

Hyperlink

|  |  |
| --- | --- |
|  |  |

Set **ObjectDataSource**, named it **ObjectDataSource1** to retrieve information of Company, and set it to the GridView.

<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"

**DataKeyNames="SupplierID"** **DataSourceID="ObjectDataSource1"** EnableViewState="False">

<Columns>

<**asp:HyperLinkField**

**DataNavigateUrlFields = "SupplierID"**

**DataNavigateUrlFormatString= "ProductsForSupplierDetails.aspx?SupplierID={0}"**

**Text = "View Products"** />

<asp:BoundField DataField="CompName" HeaderText="CIA" SortExpression="CompName" />

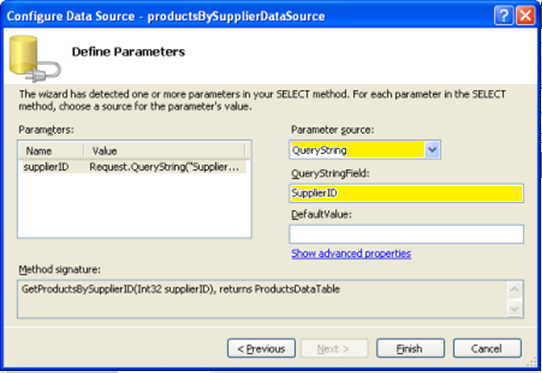
<asp:BoundField DataField="City" HeaderText="City" SortExpression="City" />

<asp:BoundField DataField="Country" HeaderText="Country" SortExpression="Country" />

</Columns>

</asp:GridView>

Page called use a **SQLDataObject** named it **suppliersDataSource** , and set the **QueryString**



Or use in the load process

protected void btnSend\_Click(object sender, EventArgs e)

{

Response.Redirect("Default2.aspx?UserId="+txtUserId.Text+"&UserName="+txtUserName.Text);

}

protected void Page\_Load(object sender, EventArgs e)

{

if(!IsPostBack)

{

lblUserId.Text = Request.QueryString[0];

lblUserName.Text = Request.QueryString[1];

}

}

It was created **ObjectDataSource**, and we assign the **QueryString** to the control so when the page is open it triggers the **ObjectDataSource.**

We drag a **FormView** and we bind it to the **ObjectDataSource**, Visual Studio will automatically create the FormView's **ItemTemplate, InsertItemTemplate**, and **EditItemTemplate** with Label and TextBox Web controls for each of the data fields returned by the ObjectDataSource.

In this case we just want to display data, so remove the **InsertItemTemplate** and **EditItemTemplate**.

Edit the ItemTemplate:

<asp:FormView ID="FormView1" runat="server" **DataKeyNames="SupplierID"**

DataSourceID="**suppliersDataSource**" EnableViewState="False">

<ItemTemplate> <h3><%# Eval("CompanyName") %></h3> <p>

<asp:Label ID="AddressLabel" runat="server" Text='<%# Bind("Address") %>'>

</asp:Label>

<br />

<asp:Label ID="CityLabel" runat="server" Text='<%# Bind("City") %>'>

</asp:Label>,

<asp:Label ID="CountryLabel" runat="server" Text='<%# Bind("Country") %>'>

</asp:Label>

<br /> Phone: <asp:Label ID="PhoneLabel" runat="server" Text='<%# Bind("Phone") %>'>

</asp:Label>

</p>

</ItemTemplate>

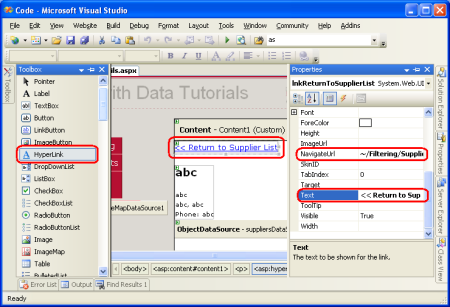
</asp:FormView>

Figure 16 shows a screen shot of the ProductsForSupplierDetails.aspx page after the supplier information detailed above has been included.



**Figure 16**: The List of Products Includes a Summary About the Supplier

Let's add a HyperLink control to that links back.

[](http://i1.asp.net/asp.net/images/dataaccess/09fig17cs.png)

**Figure 17**: Add a HyperLink Control to Take the User Back to SupplierListMaster.aspx

If the user clicks on the View Products link for a supplier that doesn't have any products, the ProductsBySupplierDataSource ObjectDataSource in ProductsForSupplierDetails.aspx won't return any results. The GridView bound to the ObjectDataSource won't render any markup resulting in a blank region on the page in the user's browser. To more clearly communicate to the user that there are no products associated with the selected supplier we can set the GridView's EmptyDataText property to the message we want displayed when such a situation arises. I've set this property to "There are no products provided by this supplier"

By default, all suppliers in the Northwinds database provide at least one product. However, for this tutorial I've manually modified the Products table so that the supplier Escargots Nouveaux is no longer associated with any products. Figure 18 shows the details page for Escargots Nouveaux after this change has been made.



**Figure 18**: Users are Informed that the Supplier Doesn't Provide Any Products