List View and Data Pager Controls

Contents

The listview control is an extremely flexible control. It can be used for the same situations as the gridview, datalist, repeater and formview controls.

The datapager control works with the listview control to enable paging.

The ListView Control

The listview control can be used to edit, delete, select, page, sort and display data like a gridview. However it also has the flexibility of a repeater control in that it is completely template driven. It also allows the insertion of data which the gridview control cannot do.

The listView control supports the following templates:

LayoutTemplate	Used to specify the containing element for the contents of the ListView. Rendered only once.
ItemTemplate	Used to format each item
ItemSeperatorTemplate	Used to display content between each item
GroupTemplate	Used to specify the containing element for a group of items
GroupSeperatorTemplate	Used to display content between each group of items
EmptyItemTemplate	Used to render content of the remaining items in the a group template
EmptyDataTemplate	Used to specify content that is displayed when no items are returned from the listView control's data source
SelectedItemTemplate	Used to specify the content displayed for the selected item
AlternatingItemTemplate	Used to render different content for alternating items
EditItemTemplate	Used for editing
InsertItemTemplate	Used for inserting

Templates

Using the LayoutTemplate and ItemTemplate

The layout template is rendered only once. It usually contains a placeholder. The placeholder is never rendered to the browser it is replaced by the contents of the itemTemplate. It must have the ID itemPlaceHolder.

The item template is used to render each item from the data source.

Activity 1. Using a listView control

default.aspx

Create a web page and place an SQLDataSource control on it. Set it up to retrieve the categoryName from the categories table in the northwind database.

Drag a listView control onto the page and link it to the datasource control. Switch over to source view and type in the following code:

```
<asp:ListView ID="lvCategories" runat="server" DataSourceID="sdsCategories">
      <LayoutTemplate>
            <div>
            <asp:PlaceHolder ID="itemPlaceholder" runat="server">
            </asp:PlaceHolder>
            </div>
      </LayoutTemplate>

      <ItemTemplate>
            <div>
            <%# Eval("CategoryName") %>
            </div>
      </ItemTemplate>
</asp:ListView>
<br />
<asp:SqlDataSource ID="sdsCategories" runat="server"
ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
SelectCommand="SELECT [CategoryName] FROM [Categories]">
</asp:SqlDataSource>
```

Itemseperator template

This can be used to display some content between items.

Activity 2. ItemSeperator template

separator.aspx

Modify the previous exercise so that a horizontal rule is displayed between each item.

```
<ItemSeparatorTemplate>
<hr />
</ItemSeparatorTemplate>
```

Alternating Item template

This template can be used to specify how every second item will be displayed. If there is no alternating item each item will appear according to the itemTemplate.

Activity 3. Alternating Item template

alternatingItem.aspx

Display every second row with a different background colour.

```
<AlternatingItemTemplate>
      <div class="alternateStyle">
      <%# Eval("CategoryName") %>
      </div>
</AlternatingItemTemplate>
```

You will need to add the alternateStyle class to your stylesheet.

Using the groupTemplate

You can use the group template to group multiple items together. This is useful if you want to display items in multiple columns.

To display the items in multiple columns you need to set the GroupItemCount property to the number of columns you would like displayed. You then need a group template to specify how each group will be displayed.

Activity 4. Group template

GroupTemplate.aspx

Create a web page with a datasource control on it. Configure it to retrieve the product name and the unit price from the products table.

