Database Programming in Tiers

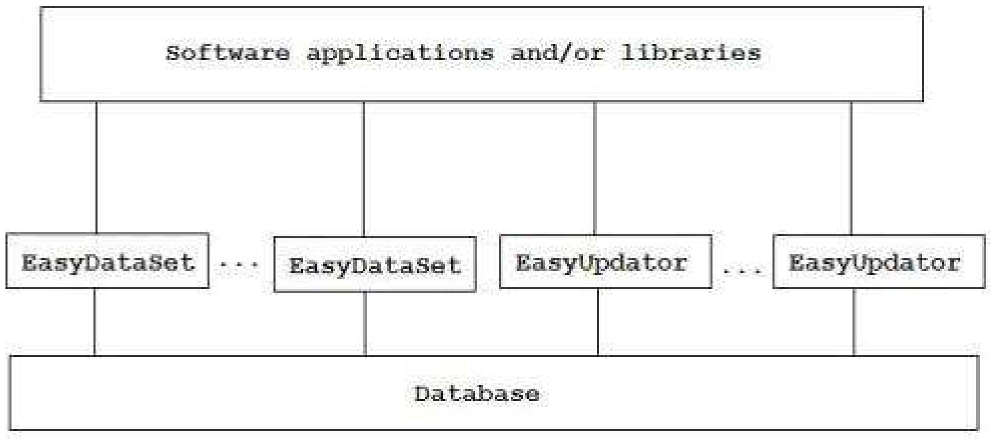
Contents

Introduction

For designing large software systems, designing software modules in layers may provide benefits in code reusing and maintenance. See http://en.wikipedia.org/wiki/Multitier_architecture , http://www.exforsys.com/tutorials/client-server/n-tier-client-server-architecture/1.html and other references.

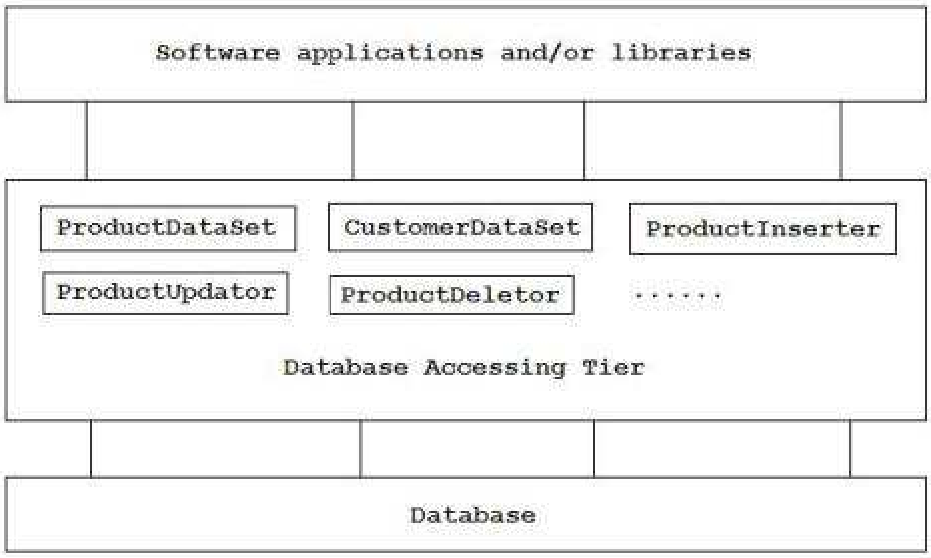
This document does not discuss multitier programming in general. It shows examples of using components, EasyDataSet and EasyUpdator, to develop a database accessing layer.

For s system not designed with layers, EasyDataSet and EasyUpdator are used directly.



Database application without tiers

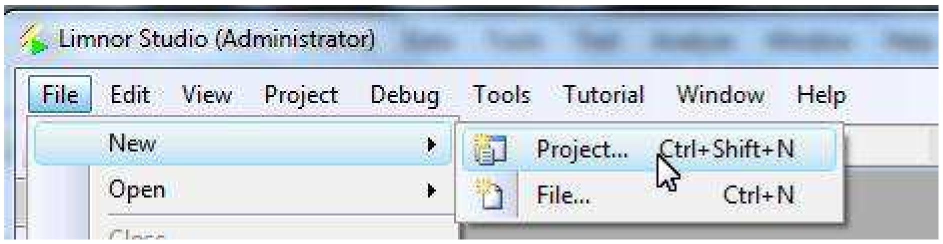
If software is designed in tiers then database accessing can be in a separate layer.

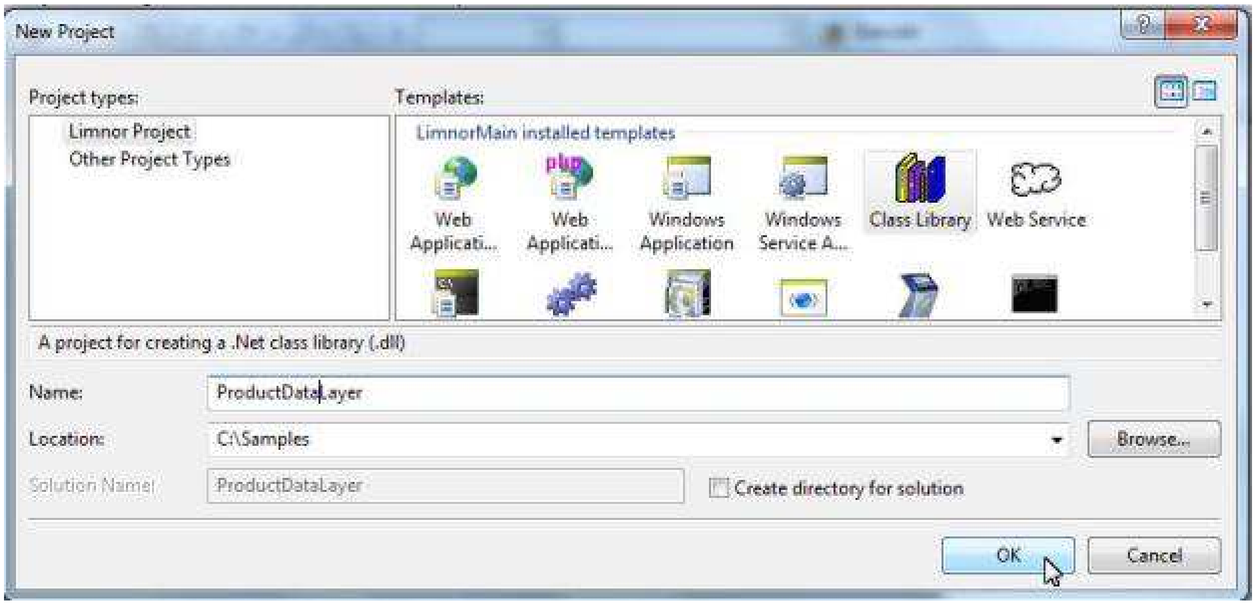


Software with tiers

Each box in the "Database Accessing Tier" represents one class derived from EasyDataSet or EasyUpdator. If the database accessing layer is in a separate DLL file then the DLL can be used for all applications. It can be used for standalone applications and also for web applications.

Create Class Library Project





A project for creating a .Net class library (.dll)

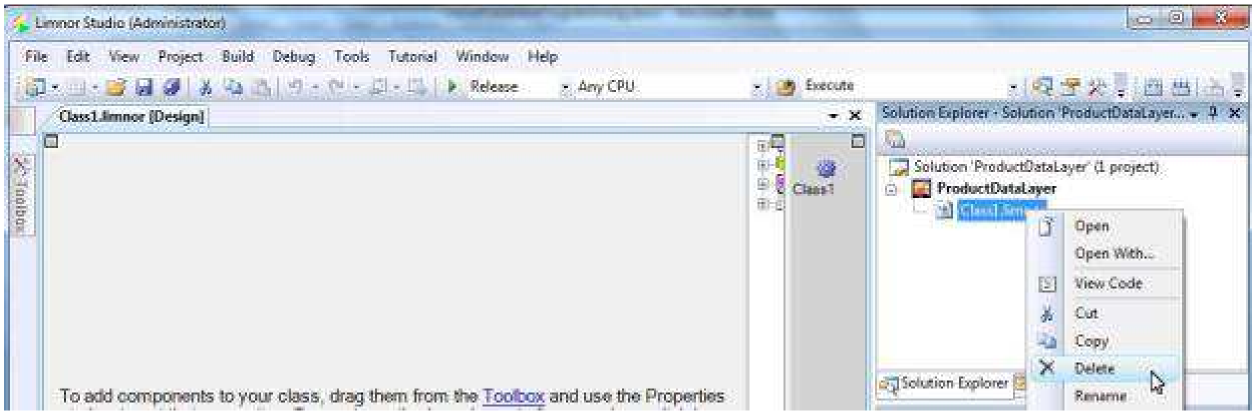
Name: ProductDataLayer

Location: C:\Samples

Solution Name: ProductDataLayer

Create directory for solution

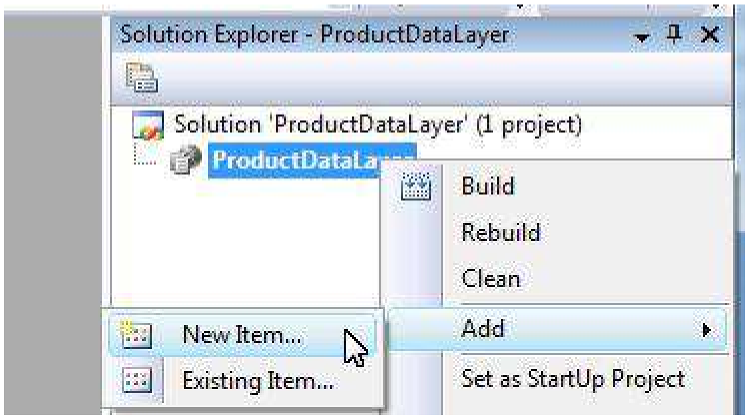
Delete the default class, Class1, because we do not need:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window Help
Release
Any CPU
Execute
Class1.limnor [Design]
Class1
Solution Explorer - Solution 'ProductDataLayer...
Solution 'ProductDataLayer' (1 project)
ProductDataLayer
Class1.limnor
Open
Open With...
View Code
Cut
Copy
Delete
Rename
Solution Explorer
Toolbox

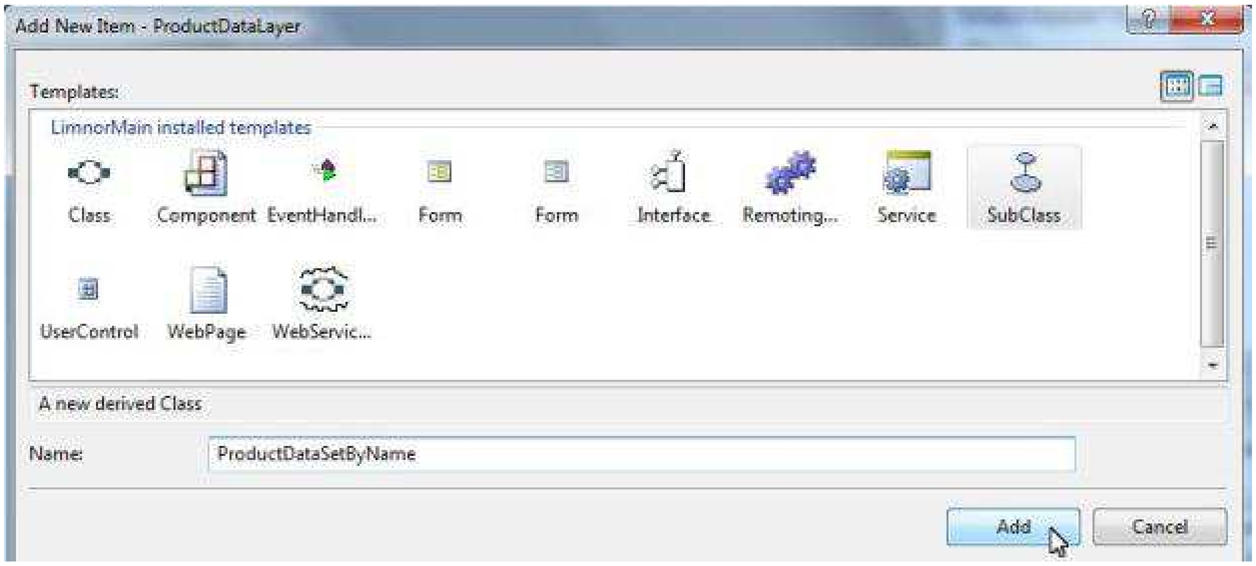
To add components to your class, drag them from the Toolbox and use the Properties

Derive from EasyDataSet

Suppose we want to provide product query functionality. We may derive a new class from EasyDataSet to provide such functionality.



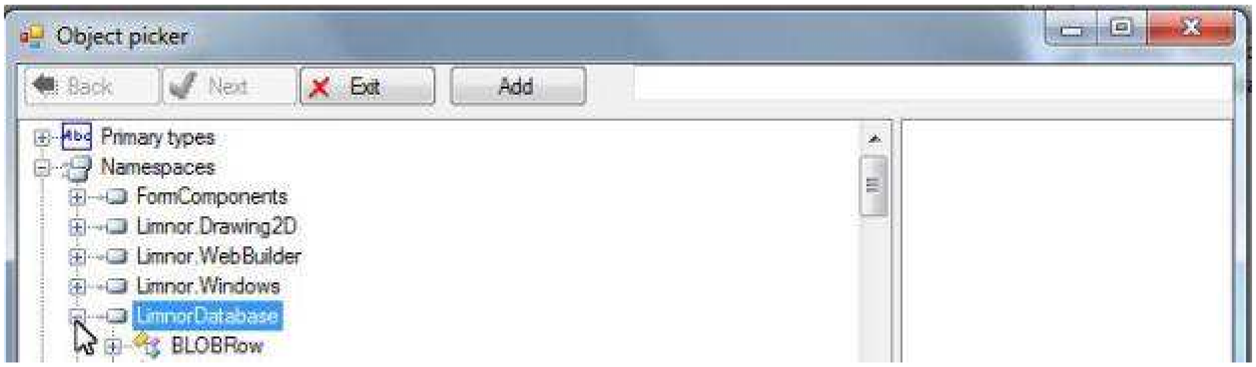
Name the new class ProductDataSetByName so that we know what the class is used for:

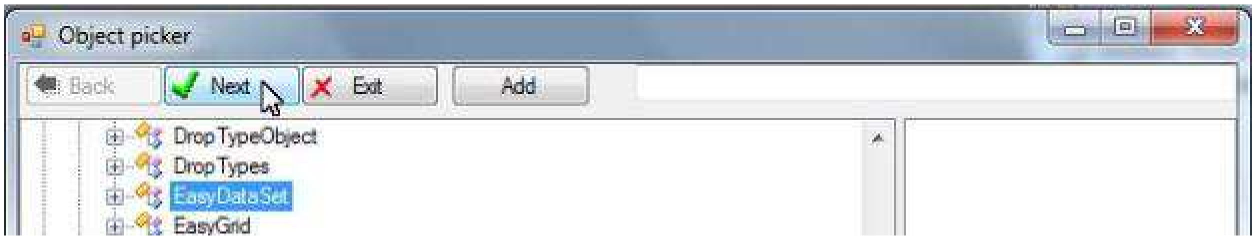


A new derived Class

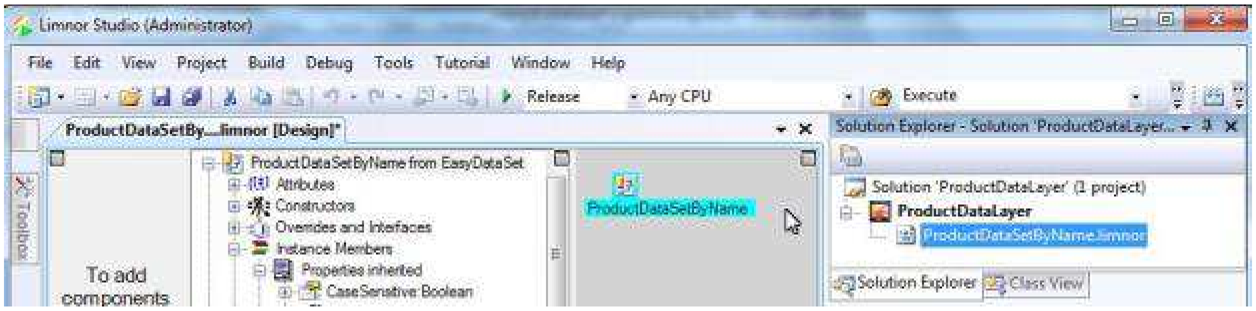
Name: ProductDataSetByName

Select EasyDataSet as the base class for the new class:

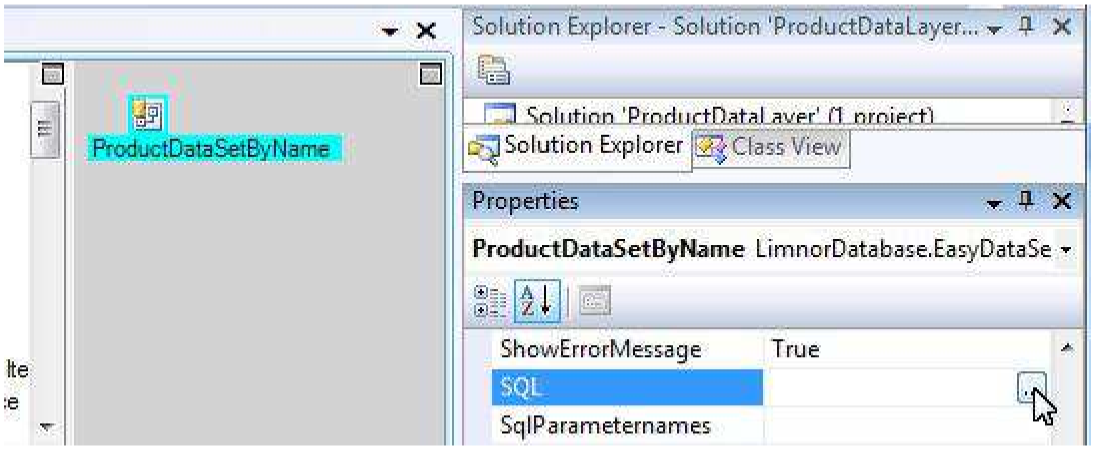




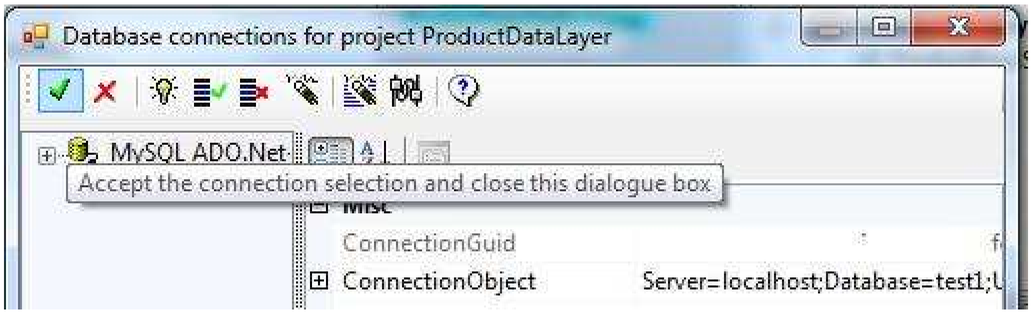
The new class is created:


Limnor Studio (Administrator)
File Edit View Project Build Debug Tools Tutorial Window Help
Release
Any CPU
Execute
ProductDataSetBy....limnor [Design]*
ProductDataSetByName from EasyDataSet
Attributes
Constructors
Overrides and Interfaces
Instance Members
Properties inherited
CaseSensitive Boolean
To add components
ProductDataSetByName
Solution Explorer - Solution 'ProductDataLayer...
Solution 'ProductDataLayer' (1 project)
ProductDataLayer
ProductDataSetByName.limnor
Solution Explorer
Class View

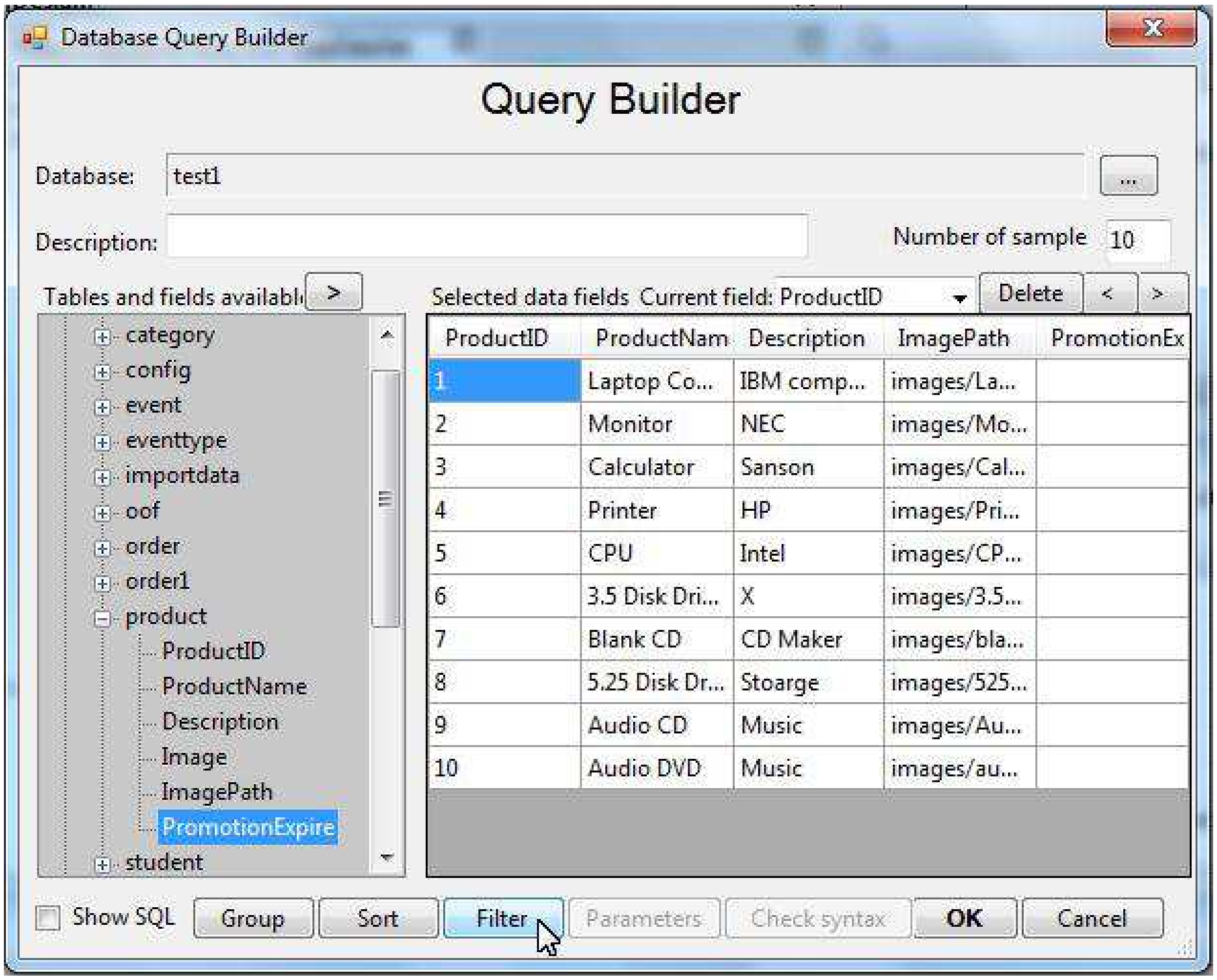
Set its SQL property to create desired database query:


ProductDataSetByName
Solution Explorer - Solution 'ProductDataLayer...
Solution 'ProductDataLayer' (1 project)
Solution Explorer
Class View
Properties
ProductDataSetByName LimnorDatabase.EasyDataSe
ShowErrorMessage True
SQL
SqlParameternames

Set the database connection:


Database connections for project ProductDataLayer
MySQL ADO.Net
Accept the connection selection and close this dialogue box
Misc
ConnectionGuid
ConnectionObject Server=localhost;Database=test1;U

Construct a query for searching products by name:



Database: test1

Number of sample: 10

ProductID	ProductNam	Description	ImagePath	PromotionEx
1	Laptop Co...	IBM comp...	images/La...	
2	Monitor	NEC	images/Mo...	
3	Calculator	Sanson	images/Cal...	
4	Printer	HP	images/Pri...	
5	CPU	Intel	images/CP...	
6	3.5 Disk Dri...	X	images/3.5...	
7	Blank CD	CD Maker	images/bla...	
8	5.25 Disk Dr...	Stoarge	images/525...	
9	Audio CD	Music	images/Au...	
10	Audio DVD	Music	images/au...	

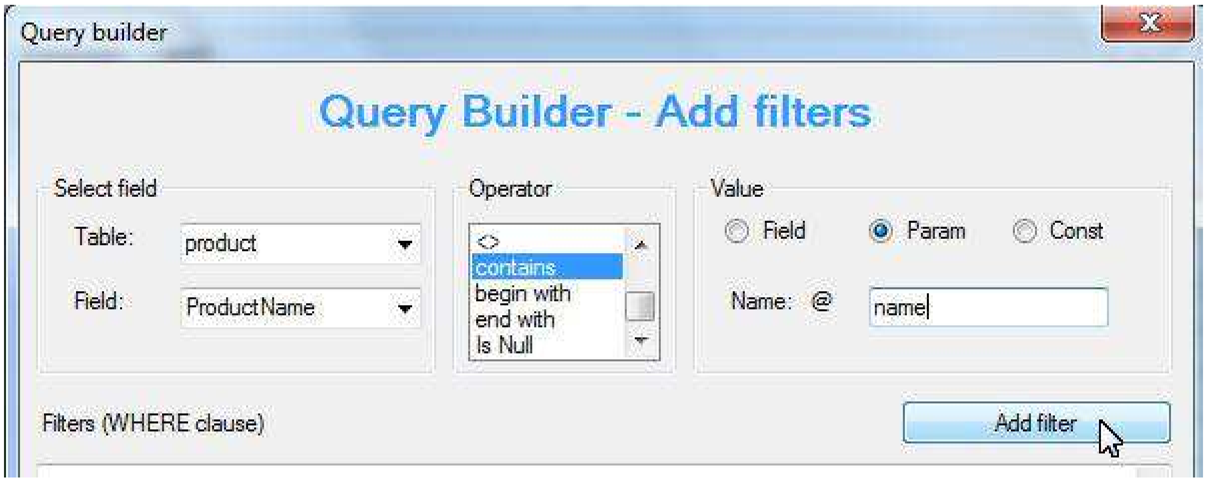


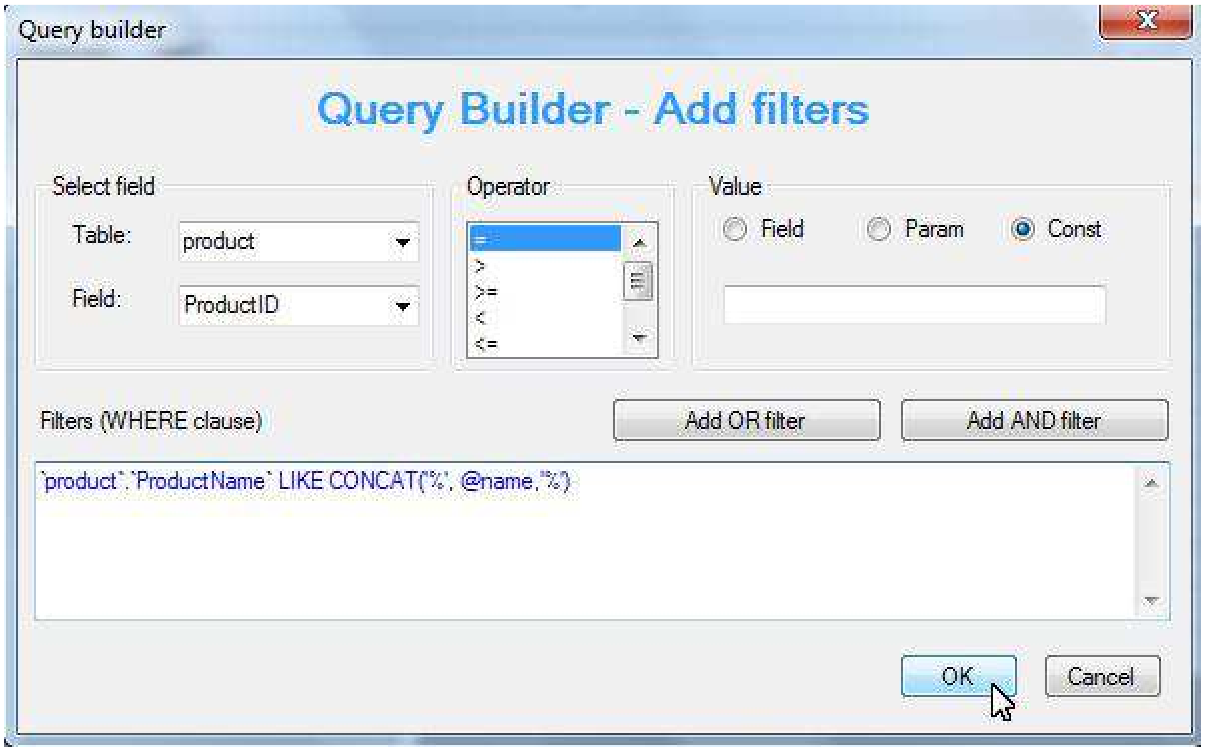
Table: product

Field: ProductName

Operator: contains

Value: Param

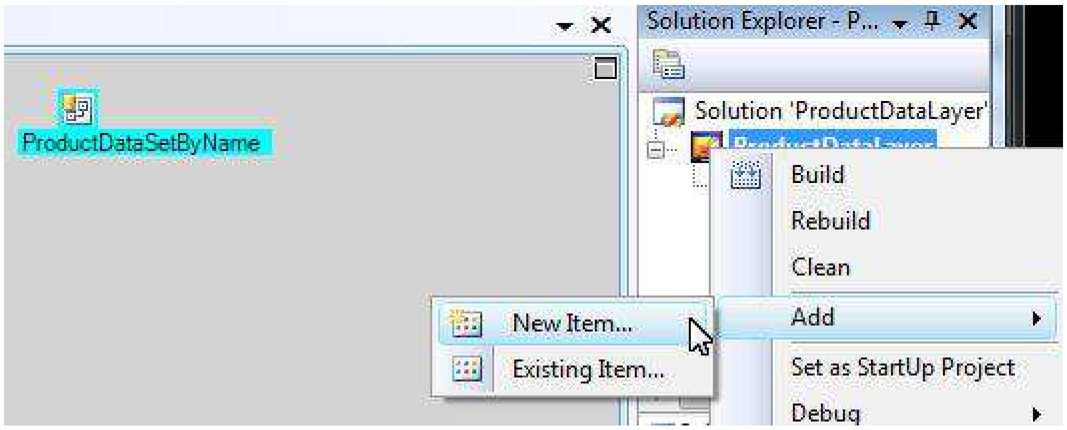
Name: @ name



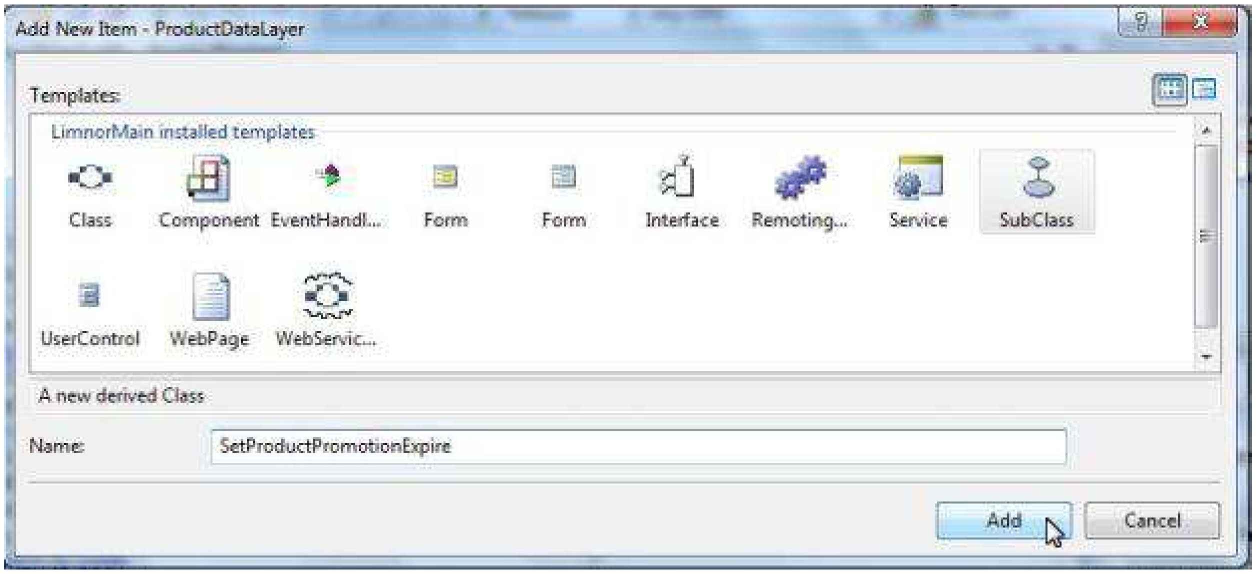
`product`.`ProductName` LIKE CONCAT('%', @name,'%')

Derive from EasyUpdator

Our sample Product table has a field, PromotionExpire. Suppose we want to provide a functionality of setting this field by productID. We'll derive a new class from EasyUpdator to do it.



