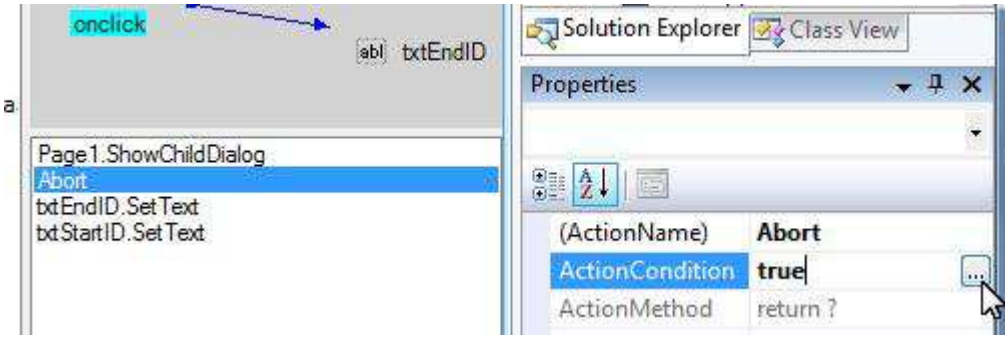
If the user cancels the dialogue box then we do not want to execute the two “SetText” actions. From the dialogResult property we know if the user cancels the dialogue box or not. For example, we may use an “Abort” action to abort the event handling if the dialogResult is “cancel”.

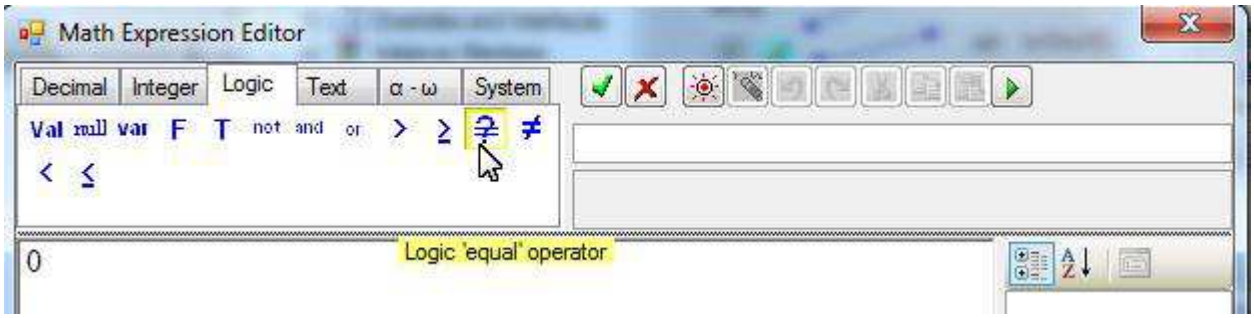


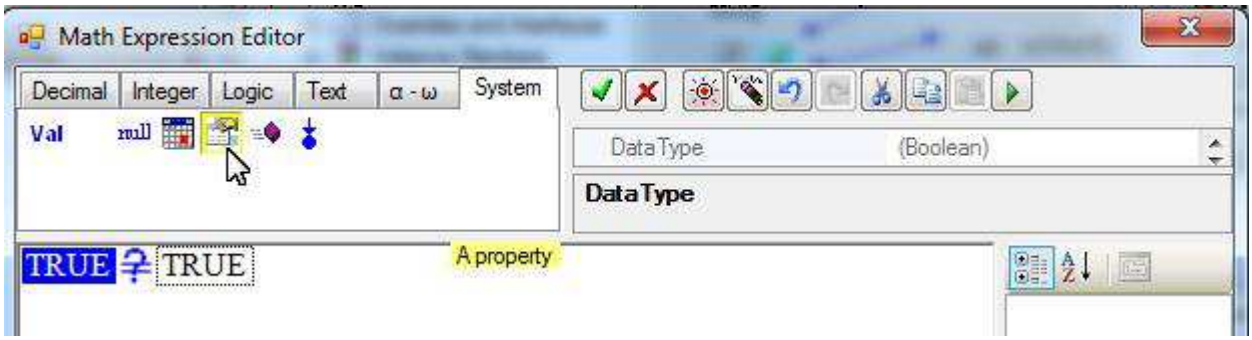
Move Abort action up to follow the dialogue box launch action:

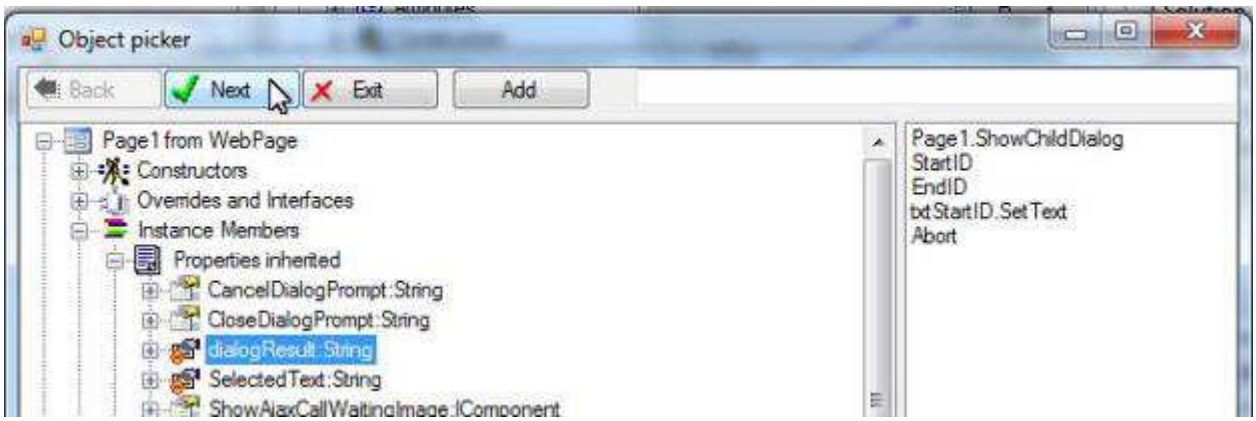


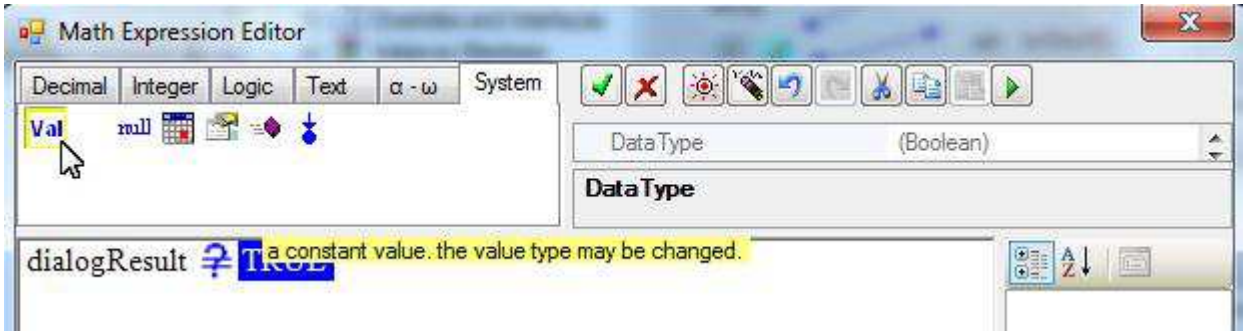
Set its ActionCondition to be that the dialogResult is “cancel”:

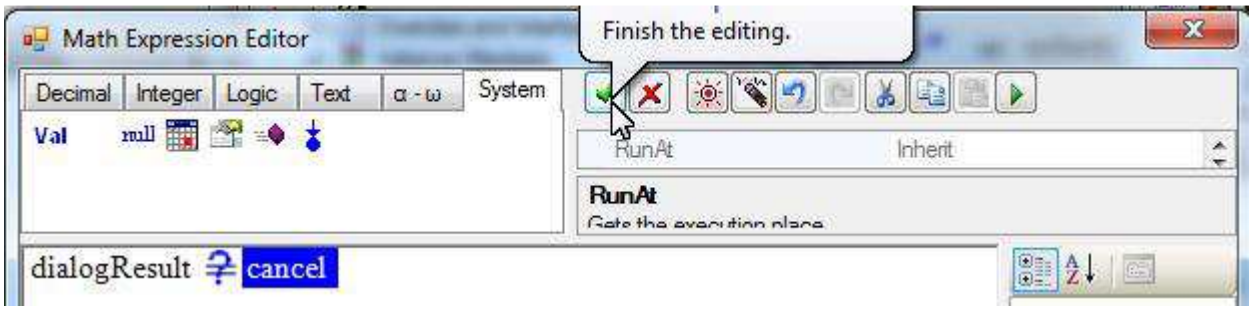

onclick
abl txtEndID
Page1.ShowChildDialog
Abort
txtEndID.SetText
txtStartID.SetText
Solution Explorer
Class View
Properties
(ActionName) Abort
ActionCondition true
ActionMethod return ?


Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
Val null var F T not and or
0
Logic 'equal' operator


Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
Val null
DataType (Boolean)
DataType
TRUE TRUE
A property

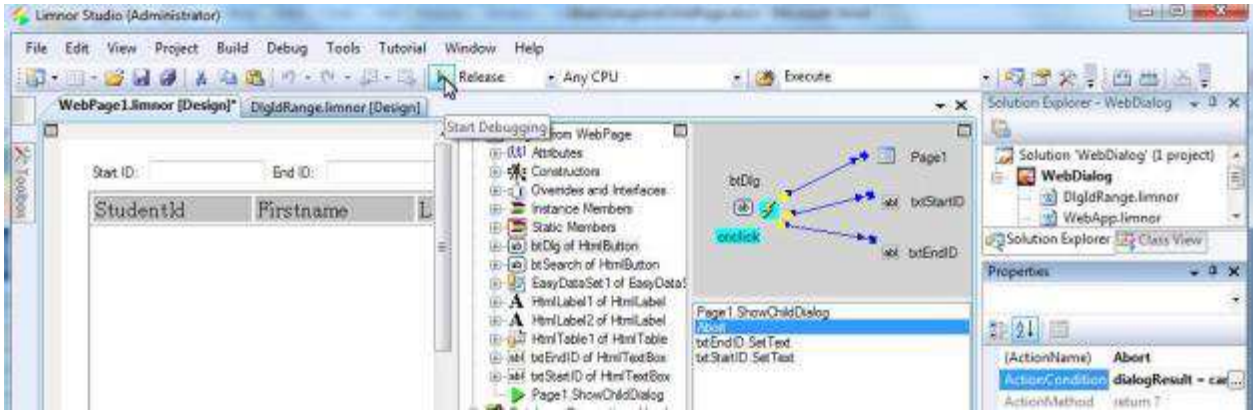

Object picker
Back
Next
Exit
Add
Page1 from WebPage
Constructors
Overrides and Interfaces
Instance Members
Properties inherited
CancelDialogPrompt:String
CloseDialogPrompt:String
dialogResult:String
SelectedText:String
Page1.ShowChildDialog
StartID
EndID
txtStartID.SetText
Abort