Configure the listview control to contain a group:

```
<asp:ListView ID="lvProducts" runat="server" DataSourceID="sdsProducts"
GroupItemCount="4">
<LayoutTemplate>
   <asp:PlaceHolder ID="groupPlaceholder" runat="server">
   </asp:PlaceHolder>
</LayoutTemplate>

<GroupTemplate>
   <div>
   <asp:PlaceHolder ID="itemPlaceholder" runat="server">
   </asp:PlaceHolder>
   </div>
</GroupTemplate>

<GroupSeparatorTemplate>
   <hr />
</GroupSeparatorTemplate>

<ItemTemplate>
   <span>
   <%# Eval("productName") %>
   <%# Eval("unitPrice", "{0:c}") %>
   </span>
</ItemTemplate>

<ItemSeparatorTemplate>
   ,
</ItemSeparatorTemplate>
</asp:ListView>
```

The listview control also supports an emptyItemTemplate which can be used to render content for the left over items in a group template. So if the groupItemCount property is set to 3 and there are 4 items the contents of the emptyItemTemplate and displayed for the last 2 items.

Selecting a row

You can set up the list view control so that you can use it select items.

Activity 5. Master/detail

selectingRows.aspx

In this activity we will display the categories in the first listview and display the products for that category in the second listview.

Drag a sqlDatasource control onto the page and configure it to retrieve the category ID and category name from the northwind database.

Add a list view control to the page which will display the category name as a hyperlink:

```
<asp:ListView ID="lvCategories" runat="server" DataSourceID="sdsCategories"
DataKeyNames="categoryID">
      <LayoutTemplate>
            <div  id="categoryList">
                <asp:PlaceHolder ID="itemPlaceholder" runat="server">
                </asp:PlaceHolder>
                </div>
      </LayoutTemplate>

      <ItemTemplate>
            <div>
            <asp:LinkButton ID="lkbSelect"
            Text='<%# Eval("categoryName") %>' CommandName="Select"
            runat="server"></asp:LinkButton>
            </div>
      </ItemTemplate>

      <SelectedItemTemplate>
                <div id="selected">
```

```
                <%# Eval("categoryName") %>
                </div>
        </SelectedItemTemplate>
</asp:ListView>
```

Setting the dataKeyNames property causes the listview control to build a hidden collection of primary key values when the listView is bound to its data source. Each row has a category ID associated with it even if it is not displayed.

The command name of the link button is select. This is a keyword which causes the listView to make that row the selected item.

You may want to add some styling to the listview so that the selected row is displayed differently to the other rows:

```
#categoryList
{
    background-color: #E6ECFF;
    float: left;
    width: 200px;
}

#categoryList a
{
    text-decoration: none;
    color: Black;
}

#categoryList a:hover
{
    text-decoration: underline;
    color: Black;
}

#selected
{
    background-color: #CCFF80;
}
```

Next we need to add a second data source control and list view. Used to display the products for the selected category.

Make sure you add a where clause to the data source control so that only the products from the selected category are displayed.

```
<asp:ListView ID="lvProducts" runat="server" DataSourceID="sdsProducts">
        <LayoutTemplate>
              <div  id="productList">
                  <asp:PlaceHolder ID="itemPlaceholder" runat="server">
                  </asp:PlaceHolder>
                  </div>
        </LayoutTemplate>

        <ItemTemplate>
              <div>
              <%# Eval("ProductName") %>
              <%# Eval("UnitPrice", "{0:C}") %>
              </div>
        </ItemTemplate>
</asp:ListView>
```

Sorting

You can sort the items in a listview control by adding one or more button controls to the listview that have a command name property set to “sort” and a command argument property se to the name of a property to sort by.

Activity 6. Sorting

sortingData.aspx

Create a web page which displays all the employees. Display the first name, last name, title of courtesy, hire date, title, date of birth and extension.