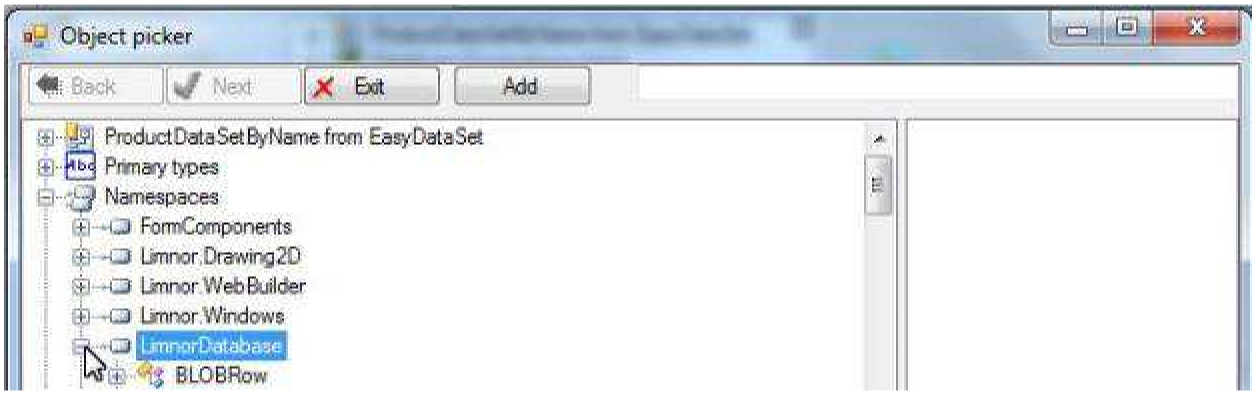
Select SubClass and give the new class a meaningful name:

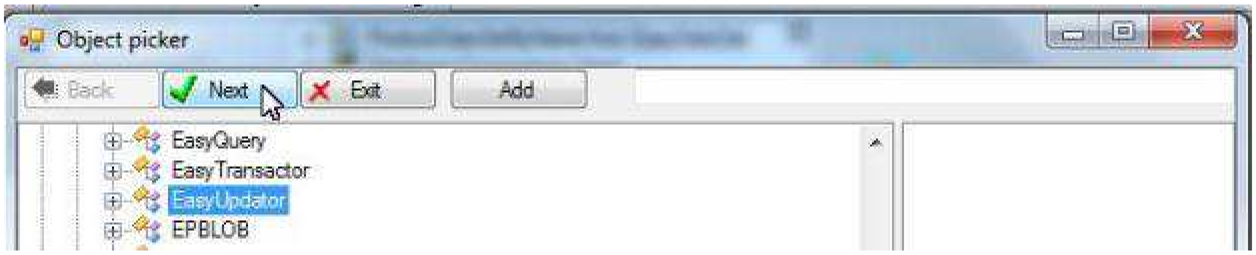


A new derived Class

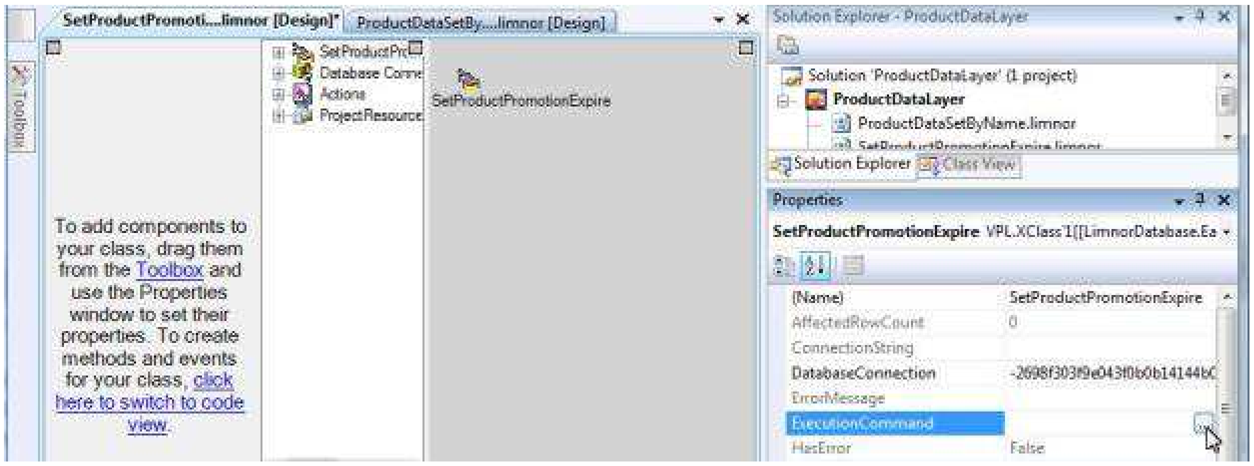
Name: SetProductPromotionExpire

Select EasyUpdator as the base class for the new class:





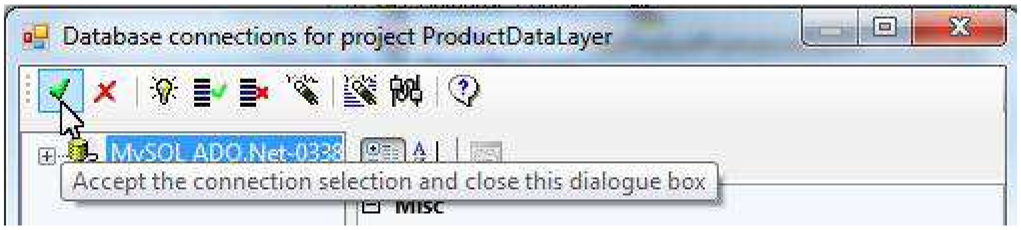
The new class appears. Set its ExecuteCommand property to update PromotionExpire field:


SetProductPromoti....limnor [Design]*
ProductDataSetBy....limnor [Design]
Toolbox
SetProductPr
Database Conne
Actions
ProjectResource
SetProductPromotionExpire
Solution Explorer - ProductDataLayer
Solution 'ProductDataLayer' (1 project)
ProductDataLayer
ProductDataSetByName.limnor
Solution Explorer
Class View
Properties
SetProductPromotionExpire VPL.XClass'1[[LimnorDatabase.Ea

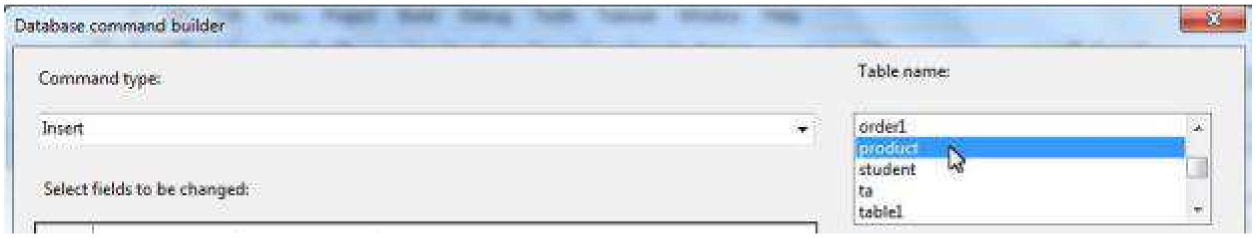
To add components to your class, drag them from the Toolbox and use the Properties window to set their properties. To create methods and events for your class, click here to switch to code view.

(Name)	SetProductPromotionExpire
AffectedRowCount	0
ConnectionString	
DatabaseConnection	-2698f303f9e043f0b0b14144b(
ErrorMessage	
ExecutionCommand	
HasError	False

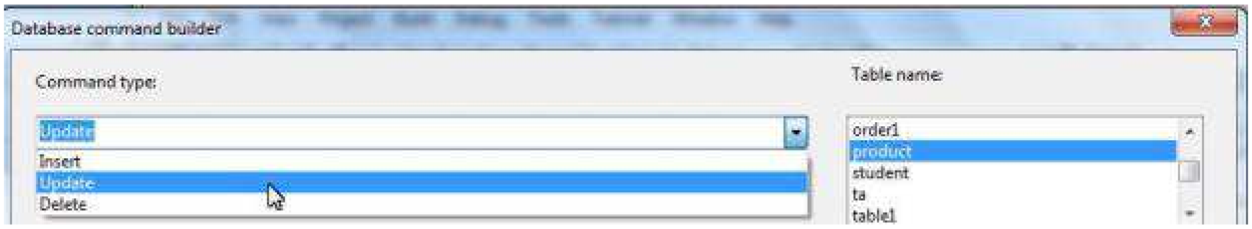
Select the database connection:


Database connections for project ProductDataLayer
MySQL ADO.Net-0328
Accept the connection selection and close this dialogue box
Misc

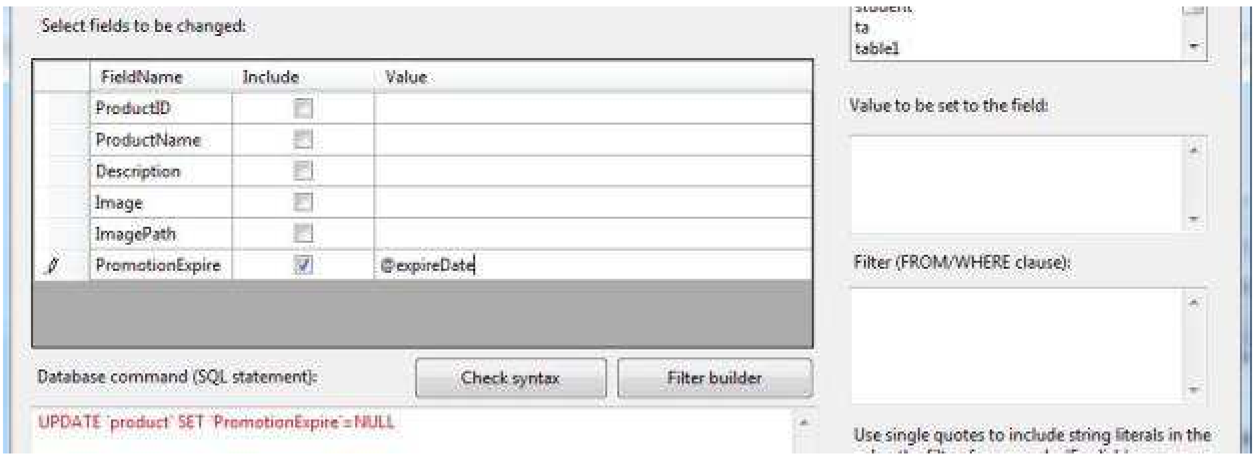
Select Product table:



Select Update command:



Check field PromotionExpire and enter a parameter name, @expireDate, for its value:



Select fields to be changed:

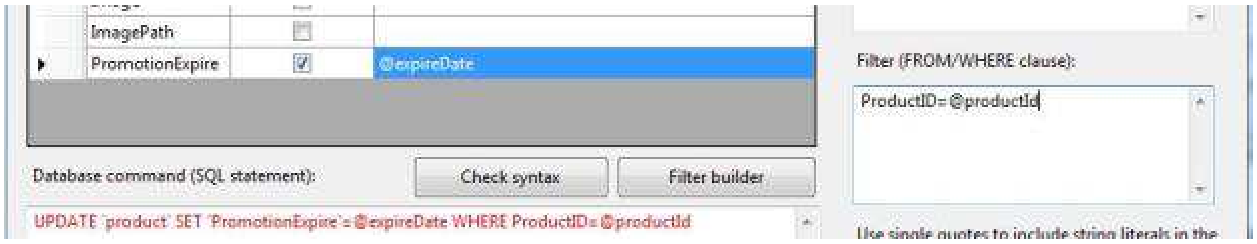
	FieldName	Include	Value
	ProductID	☐	
	ProductName	☐	
	Description	☐	
	Image	☐	
	ImagePath	☐	
✎	PromotionExpire	✓	@expireDate

Database command (SQL statement):

```
UPDATE `product` SET `PromotionExpire`=NULL
```

Check syntax
Filter builder
student
ta
table1
Value to be set to the field:
Filter (FROM/WHERE clause):
Use single quotes to include string literals in the

Enter a filter for the command:



	FieldName	Include	Value
	ImagePath	☐	
▸	PromotionExpire	✓	@expireDate

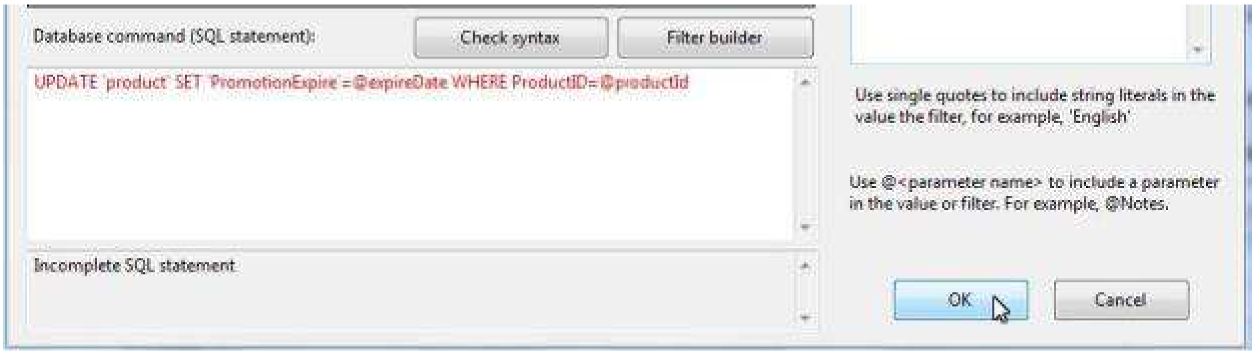
Filter (FROM/WHERE clause):

```
ProductID=@productId
```

Database command (SQL statement):

```
UPDATE `product` SET `PromotionExpire`=@expireDate WHERE ProductID=@productId
```

Click OK to finishing creating this command:



Database command (SQL statement):

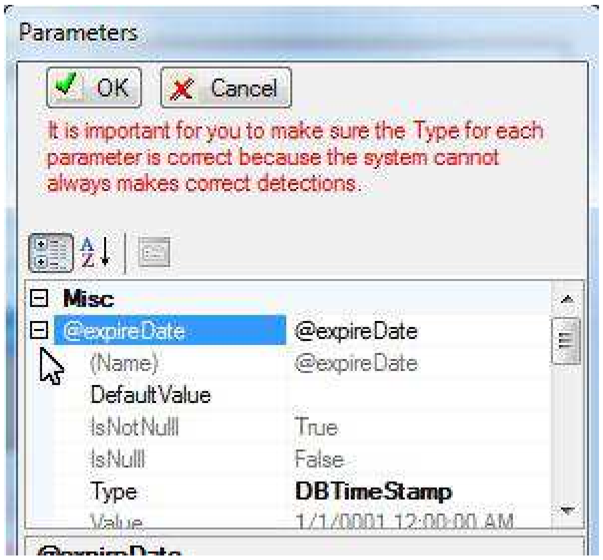
```
UPDATE `product` SET `PromotionExpire`=@expireDate WHERE ProductID=@productId
```

Incomplete SQL statement

Use single quotes to include string literals in the value the filter, for example, 'English'

Use @<parameter name> to include a parameter in the value or filter. For example, @Notes.

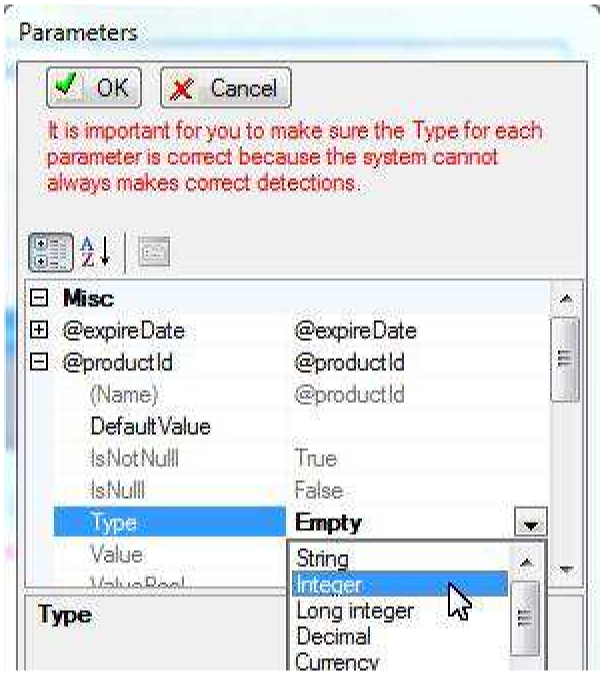
Examine the data types for the parameters to make sure they are correct:


Parameters

OK Cancel

It is important for you to make sure the Type for each parameter is correct because the system cannot always makes correct detections.

Misc	
@expireDate	@expireDate
(Name)	@expireDate
DefaultValue	
IsNotNulll	True
IsNulll	False
Type	DBTimeStamp
Value	1/1/0001 12:00:00 AM


Parameters

OK Cancel

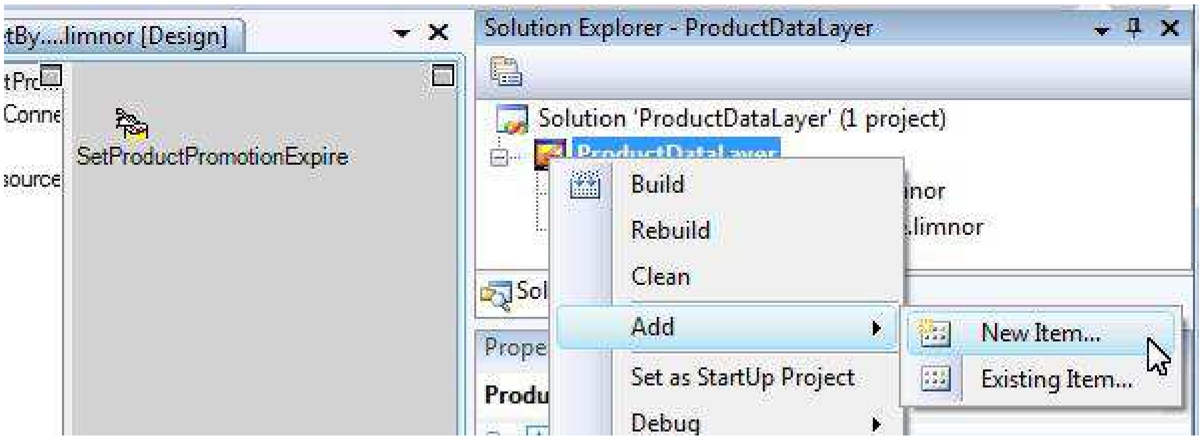
It is important for you to make sure the Type for each parameter is correct because the system cannot always makes correct detections.

Misc	
@expireDate	@expireDate
@productId	@productId
(Name)	@productId
DefaultValue	
IsNotNulll	True
IsNulll	False
Type	Empty
Value	

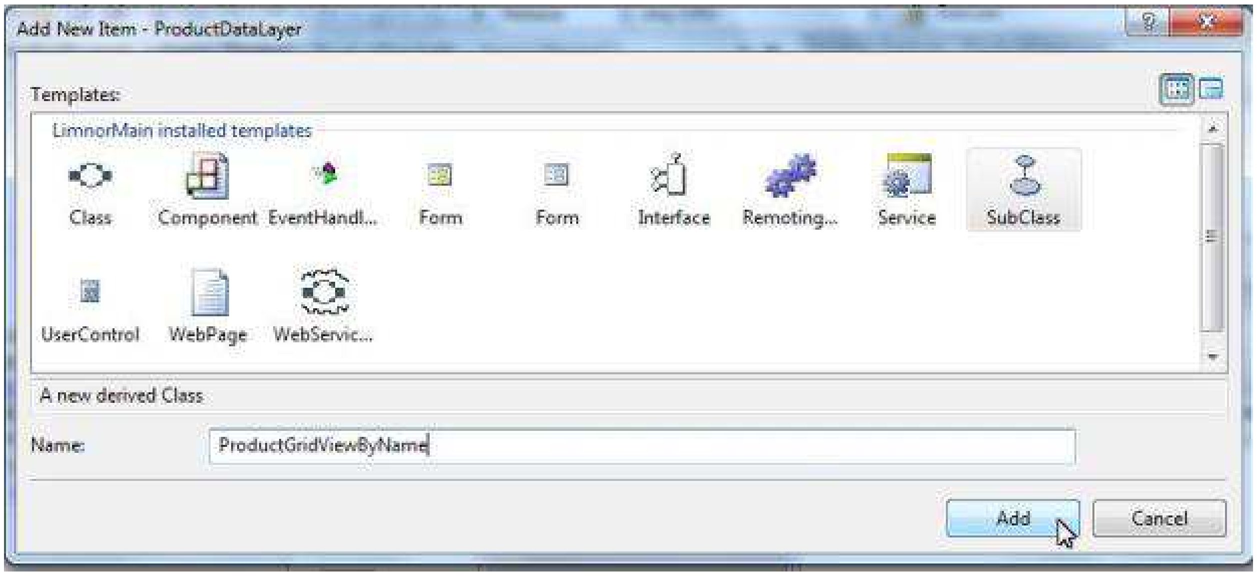
That is all for developing this new class.

Derive from EasyGrid

We may also derive from EasyGrid to make data views. Suppose we want to create a grid view for searching product records by product name.



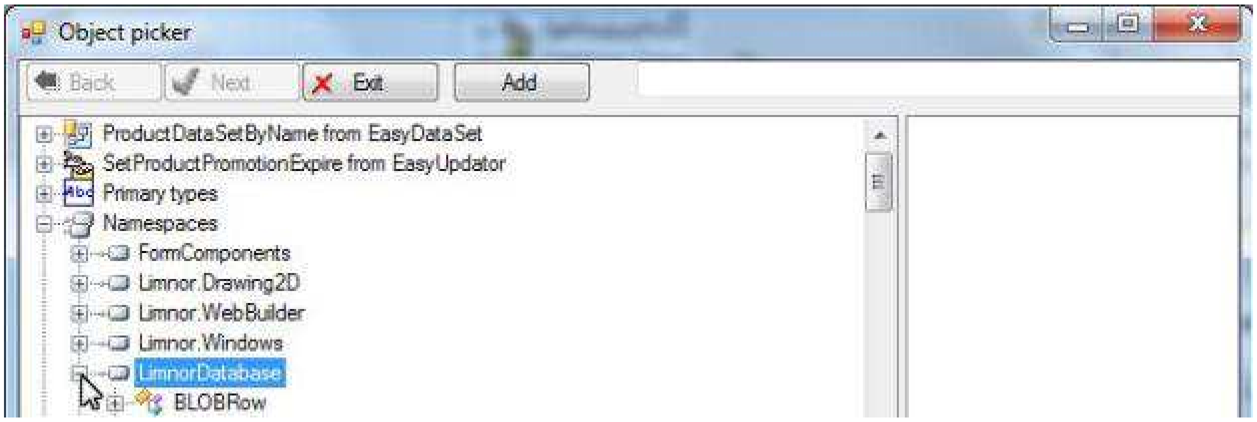
Select SubClass and give a meaningful name for the new class:

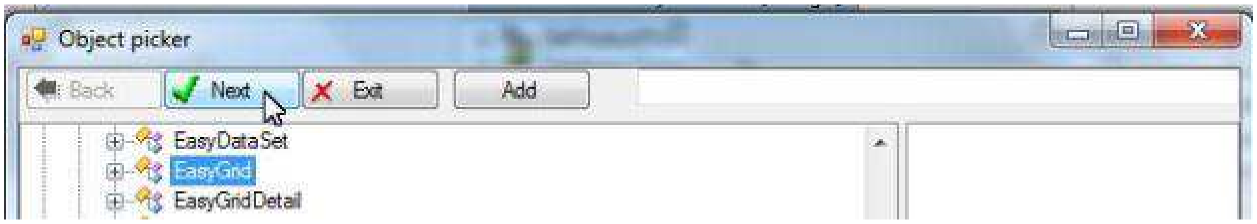


A new derived Class

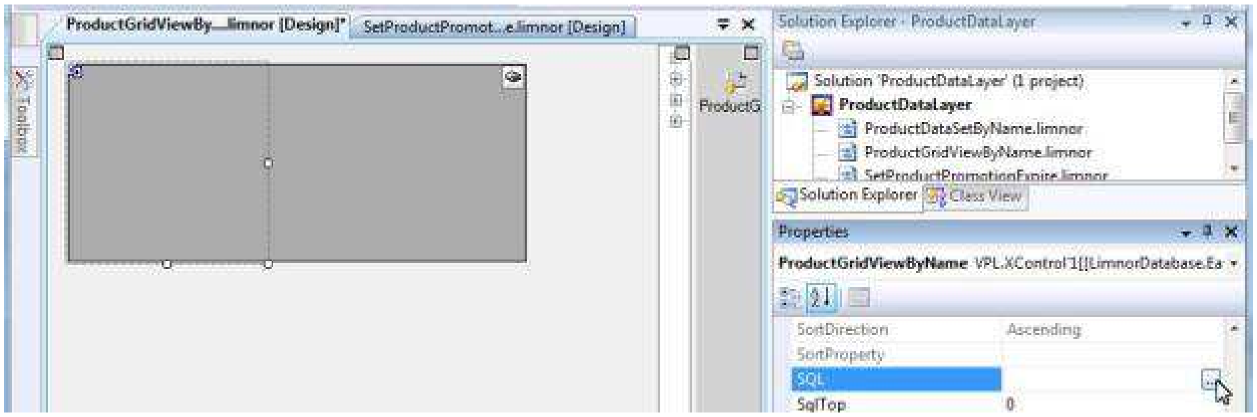
Name: ProductGridViewByName

Select EasyGrid as the base class for the new class:

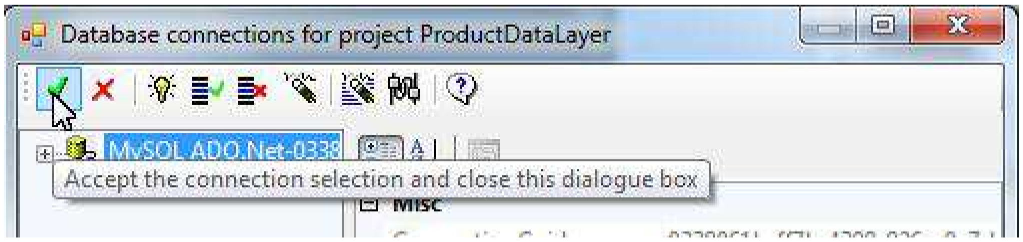




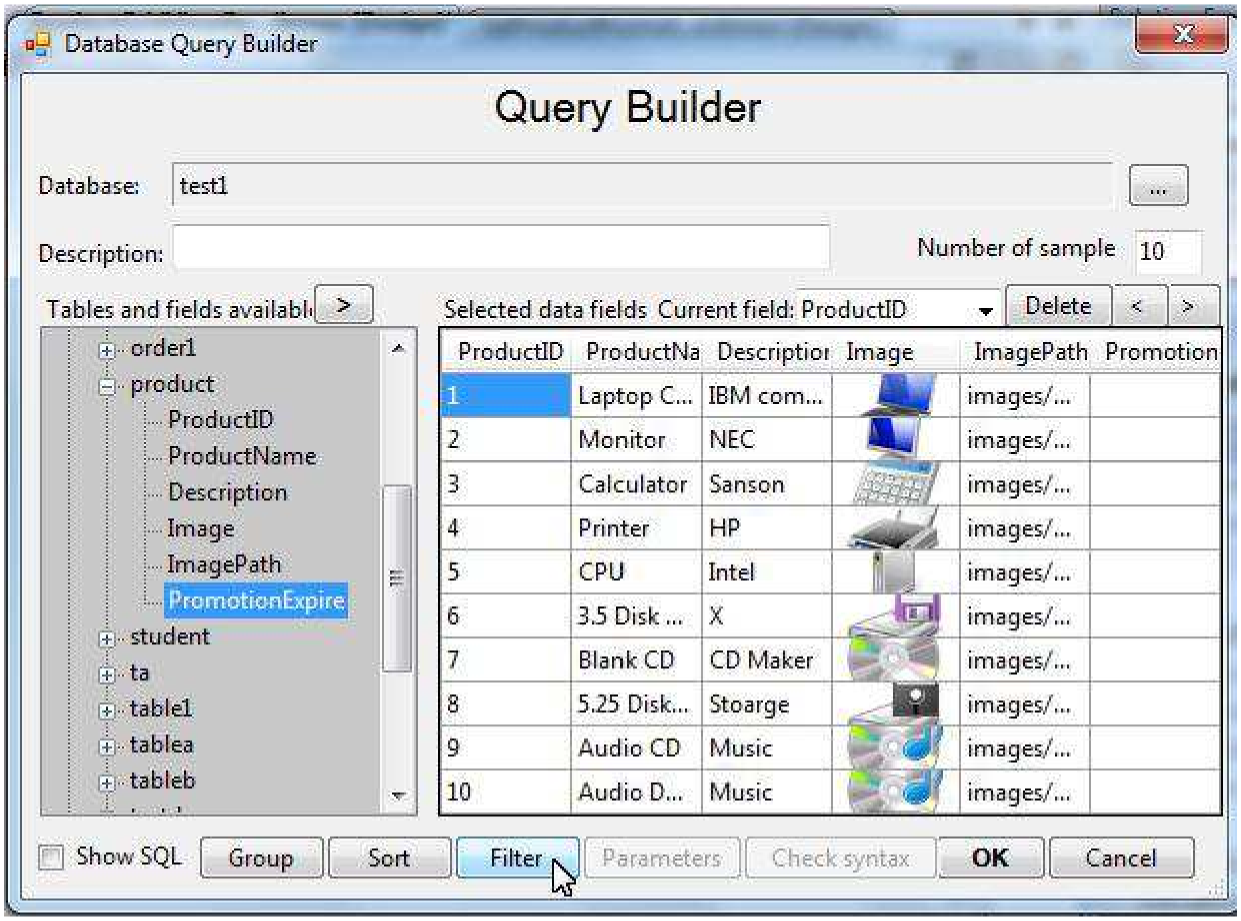
Set the SQL property of the new class for searching product records by name:


ProductGridViewBy....limnor [Design]*
SetProductPromot...e.limnor [Design]
Toolbox
ProductG
Solution Explorer - ProductDataLayer
Solution 'ProductDataLayer' (1 project)
ProductDataLayer
ProductDataSetByName.limnor
ProductGridViewByName.limnor
SetProductPromotionExpire.limnor
Solution Explorer
Class View
Properties
ProductGridViewByName VPL.XControl'1[[LimnorDatabase.Ea
SortDirection
Ascending
SortProperty
SQL
SqlTop
0

Accept the default database connection:


Database connections for project ProductDataLayer
MySQL ADO.Net-0338
Accept the connection selection and close this dialogue box
Misc

Select fields from table Product. Click Filter button to add filters:


Database Query Builder

Query Builder

Database: test1

Description:

Number of sample 10

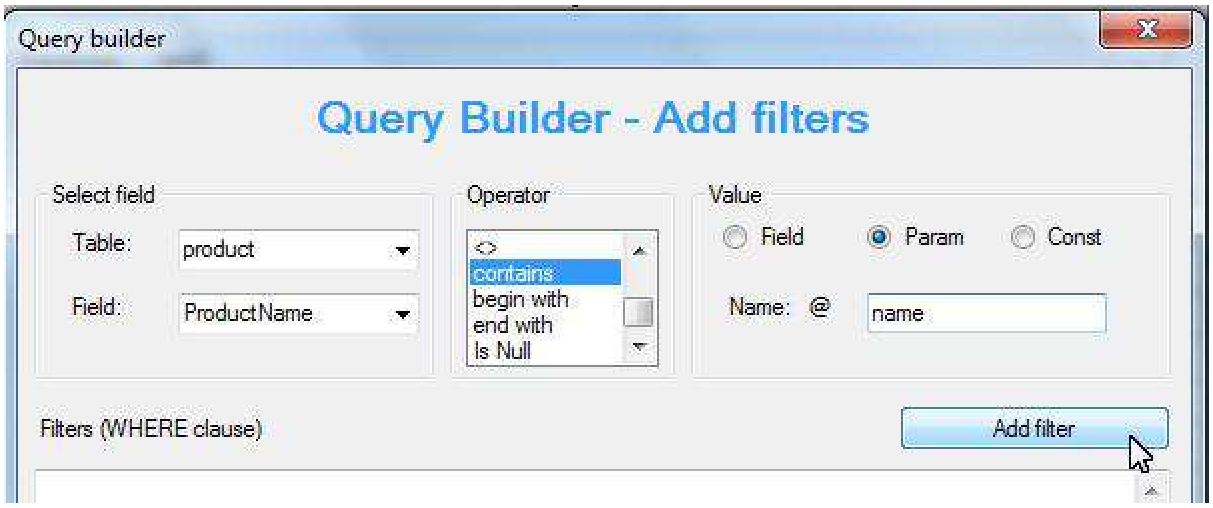
Tables and fields availabl

Selected data fields Current field: ProductID

```
order1
product
    ProductID
    ProductName
    Description
    Image
    ImagePath
    PromotionExpire
student
ta
table1
tablea
tableb
```

ProductID	ProductNa	Descriptio	Image	ImagePath	Promotion
1	Laptop C...	IBM com...		images/...	
2	Monitor	NEC		images/...	
3	Calculator	Sanson		images/...	
4	Printer	HP		images/...	
5	CPU	Intel		images/...	
6	3.5 Disk ...	X		images/...	
7	Blank CD	CD Maker		images/...	
8	5.25 Disk...	Stoarge		images/...	
9	Audio CD	Music		images/...	
10	Audio D...	Music		images/...	

Use a parameter, @name, for searching products by name:


Query builder

Query Builder - Add filters

Select field

Table: product

Field: ProductName

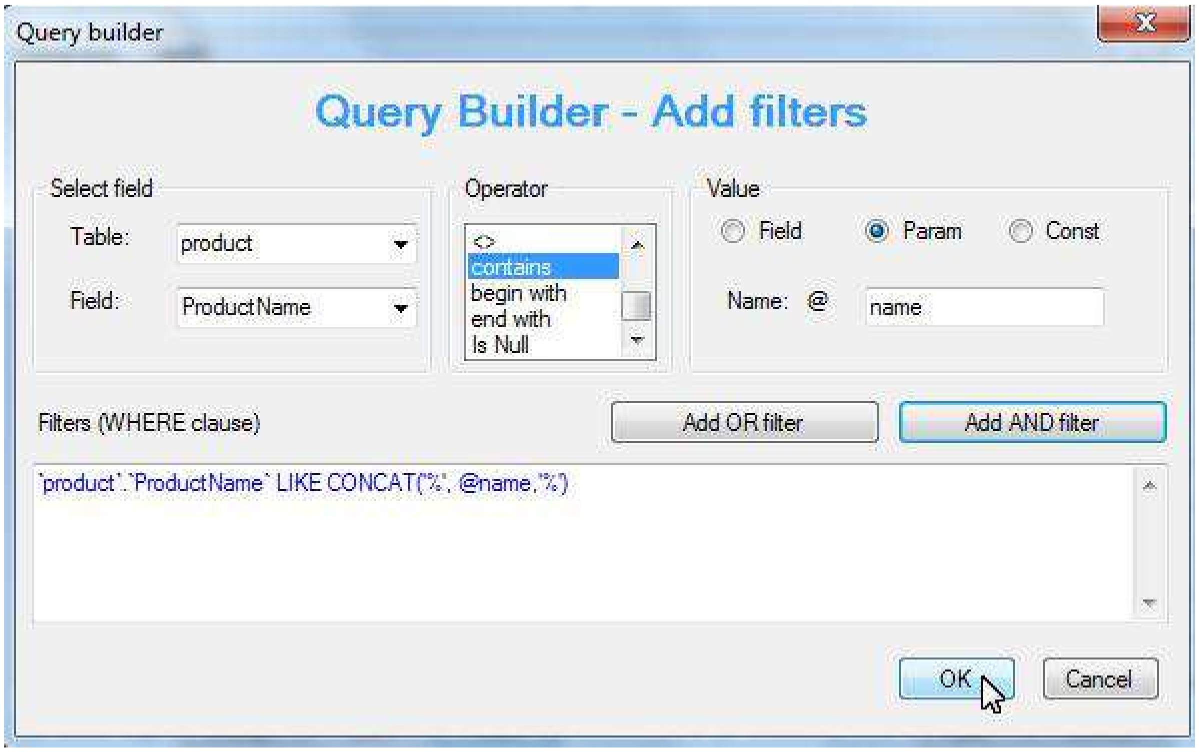
Operator: contains

Value: Field Param [✓] Const

Name: @ name

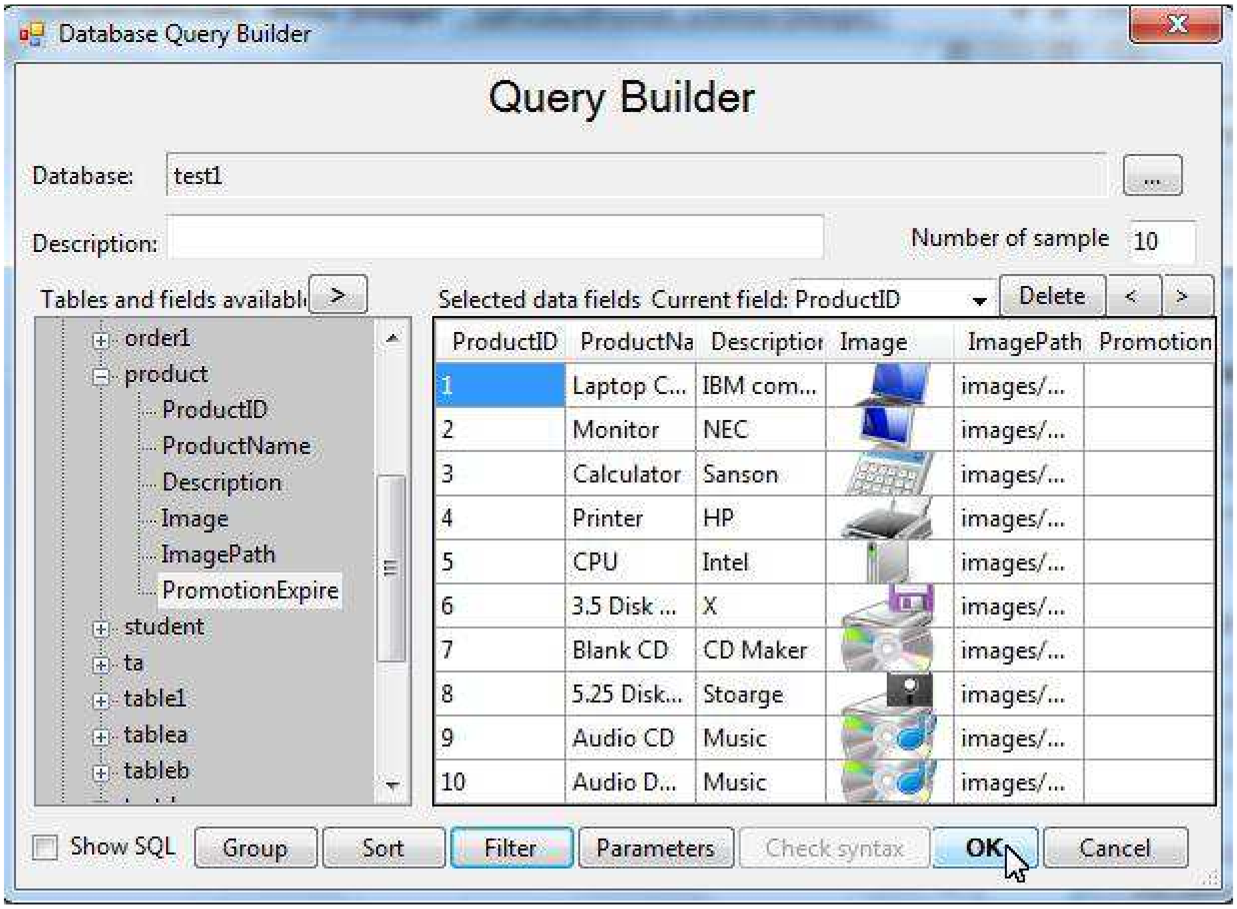
Filters (WHERE clause)

For this sample, we just need this one filter:



`product`.`ProductName` LIKE CONCAT('%', @name,'%')

Click OK to finish creating this query:



Database: test1

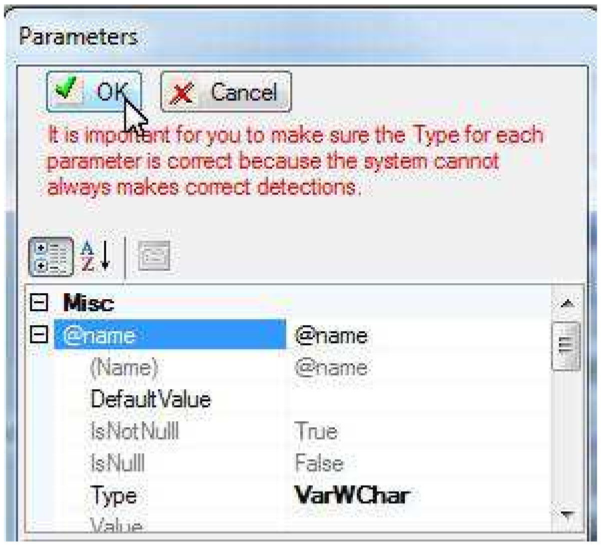
Number of sample: 10

Current field: ProductID

Tables and fields available:
- order1
- product
  - ProductID
  - ProductName
  - Description
  - Image
  - ImagePath
  - PromotionExpire
- student
- ta
- table1
- tablea
- tableb

ProductID	ProductNa	Descriptior	Image	ImagePath	Promotion
1	Laptop C...	IBM com...		images/...	
2	Monitor	NEC		images/...	
3	Calculator	Sanson		images/...	
4	Printer	HP		images/...	
5	CPU	Intel		images/...	
6	3.5 Disk ...	X		images/...	
7	Blank CD	CD Maker		images/...	
8	5.25 Disk...	Storage		images/...	
9	Audio CD	Music		images/...	
10	Audio D...	Music		images/...	

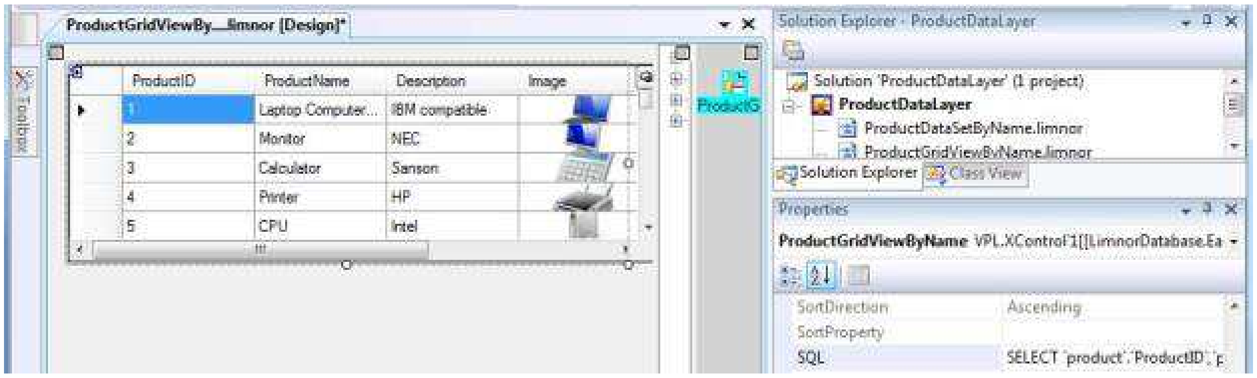
Make sure the parameter data type is correct:



It is important for you to make sure the Type for each parameter is correct because the system cannot always makes correct detections.

@name	@name
(Name)	@name
DefaultValue	
IsNotNulll	True
IsNulll	False
Type	VarWChar
Value	

Product records appear in the grid view:

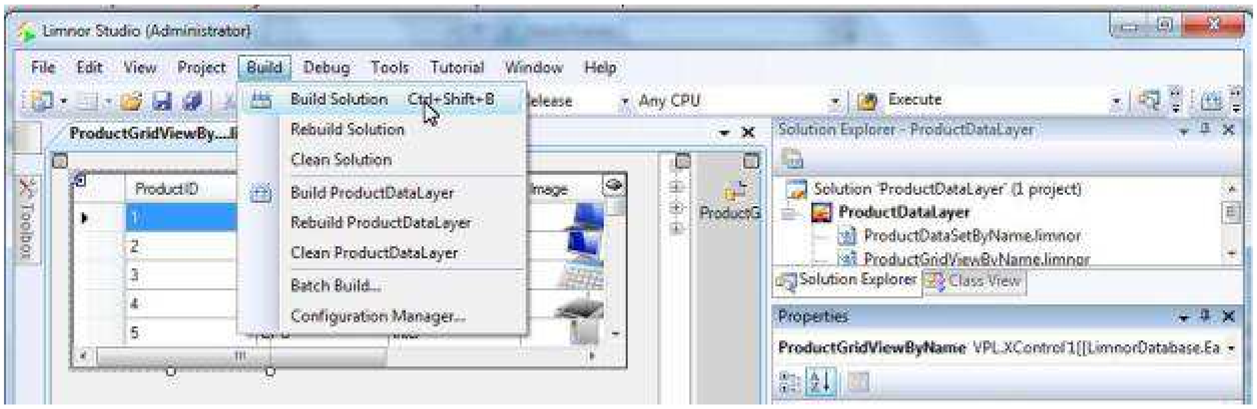

ProductGridViewBy....limnor [Design]*
Toolbox
Solution Explorer - ProductDataLayer
Solution 'ProductDataLayer' (1 project)
ProductDataLayer
ProductDataSetByName.limnor
ProductGridViewByName.limnor
Solution Explorer
Class View
Properties
ProductG

ProductID	ProductName	Description	Image
1	Laptop Computer...	IBM compatible	
2	Monitor	NEC	
3	Calculator	Sanson	
4	Printer	HP	
5	CPU	Intel	

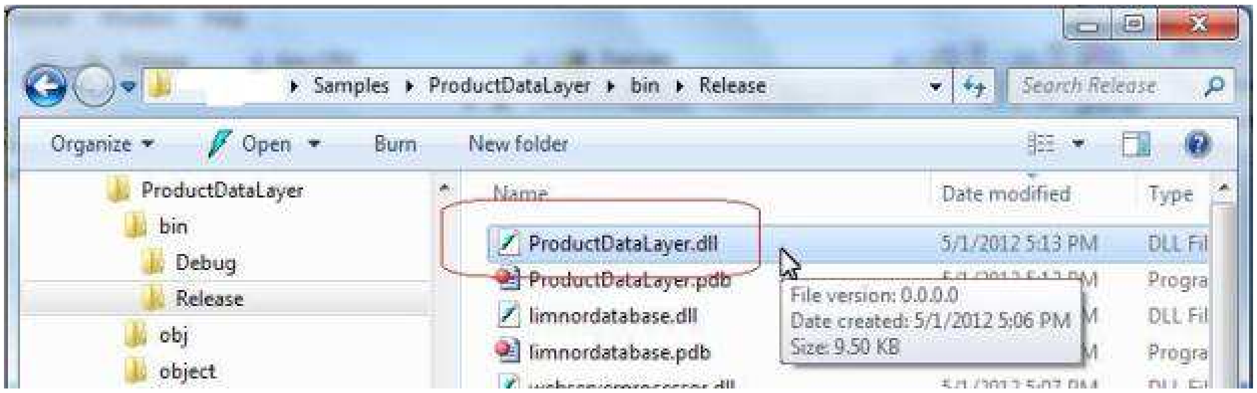
ProductGridViewByName VPL.XControl'1[[LimnorDatabase.Ea

SortDirection	Ascending
SortProperty	
SQL	SELECT `product`.`ProductID`,`p

We may compile this project:



A DLL file, ProductDataLayer.DLL, is created:



Name	Date modified	Type
ProductDataLayer.dll	5/1/2012 5:13 PM	DLL Fil
ProductDataLayer.pdb		Progra
limnordatabase.dll		DLL Fil
limnordatabase.pdb		Progra

File version: 0.0.0.0
Date created: 5/1/2012 5:06 PM
Size: 9.50 KB

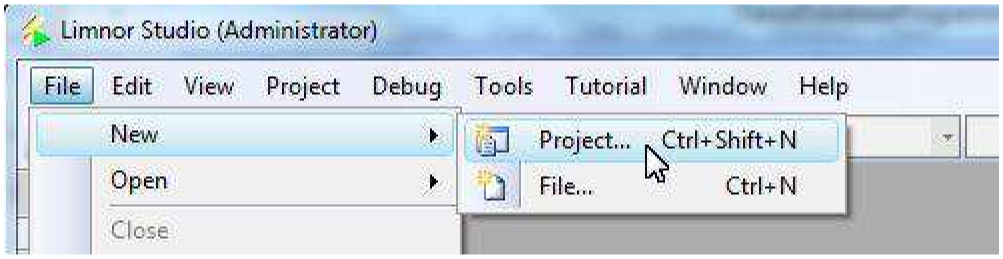
This DLL acts as a software layer for providing database accessing. Currently it only contains 3 classes. According to business requirements, more and more classes can be added to this DLL.

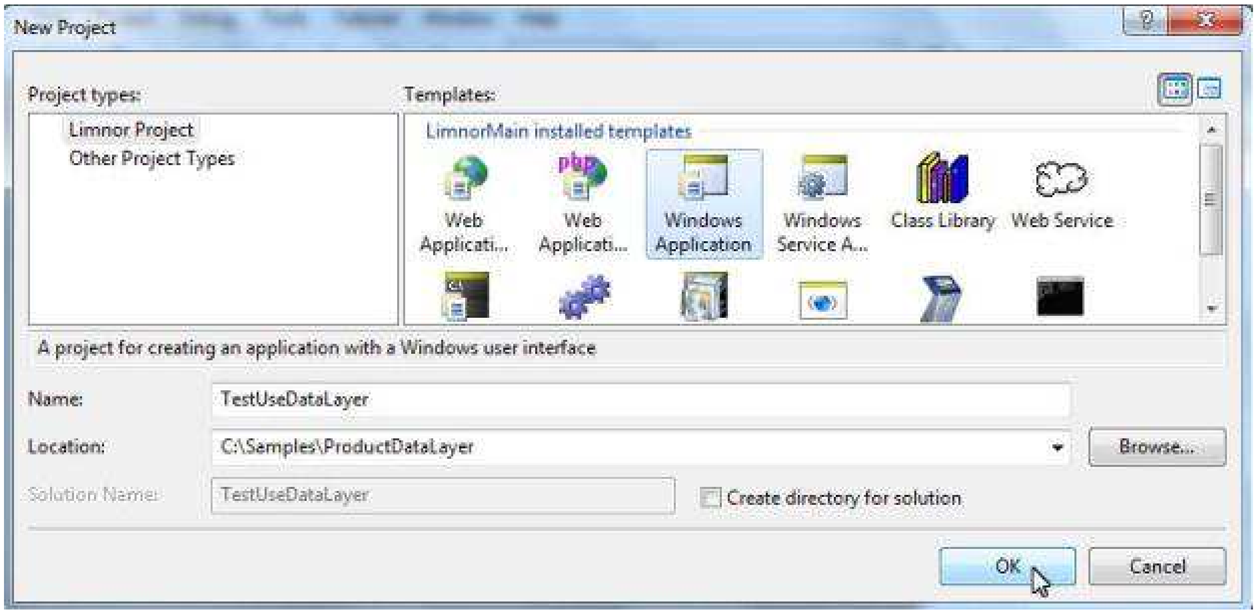
Use Data Accessing Layer

Now let's show how to use this software layer.

Create application project

Create a Window Form application as a sample using the database accessing layer.





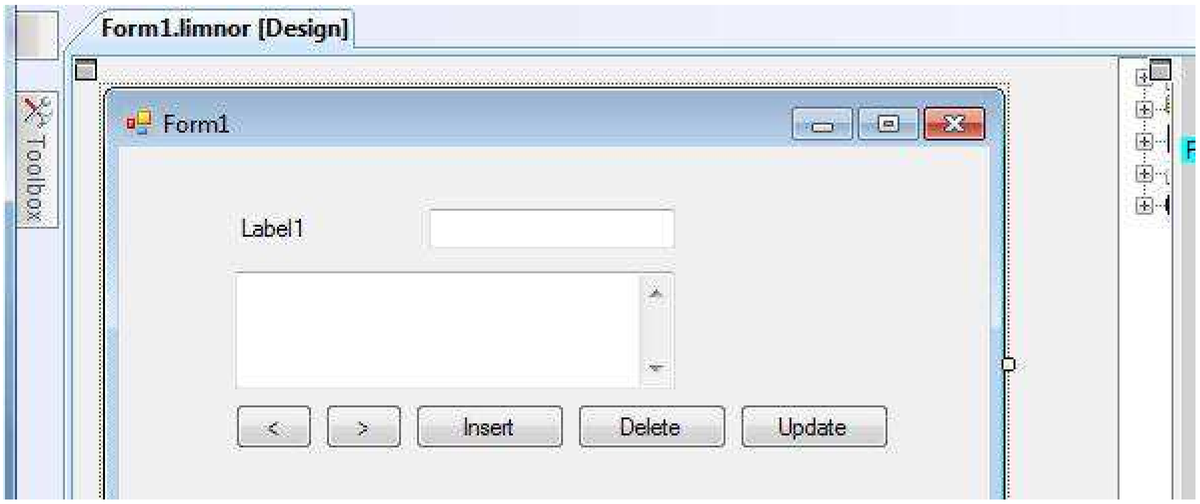
A project for creating an application with a Windows user interface

Name: TestUseDataLayer

Location: C:\Samples\ProductDataLayer

Solution Name: TestUseDataLayer

Add a few controls for displaying and updating data:

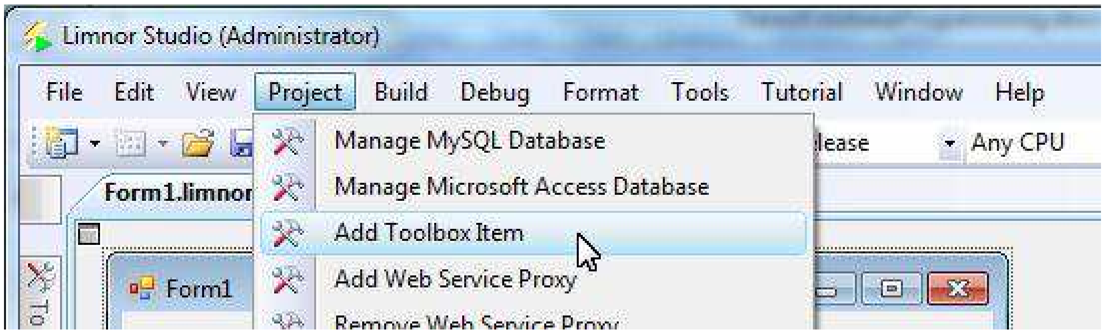


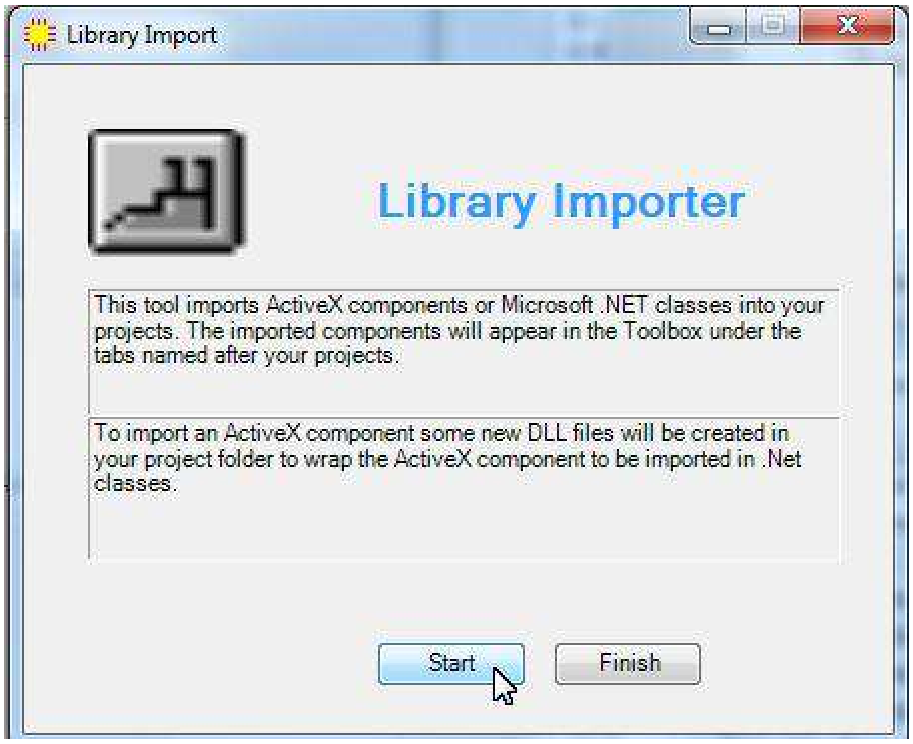
Use Derived Dataset

Instead of using EasyDataSet, we use classes derived from EasyDataSet. In this sample, we use ProductDataSetByName.

Access database layer

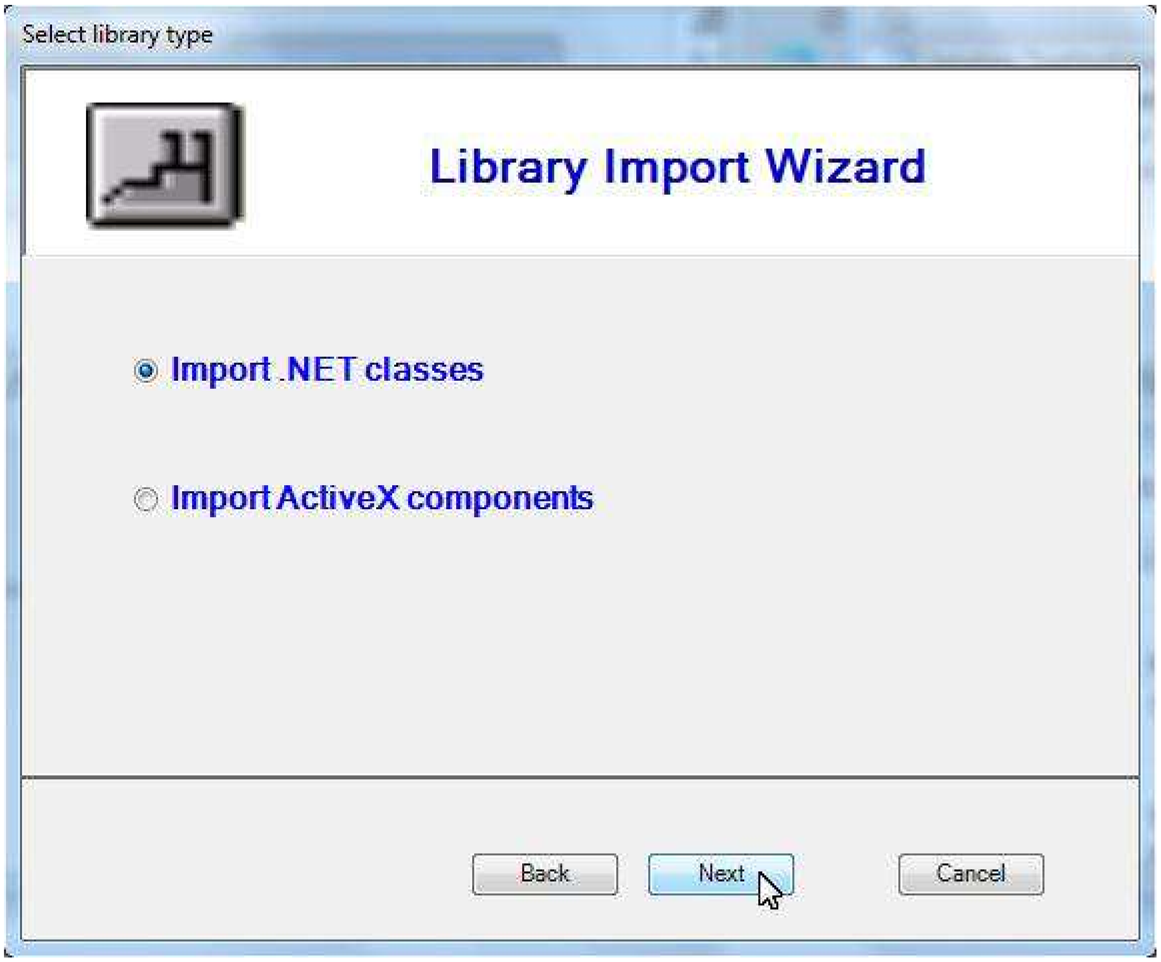
To use the data layer, add classes from the data layer to the Toolbox:





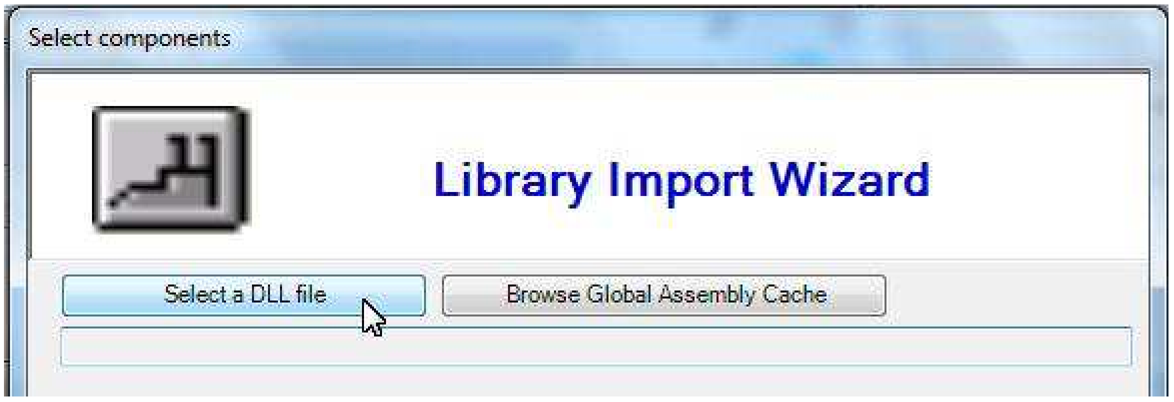
This tool imports ActiveX components or Microsoft .NET classes into your projects. The imported components will appear in the Toolbox under the tabs named after your projects.