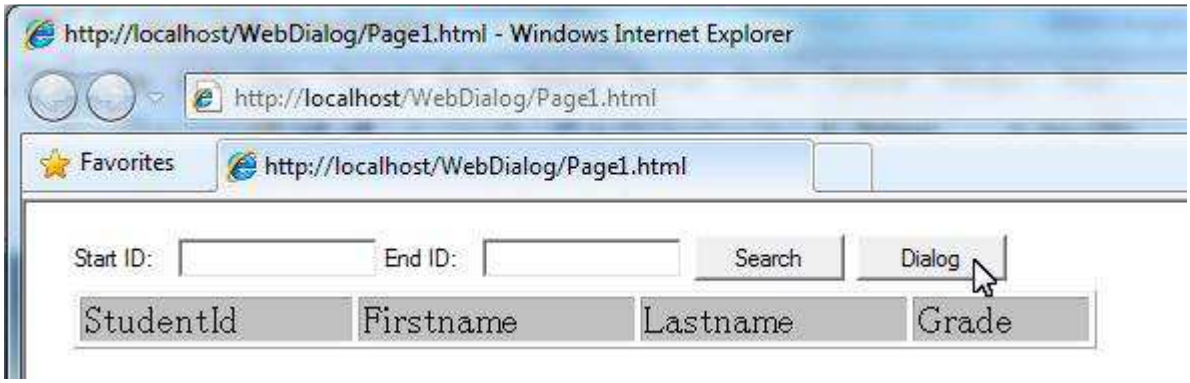


Test

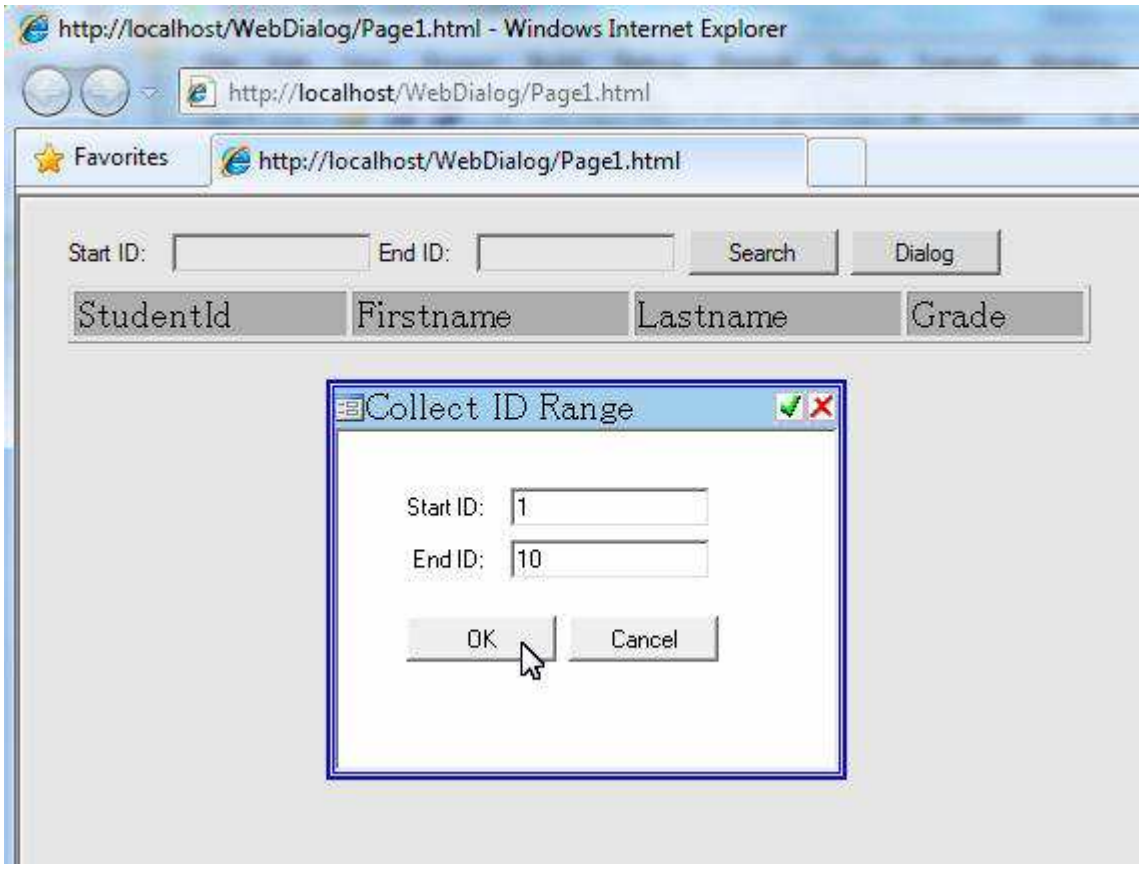
We may test our programming now.



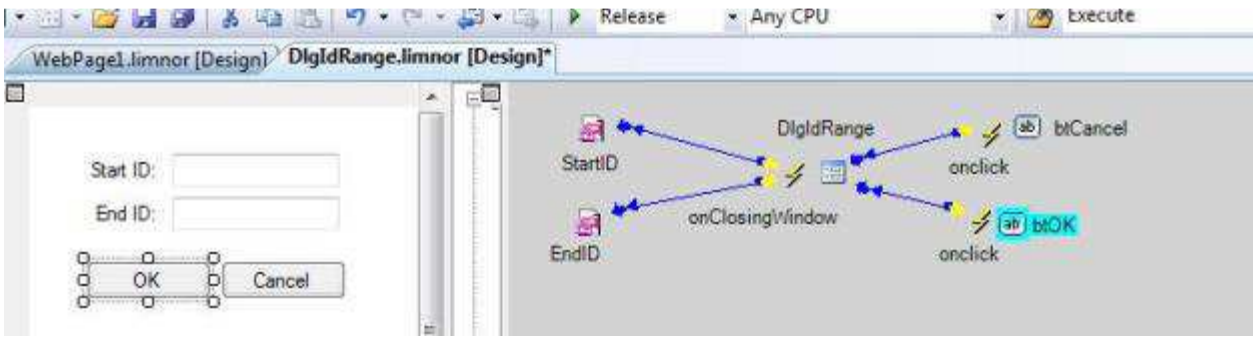
Click the dialog button:



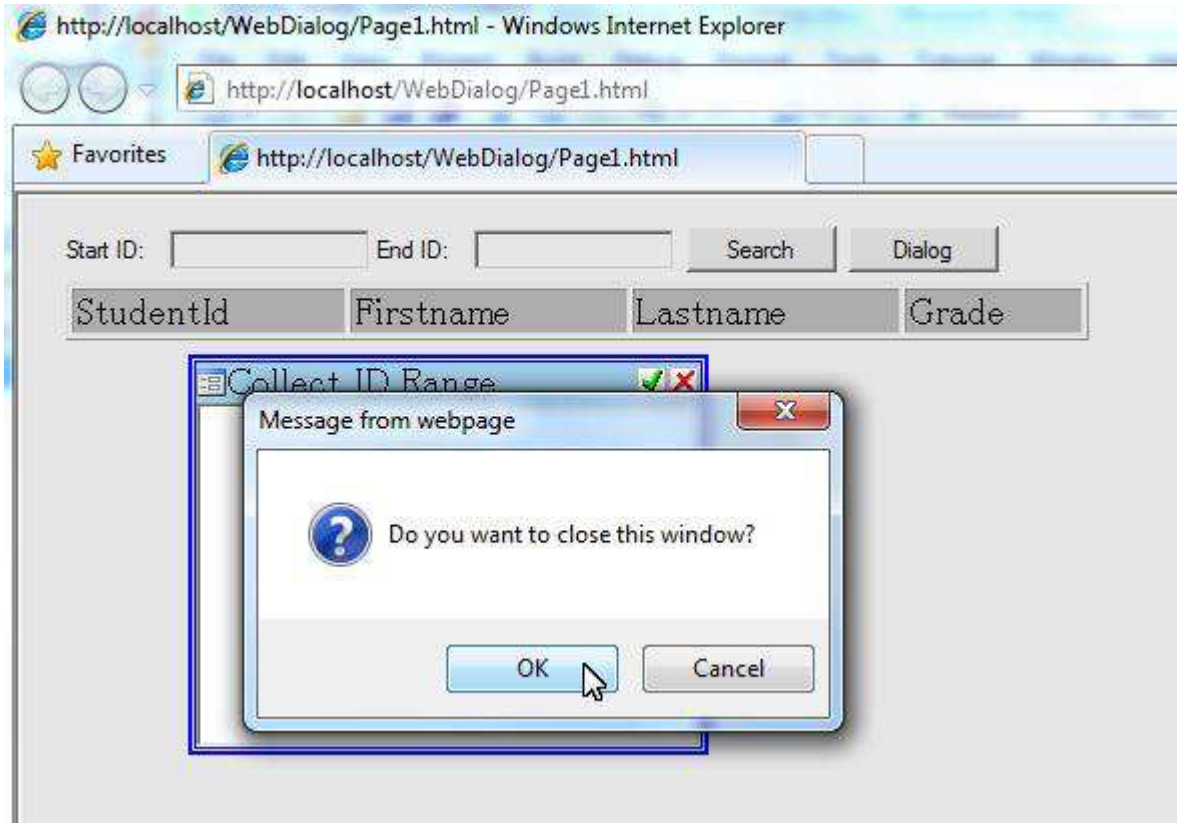
A dialogue box appears. Enter data and click OK:

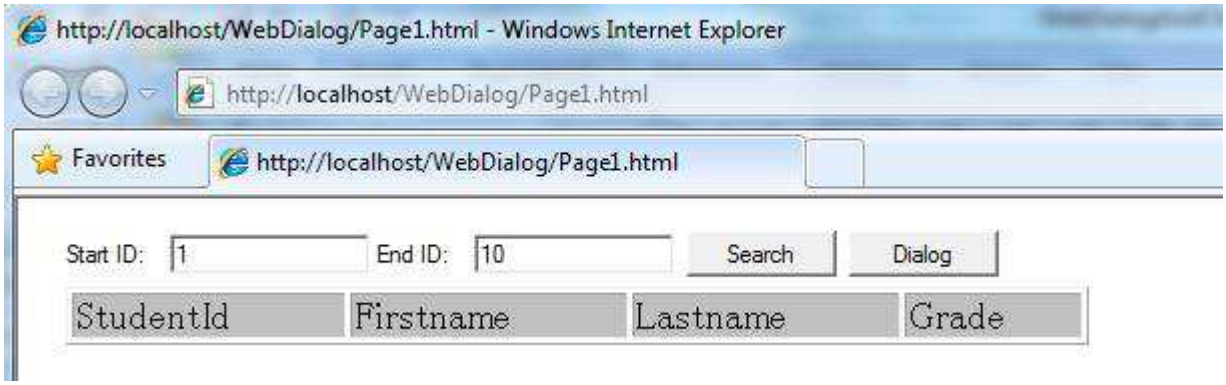


We did not show that a ConfirmDialog action was assigned to the OK button, and a ClosePage action was assigned to the Cancel button. If you do not know how to do it then please read "Users' Guide for Beginners": http://www.limnor.com/support/UsersGuideForBeginners.pdf

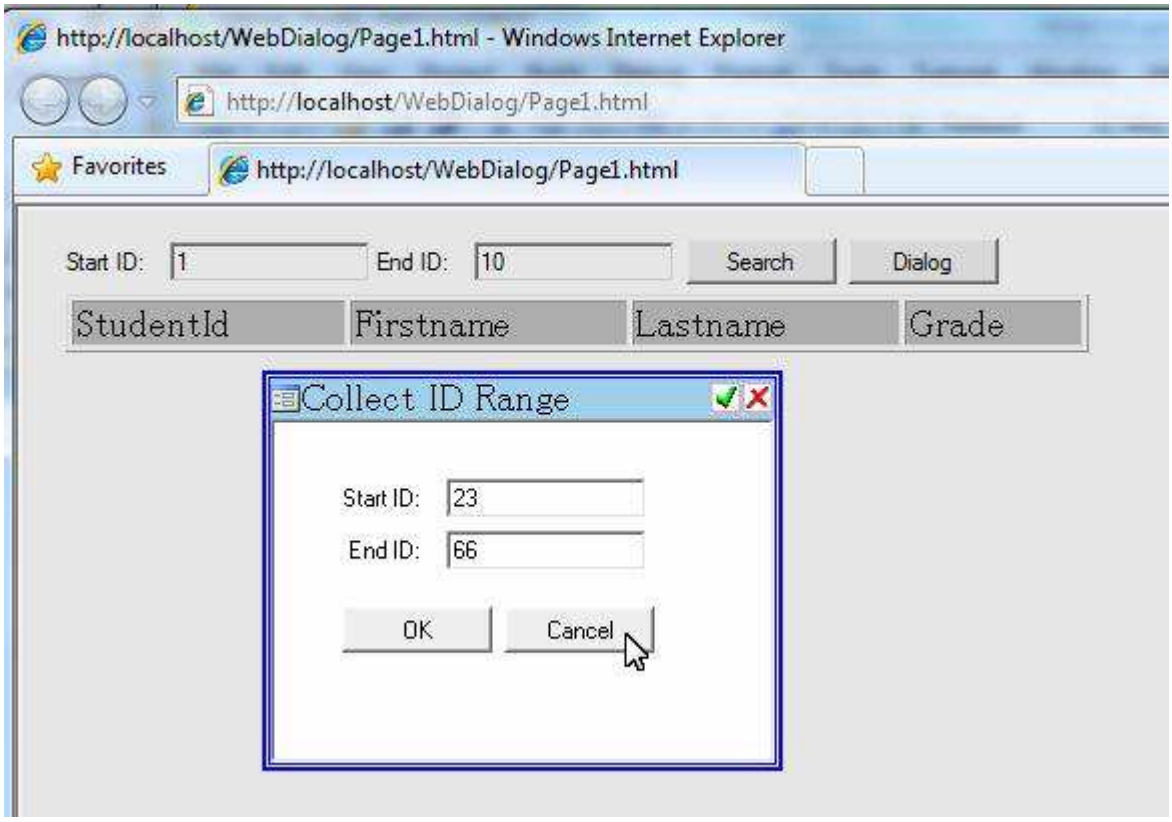


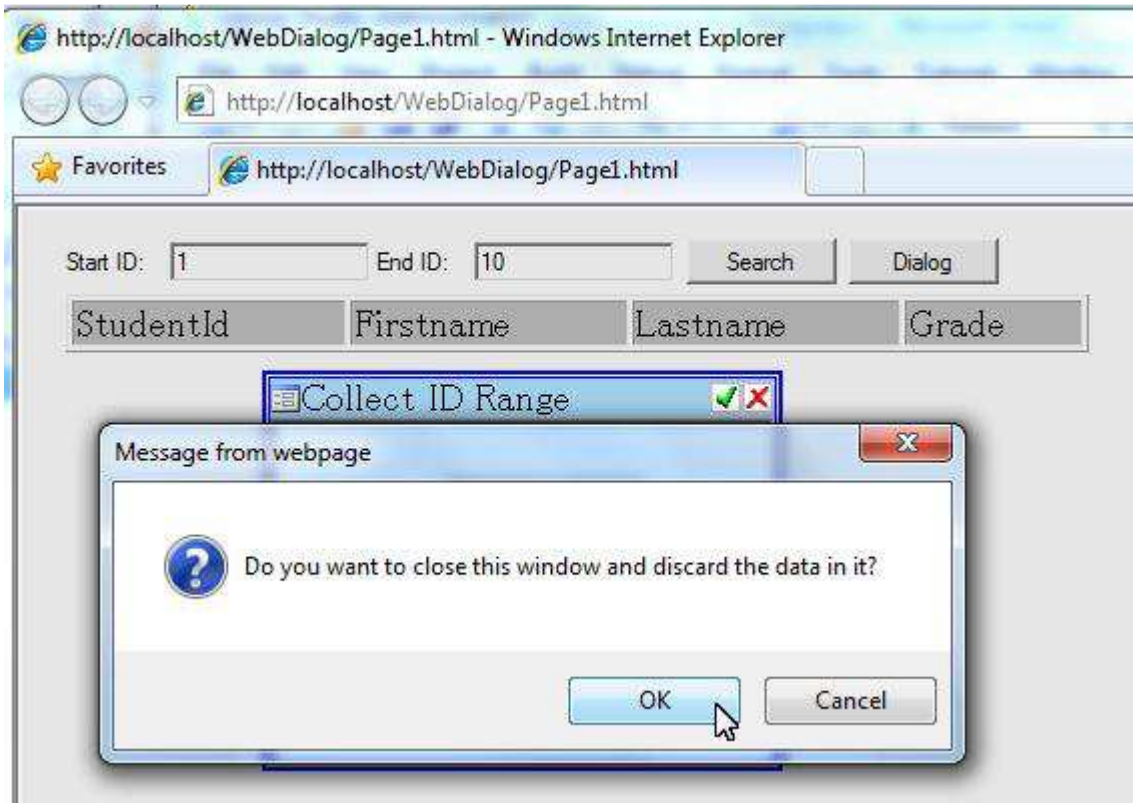
By clicking OK, the data are transferred from the dialogue box to the parent page:





Let's try the Cancel button.