Add sorting for the first name, last name, hire date and title.

```
<LayoutTemplate>
      <div id="employeeList">
            <h2>Sort By:</h2>
            <div id="sortEmployee">
            <asp:LinkButton ID="lkbFirstName" Text="First Name"
            CommandArgument="firstName" CommandName="sort"
            runat="server"></asp:LinkButton>
            <asp:LinkButton ID="lkLastName" Text="Last Name"
            CommandArgument="lastName" CommandName="sort"
            runat="server"></asp:LinkButton>
            <asp:LinkButton ID="lkbTitle" Text="Position"
            CommandArgument="title" CommandName="sort" runat="server"></asp:LinkButton>
            <asp:LinkButton ID="lkbHire" Text="Hire Date"
            CommandArgument="hireDate" CommandName="sort"
            runat="server"></asp:LinkButton>
            </div>
            <asp:PlaceHolder ID="itemPlaceholder"
            runat="server"></asp:PlaceHolder>
      </div>
</LayoutTemplate>

<ItemTemplate>
      <div>
            <%# Eval("TitleOfCourtesy")%>
            <%# Eval("firstName")%>
            <%# Eval("lastName")%>
            <br />
            DOB: <%# Eval("birthdate")%>
            <br />
            <br />
            <%# Eval("title")%>
            <br />
            Date Hired: <%# Eval("hireDate")%>
            <br />
            Ext: <%# Eval("extension")%>
      </div>
</ItemTemplate>
```

Editing Data

You can use the listview control to update, delete and insert data. This is achieved by adding link buttons having the command name “Edit”, “Delete”, Update”, Cancel”, “Insert”.

When editing the listview control the data keyNames must be set to the primary of the table you will be modifying.

Deleting data

To delete a row you need to add a delete button or link, in the item template, with the command name of “Delete”. It may be a good idea to also add some client side code to make sure they really do want to delete the row.

```
<asp:LinkButton ID="lkbDelete" runat="server" Text="Delete" CommandName="Delete"
OnClientClick="return window.confirm('Are you Sure?');"></asp:LinkButton>
```

Editing data

To edit a row you need to add an edit button or link, in the item template, with the command name of “Edit”.

```
<asp:LinkButton ID="lkbEdit" runat="server" Text="Edit"
CommandName="Edit"></asp:LinkButton>
```

You will also need to add the edit item template.

```
<EditItemTemplate>
      <div id="selected">
      <asp:Label ID="lblCategoryName"
      AssociatedControlID="txtCategoryName" runat="server"
      Text="Category Name:"></asp:Label>
      <asp:TextBox ID="txtCategoryName" runat="server"
      Text='<%# Bind("categoryName") %>'></asp:TextBox>
```

```
        <br />
        <asp:Label ID="lblDescription"
        AssociatedControlID="txtDescription" runat="server"
        Text="Description:"></asp:Label>
        <asp:TextBox ID="txtDescription" runat="server"
        Text='<%# Bind("description") %>'></asp:TextBox>
        <br />
        <asp:LinkButton ID="lkbUpdate" runat="server"
        Text="Update" CommandName="Update"></asp:LinkButton>
        <asp:LinkButton ID="lkbCancel" runat="server" Text="Cancel"
        CommandName="Cancel"></asp:LinkButton>
        </div>
</EditItemTemplate>
```

Notice the 2 link buttons with a command name of Update and Cancel.

Inserting data

To insert a row you need to supply an insert item template.

```
<InsertItemTemplate>
        <div class="insertStyle">
        <h2>Add new Category</h2>
        <asp:Label ID="lblCategoryName"
        AssociatedControlID="txtCategoryName" runat="server"
        Text="Category Name:"></asp:Label>
        <asp:TextBox ID="txtCategoryName" runat="server"
        Text='<%# Bind("categoryName") %>'></asp:TextBox>
        <br />
        <asp:Label ID="lblDescription"
        AssociatedControlID="txtDescription" runat="server"
        Text="Description:"></asp:Label>
        <asp:TextBox ID="txtDescription" runat="server"
        Text='<%# Bind("description") %>'></asp:TextBox>
        <br />
        <asp:LinkButton ID="lkbInsert" runat="server" Text="Insert"
        CommandName="Insert"></asp:LinkButton>
        </div>
</InsertItemTemplate>
```

Notice the link button with the command name of Insert.