To import an ActiveX component some new DLL files will be created in your project folder to wrap the ActiveX component to be imported in .Net classes.



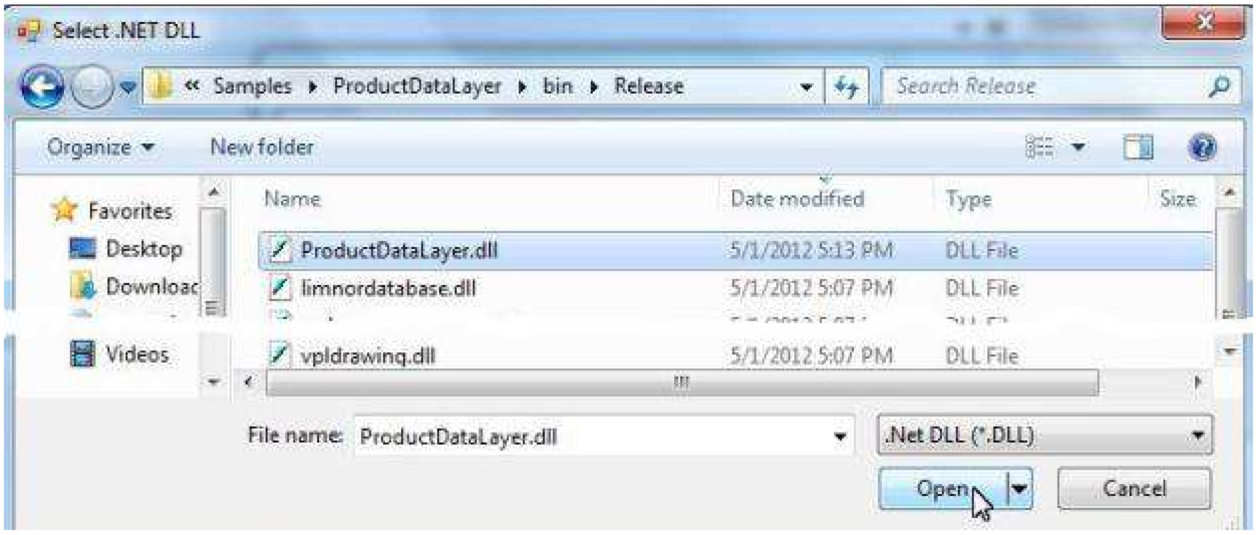
Select library type

[✓ Import .NET classes] Import ActiveX components

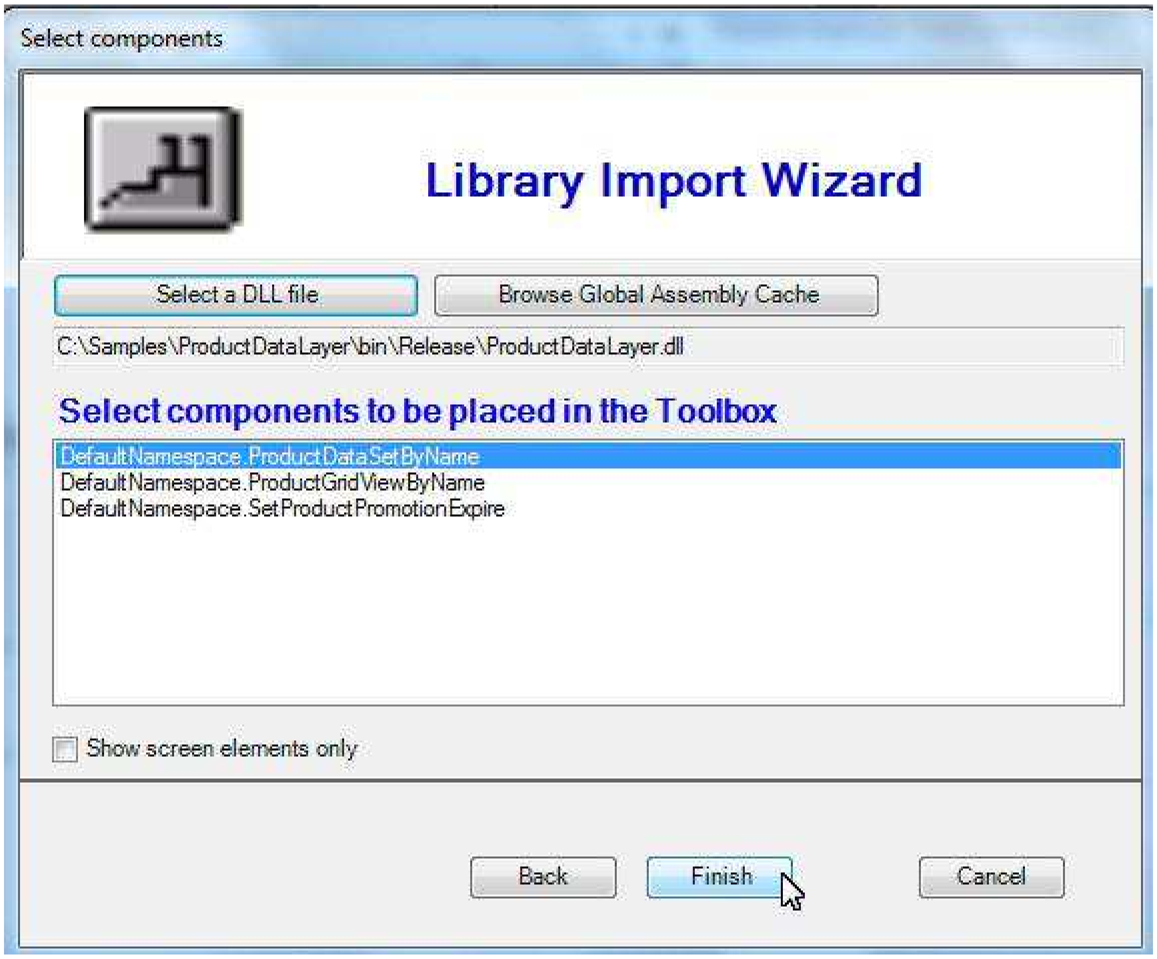


Select components

Select the data accessing layer DLL:


Select .NET DLL
« Samples ▸ ProductDataLayer ▸ bin ▸ Release
Search Release
Organize
New folder
Favorites
Desktop
Downloads
Videos
Name
Date modified
Type
Size
ProductDataLayer.dll 5/1/2012 5:13 PM DLL File
limnordatabase.dll 5/1/2012 5:07 PM DLL File
vpldrawing.dll 5/1/2012 5:07 PM DLL File
File name: ProductDataLayer.dll
.Net DLL (*.DLL)
Open
Cancel

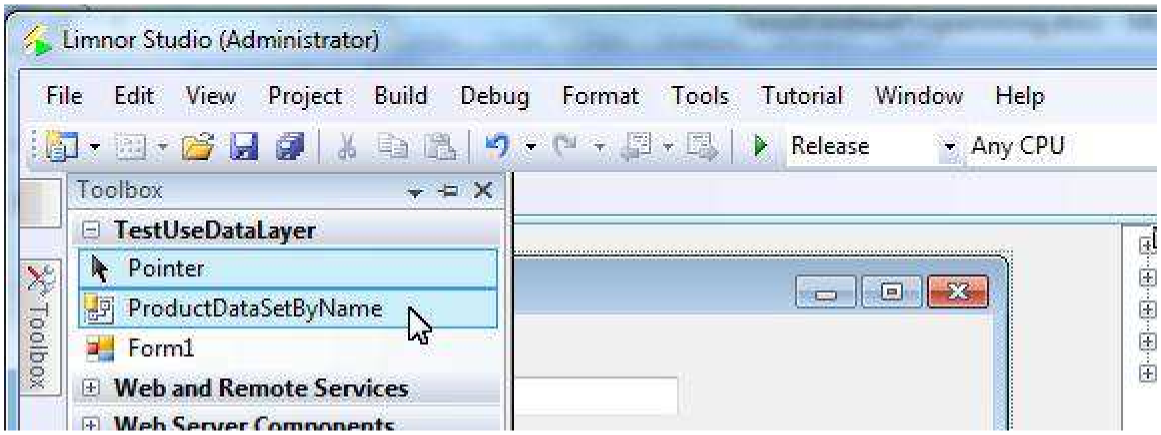
Currently the DLL contains only 3 classes. Select class, ProductDataSetByName, click Finish:


Select components
Library Import Wizard
Select a DLL file
Browse Global Assembly Cache
Select components to be placed in the Toolbox
Show screen elements only
Back
Finish
Cancel

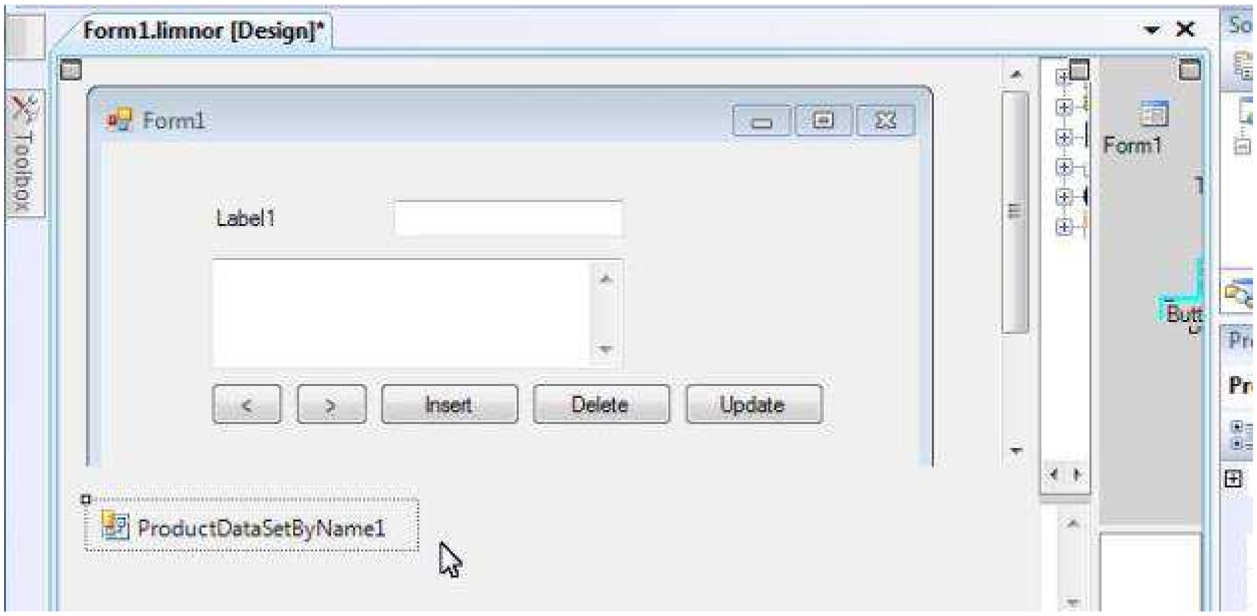
C:\Samples\ProductDataLayer\bin\Release\ProductDataLayer.dll

DefaultNamespace.ProductDataSetByName
DefaultNamespace.ProductGridViewByName
DefaultNamespace.SetProductPromotionExpire

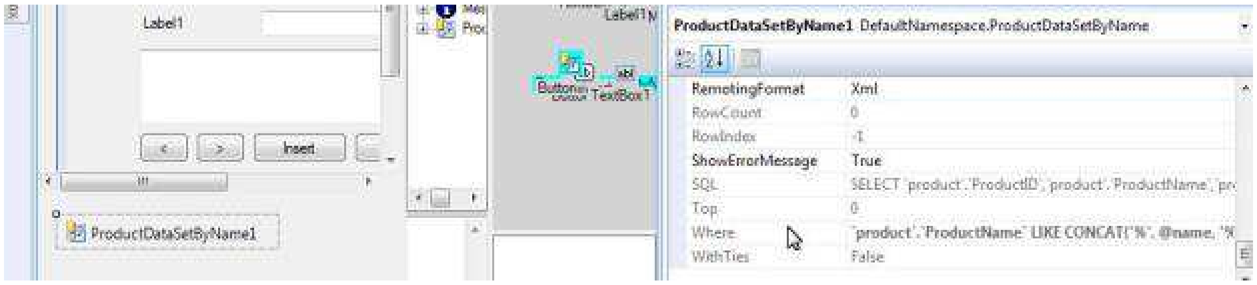
The class from the data accessing layer appears in the Toolbox. Drop it to the form:



An instance of ProductDataSetByName appears in the designer tray:



Examining the properties of ProductDataSetByName1, we can see that the properties defining the database query are read-only. It saves the efforts of creating database query and also ensures that any developers using it are using the correct database query.

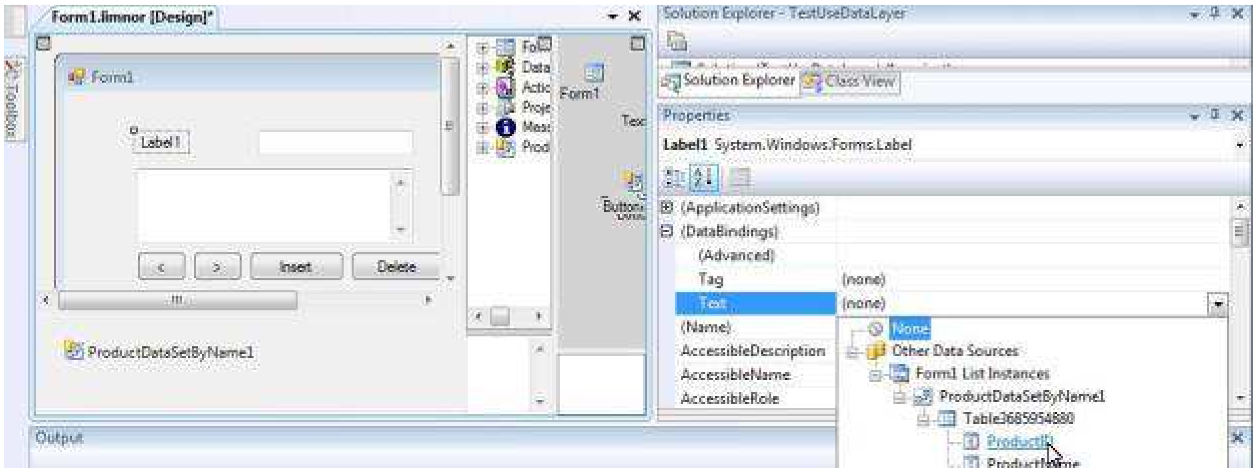


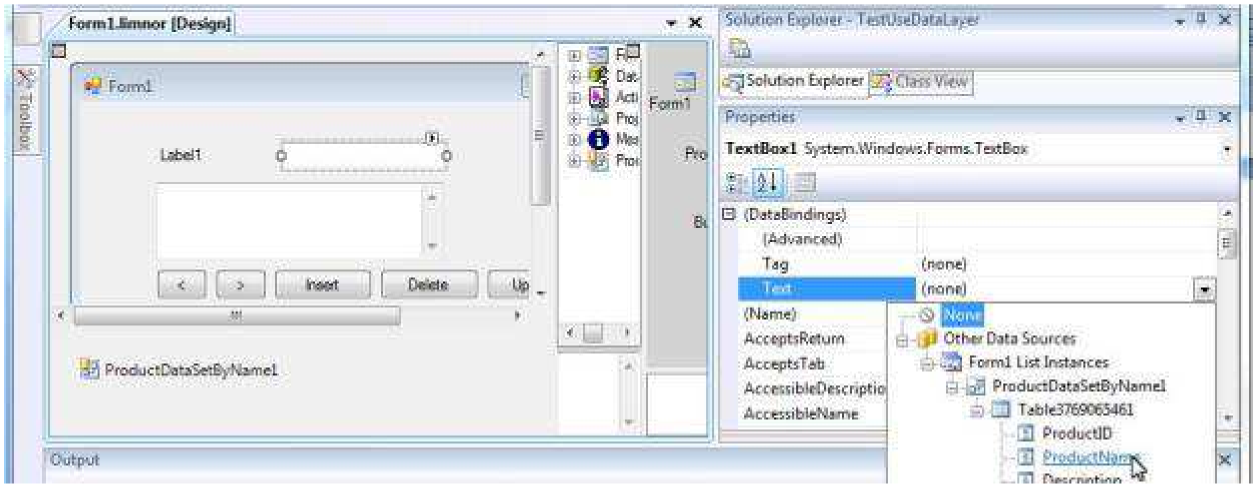
ProductDataSetByName1 DefaultNamespace.ProductDataSetByName

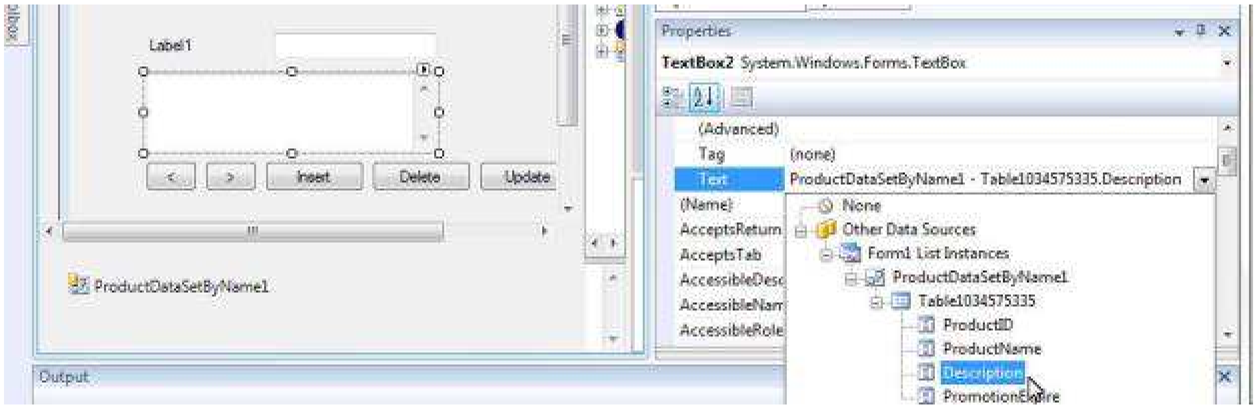
RemotingFormat	Xml
RowCount	0
RowIndex	-1
ShowErrorMessage	True
SQL	SELECT `product`.`ProductID`, `product`.`ProductName`, `pr
Top	0
Where	`product`.`ProductName` LIKE CONCAT('%', @name, '%
WithTies	False

Data-binding with database layer

Bind the controls to ProductDataSetByName1:

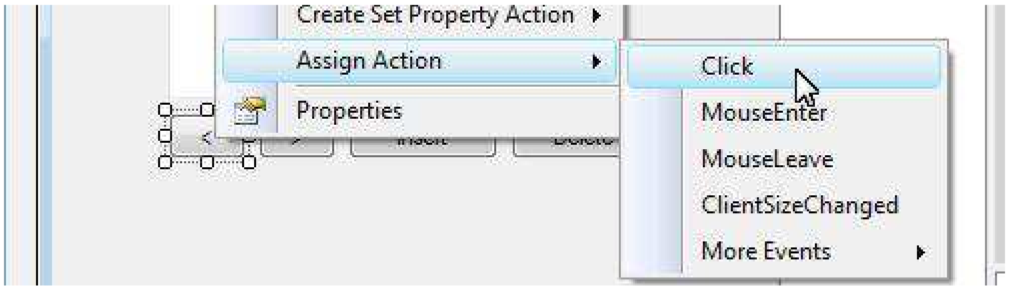




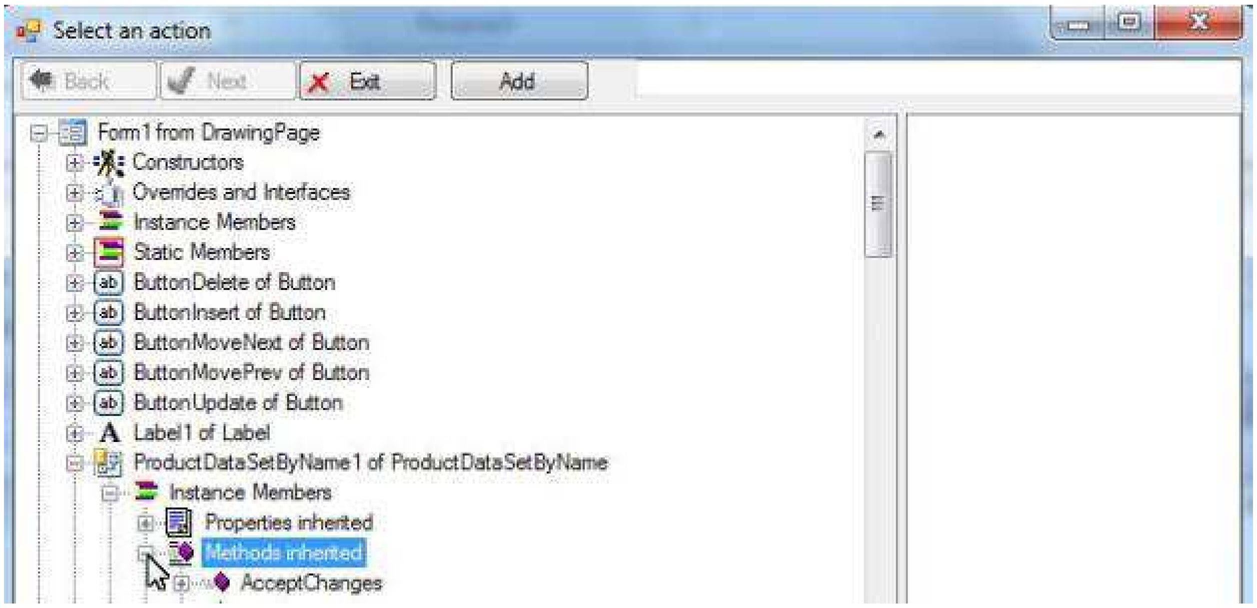


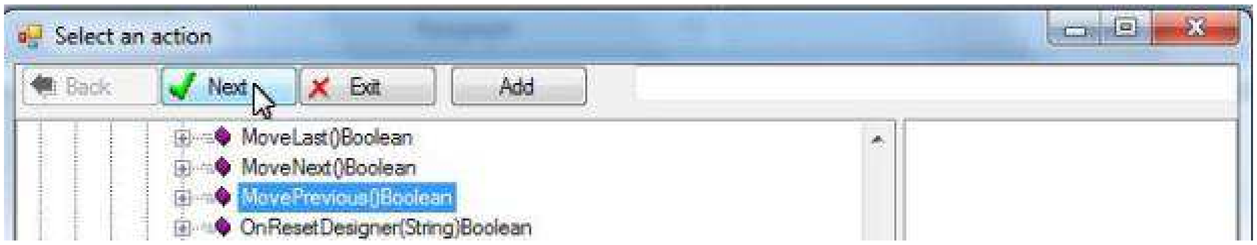
Create database actions

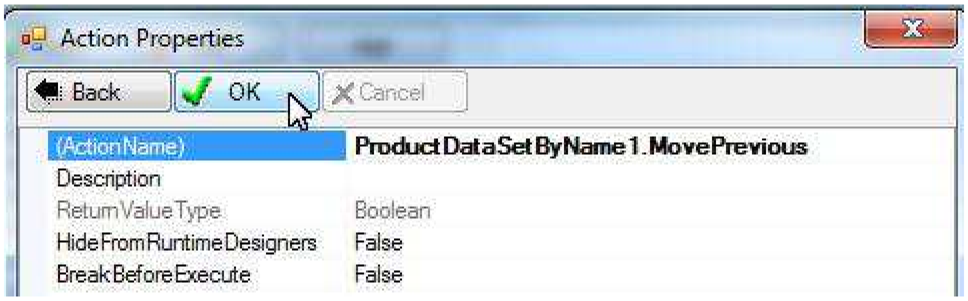
Right-click button “<”; choose “Assign action”; choose “Click”:



Select MovePrevious of ProductDataSetByName1:

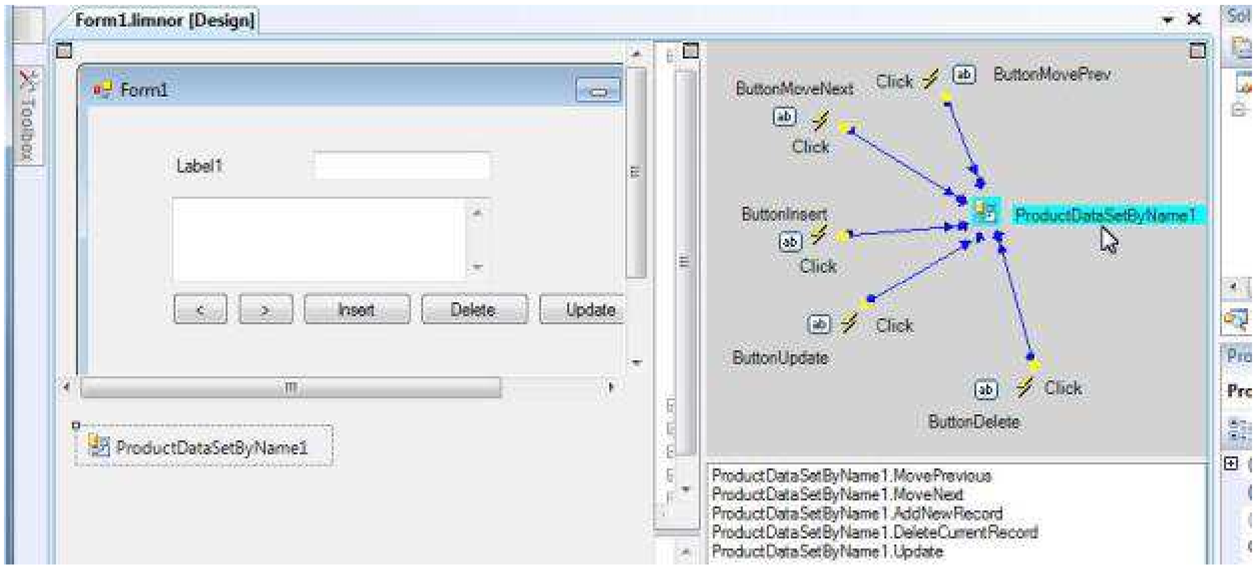






(ActionName)	ProductDataSetByName1.MovePrevious
Description	
ReturnValueType	Boolean
HideFromRuntimeDesigners	False
BreakBeforeExecute	False

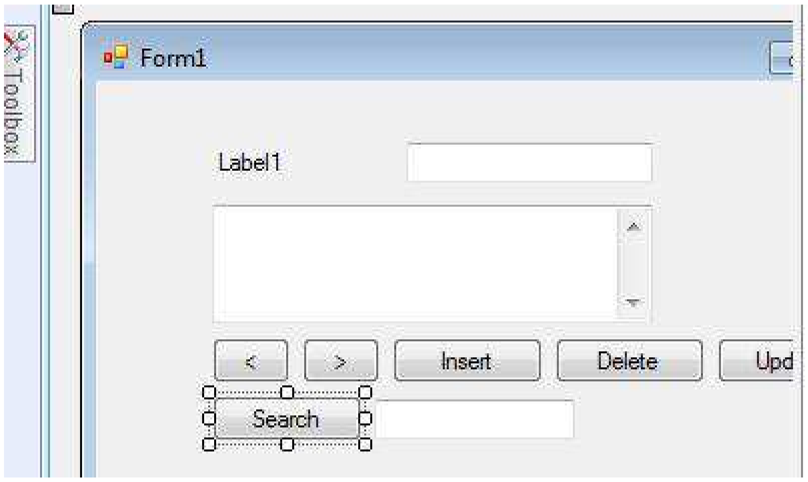
In a similar way, each button is assigned a corresponding action made by a method of ProductDataSetByName1:

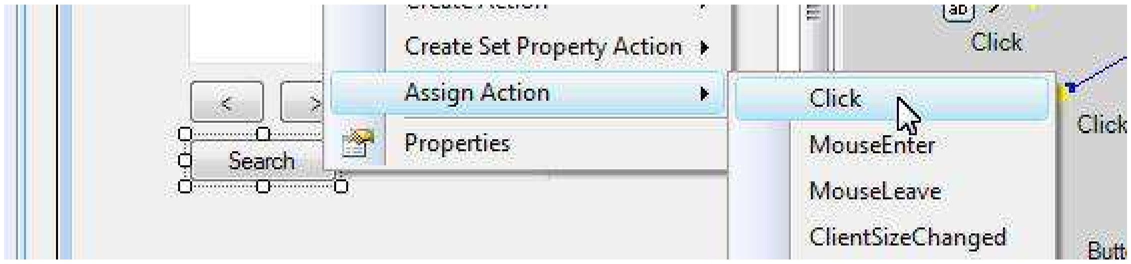


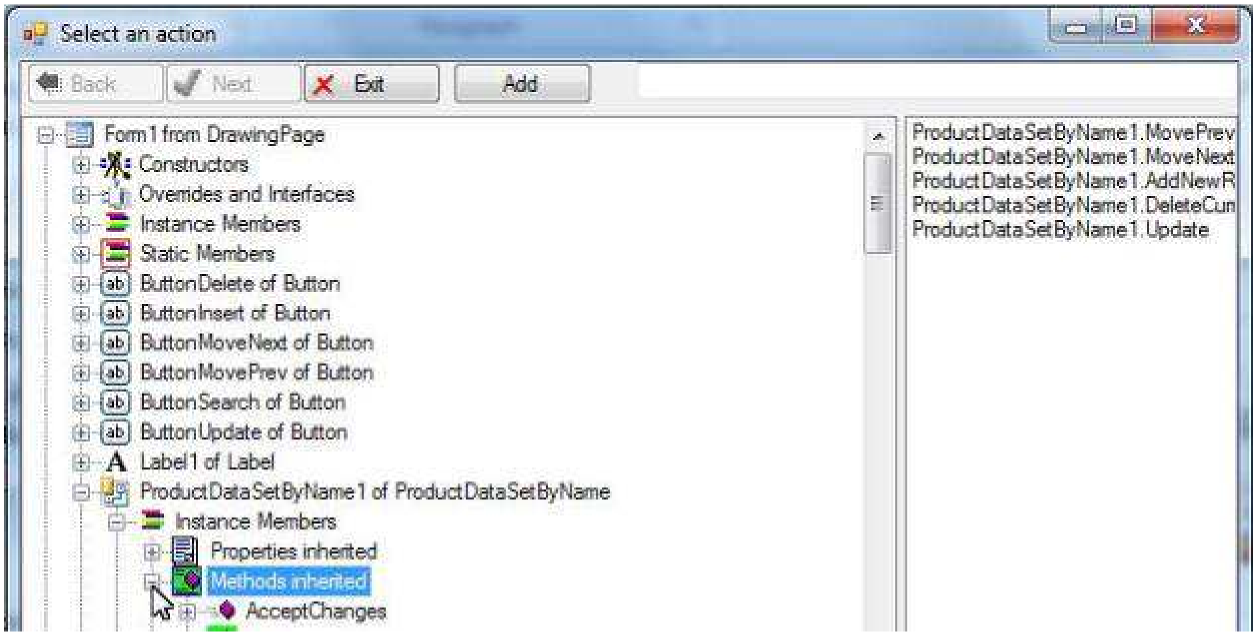
ProductDataSetByName1.MovePrevious
ProductDataSetByName1.MoveNext
ProductDataSetByName1.AddNewRecord
ProductDataSetByName1.DeleteCurrentRecord
ProductDataSetByName1.Update

Search database

ProductDataSetByName uses a parameter, @name, for searching Product records by product name. We use a text box for entering search text, and use a button for executing the search.