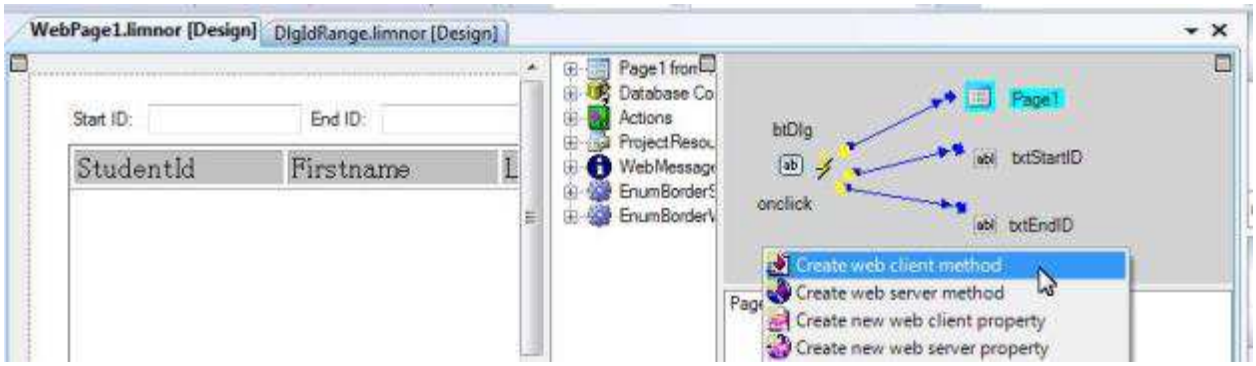
We can see that the data in the parent page are not changed.

Action execution via client methods

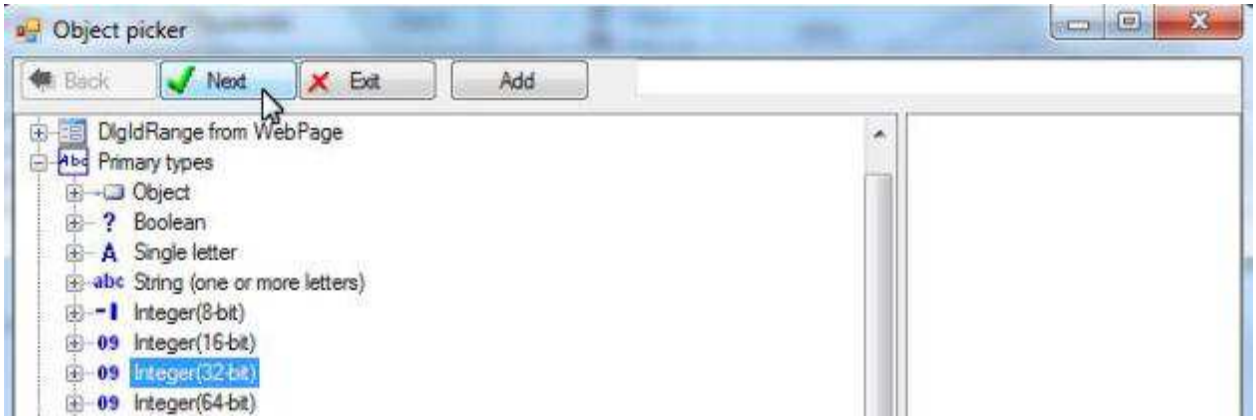
Suppose we want to do data search on the parent page from the dialogue page.

Publish functions via client methods

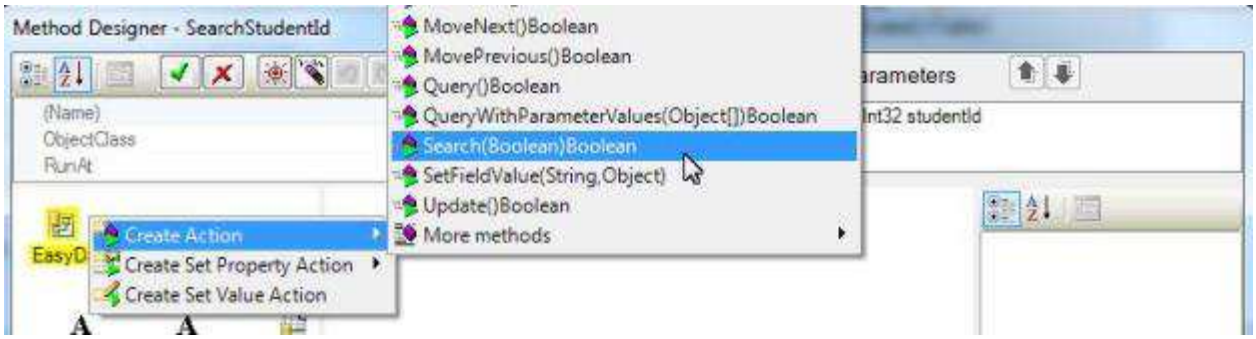
The parent page has an EasyDataSet for providing data from a database. The EasyDataSet has a Search method for locating a record on the web page. We may publish this feature to other pages by creating client methods.

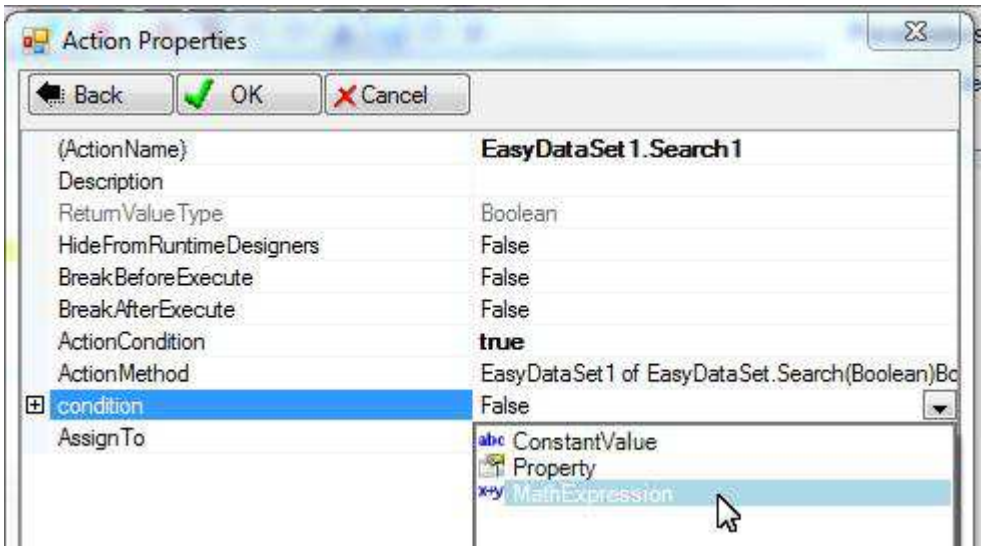


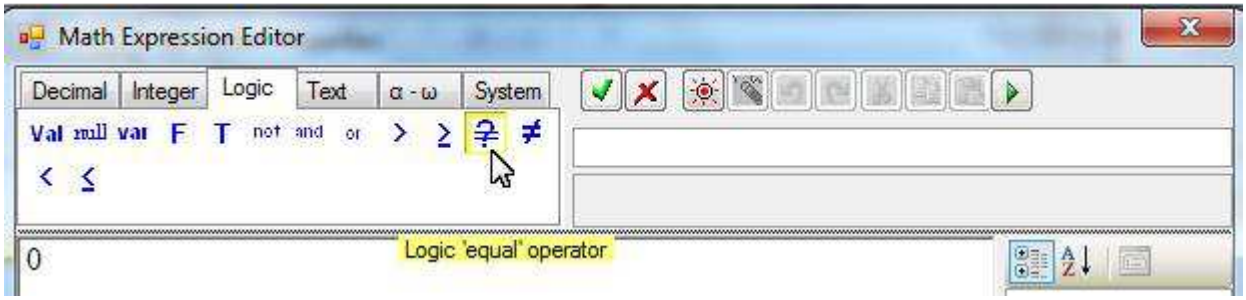
Name the method “SearchStudentId”. Add a parameter named studentId as the value to be searched for.

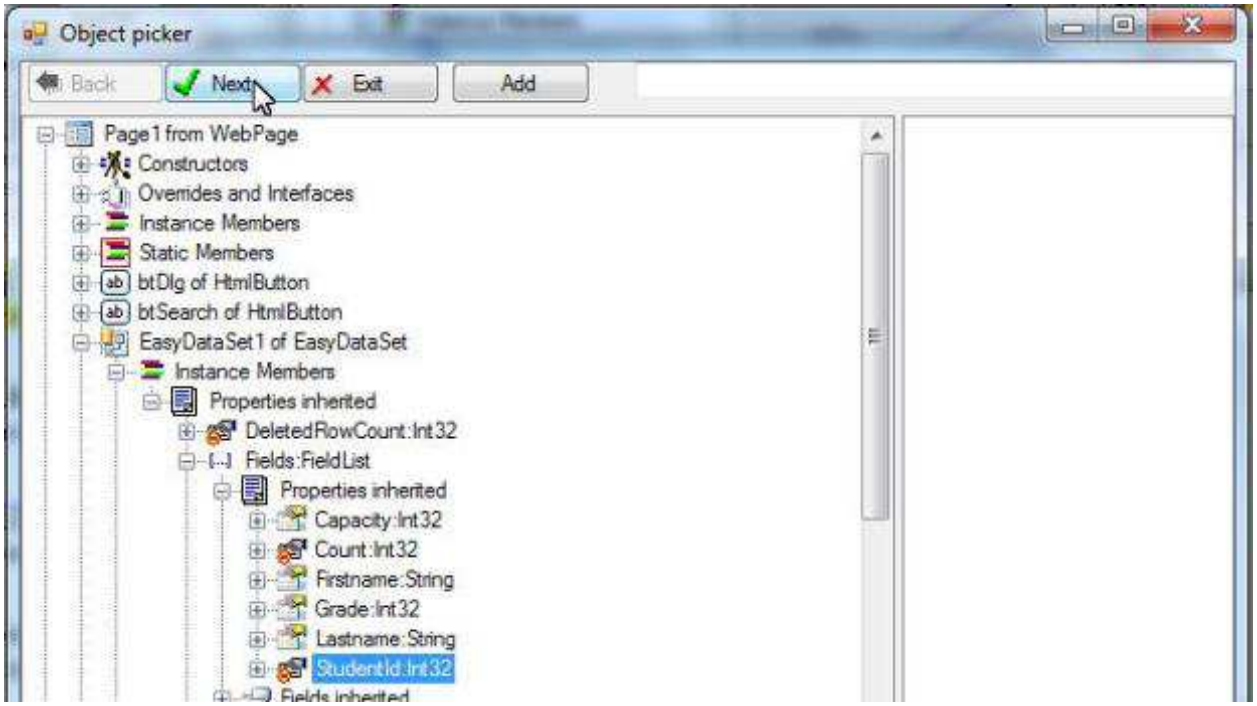


Create an action to do the search:

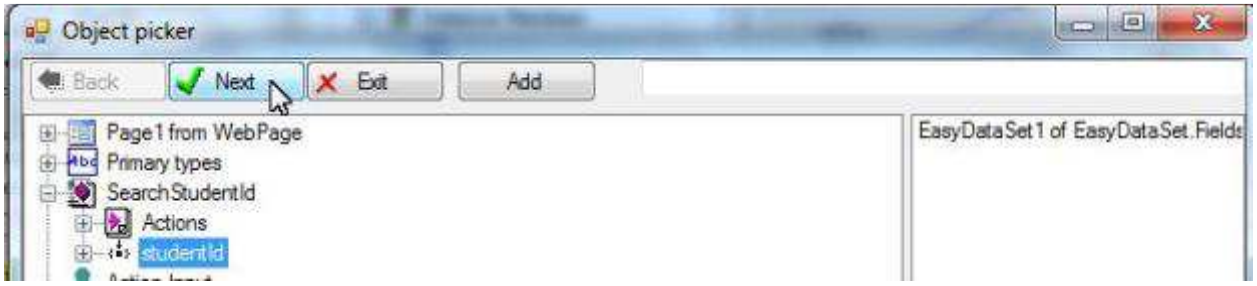


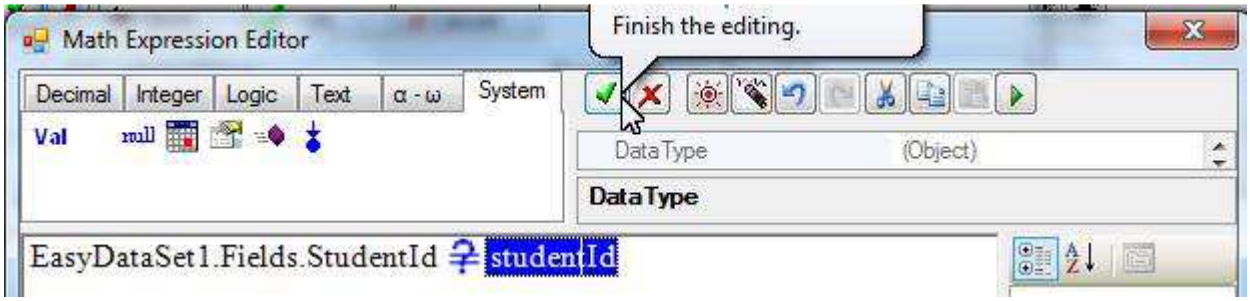


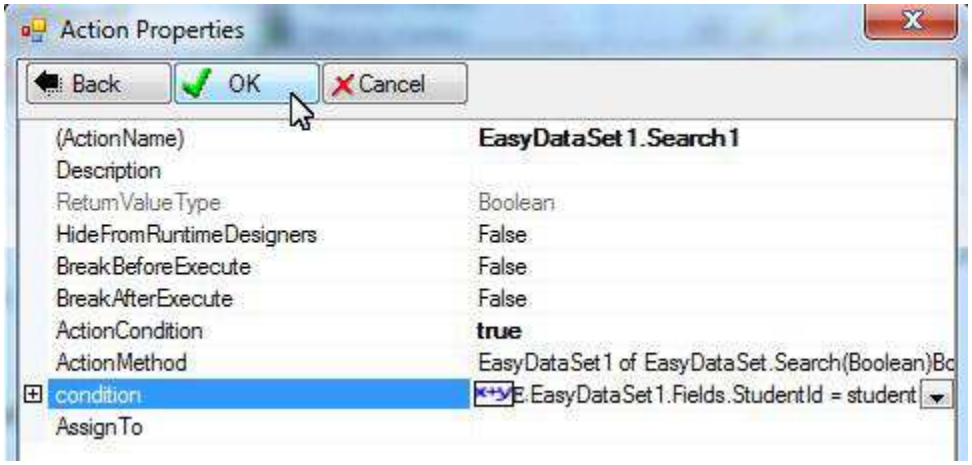


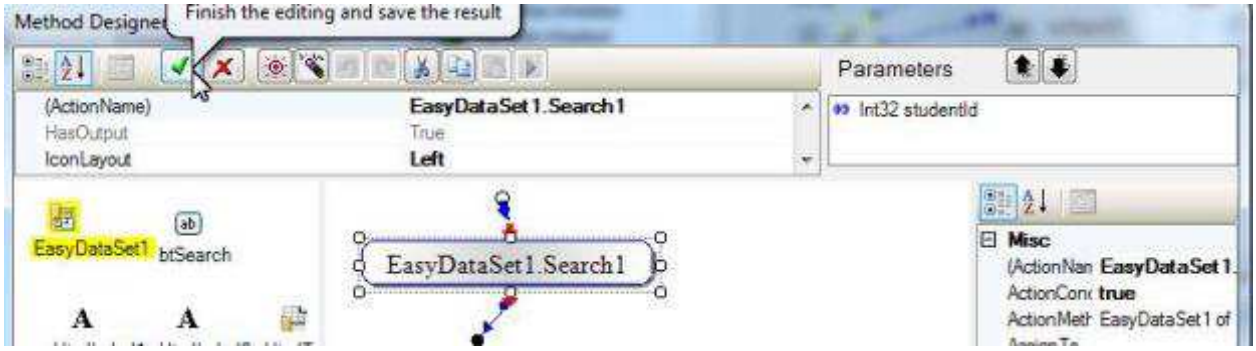

Object picker
Back
Next
Exit
Add
Page1 from WebPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
btDlg of HtmlButton
btSearch of HtmlButton
EasyDataSet1 of EasyDataSet
Instance Members
Properties inherited
DeletedRowCount:Int32
Fields:FieldList
Properties inherited
Capacity:Int32
Count:Int32
Firstname:String
Grade:Int32
Lastname:String
StudentId:Int32


Math Expression Editor
Decimal
Integer
Logic
Text
α - ω
System
Val
null
DataType
(Boolean)
DataType
EasyDataSet1.Fields.StudentId
A property


Object picker
Back
Next
Exit
Add
Page1 from WebPage
Primary types
SearchStudentId
Actions
studentId
EasyDataSet1 of EasyDataSet.Fields


Math Expression Editor
Finish the editing.
Decimal
Integer
Logic
Text
α - ω
System
Val
null
DataType
(Object)
DataType
EasyDataSet1.Fields.StudentId
studentId

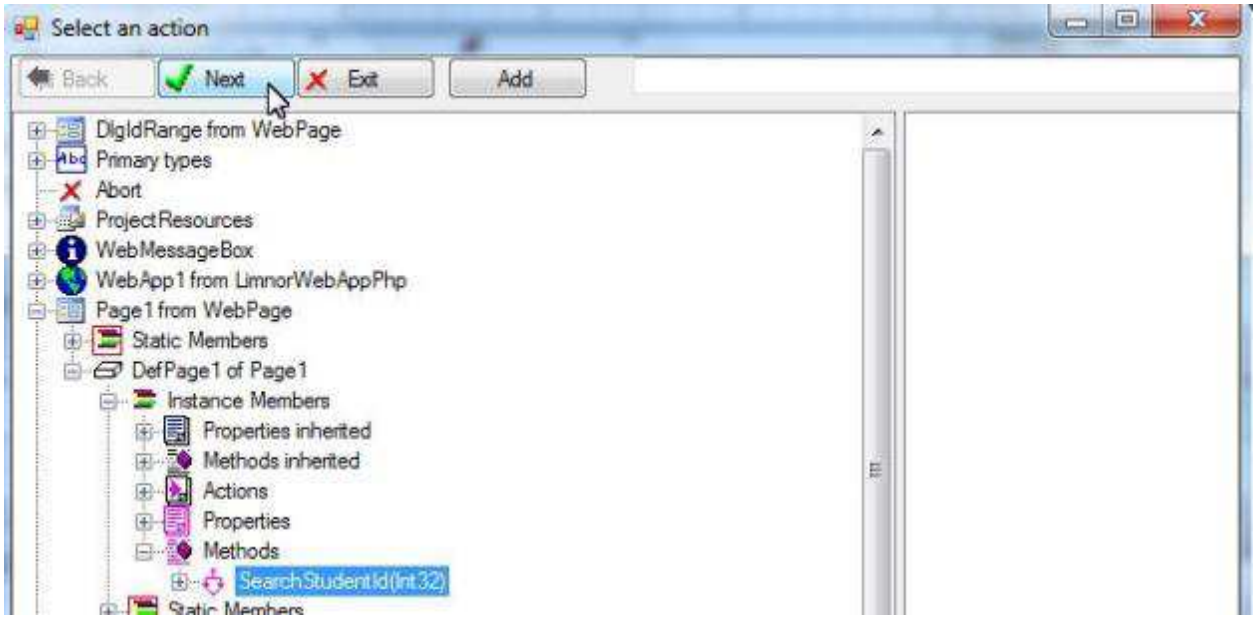




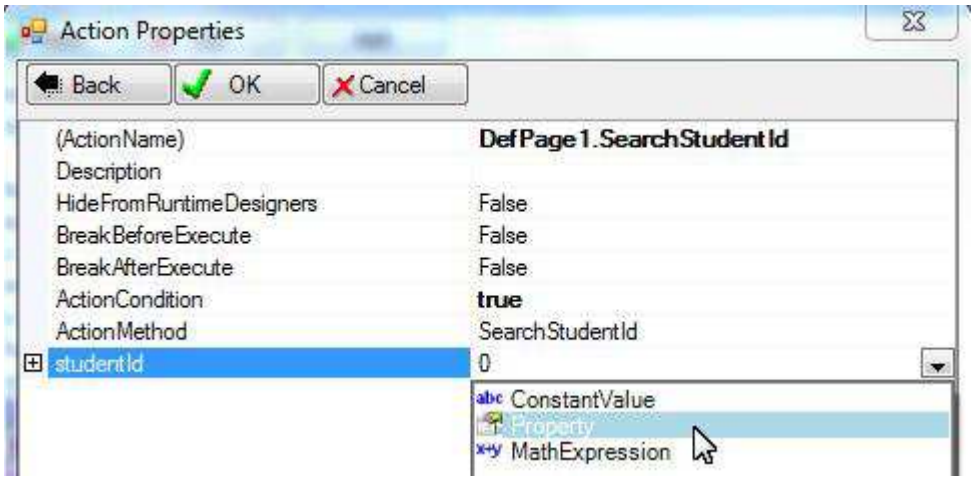
Execute client method from other web pages

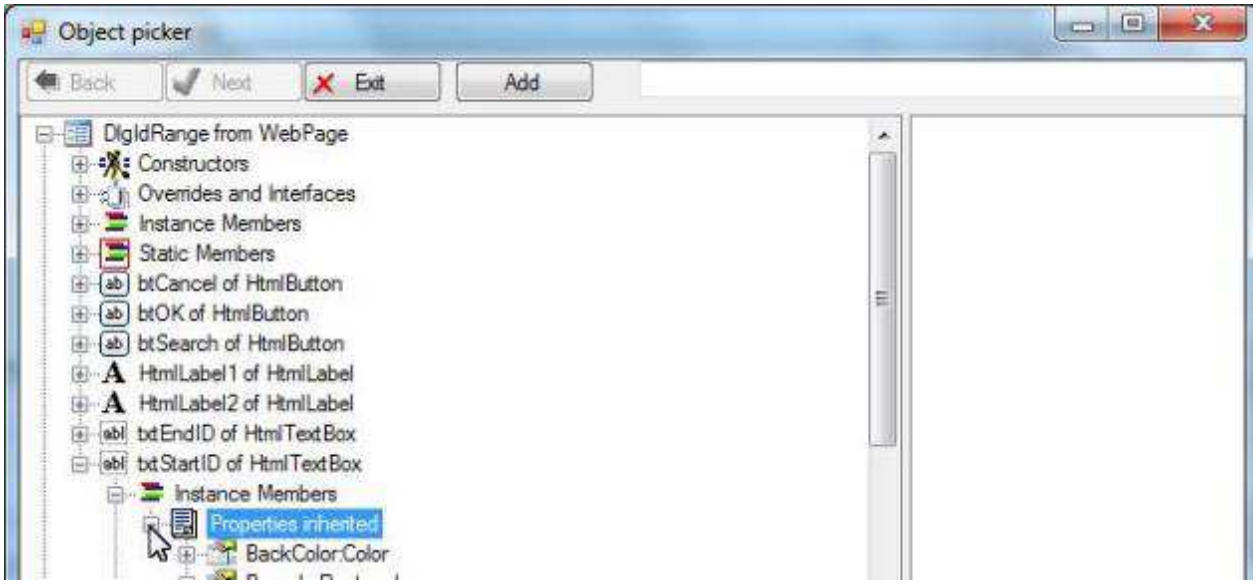
Let's execute the above method from the dialogue box page.

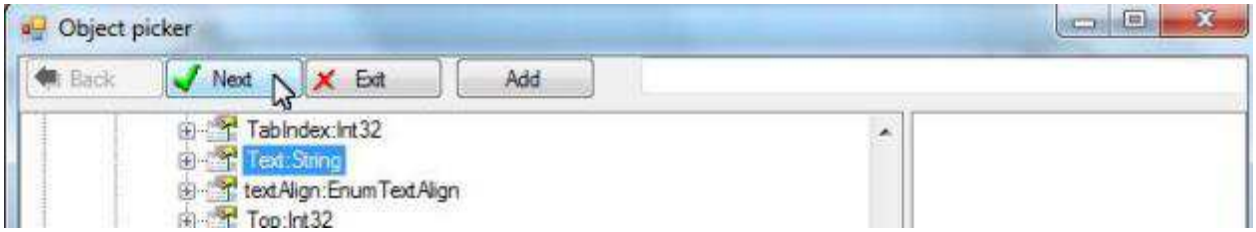
Select the search method under the default instance of page 1:

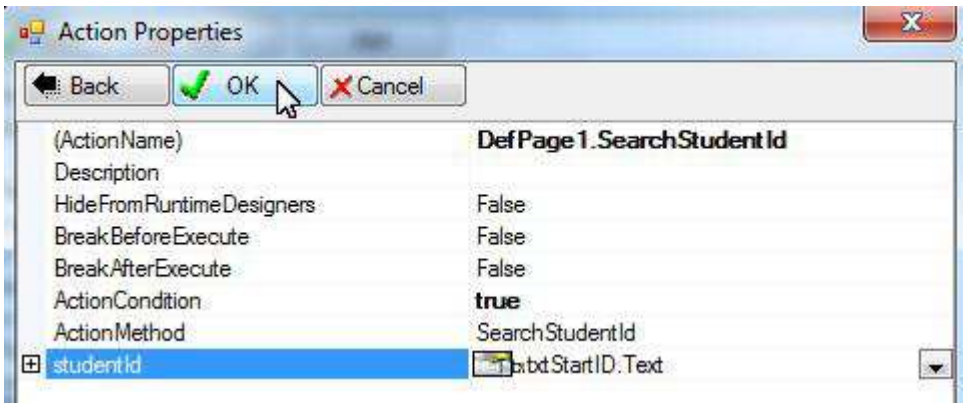


Set the parameter "studentId" to the Text property of the first text box:





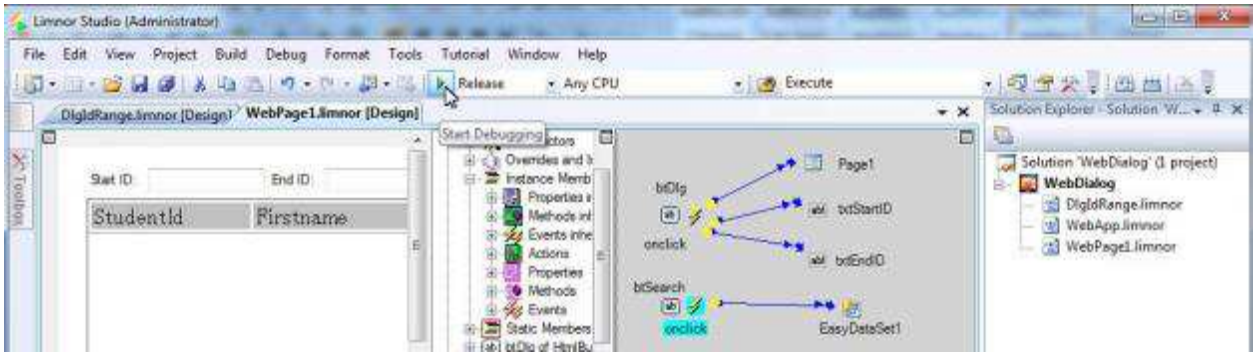




Note that the above action executes a method of Page1 from another web page and pass in a parameter from another page.