You also need to specify where the insert item template will be displayed. This is done by setting the InsertItemPosition property to “FirstItem”, “LastItem” or “None”:

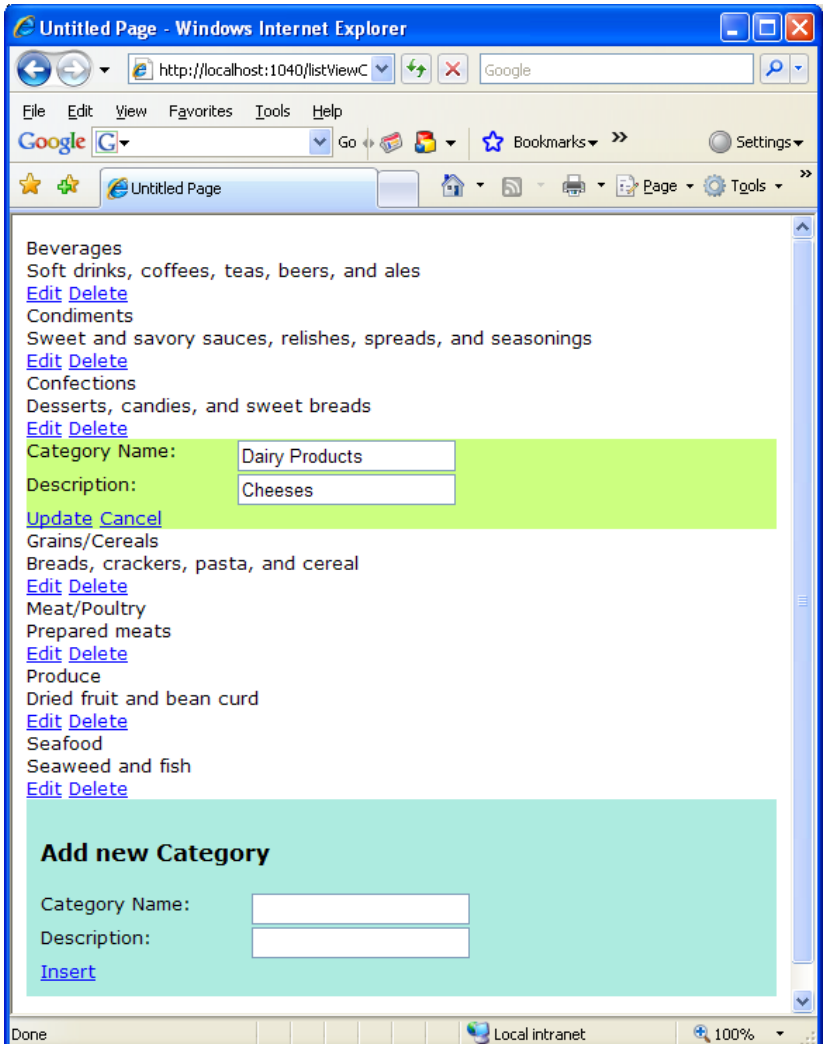
```
<asp:ListView ID="lvCategories" runat="server" DataSourceID="sdsCategories"
DataKeyNames="CategoryID" InsertItemPosition="LastItem">
```

Activity 7. Insert, update and delete

editingData.aspx

In this activity we will create a listview which will provide the ability to insert, update and delete categories.

The result should look something like this:



The dataPager control

The data pager control displays a user interface for navigating through multiple pages of items. The data pager control can work with any control that supports the IPageableItemContainer interface. At this stage the only control that supports this is the listview control. However this does hint that in future releases there will be others.

The data pager control includes the following properties:

- PageSize - gets or sets the number of items to display at a time
- PagedControlId - gets or sets the control to page the control at this stage can only be a listView control)
- Fields - Gets the fields contained by the DataPager
- StartRowIndex - Gets the index of the first item to show
- MaximumRows - Gets the maximum number of rows to retrieve from the data source
- TotalRowCount - Gets the total number of items available from the data source

To use the data pager control you need to set the pageSize property to the number of rows you would like to display at a time. The pagerControlId can be left out if the datapager control is placed inside the listview control.