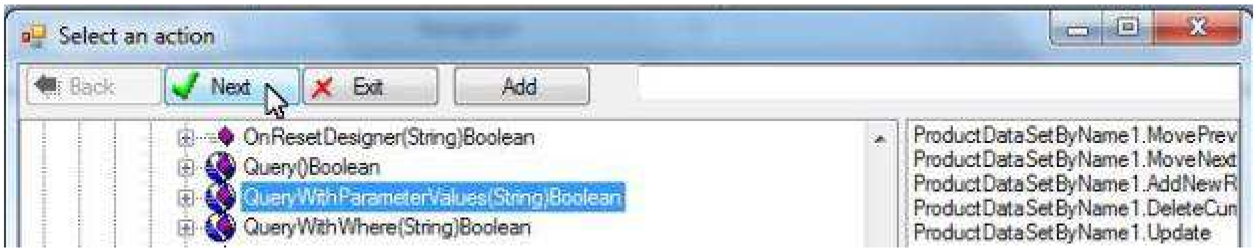






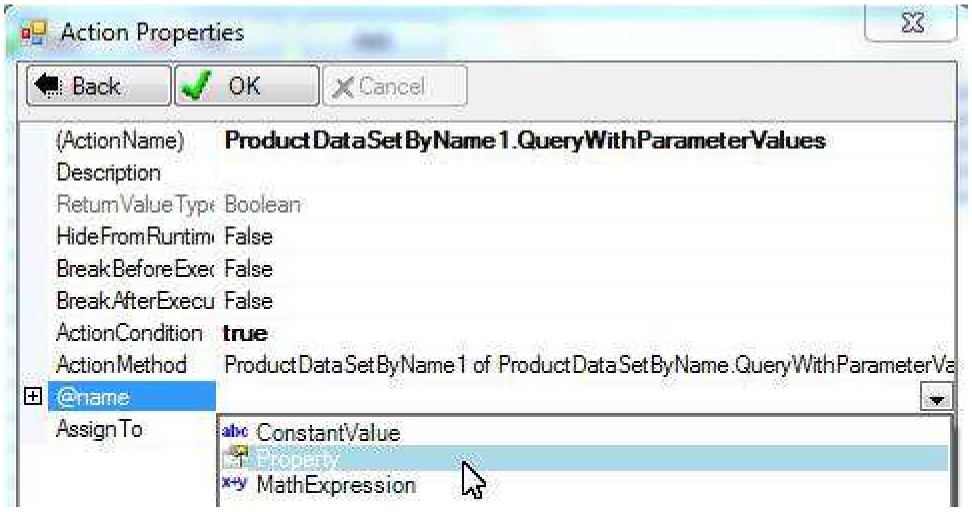
Form1 from DrawingPage
Constructors
Overrides and Interfaces
Instance Members
Static Members
ButtonDelete of Button
ButtonInsert of Button
ButtonMoveNext of Button
ButtonMovePrev of Button
ButtonSearch of Button
ButtonUpdate of Button
Label1 of Label
ProductDataSetByName1 of ProductDataSetByName
Instance Members
Properties inherited
Methods inherited
AcceptChanges

ProductDataSetByName1.MovePrev
ProductDataSetByName1.MoveNext
ProductDataSetByName1.AddNewR
ProductDataSetByName1.DeleteCurr
ProductDataSetByName1.Update

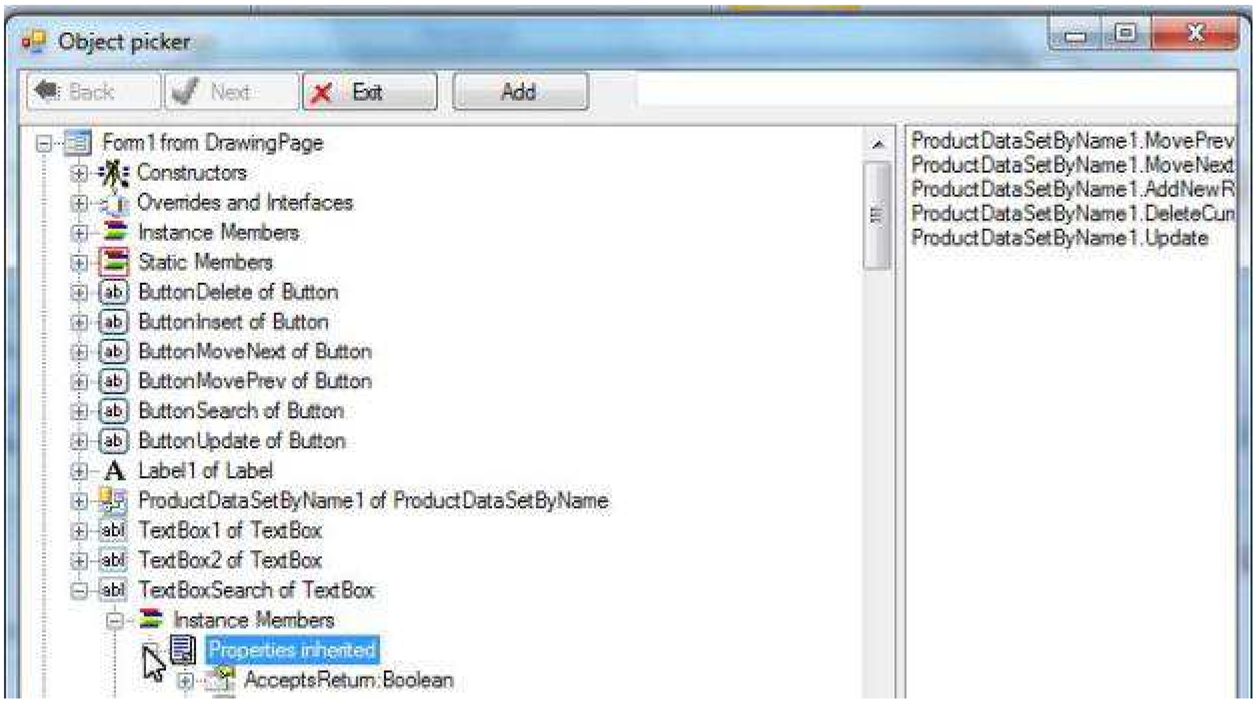


OnResetDesigner(String)Boolean
Query()Boolean
QueryWithParameterValues(String)Boolean
QueryWithWhere(String)Boolean

ProductDataSetByName1.MovePrev
ProductDataSetByName1.MoveNext
ProductDataSetByName1.AddNewR
ProductDataSetByName1.DeleteCurr
ProductDataSetByName1.Update

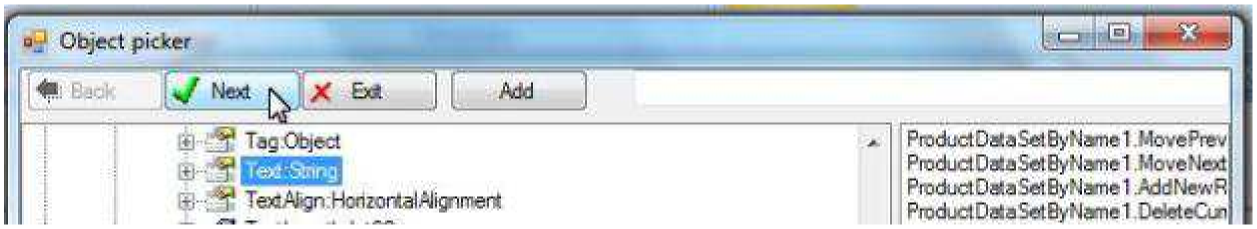


(ActionName)	**ProductDataSetByName1.QueryWithParameterValues**
Description	
ReturnValueType	Boolean
HideFromRuntim	False
BreakBeforeExec	False
BreakAfterExecu	False
ActionCondition	**true**
ActionMethod	ProductDataSetByName1 of ProductDataSetByName.QueryWithParameterVa
@name	
AssignTo	



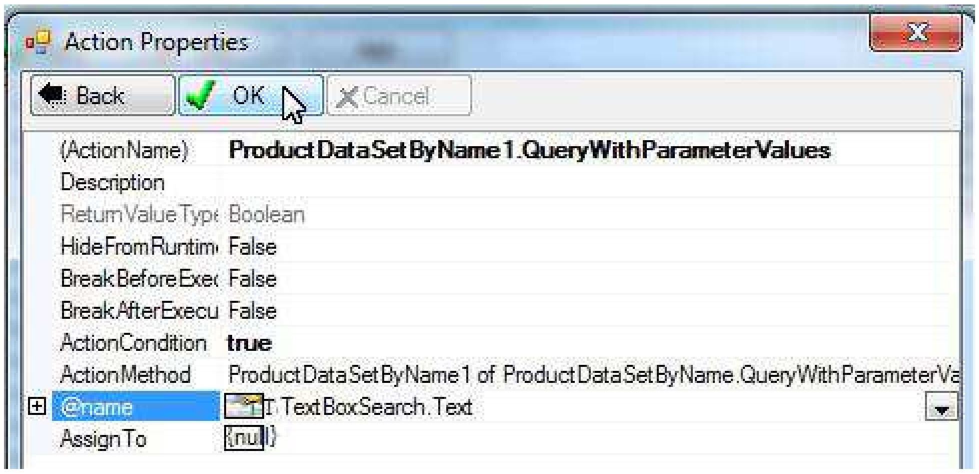
```
Form1 from DrawingPage
    Constructors
    Overrides and Interfaces
    Instance Members
    Static Members
    ButtonDelete of Button
    ButtonInsert of Button
    ButtonMoveNext of Button
    ButtonMovePrev of Button
    ButtonSearch of Button
    ButtonUpdate of Button
    Label1 of Label
    ProductDataSetByName1 of ProductDataSetByName
    TextBox1 of TextBox
    TextBox2 of TextBox
    TextBoxSearch of TextBox
        Instance Members
            Properties inherited
                AcceptsReturn:Boolean
```

```
ProductDataSetByName1.MovePrev
ProductDataSetByName1.MoveNext
ProductDataSetByName1.AddNewR
ProductDataSetByName1.DeleteCurr
ProductDataSetByName1.Update
```

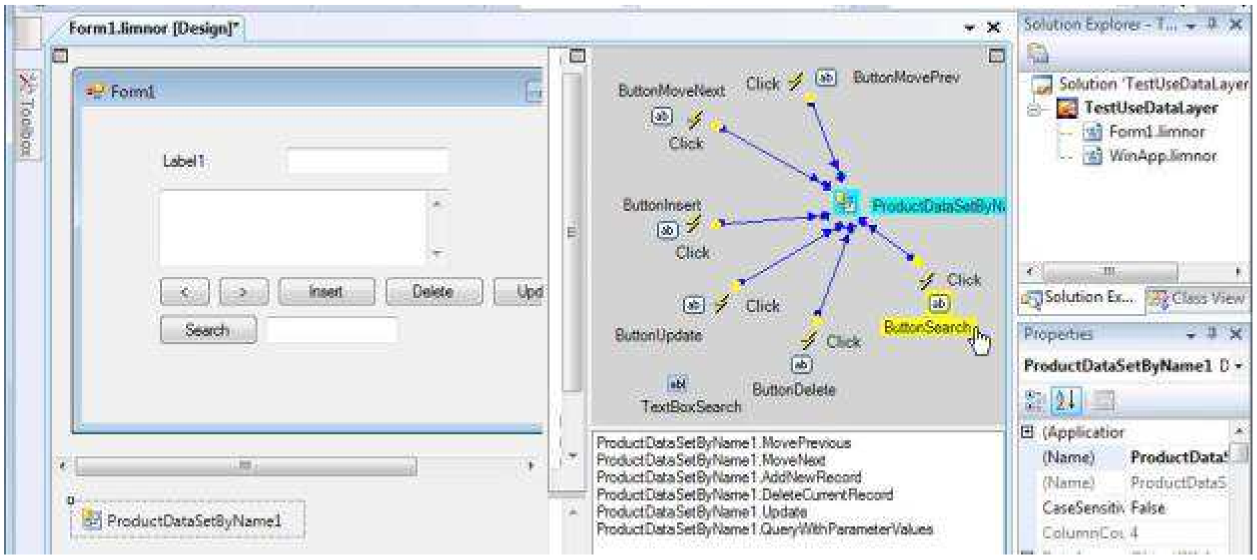


```
Tag:Object
Text:String
TextAlign:HorizontalAlignment
```

```
ProductDataSetByName1.MovePrev
ProductDataSetByName1.MoveNext
ProductDataSetByName1.AddNewR
ProductDataSetByName1.DeleteCurr
```



(ActionName)	**ProductDataSetByName1.QueryWithParameterValues**
Description	
ReturnValueType	Boolean
HideFromRuntim	False
BreakBeforeExec	False
BreakAfterExecu	False
ActionCondition	**true**
ActionMethod	ProductDataSetByName1 of ProductDataSetByName.QueryWithParameterVa
@name	TextBoxSearch.Text
AssignTo	(null)



ProductDataSetByName1.MovePrevious
ProductDataSetByName1.MoveNext
ProductDataSetByName1.AddNewRecord
ProductDataSetByName1.DeleteCurrentRecord
ProductDataSetByName1.Update
ProductDataSetByName1.QueryWithParameterValues

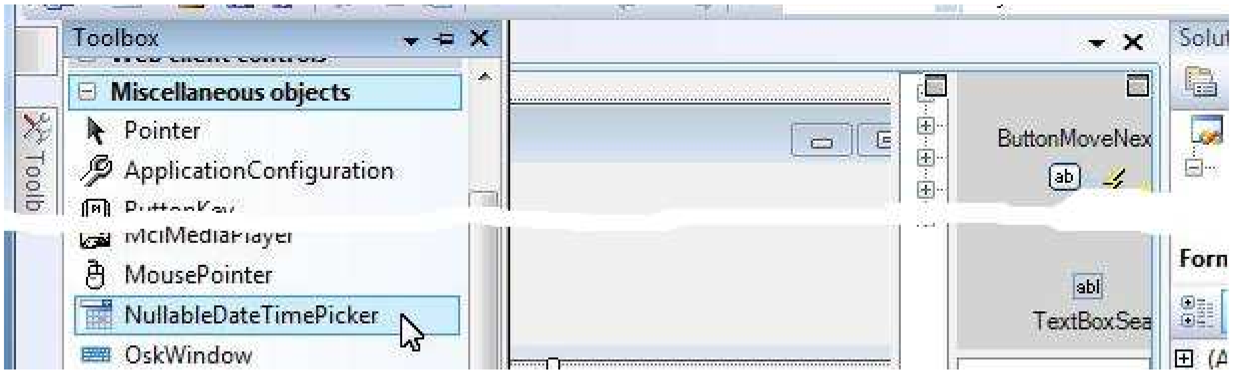
The use of ProductDataSetByName is mostly the same as that of using EasyDataSet. For details of using EasyDataSet, see http://www.limnor.com/support/Limnor%20Studio%20-%20User%20Guide%20-%20Part%20VI.pdf

Using ProductDataSetByName, instead of using EasyDataSet, we do not need to create database query.

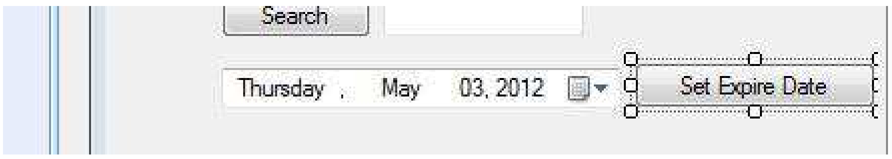
Use Derived Data Updater

UI Design

We use a Month Calendar control for picking date:

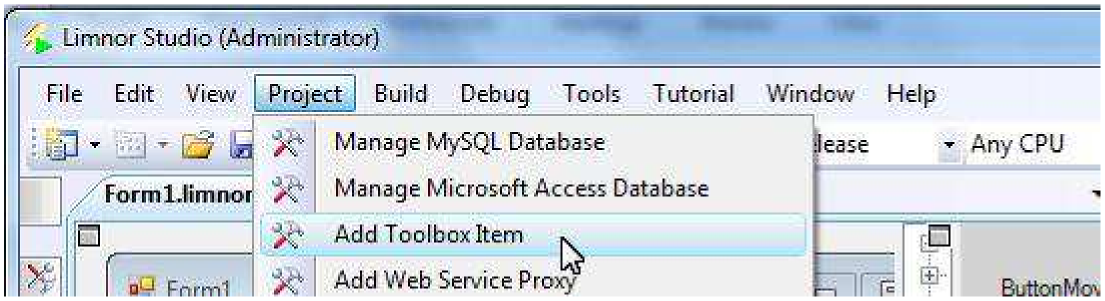


We use a button to execute a database update command:

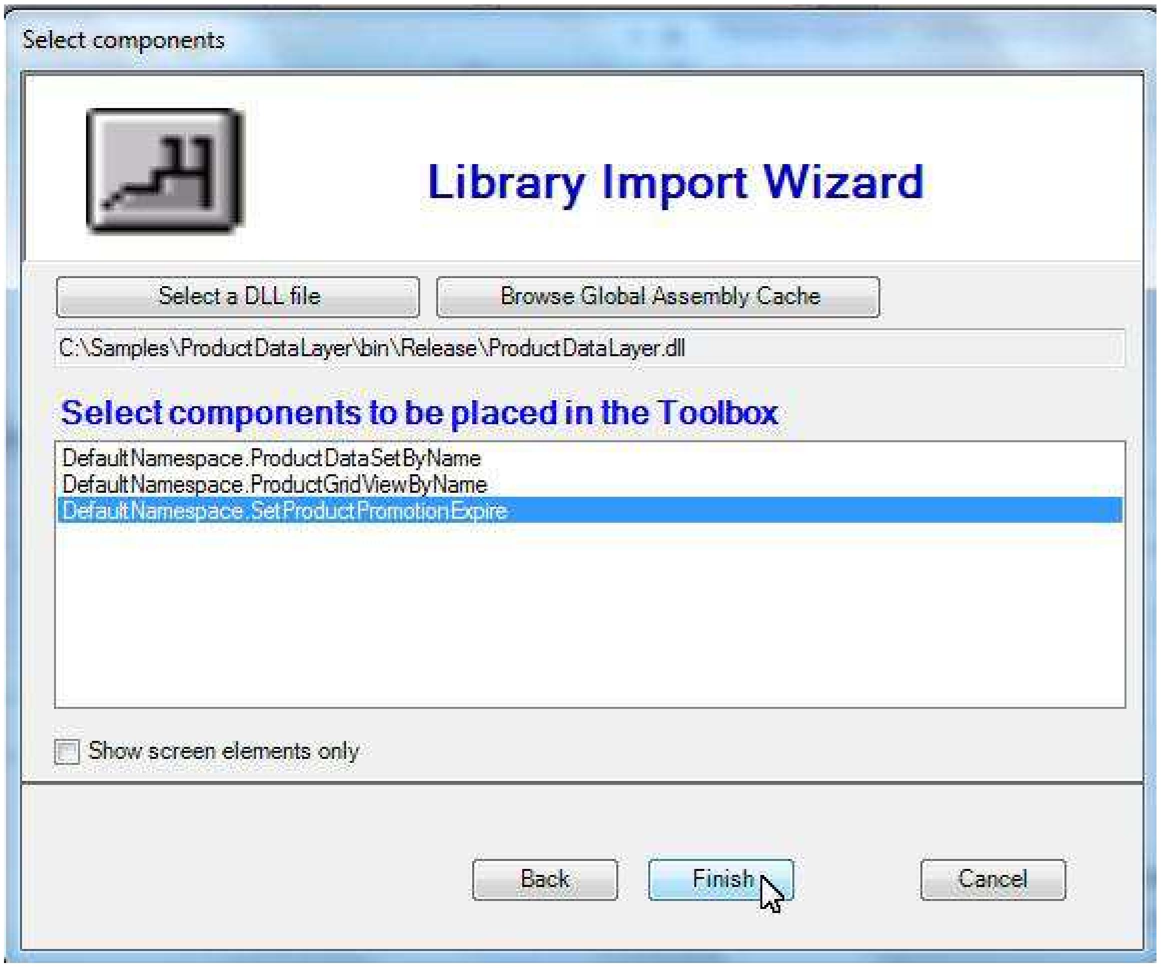


Access data layer

We want to use class SetPromotionExpire from the data layer. First, we add it to the Toolbox.



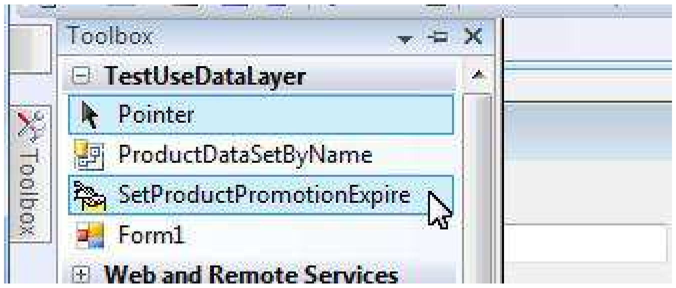
We'll skip the steps already shown in a previous section, and jump to the step of selecting class SetProductPromtionExpire from the data layer:


Select components
Library Import Wizard
Select a DLL file
Browse Global Assembly Cache
Select components to be placed in the Toolbox
Show screen elements only
Back
Finish
Cancel

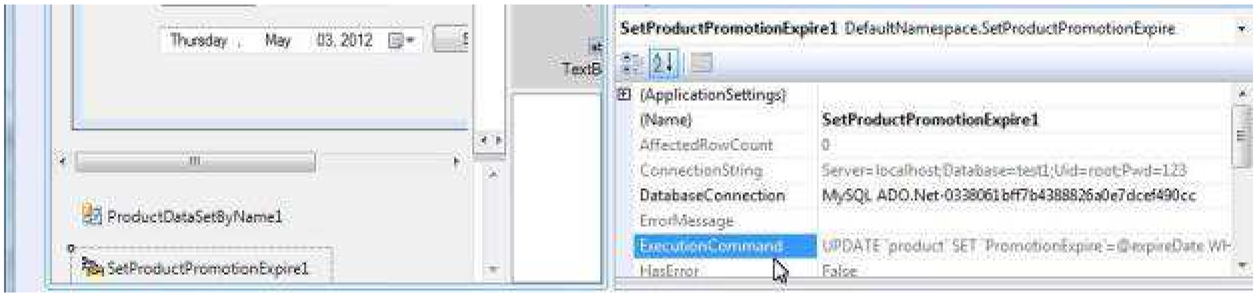
C:\Samples\ProductDataLayer\bin\Release\ProductDataLayer.dll

DefaultNamespace.ProductDataSetByName
DefaultNamespace.ProductGridViewByName
DefaultNamespace.SetProductPromotionExpire

SetProductPromotionExpire appears in the Toolbox. Drop it from the Toolbox to the form to use it.



Let’s examine the properties of the new SetProductPromotionExpire instance:

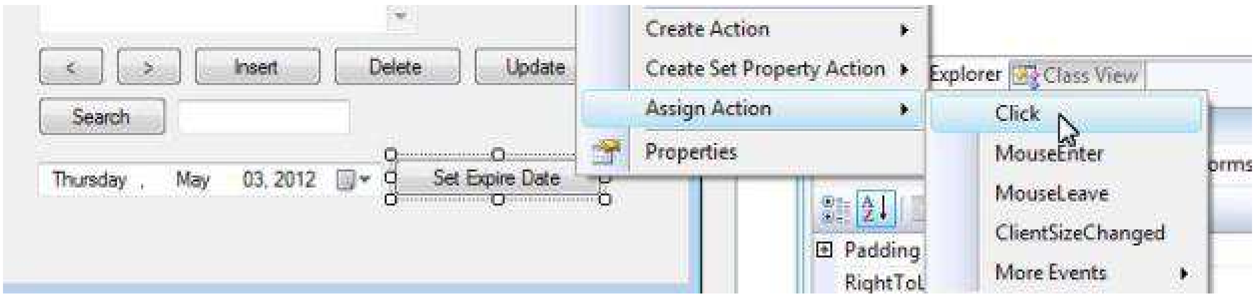

Thursday , May 03, 2012
ProductDataSetByName1
SetProductPromotionExpire1
SetProductPromotionExpire1 DefaultNamespace.SetProductPromotionExpire

(ApplicationSettings)	
(Name)	SetProductPromotionExpire1
AffectedRowCount	0
ConnectionString	Server=localhost;Database=test1;Uid=root;Pwd=123
DatabaseConnection	MySQL ADO.Net-0338061bff7b4388826a0e7dcef490cc
ErrorMessage	
ExecutionCommand	UPDATE `product` SET `PromotionExpire`=@expireDate WH
HasError	False

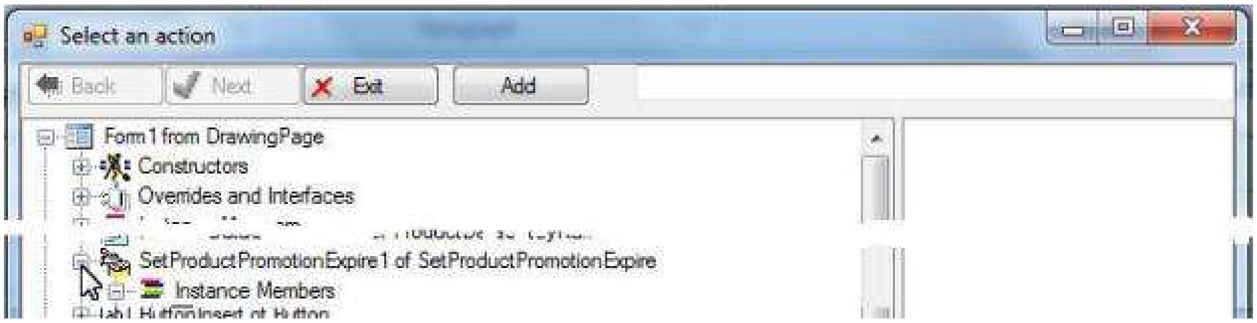
We can see that the property ExecutionCommand is read-only.