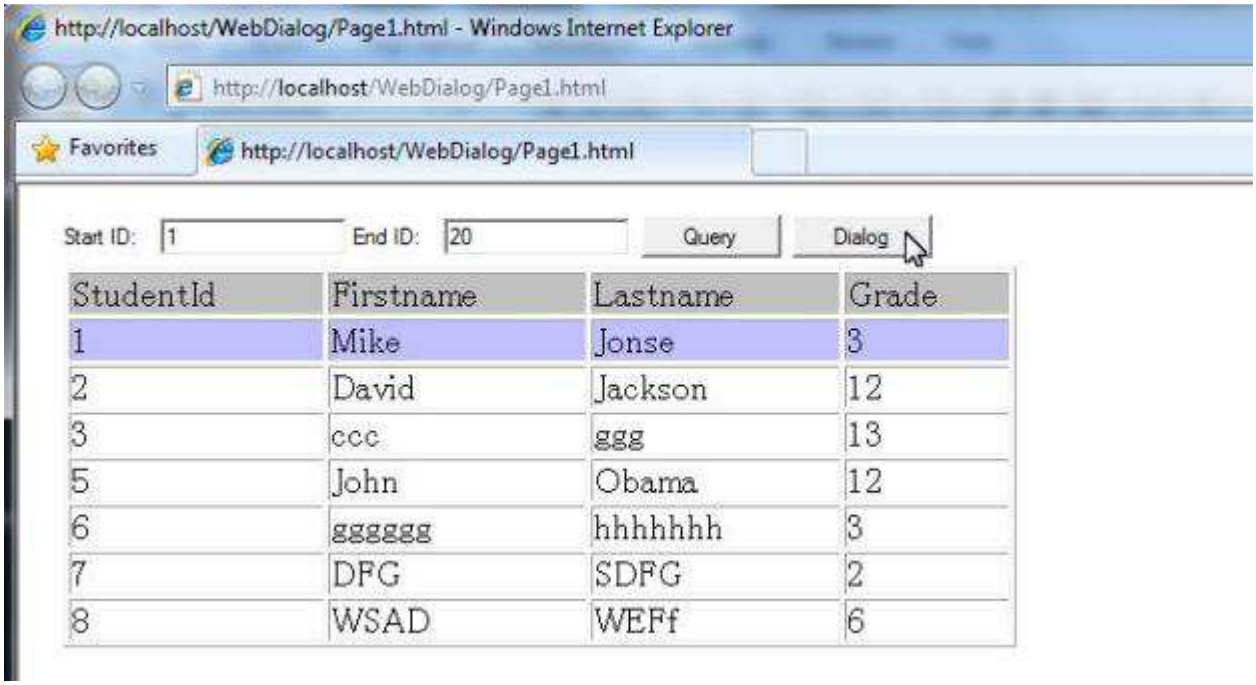
Test

Click the Run button to test it.



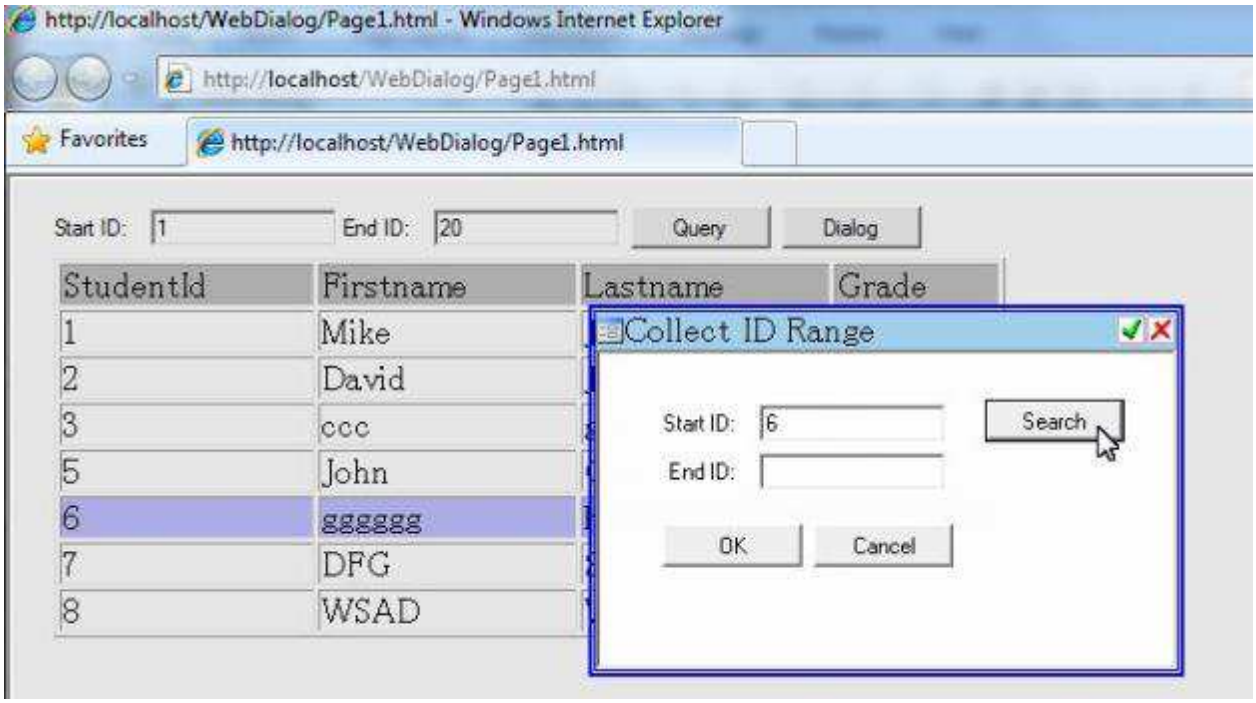
Enter Start ID and End ID and click Query button to load data from the database. We did not show this part of programming. For database programming, see
http://www.limnor.com/support/webDatabaseProgramming.pdf

Click Dialog button to launch the dialogue box.



The dialogue box appears. Enter a value for Start ID. Click the Search button to do search on the parent page.

We can see that the record corresponding to the Start ID value on the dialogue box becomes the current record on the parent web page.

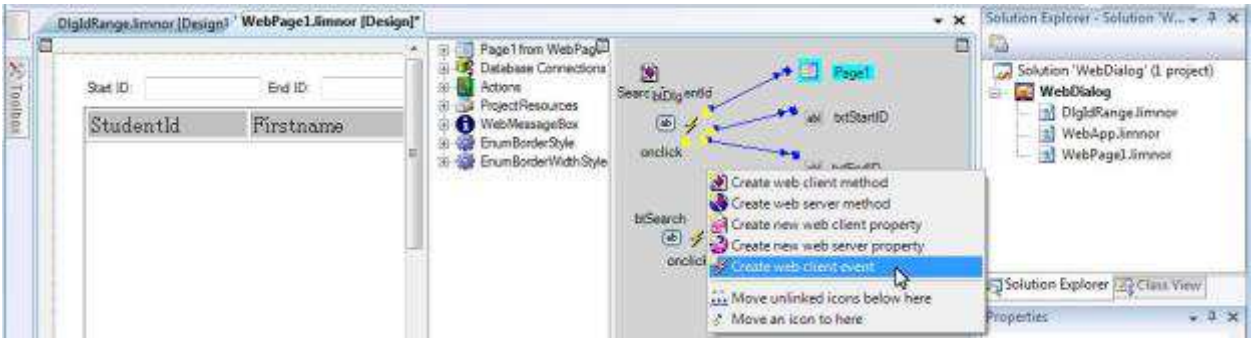


Trigger events across web pages

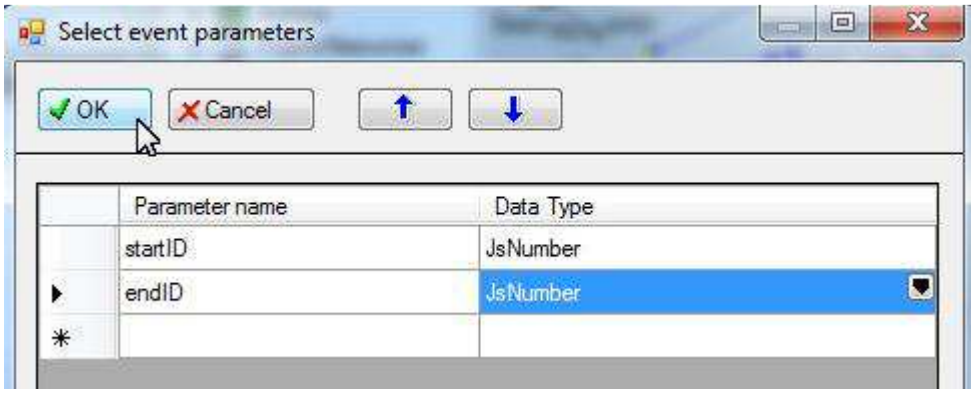
Only client methods can be executed directly across web pages. Suppose we want to do database query, which is a server operation, from another web page, we may publish such operations via client events. An event defined on one web page may be triggered from other web pages.

Publish functions via client events

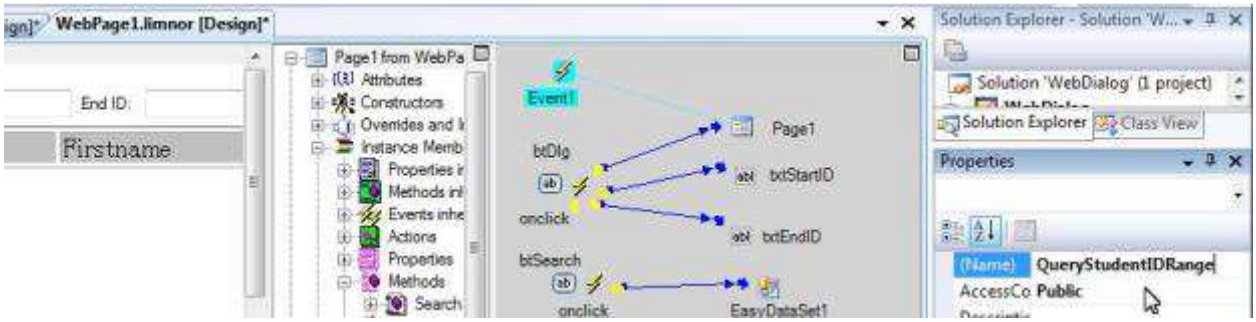
The parent page has an EasyDataSet for doing database query. We may publish the query to other pages by creating client events.



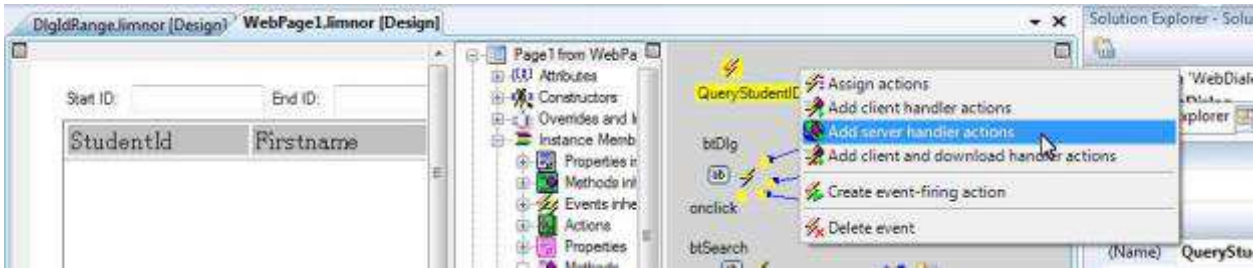
Add two parameters to the new event for database querying because the query we defined uses two parameters.



