Lesson 11-1: Use the LinqDataSource control

The easiest way to bind data to a control on a page is to use a data source control. There are a few of these in the *Data* category of the *Toolbox*, but in this lesson you're going to use the *LinqDataSource* control.

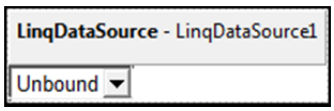
The *LinqDataSource* control retrieves data using *LINQ to SQL Classes*.

1 Open *Spark* from your sample files folder.

2 Open *newtransaction.aspx* in *Design* view.

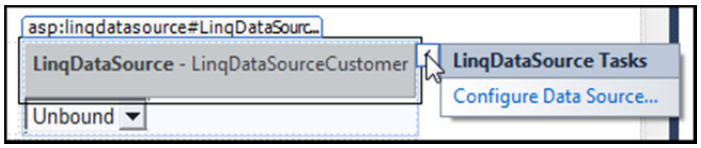
3 Add a *LinqDataSource* control to the page.

Drag a *LinqDataSource* control from the *Data* category of the *Toolbox* onto the page. Place it just before the *Customer* drop down.



4 Configure the *LinqDataSource* control.

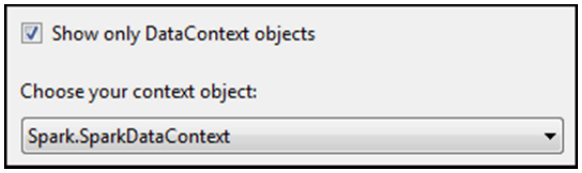
1. Set the *ID* property of the *LinqDataSource* control to: **LinqDataSourceCustomer**
2. Click *Configure Data Source…* from the *QuickTasks* menu of the *LinqDataSource* control.



This opens the configuration dialog for the control.

3. Make sure *Spark.SparkDataContext* is selected.

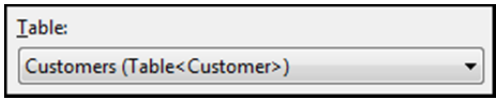
In the first step of configuration, you select which LINQ data context (ie dbml file) to use.



Since you only have one it should already be selected. If it isn't, select it.

4. Click *Next*.
5. Select the table to retrieve data from.

This data source is going to retrieve a list of customers for the *DropDownListCustomer* control to use, so select the *Customers* table from the *Table* dropdown.



note

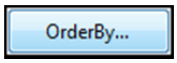
Placing data sources

In this lesson you place the *LinqDataSource* control near the *DropDownList* control it is going to populate, but the placement of a data source control doesn't matter.

Data sources aren't visible to people visiting the page, so you can put them wherever seems most logical.

If you are not completing the course incrementally use the sample file: **Lesson 11-1** to begin this lesson.

Sample files with the starting point for each lesson are also provided for all of the other lessons in this session.

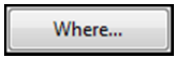


6. Click *OrderBy…*.

   This opens a dialog to configure sorting. Sorting works exactly the same way as using the *OrderBy* and *OrderByDescending* methods in LINQ (see: *Lesson 10-5: Sort results and call stored procedures using LINQ*).

7. Select *CustomerName* and *Ascending*.

8. Click *OK*.



9. Click *Where…*.

   This opens a dialog to configure query criteria, which work the same way as the *Single* and *Where* methods in LINQ (see: *Lesson 10-3: Retrieve a single row of data using LINQ*).

10. Select *CustomerID* from the *Column* drop-down.

    This tells the *LinqDataSource* control that *CustomerID* is the database column you want to use for your criteria.

11. Select *!=* from the *Operator* drop-down.

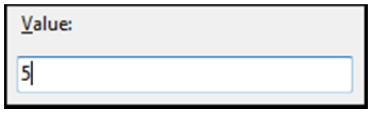
    This is the 'does not equal' operator. You learned about it in: *Lesson 7-3: Use basic logical operators*.

12. Select *None* from the *Source* drop-down.

    The *Source* drop-down allows you to tell your data source to retrieve a value from several different sources (see sidebar).

    In this case, you want a value that doesn't change. *None* allows you to type in the value you want to use.

13. Enter **5** into the Parameter Properties *Value* text box.



14. Click *Add*.

    Your new parameter has been added. This data source will now retrieve everything from the *Customer* table except for *CustomerID* 5.

15. Click *OK*.

16. Click *Finish*.

note

Other *Source* options

In this lesson, you use the *None* source to specify your own value for a parameter, but you can choose between several sources.

Control
Retrieve a value from a control on the page.

Cookie
Retrieve a value from a cookie.

Form
Retrieve a value from: *Page.Request.Form*

Profile
Retrieve a value from a user profile.

QueryString
Retrieve a value from: *Page.Request.QueryString*

Session
Retrieve a value from a *Session* variable.

Route
Retrieve a value from a URL.