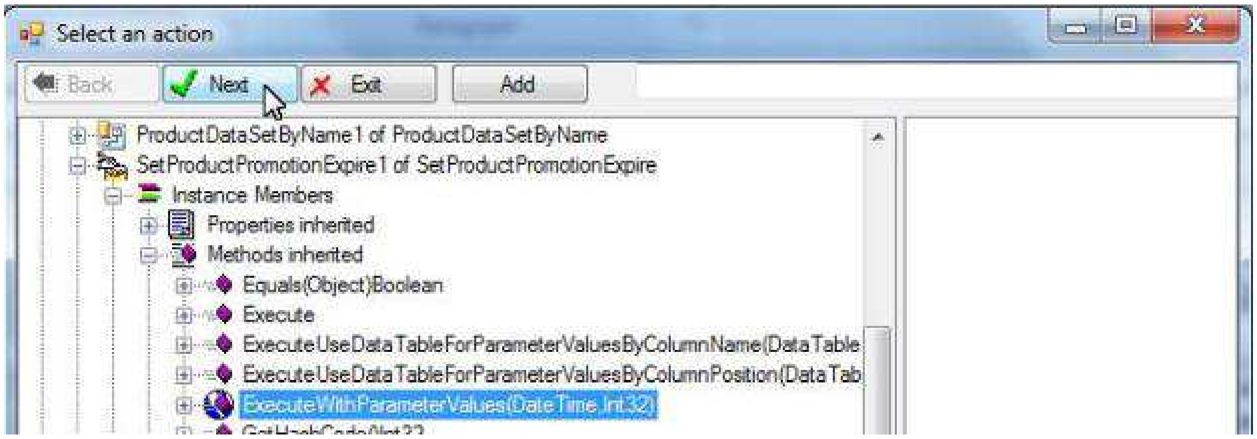
Execute data update action

Right-click button “Set expire date”; choose “Assign Action”; chose “Click” event:

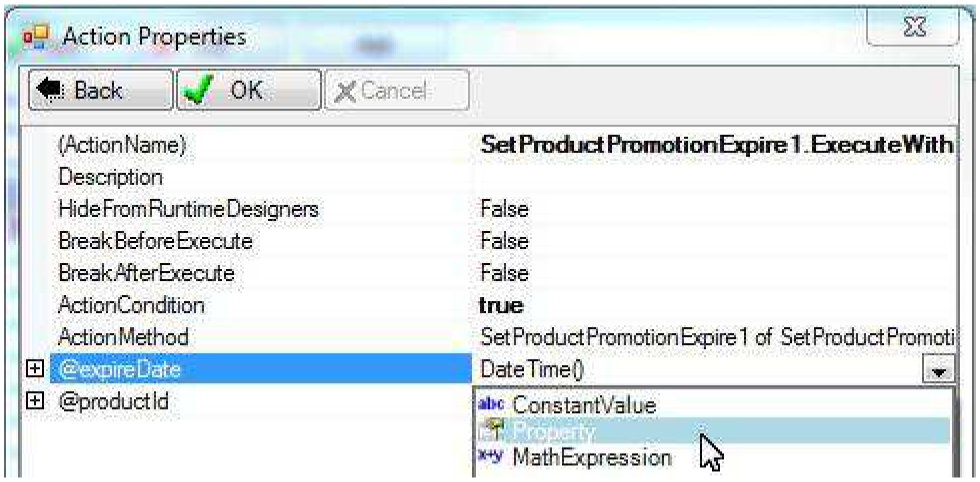


Select ExecuteWithParameterValues under SetProductPromotionExpire:

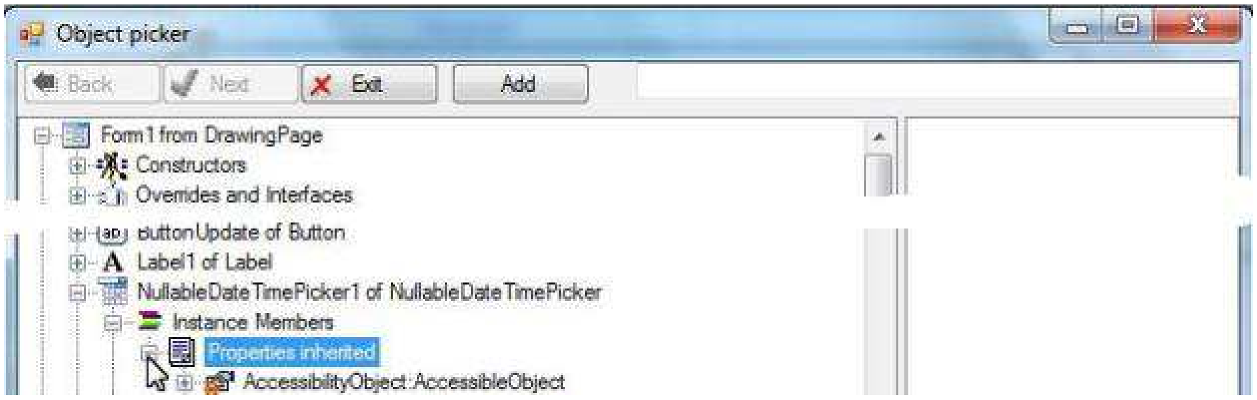


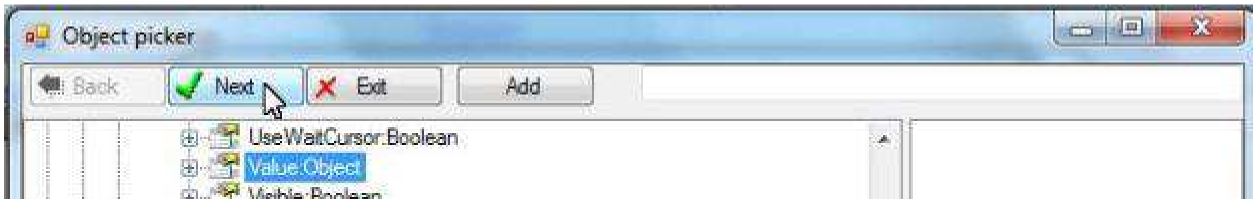

Select an action
Back
Next
Exit
Add
ProductDataSetByName1 of ProductDataSetByName
SetProductPromotionExpire1 of SetProductPromotionExpire
Instance Members
Properties inherited
Methods inherited
Equals(Object)Boolean
Execute
ExecuteUseDataTableForParameterValuesByColumnName(DataTable
ExecuteUseDataTableForParameterValuesByColumnPosition(DataTab
ExecuteWithParameterValues(DateTime,Int32)

Set @expireDate to the Value property of the date time picker:

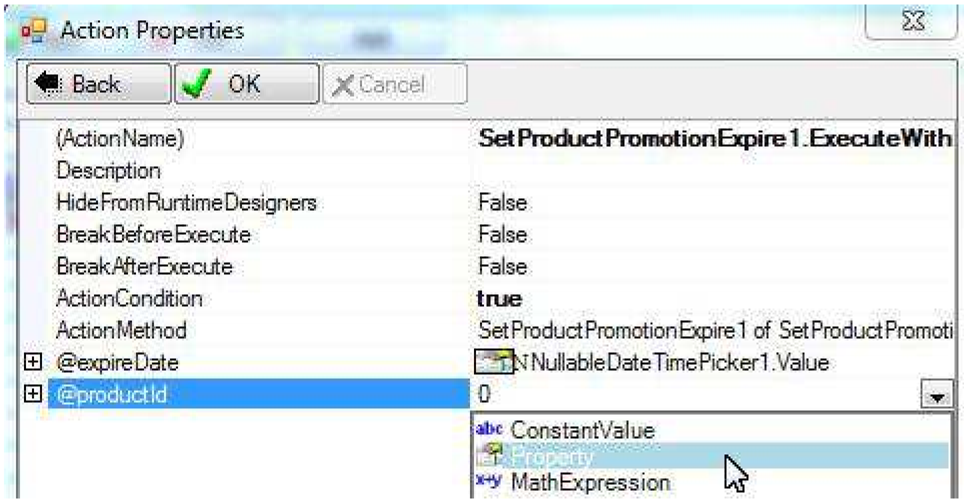


(ActionName)	SetProductPromotionExpire1.ExecuteWith
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	SetProductPromotionExpire1 of SetProductPromoti
@expireDate	DateTime()
@productId	

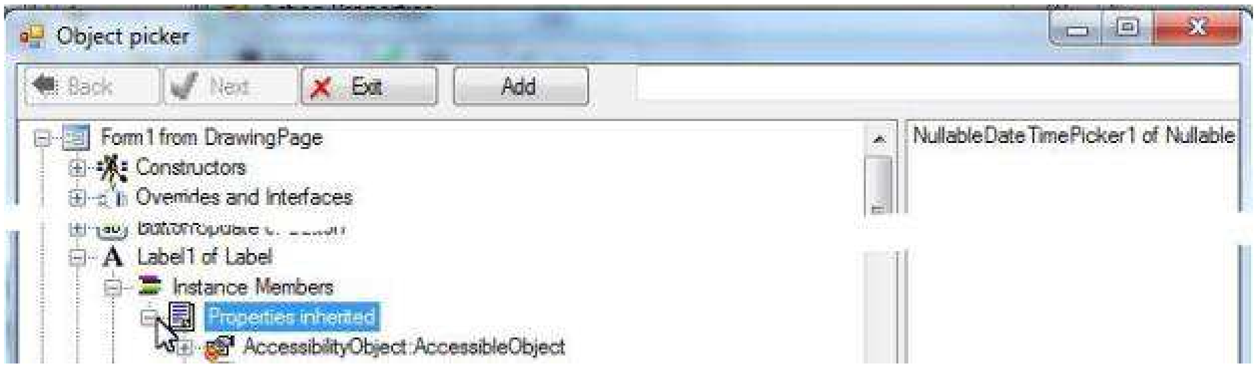


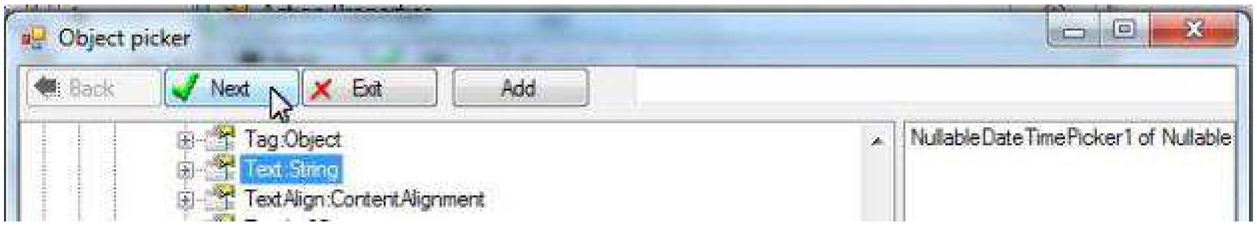


Set @productId to the Text property of the label which is bound to the ProductID field of the ProductDataSetByName:

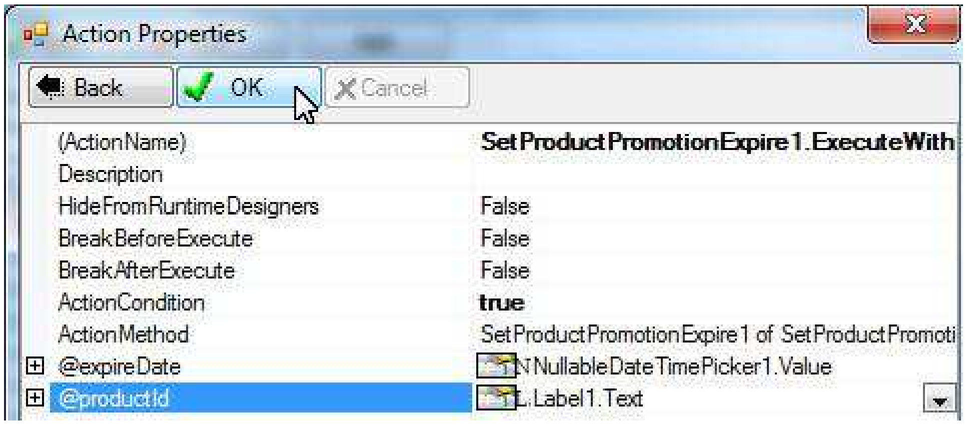


(ActionName)	**SetProductPromotionExpire1.ExecuteWith**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	SetProductPromotionExpire1 of SetProductPromoti
@expireDate	NullableDateTimePicker1.Value
@productId	0

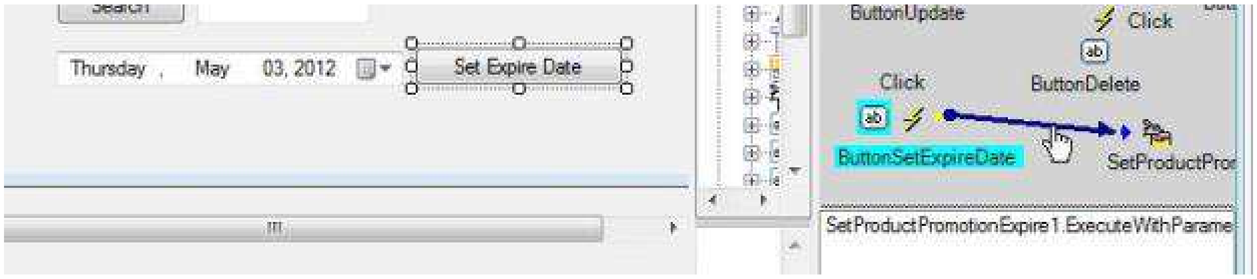



Object picker
Back
Next
Exit
Add
Tag:Object
Text:String
TextAlign:ContentAlignment
NullableDateTimePicker1 of Nullable

Click OK. The action is created and assigned to the button:



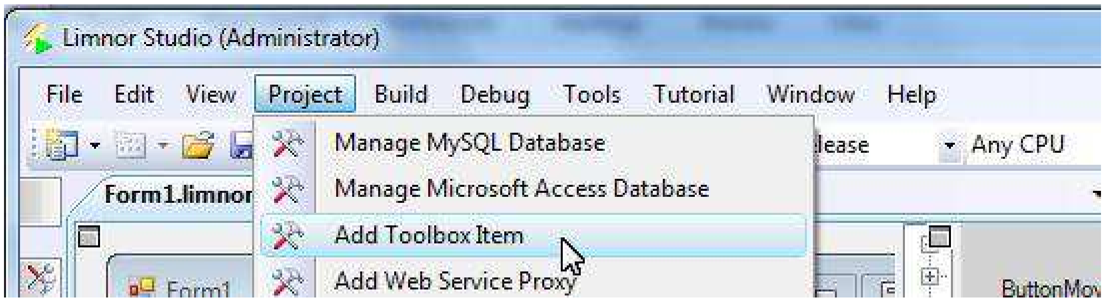
(ActionName)	**SetProductPromotionExpire1.ExecuteWith**
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	**true**
ActionMethod	SetProductPromotionExpire1 of SetProductPromoti
@expireDate	NullableDateTimePicker1.Value
@productId	Label1.Text



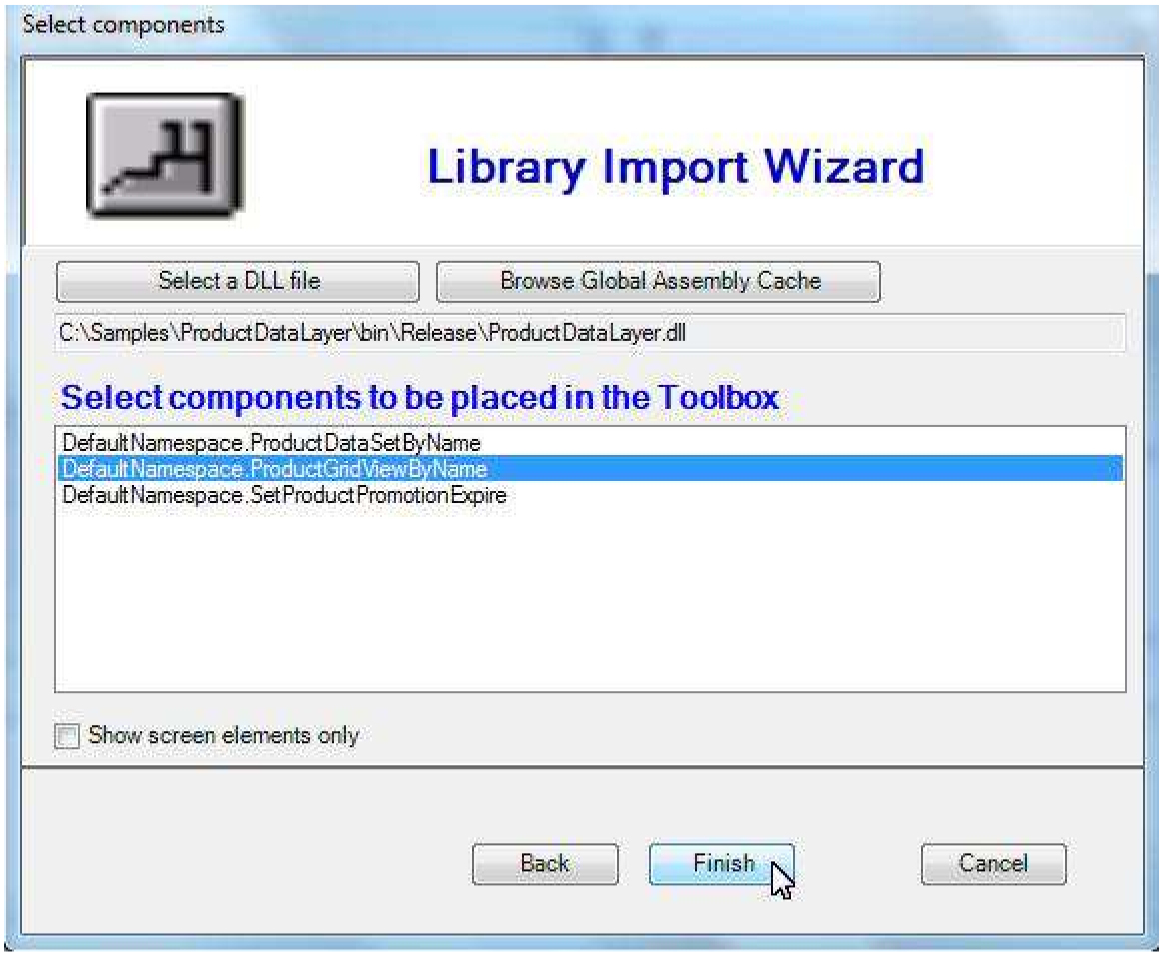
Use Derived Data Grid View

Use data layer

We'll add ProductGridViewByName to the Toolbox:



We'll skip the steps already shown in a previous section, and jump to the step of selecting class ProductGridViewByName from the data layer:

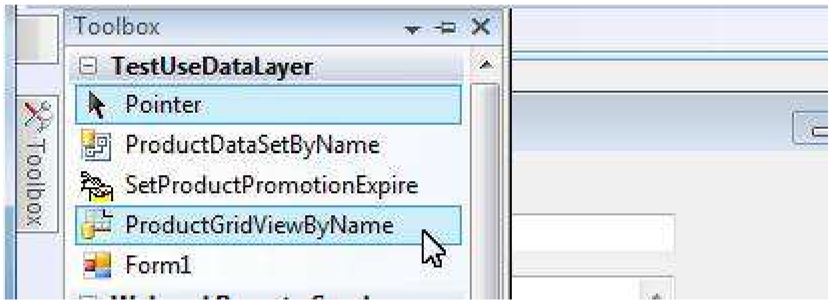


C:\Samples\ProductDataLayer\bin\Release\ProductDataLayer.dll

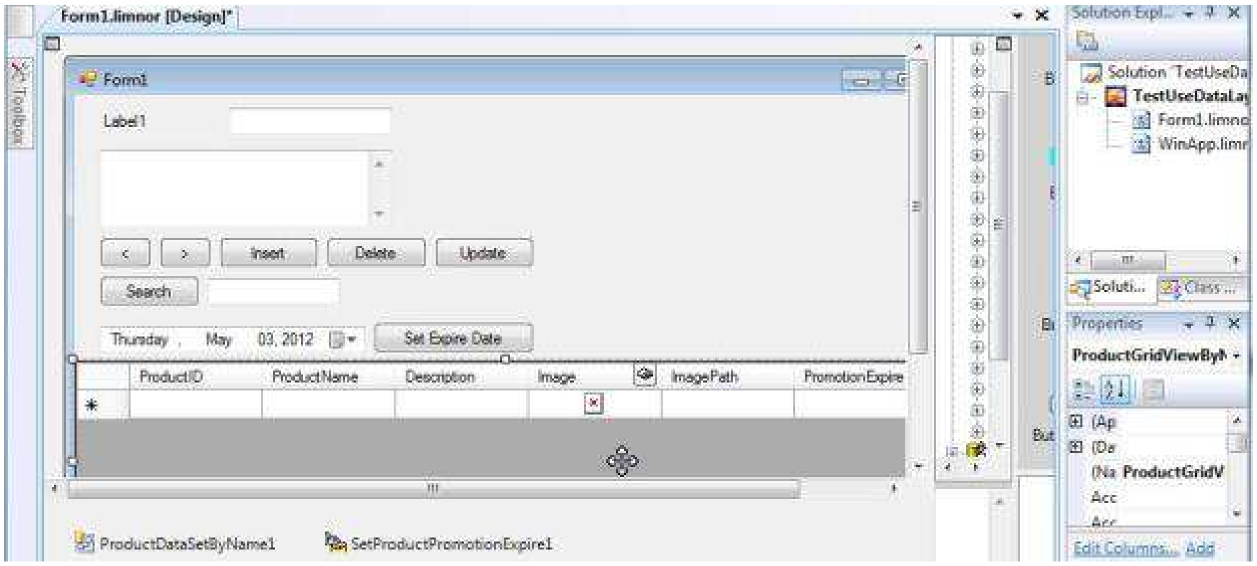
Select components to be placed in the Toolbox

DefaultNamespace.ProductDataSetByName
DefaultNamespace.ProductGridViewByName
DefaultNamespace.SetProductPromotionExpire

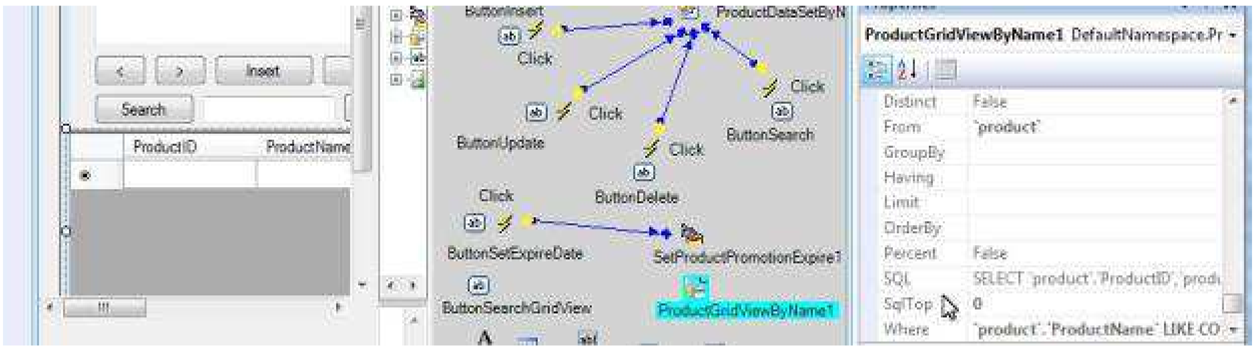
ProductGridViewByName appears in the Toolbox. Drop it from the Toolbox to the form to use it.



An instance of ProductGridViewByName appears on the form:



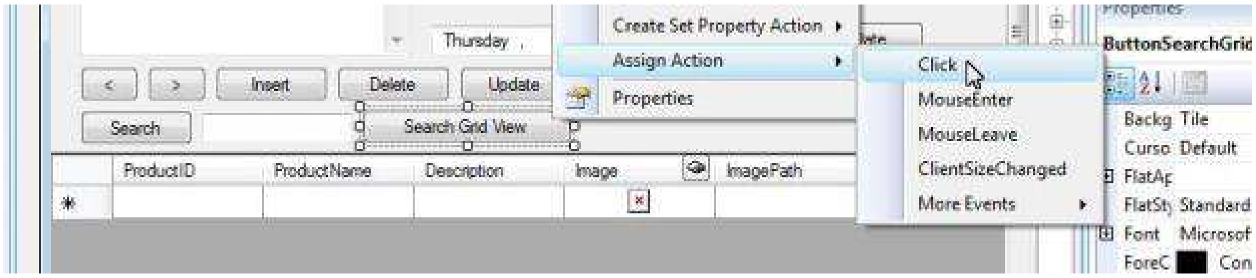
Like other classes in the data layer, the properties for database query of the class ProductGridViewByName are read-only:



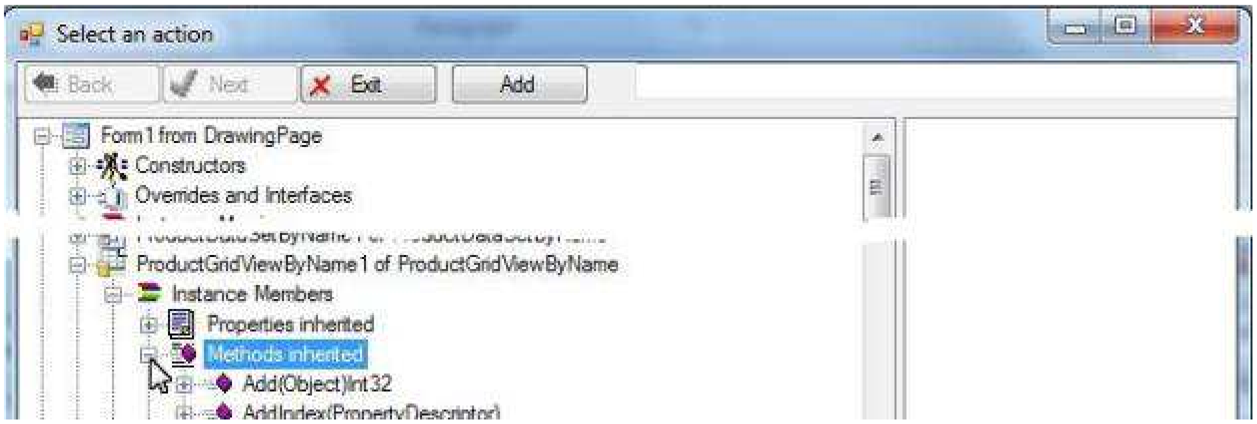
Distinct	False
From	`product`
GroupBy	
Having	
Limit	
OrderBy	
Percent	False
SQL	SELECT `product`.`ProductID`, prodt
SqlTop	0
Where	`product`.`ProductName` LIKE CO

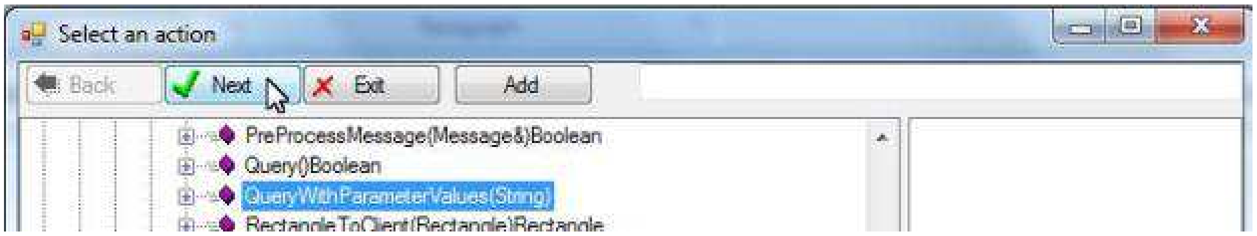
Execute actions on derived grid view

Because ProductGridViewByName allows searching product records by name, we use a button to do the search. Right-click the button; choose “Assign Action”; choose “Click” event:

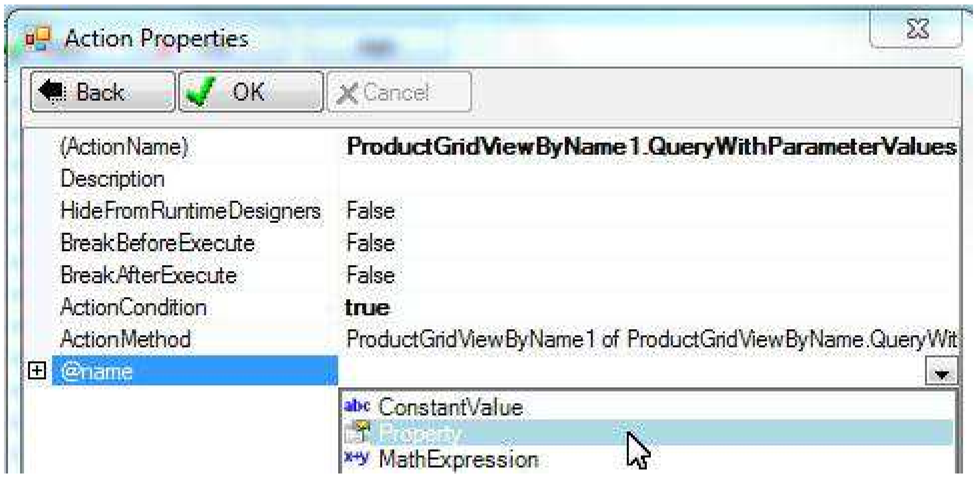


Select QueryWithParameterValues of ProductGridViewByName1:

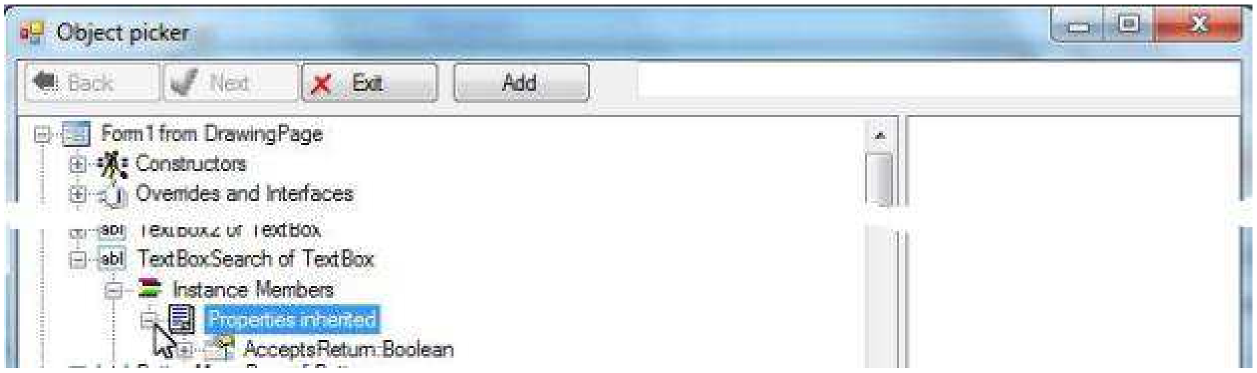


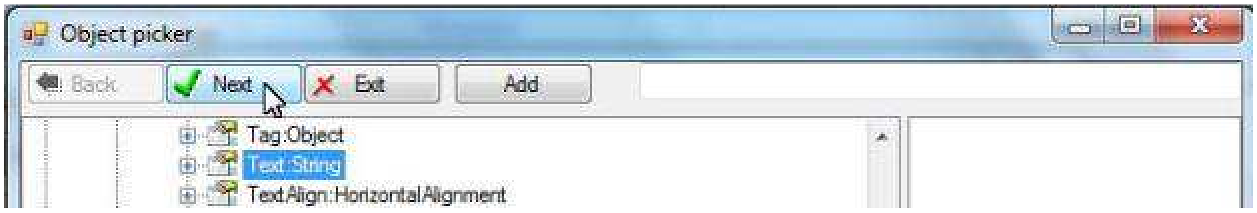

Select an action
Back
Next
Exit
Add
PreProcessMessage(Message&)Boolean
Query()Boolean
QueryWithParameterValues(String)
RectangleToClient(Rectangle)Rectangle

Select the Text property of the text box for searching product records:

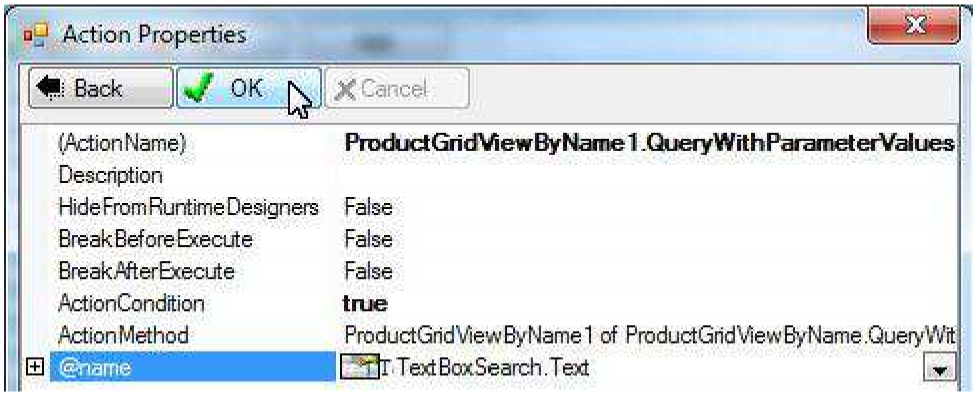


(ActionName)	ProductGridViewByName1.QueryWithParameterValues
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ProductGridViewByName1 of ProductGridViewByName.QueryWit
@name	