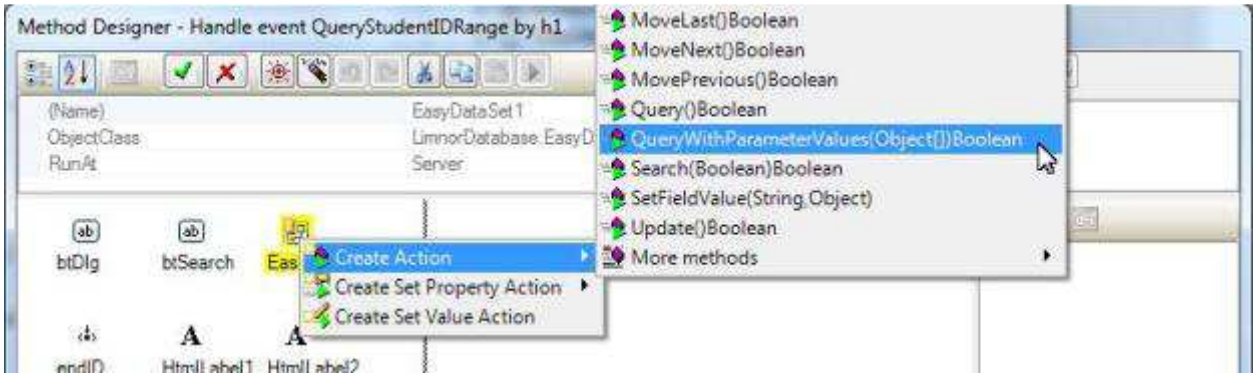
Name the event “QueryStudentIDRange”.



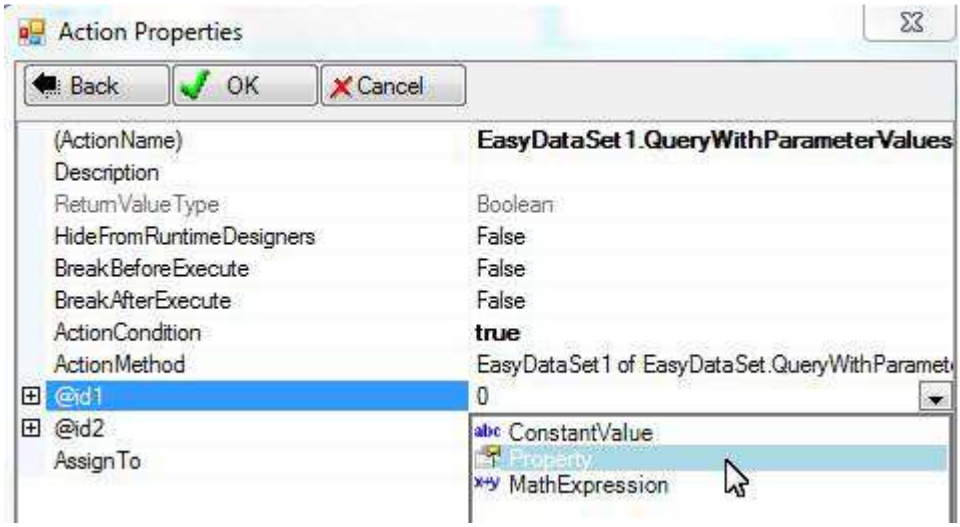
We want to do a QueryWithParameterValues action at this event. Because we want to use event parameters, we need to create an event handler method. Because QueryWithParameterValues is a server action, we need to create a server event handler:

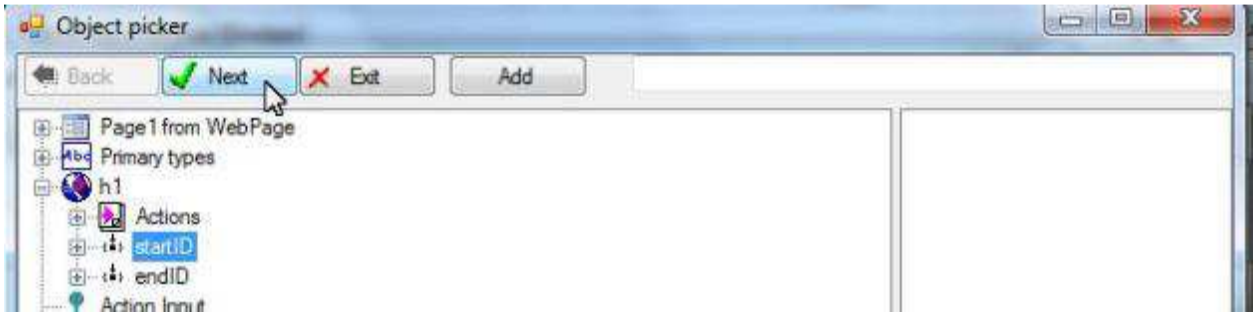


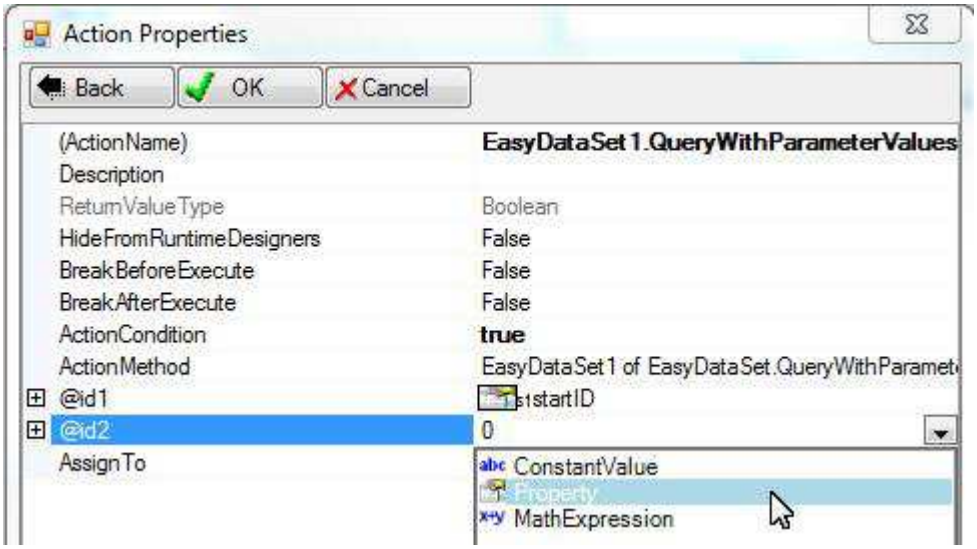
Create a QueryWithParameterValues action:



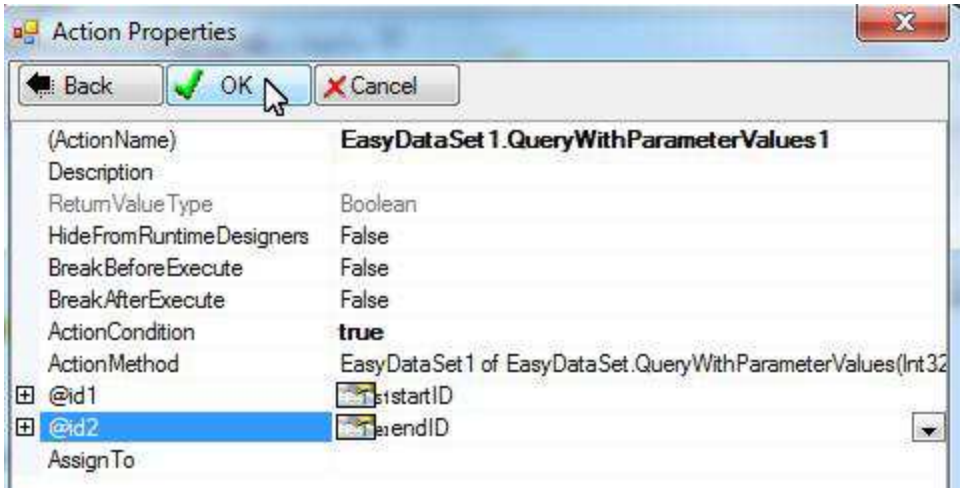
Use the event parameters for the query parameters:

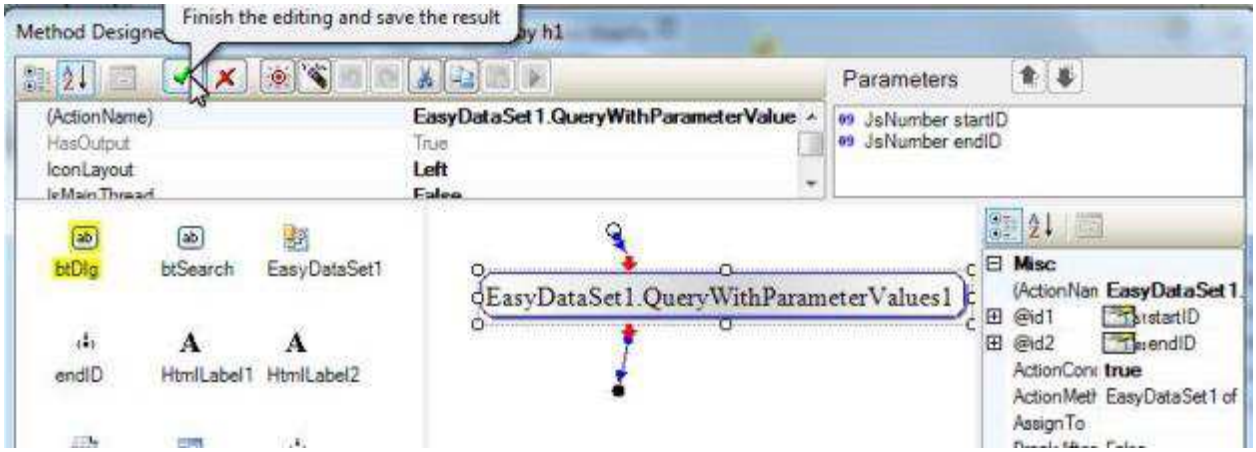


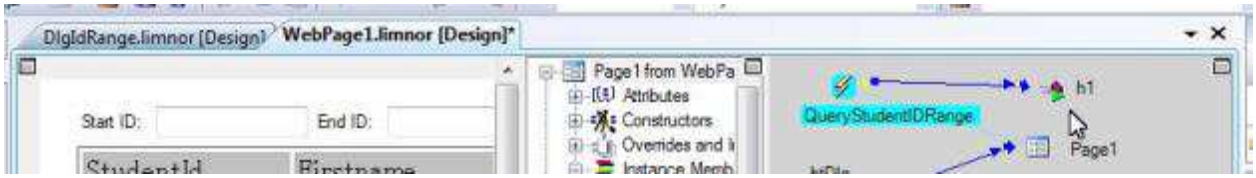



Action Properties
Back
OK
Cancel
(ActionName)
EasyDataSet1.QueryWithParameterValues
Description
ReturnValueType
Boolean
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
EasyDataSet1 of EasyDataSet.QueryWithParamet
@id1
startID
@id2
0
AssignTo
ConstantValue
Property
MathExpression


Object picker
Back
Next
Exit
Add
Page1 from WebPage
Primary types
h1
Actions
startID
endID
JsNumber startID

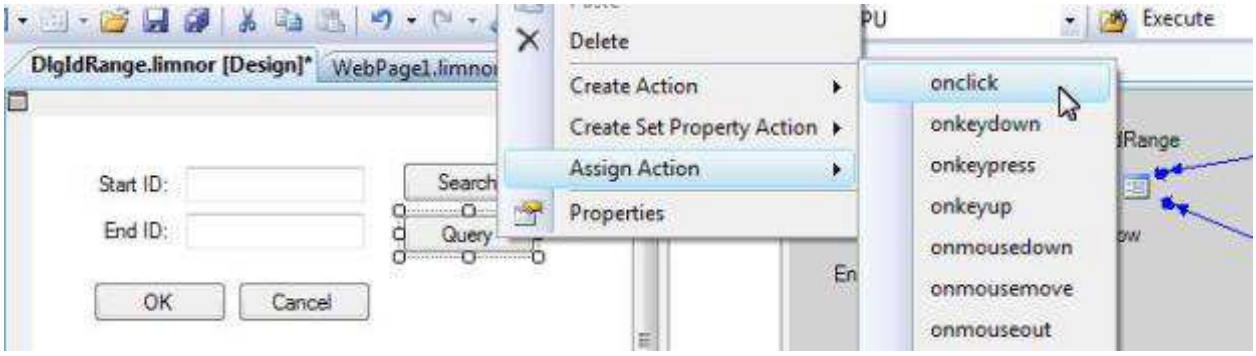

Action Properties
Back
OK
Cancel
(ActionName)
EasyDataSet1.QueryWithParameterValues1
Description
ReturnValueType
Boolean
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
EasyDataSet1 of EasyDataSet.QueryWithParameterValues(Int32
@id1
startID
@id2
endID
AssignTo