The datapager supports the following fields:

- NextPreviousPagerField - Used to display Next, Previous, First, and Last links
- NumericPagerField - Used to display Next, Previous, and page numbers links
- TemplatePagerField - Used to create a custom user interface for paging

Activity 8. Pager control

paging.aspx

In this activity we will display the product name and the price. The list view will display 8 products at a time.

```
<asp:ListView ID="lvProducts" runat="server" DataSourceID="sdsProducts">
      <LayoutTemplate>
            <div>
            <asp:PlaceHolder ID="itemPlaceholder" runat="server">
            </asp:PlaceHolder>
```

```
            <hr />
            <asp:DataPager ID="dpProducts" runat="server"
            PageSize="10">
                  <Fields>
                        <asp:NumericPagerField />
                  </Fields>
            </asp:DataPager>

            </div>
      </LayoutTemplate>

      <ItemTemplate>
            <div>
                  <%# Eval("ProductName") %>
                  <%# Eval("UnitPrice", "{0:C}") %>

            </div>
      </ItemTemplate>
</asp:ListView>
```

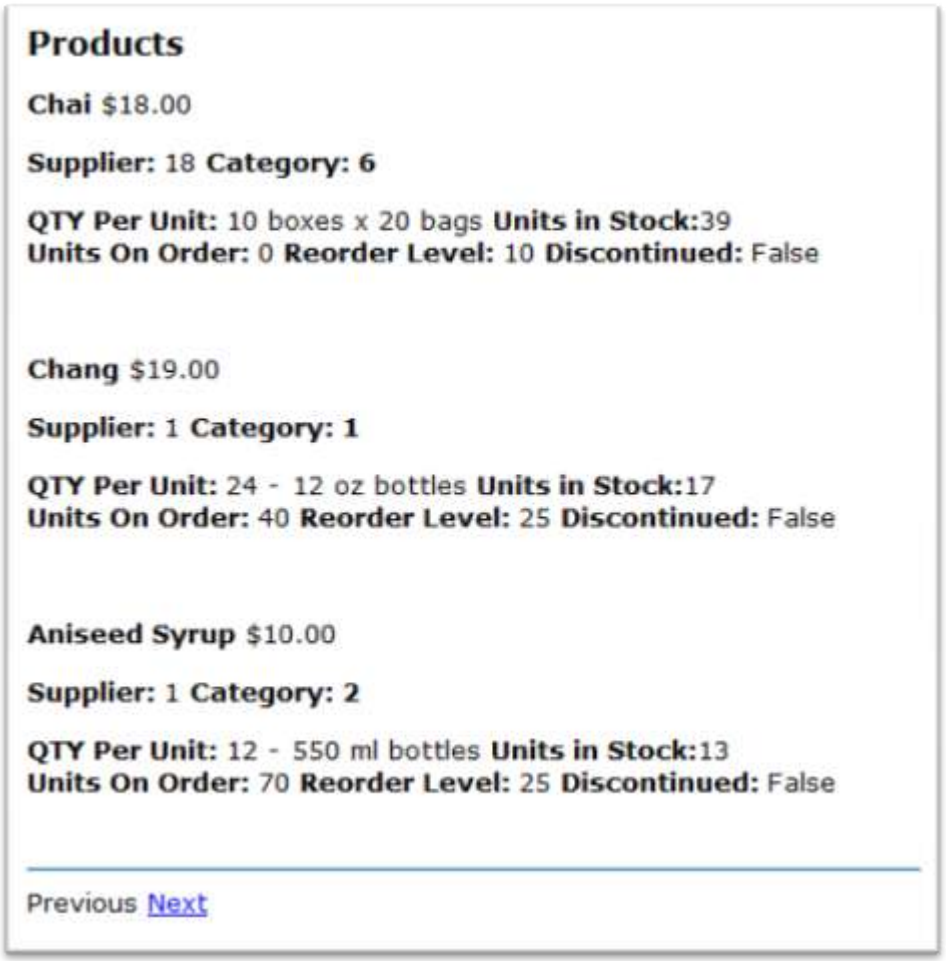
Activity 9. List view for product table

products.aspx

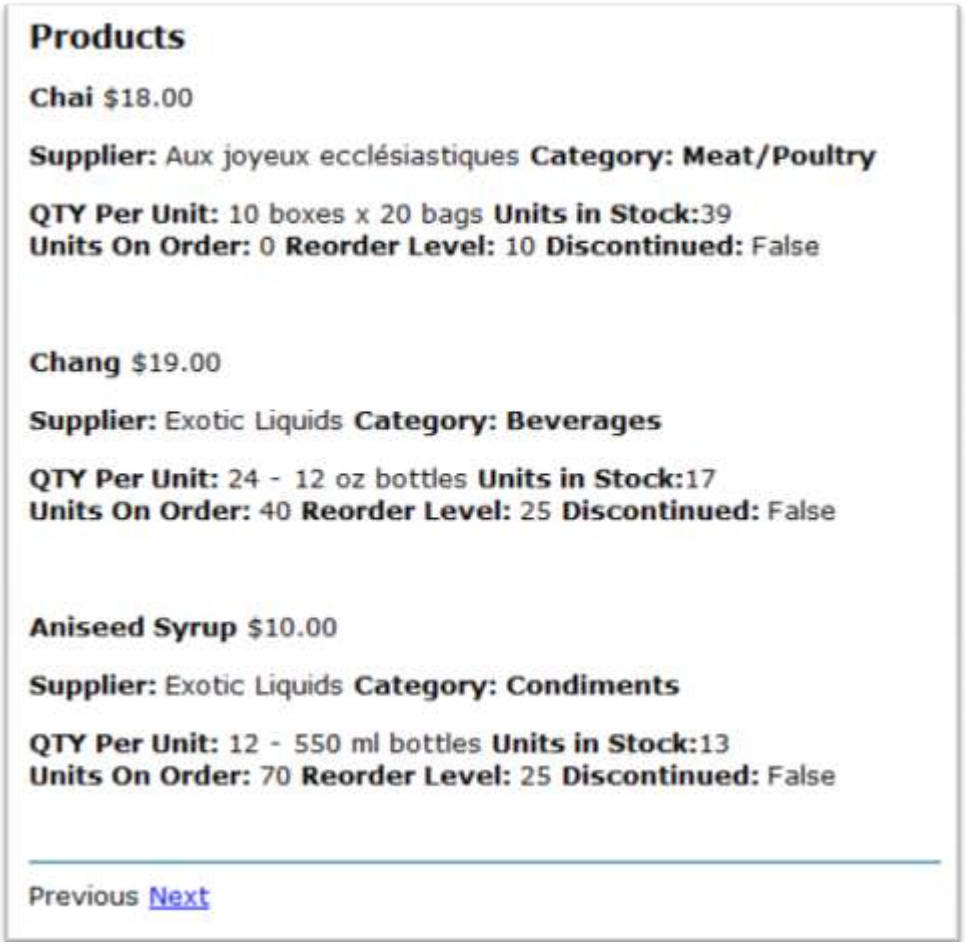
In this exercise we will create a list view control which will allow the display, edit, insert and delete of the products in the northwind database. The category and suppliers should be displayed as the category name and supplier company name when viewing the data and should be displayed in a drop down list when editing and inserting.

First create a list view that displays all the products. As we have many products add paging so that 3 products are displayed at a time. Make sure the data source control is set up to generate insert, update and delete statements.