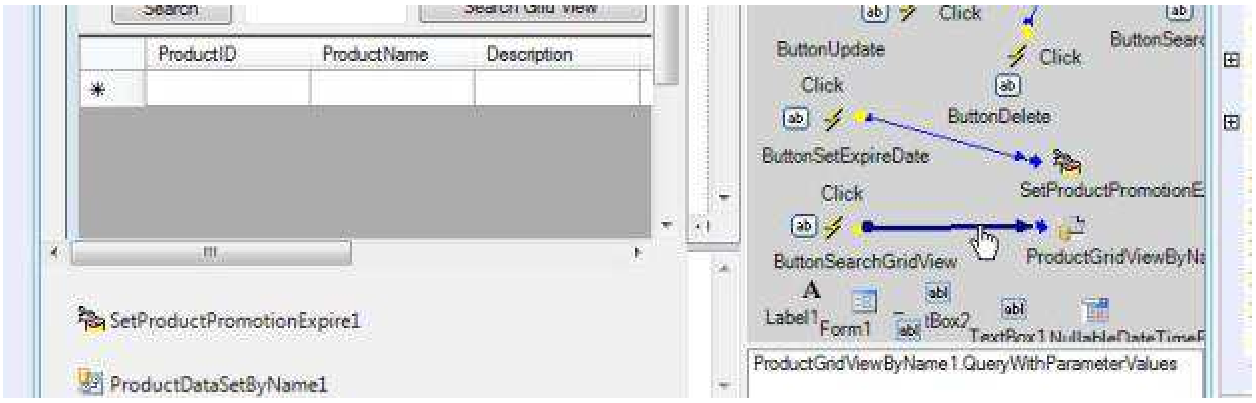




Click OK. The action is created and assigned to the button:

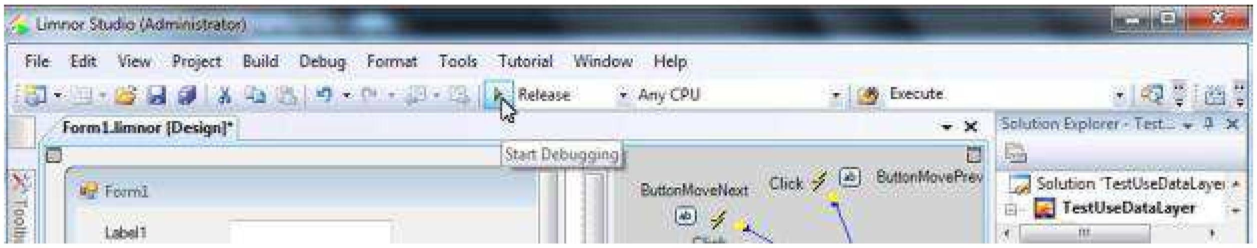


(ActionName)	ProductGridViewByName1.QueryWithParameterValues
Description	
HideFromRuntimeDesigners	False
BreakBeforeExecute	False
BreakAfterExecute	False
ActionCondition	true
ActionMethod	ProductGridViewByName1 of ProductGridViewByName.QueryWit
@name	TextBoxSearch.Text



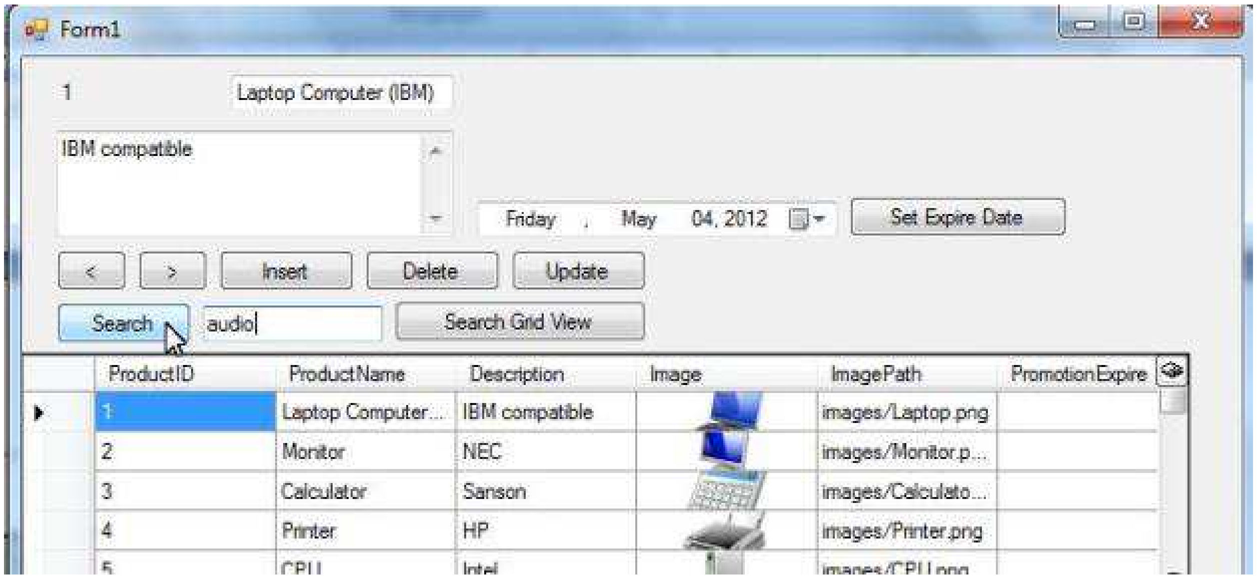
Test

We may compile and run the test application.


Limnor Studio (Administrator)
File Edit View Project Build Debug Format Tools Tutorial Window Help
Release
Any CPU
Execute
Form1.limnor [Design]*
Start Debugging
Solution Explorer - Test...
Form1
Label1
ButtonMoveNext
Click
ButtonMovePrev
Solution 'TestUseDataLayer'
TestUseDataLayer

Test searching dataset

Enter a string to search, for example, “audio”. Click “Search” button:



1 Laptop Computer (IBM)

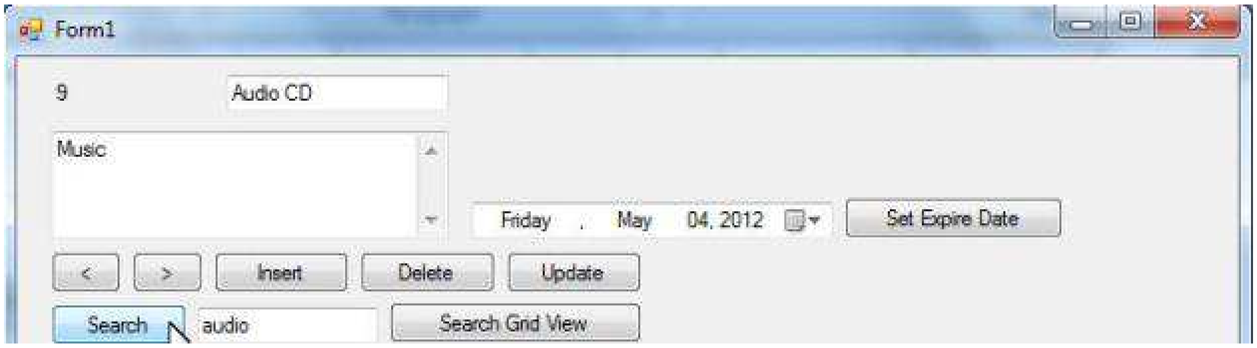
IBM compatible

Friday , May 04, 2012

audio

	ProductID	ProductName	Description	Image	ImagePath	PromotionExpire
▶	1	Laptop Computer...	IBM compatible		images/Laptop.png	
	2	Monitor	NEC		images/Monitor.p...	
	3	Calculator	Sanson		images/Calculato...	
	4	Printer	HP		images/Printer.png	
	5	CPU	Intel		images/CPU.png	

The search results appear:



9 Audio CD

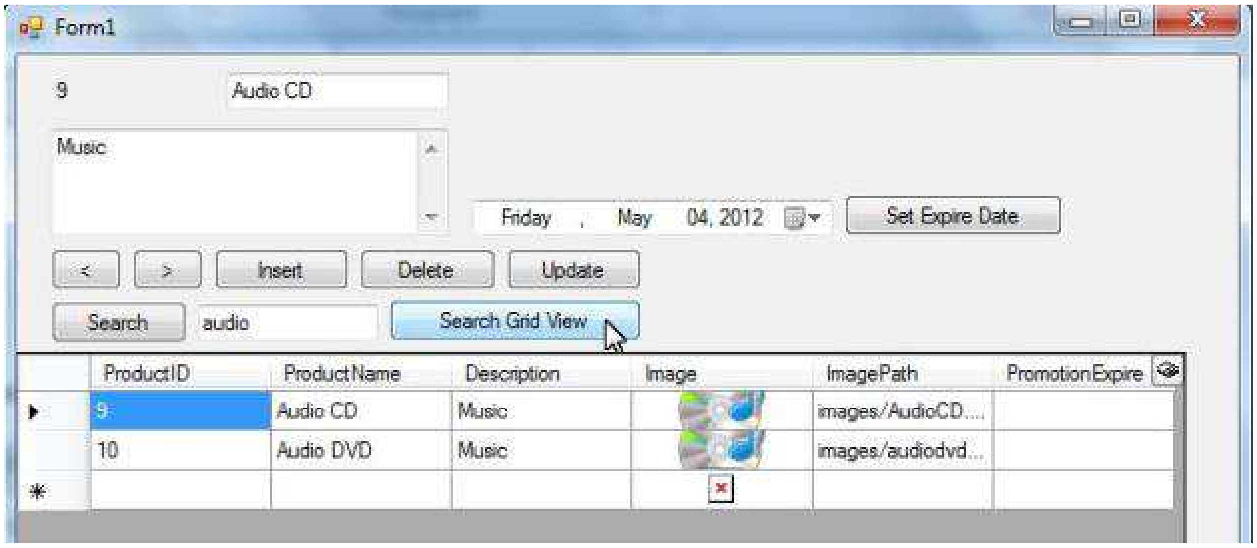
Music

Friday , May 04, 2012

audio

Test search data grid

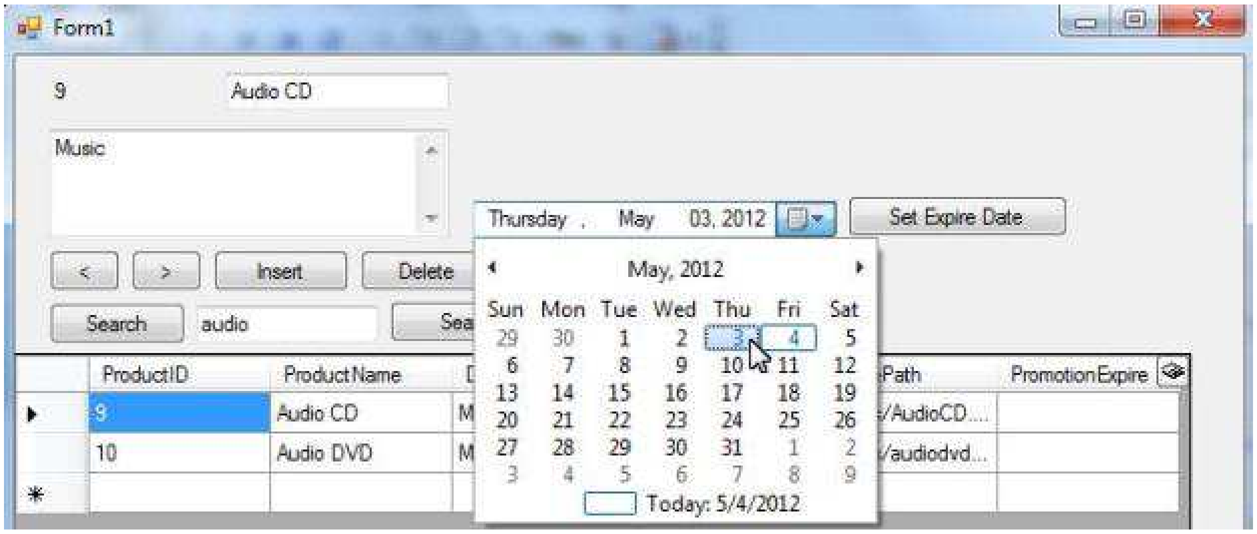
Click “Search Grid View”. The search results appear in the grid:



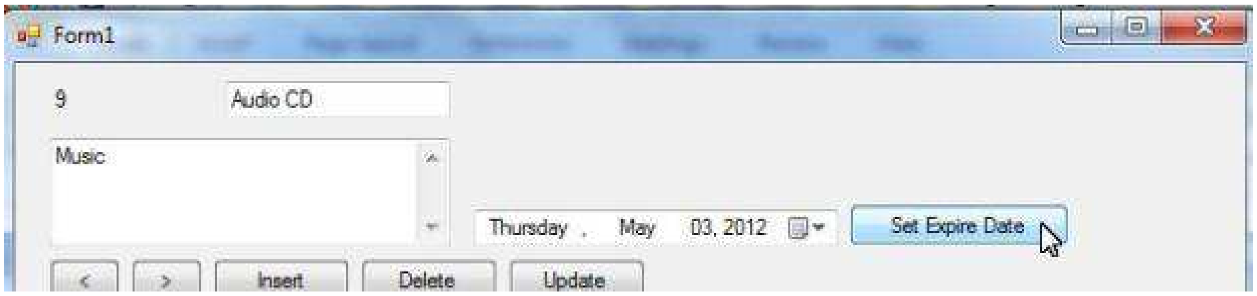
ProductID	ProductName	Description	Image	ImagePath	PromotionExpire
9	Audio CD	Music		images/AudioCD....	
10	Audio DVD	Music		images/audiodvd...	

Test data updater

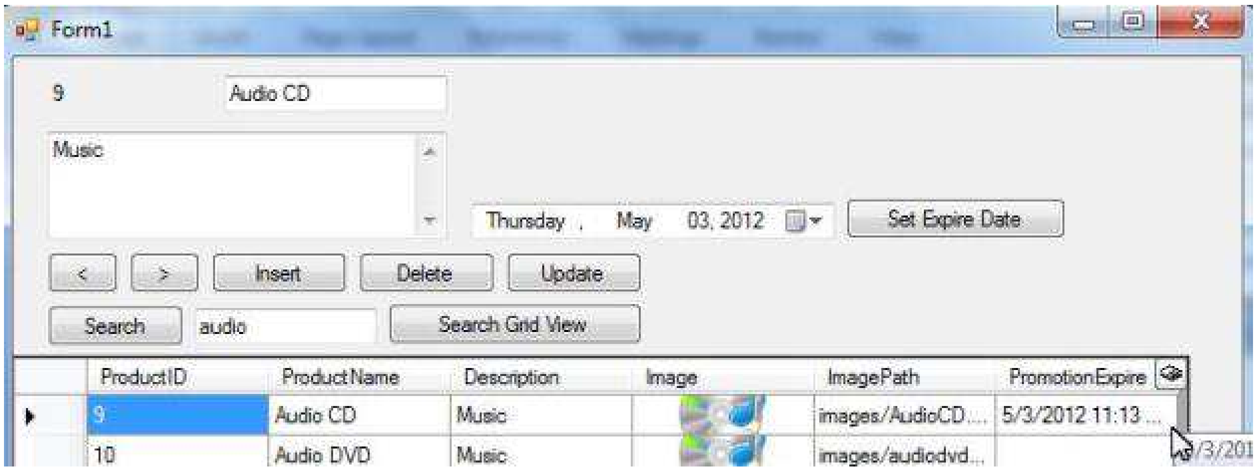
Let’s select a date time:



Click “Set Expire Date” button:



Click “Search Grid View” button again. We can see that the PromotionExpire column now has the date we selected:



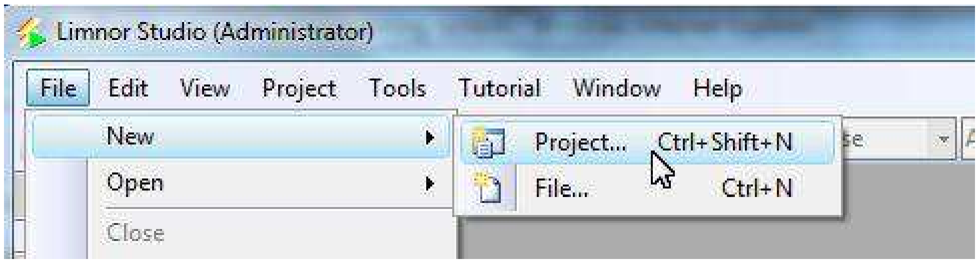
ProductID	ProductName	Description	Image	ImagePath	PromotionExpire
9	Audio CD	Music		images/AudioCD...	5/3/2012 11:13 ...
10	Audio DVD	Music		images/audiodvd...	

Note that this is for demonstration only. When a data grid is not read-only you may directly edit the column values. You may set field editors to make editing easier, for example, use date-time picker, use database lookup, etc. For details, see http://www.limnor.com/support/Limnor%20Studio%20-%20User%20Guide%20-%20Part%20VI.pdf

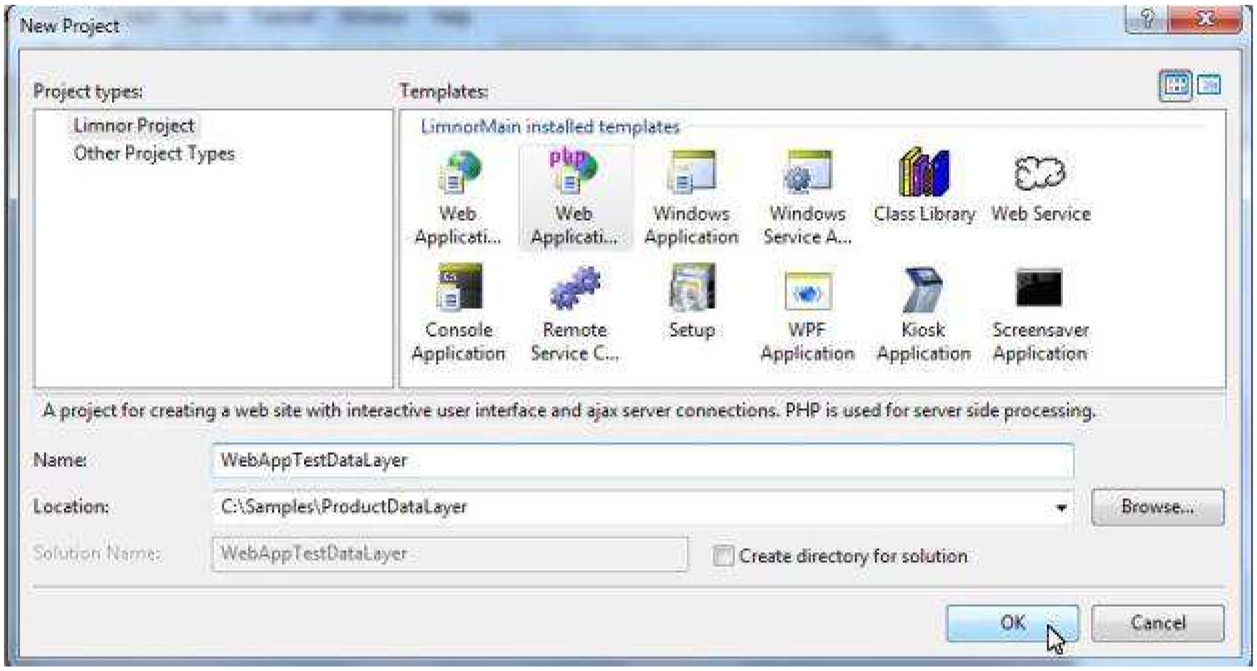
Use Data Layer in Web Applications

Let's create a web application and use the data layer.

Create web project



Create a PHP web application:


New Project

Project types:
Limnor Project
Other Project Types

Templates:
LimnorMain installed templates

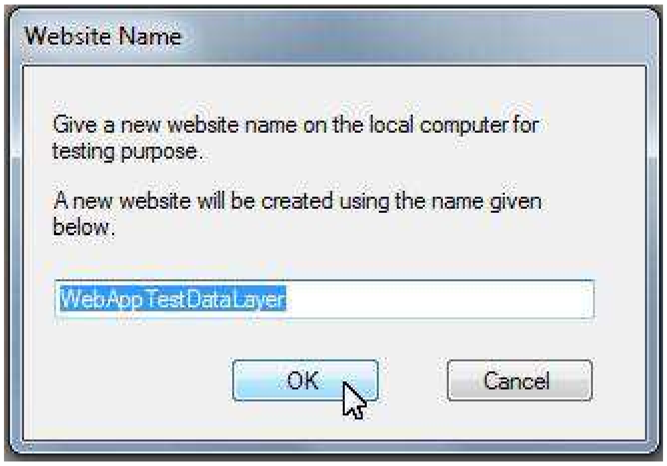
A project for creating a web site with interactive user interface and ajax server connections. PHP is used for server side processing.

Name: WebAppTestDataLayer

Location: C:\Samples\ProductDataLayer

Solution Name: WebAppTestDataLayer

Create directory for solution


Website Name

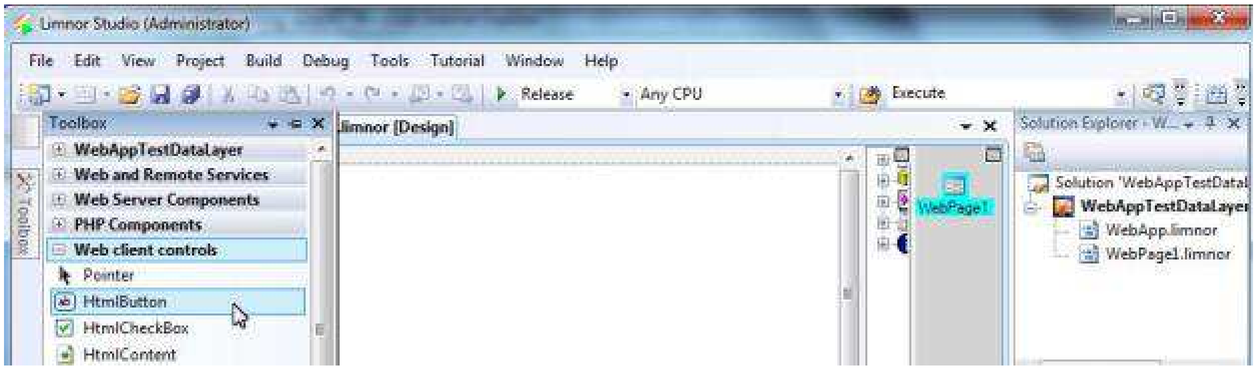
Give a new website name on the local computer for testing purpose.

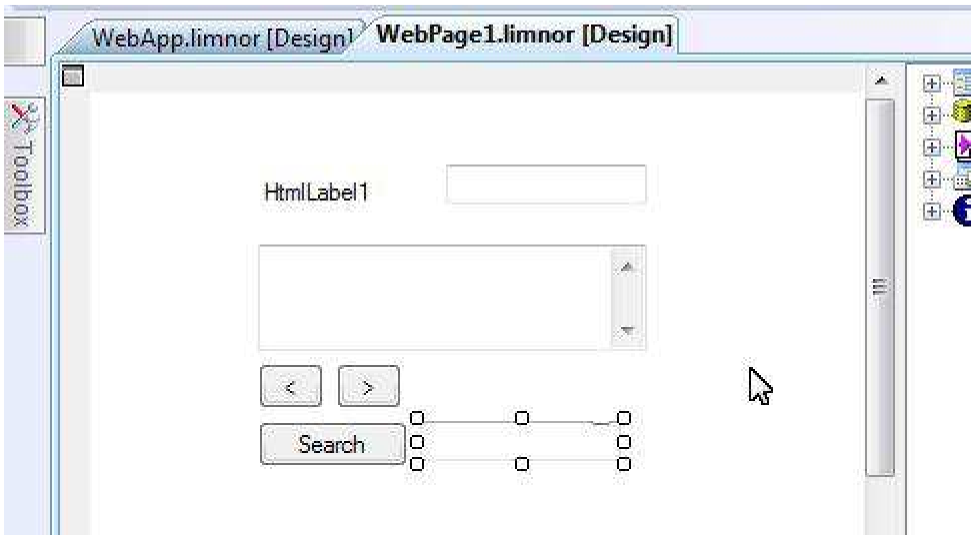
A new website will be created using the name given below.

WebAppTestDataLayer

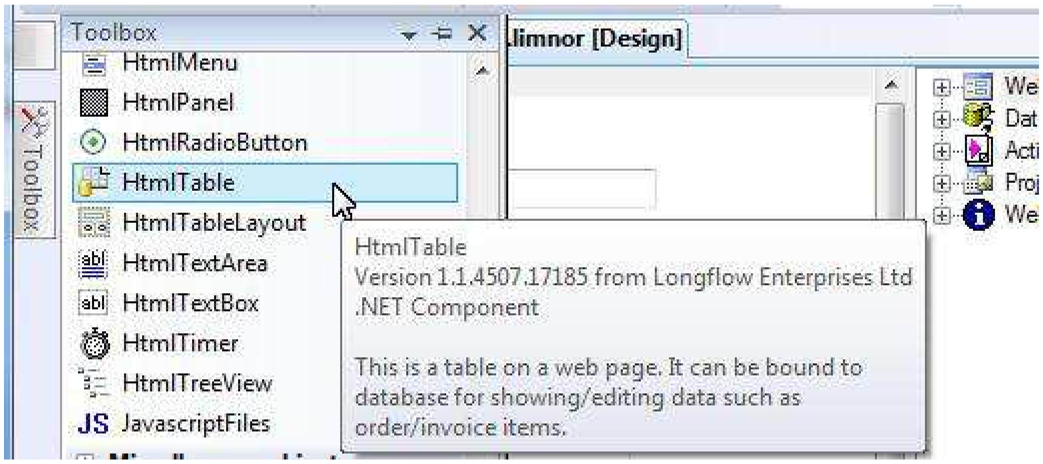
Create web page

Drop some web controls to the web page to form an UI similar to what we did for Window Form application:



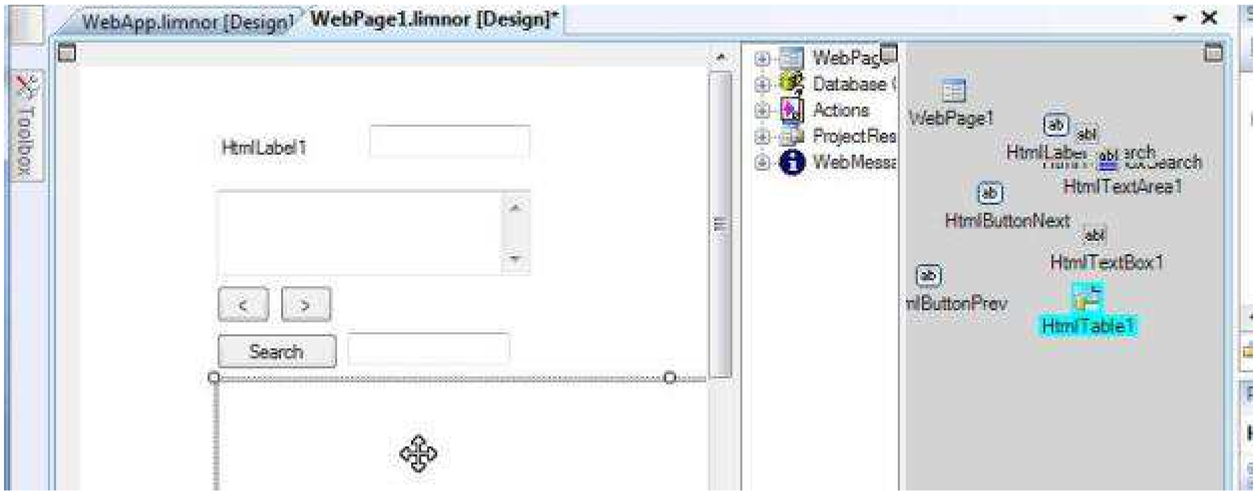


Note that EasyGrid cannot be used in a web page. We may use HtmlTable to provide a grid view:



HtmlTable
Version 1.1.4507.17185 from Longflow Enterprises Ltd
.NET Component

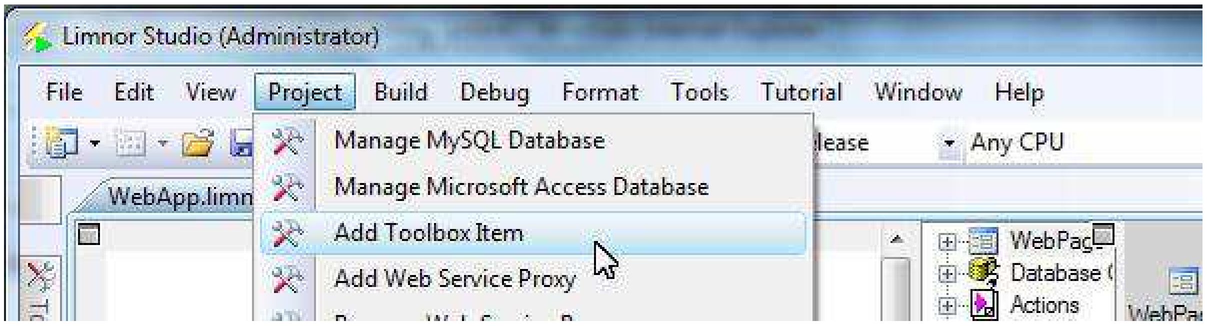
This is a table on a web page. It can be bound to database for showing/editing data such as order/invoice items.