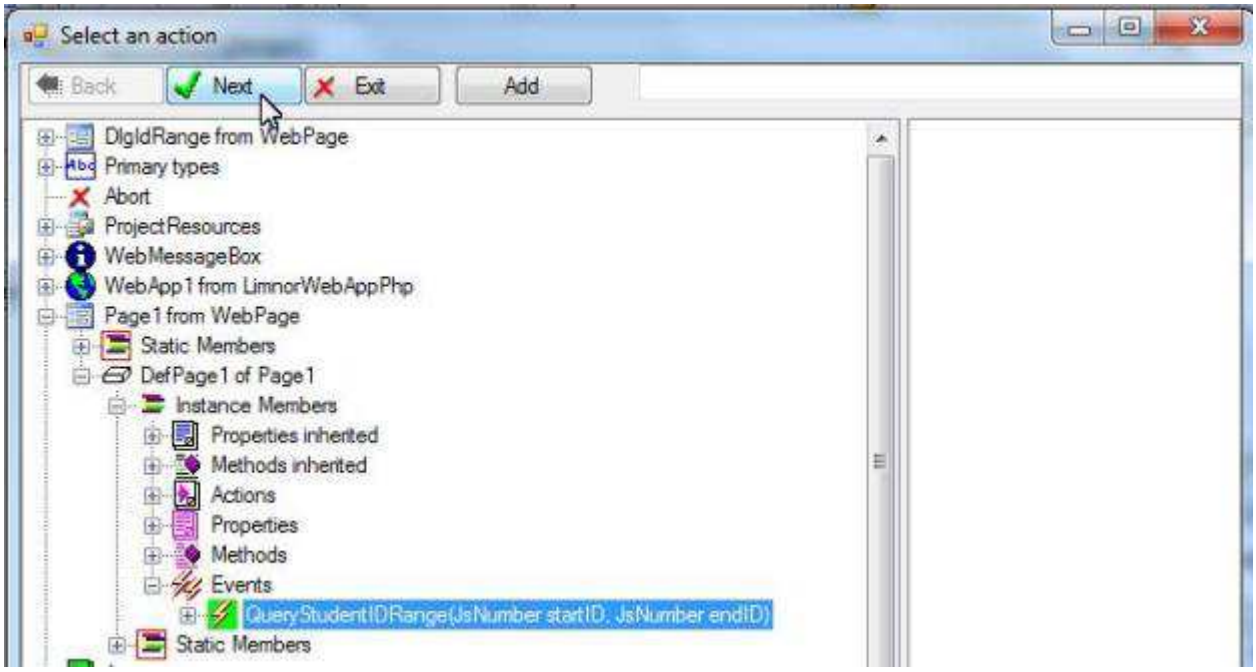


Trigger events via default page instance

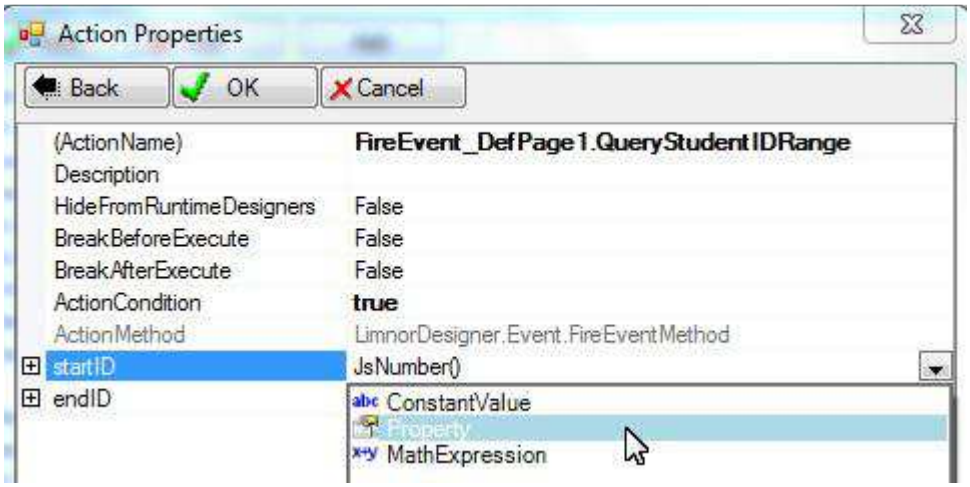
On the dialogue box page, we use a button to trigger the event on its parent page:

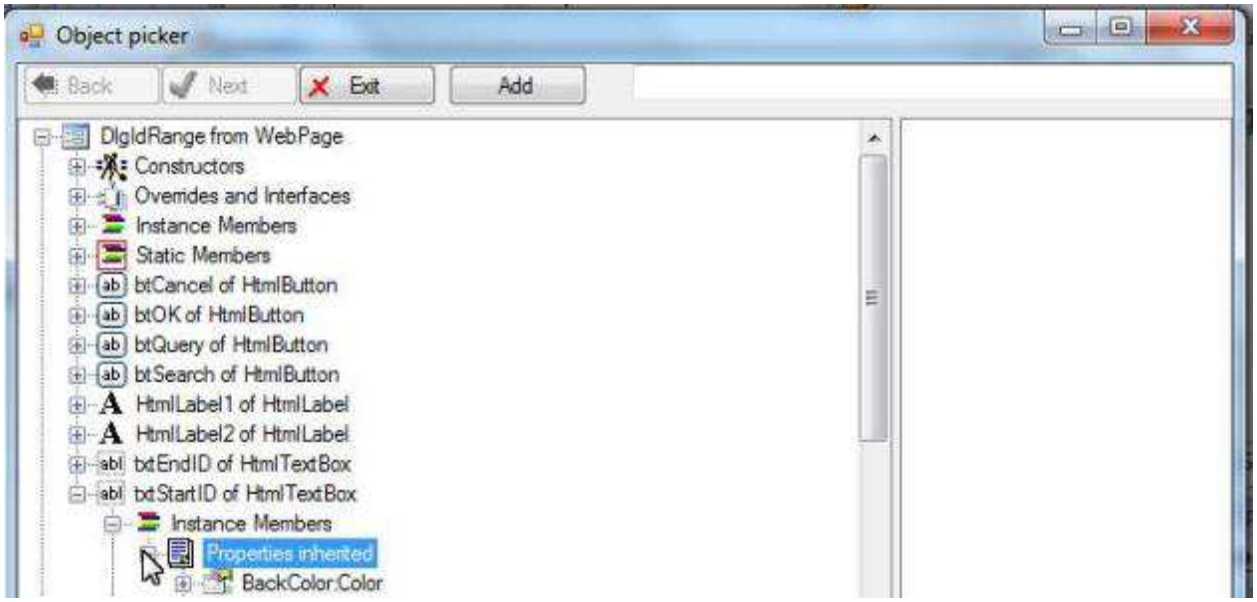


Select the event from the default instance of web page 1:

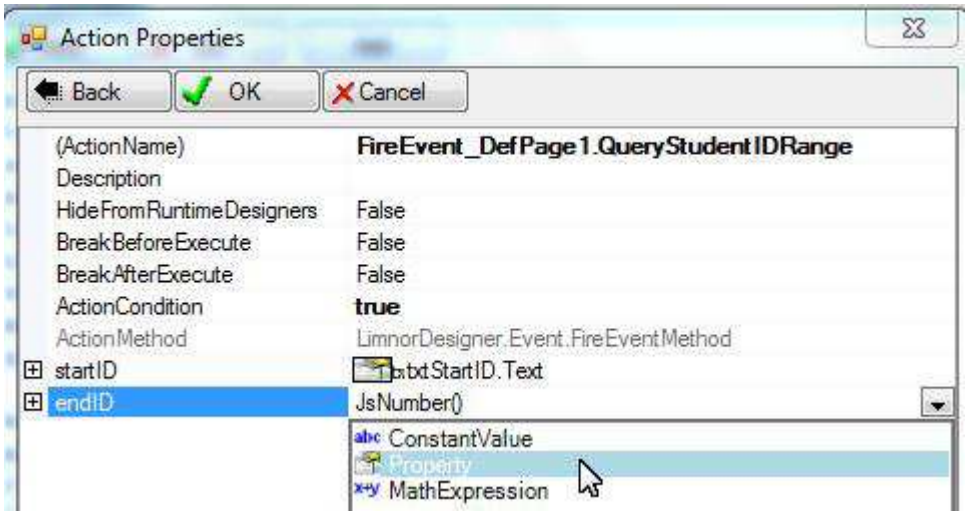


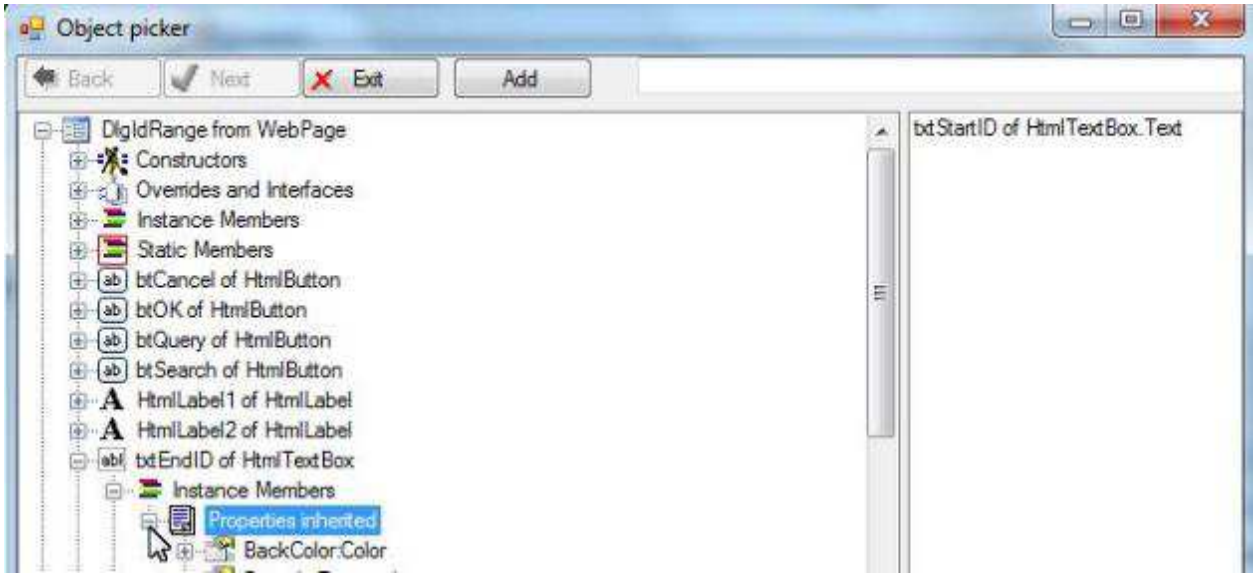
Use the values from the text boxes on the dialogue box page as the event parameters:

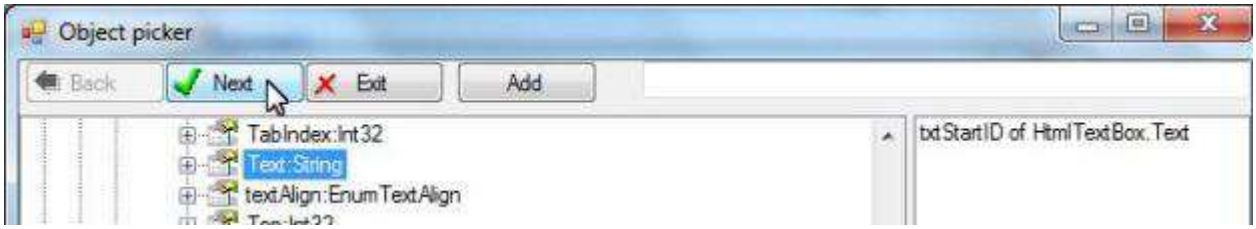


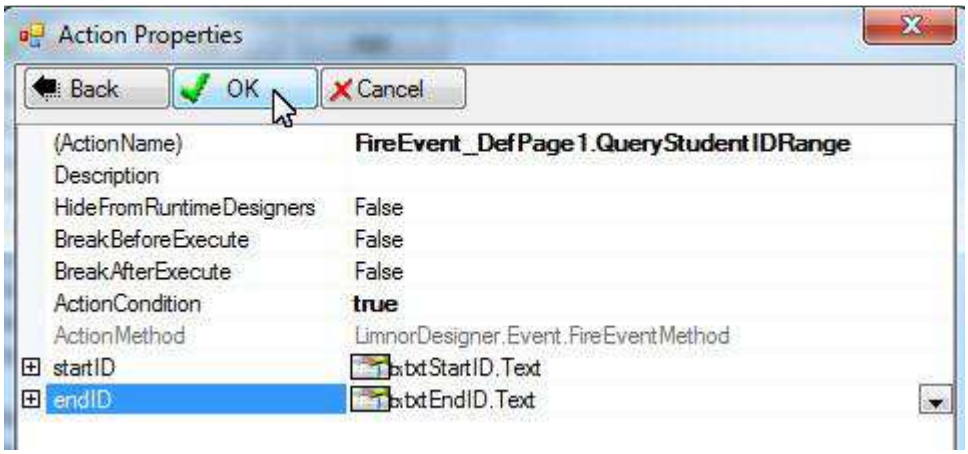

Object picker
Back
Next
Exit
Add
DlgIdRange from WebPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
btCancel of HtmlButton
btOK of HtmlButton
btQuery of HtmlButton
btSearch of HtmlButton
HtmlLabel1 of HtmlLabel
HtmlLabel2 of HtmlLabel
txtEndID of HtmlTextBox
txtStartID of HtmlTextBox
Instance Members
Properties inherited
BackColor:Color


Object picker
Back
Next
Exit
Add
TabIndex:Int32
Text:String
textAlign:EnumTextAlign