The page should look something like this:

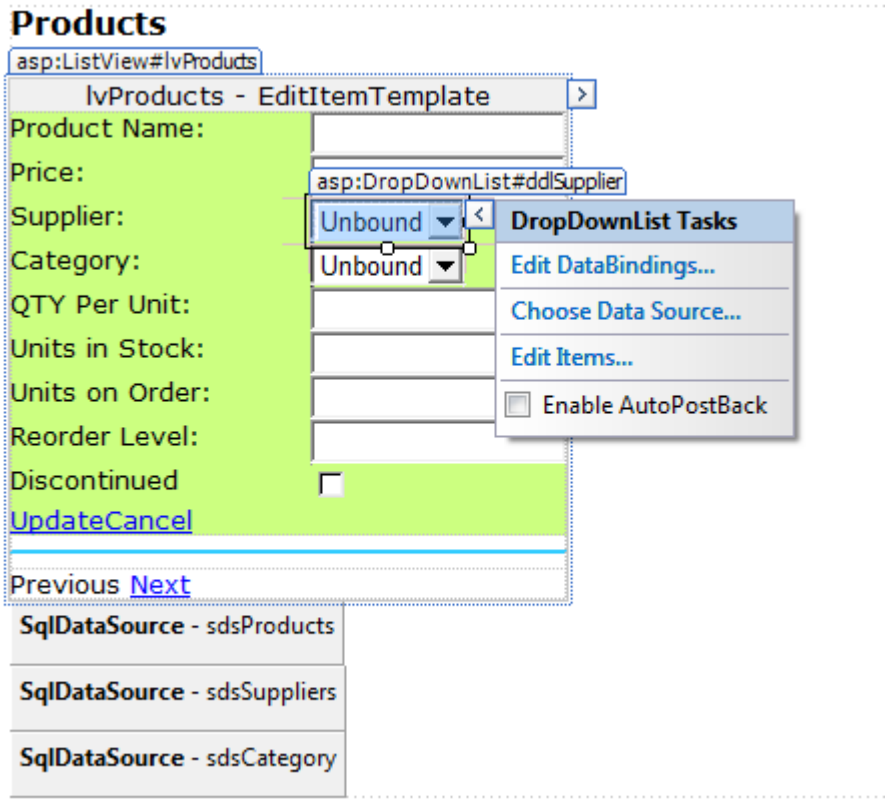


Modfify the select statement so that the category name and supplier company name are retreived as well as the ID. Change the item template so that the names are displayed:



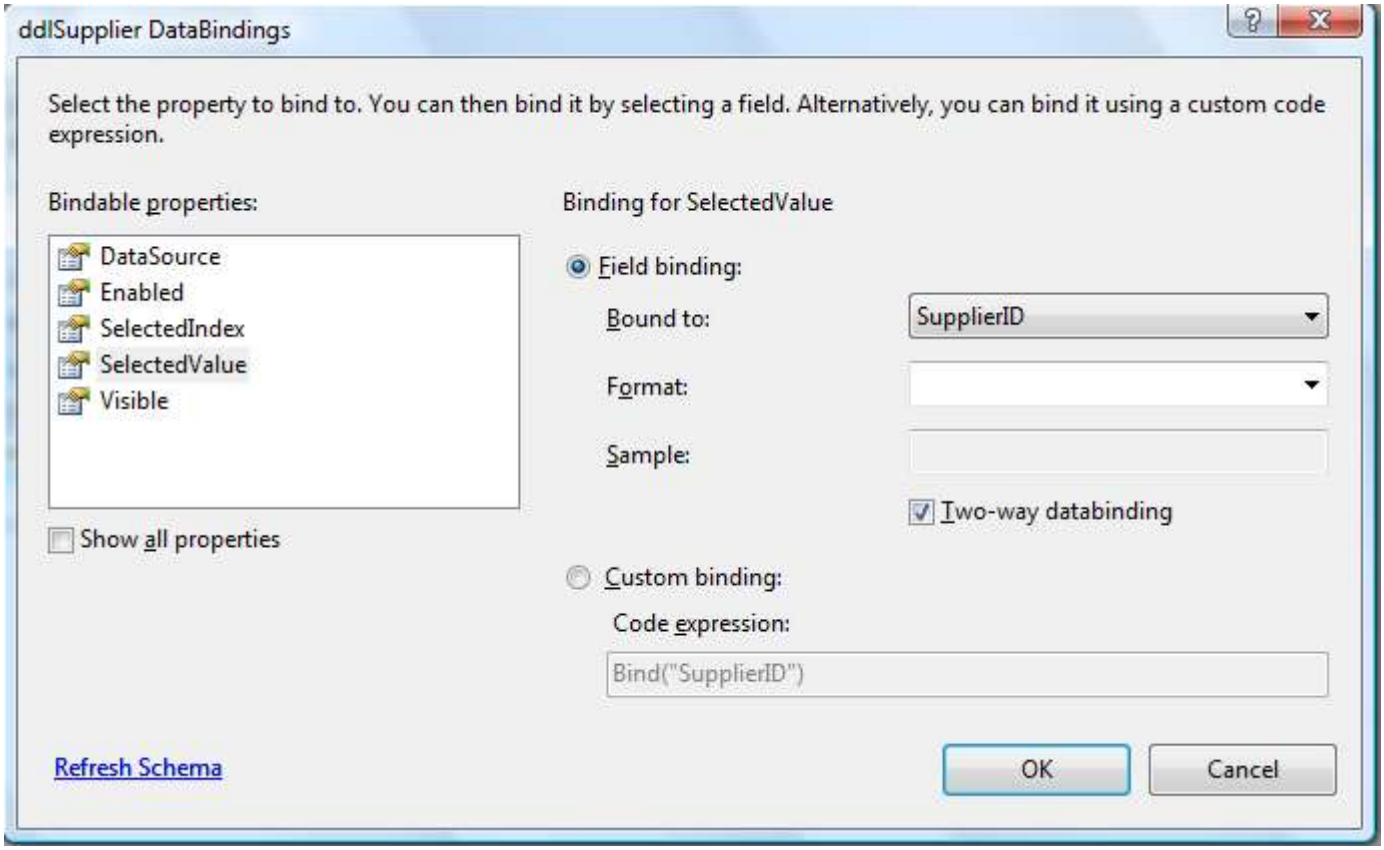
Add 2 links to allow edit and delete. These should be added as link buttons with command names of edit and delete.

Add an editItem template to be displayed when editing the product. Each field should be edited as a textbox except for category and supplier which will be displayed as drop down lists and discontinued which will be displayed as a checkbox.



You will need to add 2 new SQL data source controls, one to select all the categories categoryID and categoryName and the other to select the supplierID and supplier CompanyName. Allocate the corresponding data source to the drop down lists.

Set up the binding for the selected value for the supplier drop down list to supplierID and category to categoryID. You can do this by selecting “edit databinding” from the drop down list tasks.



This should allow the edit and delete of products.

Inserting is a bit more complex if we are using drop down lists. We need to add a few lines of code in the item inserting event. This code allocates the categoryID and supplierID to the insert parameters from the selected dropdown list values.

To enable insert add the InsertItemPosition property to the listview and allocated the value "FirstItem". Create the insert item template so that it contains the same form elements as the edit item template.

A drop down list cannot have 2 way data binding set up so in the item_inserting event for the list view add the following code:

```
protected void lvProducts_ItemInserting(object sender, ListViewInsertEventArgs e)
{
        //set category ID
        DropDownList ddlCat;

        ddlCat = (DropDownList)lvProducts.InsertItem.FindControl("ddlCategory");

        e.Values["categoryID"] = ddlCat.SelectedValue.ToString();

        //set supplier ID
        DropDownList ddlSup;

        ddlSup = (DropDownList)lvProducts.InsertItem.FindControl("ddlSupplier");

        e.Values["supplierID"] = ddlSup.SelectedValue.ToString();
}
```

References

ASP.NET 3.5 Unleashed by Stephen Walther