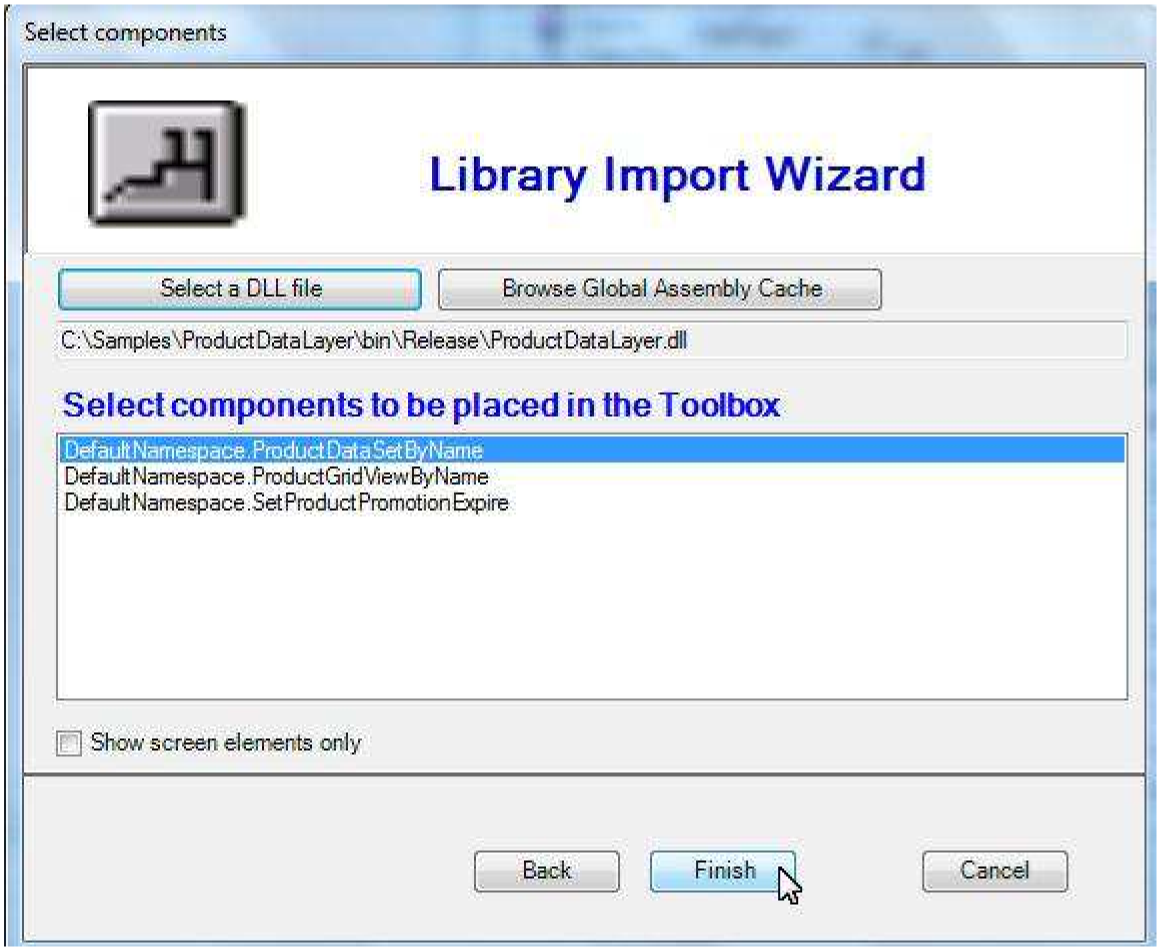
Use data layer

Access data layer

Add ProductDataSetByName to the Toolbox:



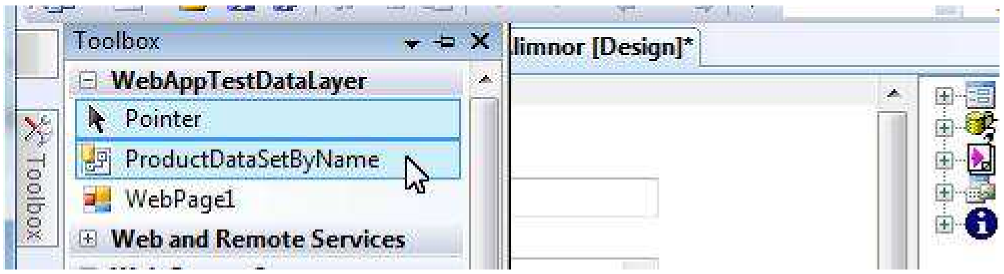
It is the same process as we did before. The last step is to select class ProductDataSetByName:


Select components
Library Import Wizard
Select a DLL file
Browse Global Assembly Cache
Select components to be placed in the Toolbox
Show screen elements only
Back
Finish
Cancel

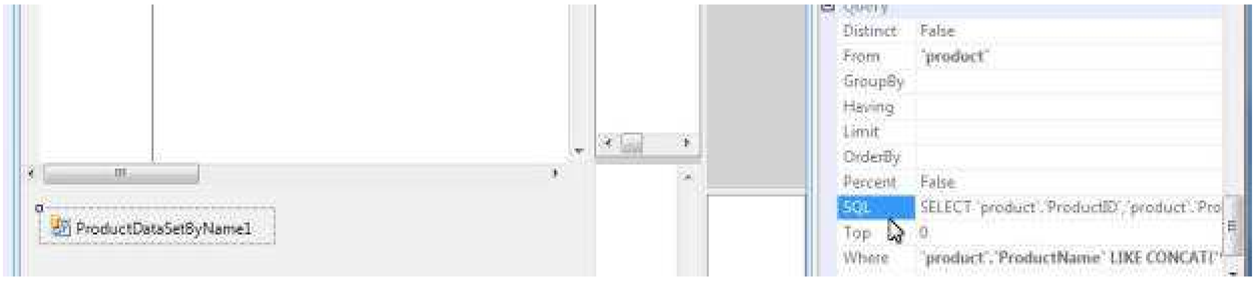
C:\Samples\ProductDataLayer\bin\Release\ProductDataLayer.dll

DefaultNamespace.ProductDataSetByName
DefaultNamespace.ProductGridViewByName
DefaultNamespace.SetProductPromotionExpire

ProductDataSetByName appears in the Toolbox. Drop it to the web page to use it:



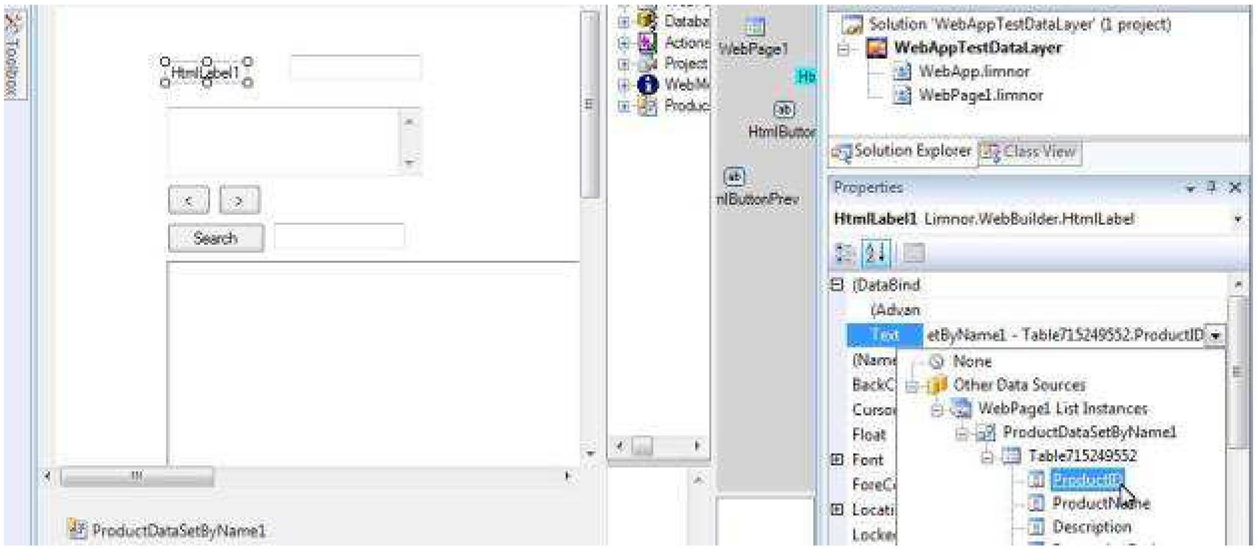
A new instance of ProductDataSetByName appears. We can see that its properties defining the database query are read-only



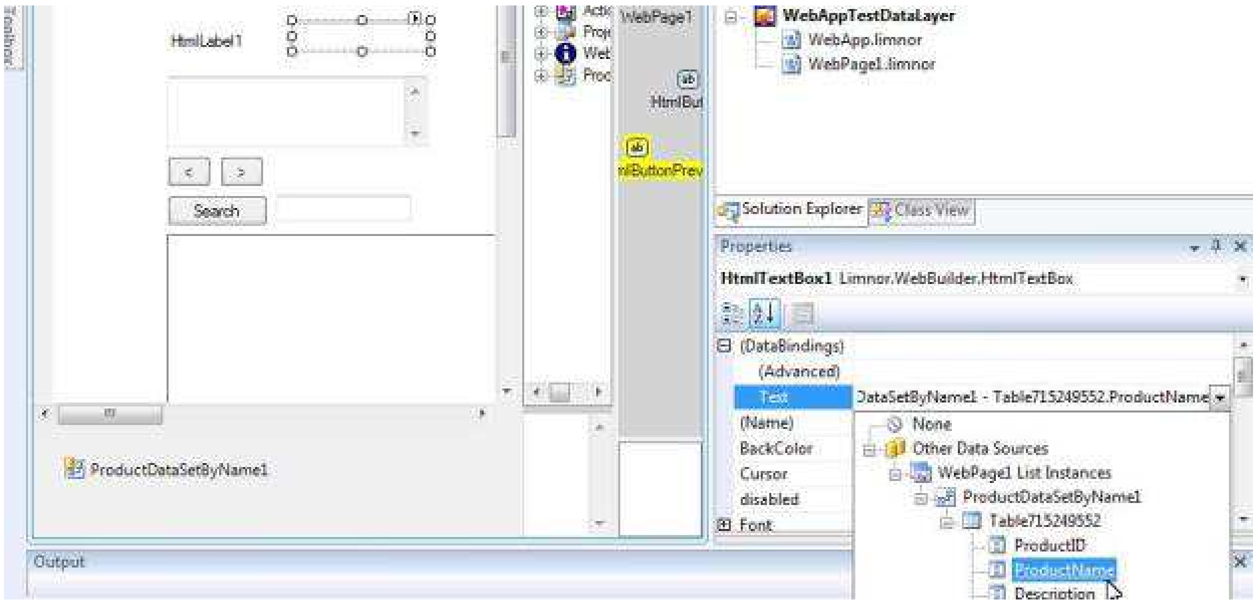
Distinct	False
From	'product'
GroupBy	
Having	
Limit	
OrderBy	
Percent	False
SQL	SELECT 'product'.'ProductID','product'.'Pro
Top	0
Where	'product'.'ProductName' LIKE CONCAT('

Data-binding

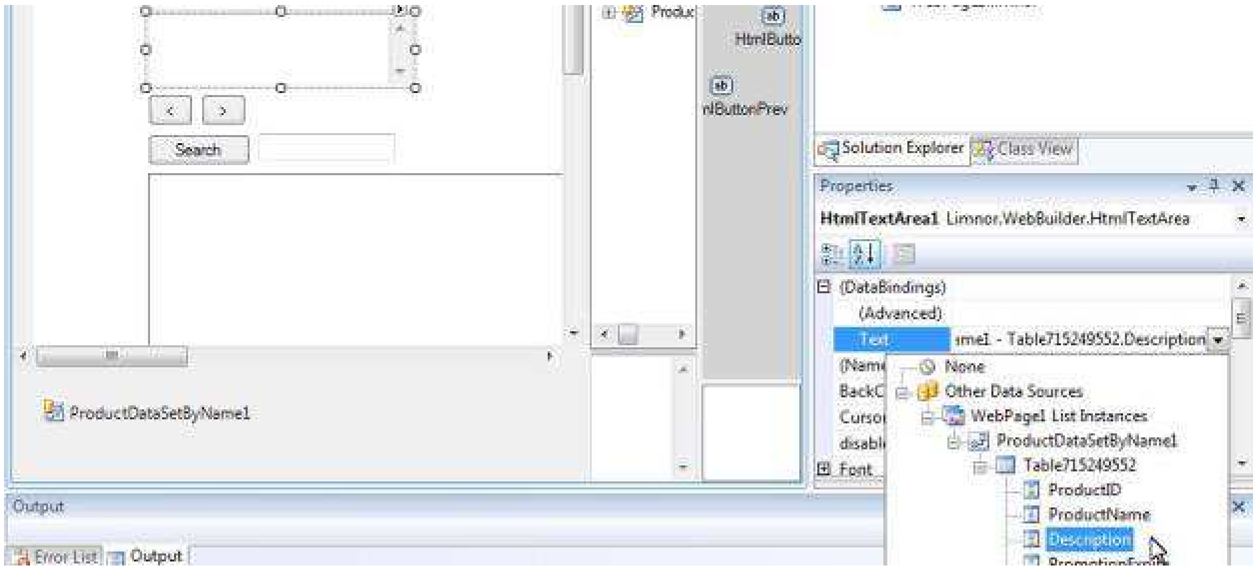
Bind the label to ProductID:



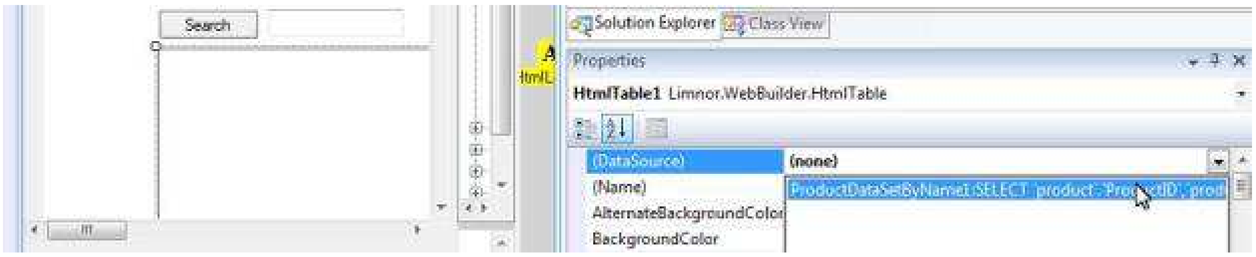
Bind the text box to ProductName:



Bind the text area to Description:

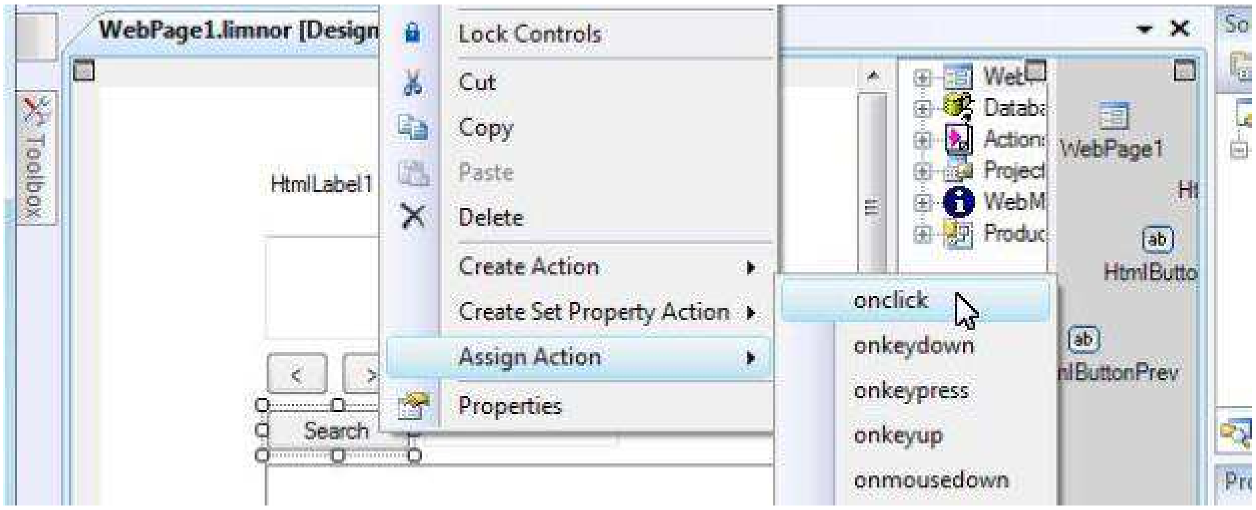


Set the DataSource property of the HtmlTable to ProductDataSetByName1:

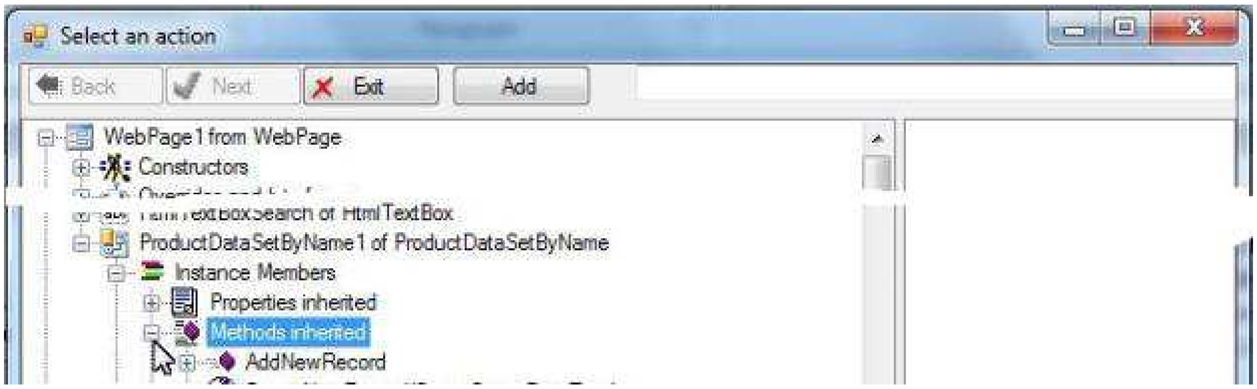


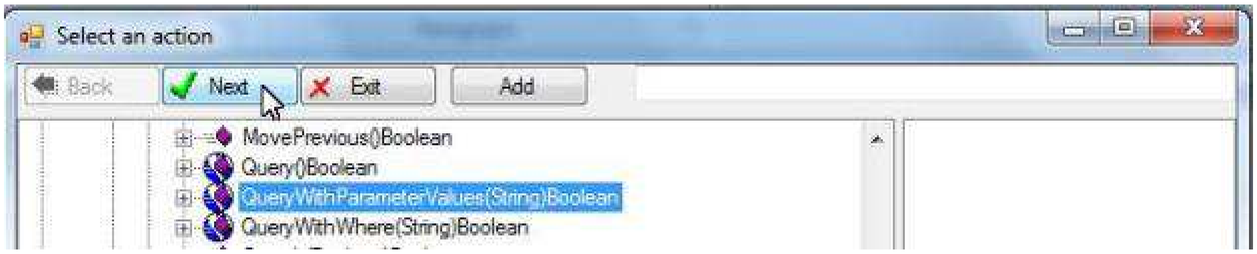
Execute search by name

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

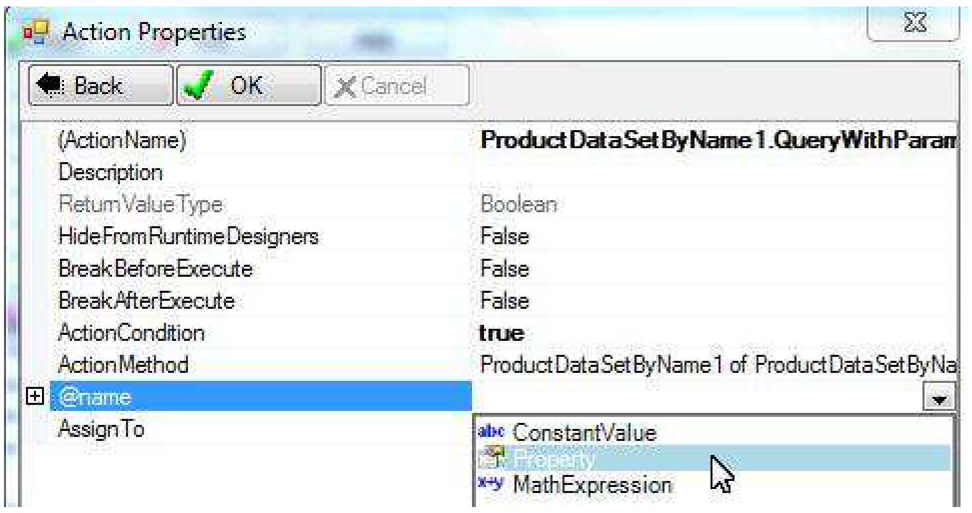


Select QueryWithParameterValues under ProductDataSetByName1:

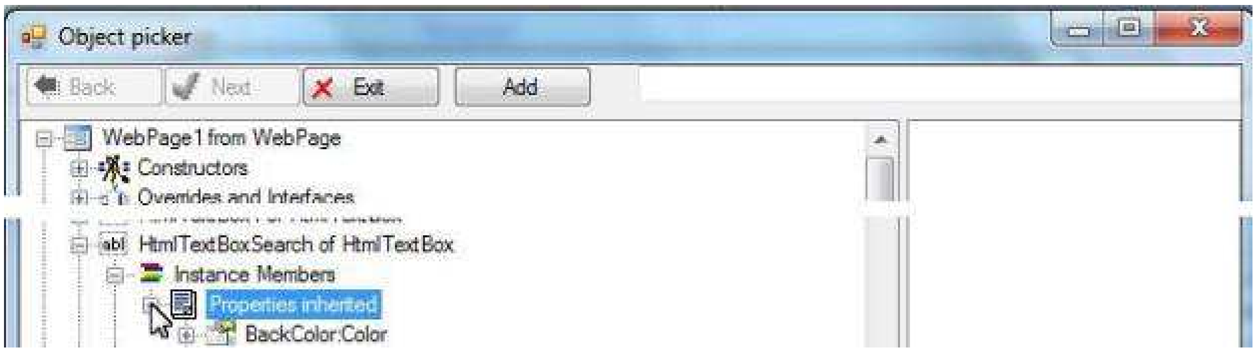


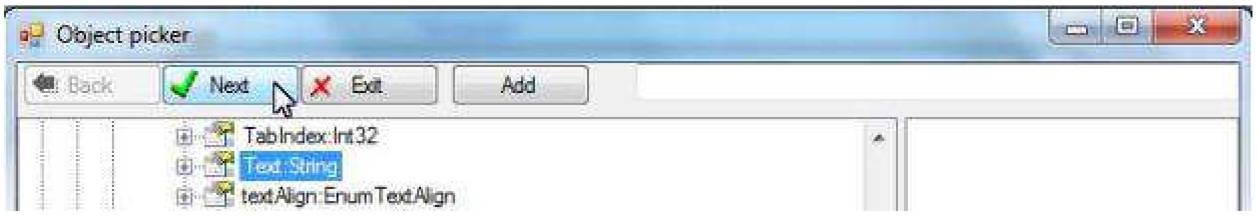

Select an action
Back
Next
Exit
Add
MovePrevious()Boolean
Query()Boolean
QueryWithParameterValues(String)Boolean
QueryWithWhere(String)Boolean

For @name parameter, select Text property of the text box for searching:

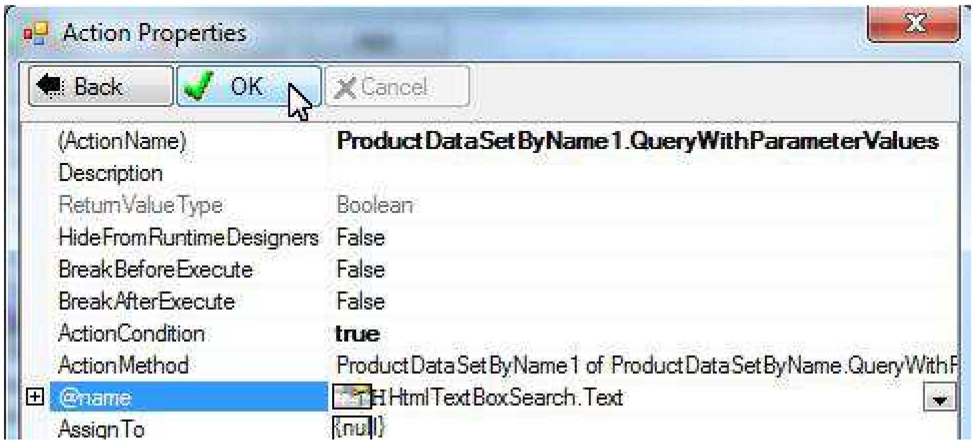


(ActionName) ProductDataSetByName1.QueryWithParam
Description
ReturnValueType Boolean
HideFromRuntimeDesigners False
BreakBeforeExecute False
BreakAfterExecute False
ActionCondition **true**
ActionMethod ProductDataSetByName1 of ProductDataSetByNa
@name
AssignTo

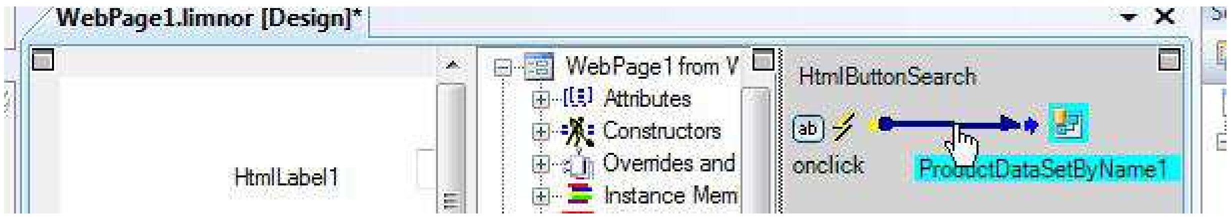




Click OK. The action is created and assigned to the button.

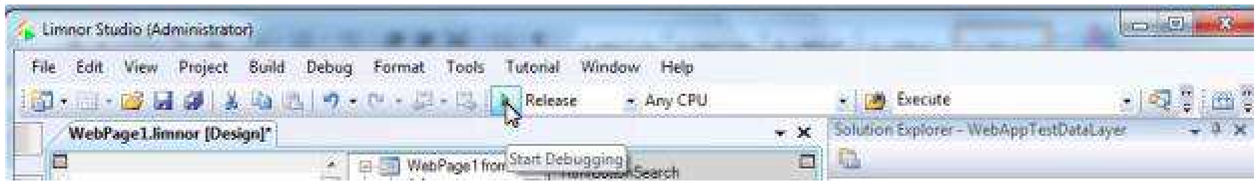


(ActionName) ProductDataSetByName1.QueryWithParameterValues
Description
ReturnValueType Boolean
HideFromRuntimeDesigners False
BreakBeforeExecute False
BreakAfterExecute False
ActionCondition **true**
ActionMethod ProductDataSetByName1 of ProductDataSetByName.QueryWithP
@name HtmlTextBoxSearch.Text
AssignTo {null}

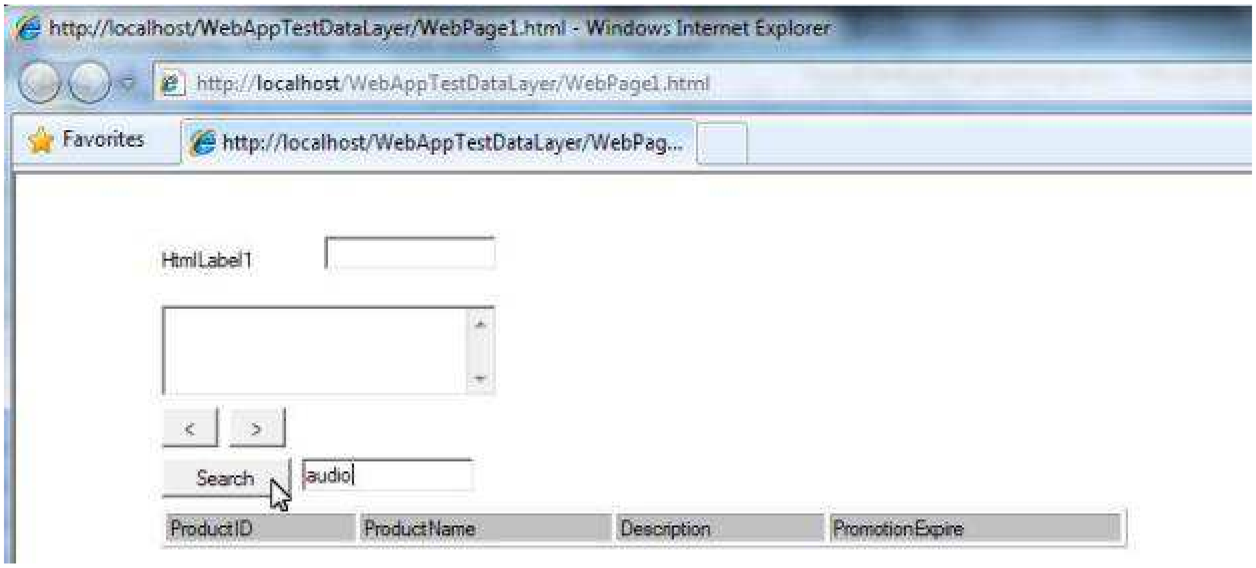


Test

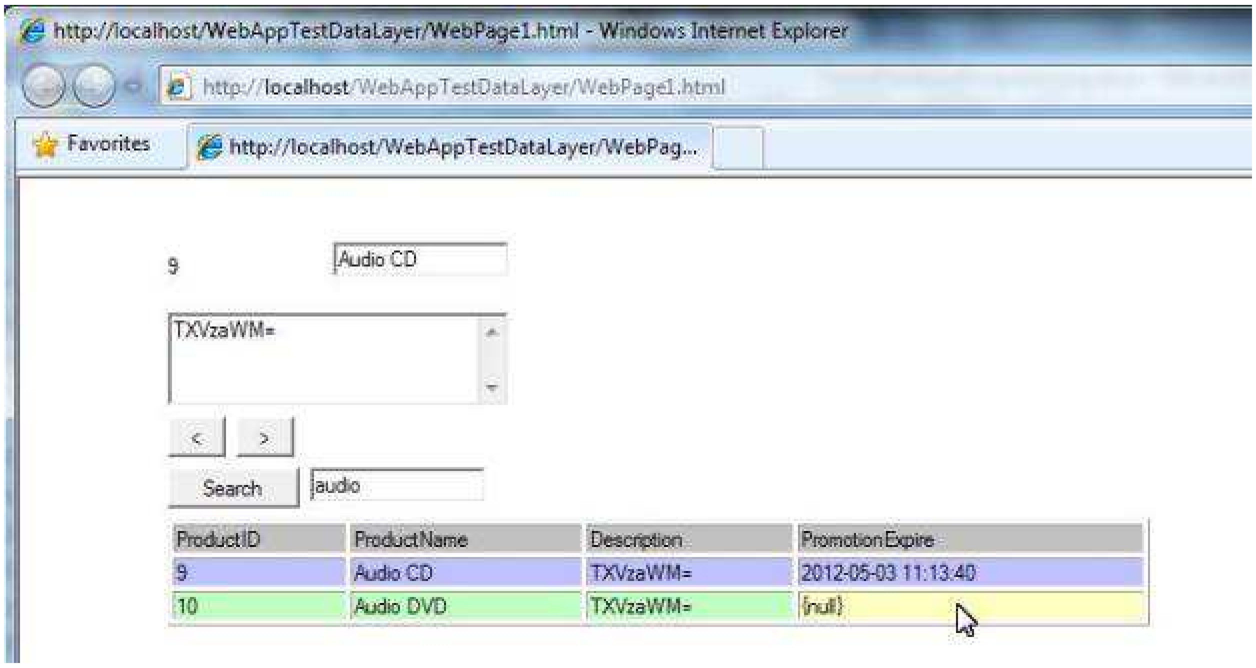
Click the Run button to compile the project and start it in the default web browser:


Limnor Studio (Administrator)
File Edit View Project Build Debug Format Tools Tutorial Window Help
Release
Any CPU
Execute
WebPage1.limnor [Design]*
Solution Explorer - WebAppTestDataLayer
Start Debugging

The web page appears. Enter a search text; click the Search button:


http://localhost/WebAppTestDataLayer/WebPage1.html - Windows Internet Explorer
http://localhost/WebAppTestDataLayer/WebPage1.html
Favorites
http://localhost/WebAppTestDataLayer/WebPag...
HtmlLabel1
<
>
Search
audio
ProductID
ProductName
Description
PromotionExpire

The search results appear:


http://localhost/WebAppTestDataLayer/WebPage1.html - Windows Internet Explorer
http://localhost/WebAppTestDataLayer/WebPage1.html
Favorites
http://localhost/WebAppTestDataLayer/WebPag...
<
>
Search

9

Audio CD

TXVzaWM=

audio

ProductID	ProductName	Description	PromotionExpire
9	Audio CD	TXVzaWM=	2012-05-03 11:13:40
10	Audio DVD	TXVzaWM=	{null}

Conclusion

The database components, EasyDataSet, EasyUpdator, and EasyGrid, shipped with Limnor Studio may be used in n-tire programming via derivation. Properties related to query and database command of derived classes are already defines and thus the applications using those classes do not need to do such work again. Those properties are read-only so that they are always correct once tests passed.