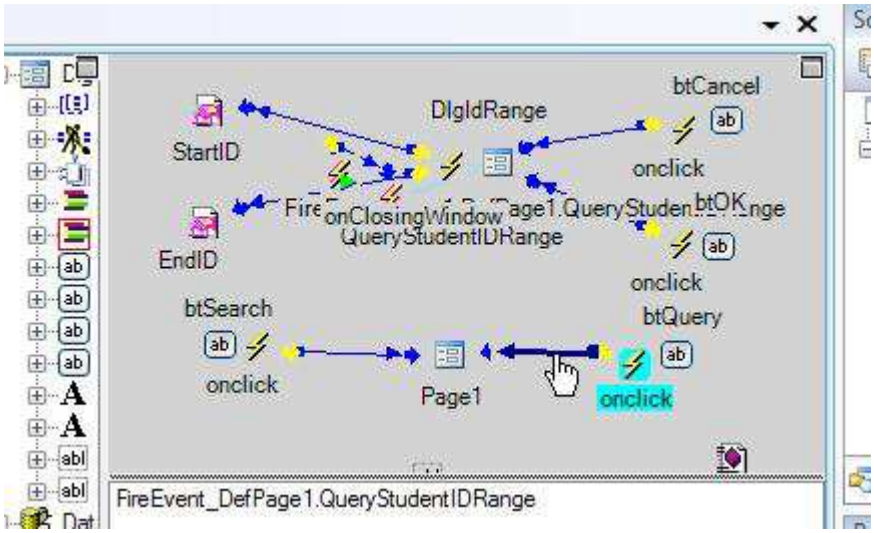

Action Properties
Back
OK
Cancel
(ActionName)
FireEvent_DefPage1.QueryStudentIDRange
Description
HideFromRuntimeDesigners
False
BreakBeforeExecute
False
BreakAfterExecute
False
ActionCondition
true
ActionMethod
LimnorDesigner.Event.FireEventMethod
startID
txtStartID.Text
endID
JsNumber()
ConstantValue
Property
MathExpression





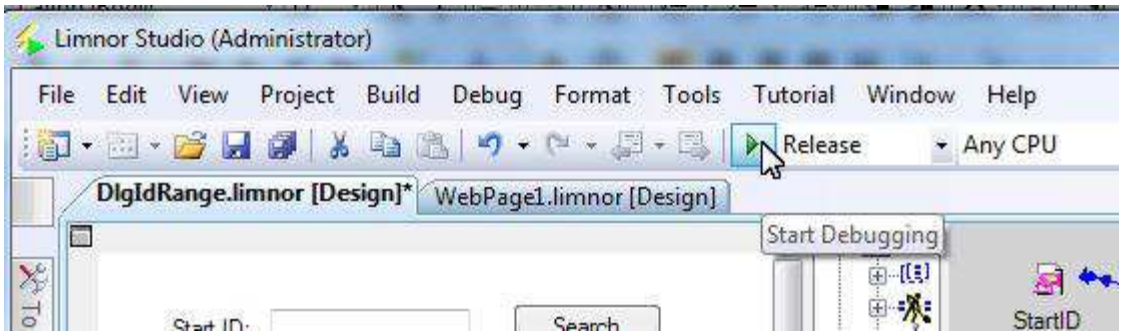


The above action triggers QueryStudentIDrange event on Page 1 from the dialogue box page, using the values from the dialogue box page.

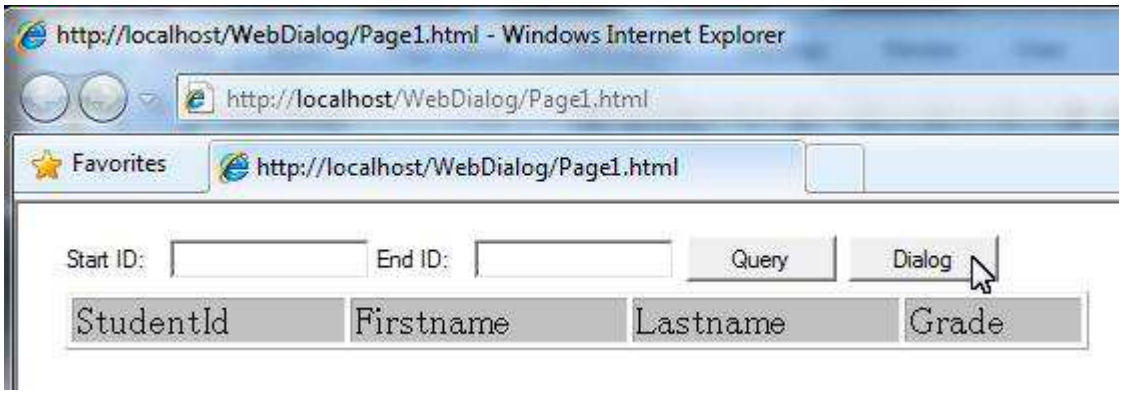


Test

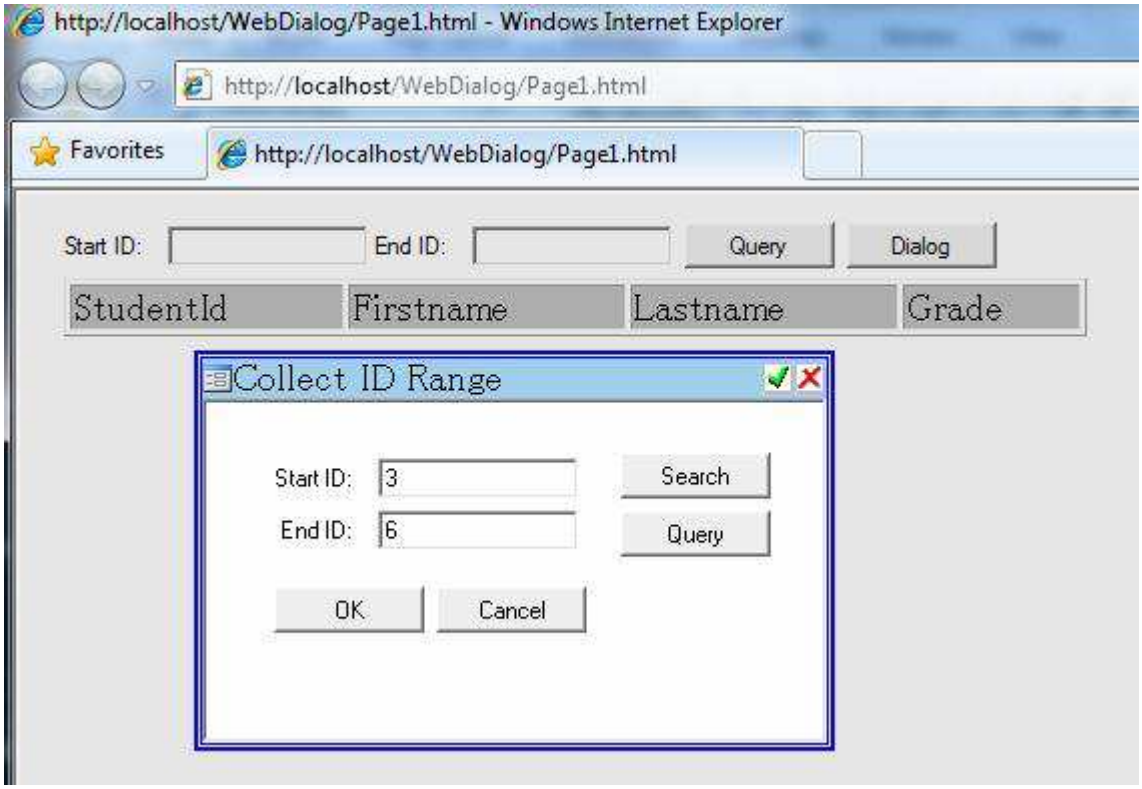
Click the Run button to launch the Start Page:



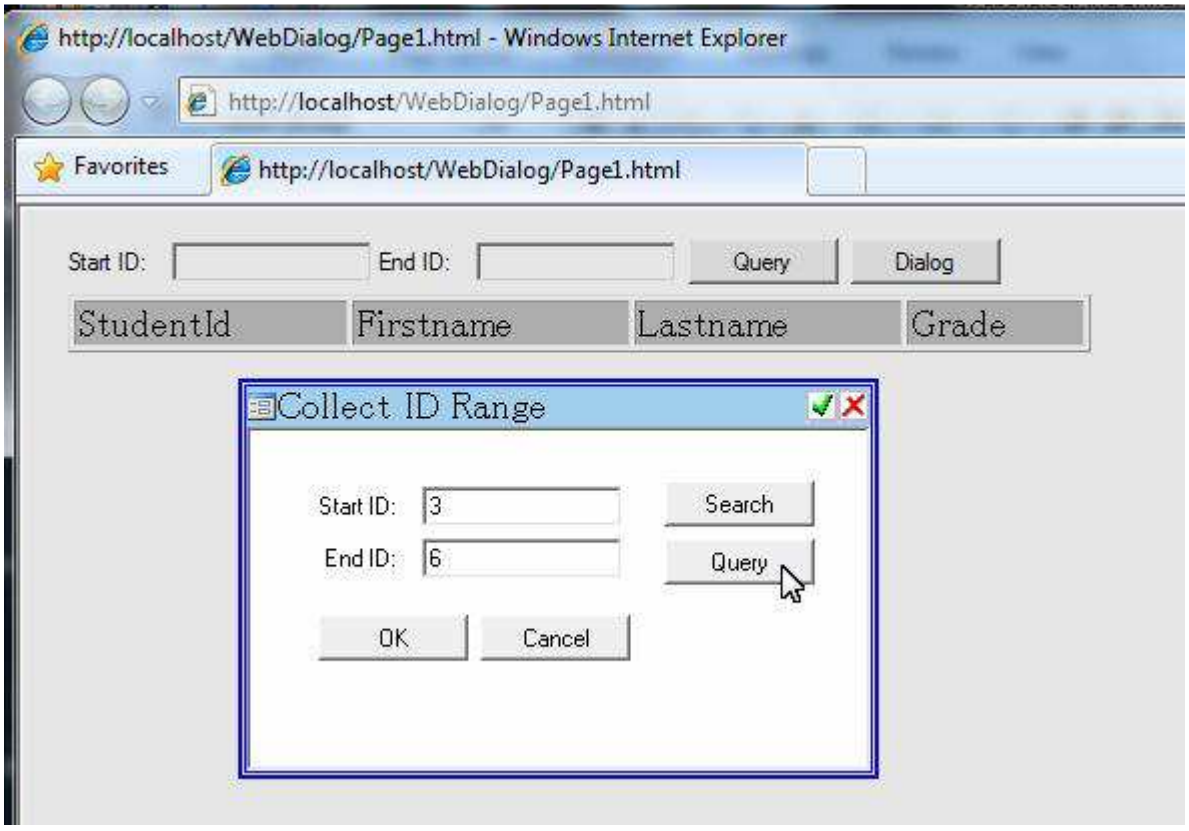
Click the Dialog button to launch the dialogue box:



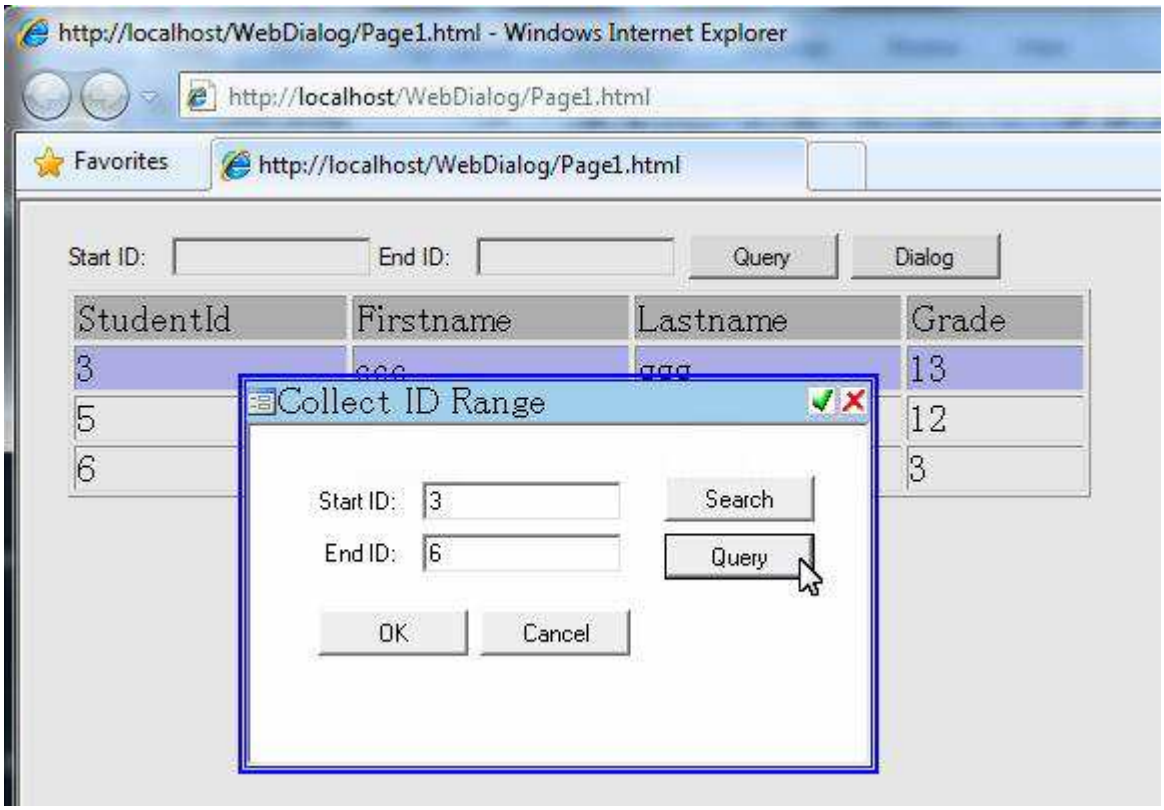
Enter start ID and end ID on the dialogue box:



Click the Query button:



We can see that the query results appear on the parent page:



This sample shows the triggering of web page events across web pages. In this sample, one server event handler is assigned to the event. Many actions and event handlers, client side and server side, can be assigned to the event, if needed.

Creating events can avoid duplicated programming. For example, suppose you have two buttons both trigger the exactly same set of actions and event handlers, but using different parameters. Instead of creating duplicated event handlers, one event can be create and assigned one set of actions and event handlers to the event. Each button is assigned one action to trigger the event, using proper event parameters.

Feedback

Please send your feedback to support@limnor.com