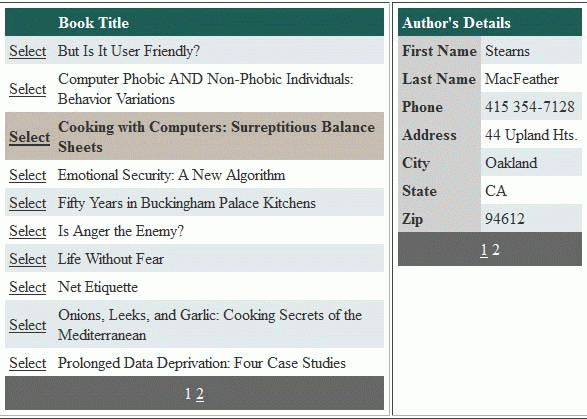
**Introduction**  
  
In this article we will create a Master-Detail form using a GridView and a DetailsView. A Master-Detail form displays a master list and the details for the currently selected item. We will use a "Titles" table of a "Pubs" database to display a master list in a GridView and an "Authors" table to display author details of the currently selected book in the master list.  
  
  
  
**Steps:**

* Create a new ASP.NET Web Application.
* Add a GridView and a DetailsView in the Default.aspx.
* Add a ConnectionString in the Web.config file for database connection.
* In the code view of Default.aspx, add the "System.Data.SqlClient" namespace. Declare the following variables inside the class declaration:  
    
  SqlConnection con;  
  SqlCommand cmd;  
  SqlDataAdapter sda;  
  DataTable dt;  
  string ConString, CmdString, TitleID;
* Add the following code in the page load event:  
    
  protected void Page\_Load(object sender, EventArgs e)  
  {  
      ConString = ConfigurationManager.ConnectionStrings["ConString"].ConnectionString;  
      if (!IsPostBack)  
      {  
          BindGridView();  
      }  
  }

ConString retrieves connection string from the Web.config file named "ConString". BindGridView() method binds data to the GridView.  
  
private void BindGridView()  
{  
    con = new SqlConnection(ConString);  
    CmdString = "SELECT Title\_Id, Title FROM Titles";  
    cmd = new SqlCommand(CmdString, con);  
    sda = new SqlDataAdapter(cmd);|  
    dt = new DataTable();  
    sda.Fill(dt);  
    GridView1.DataSource = dt.DefaultView;  
    GridView1.DataBind();  
}

BindGridView sets Title\_Id and Title of Titles table as the data source of the GridView.   
  
Add the following code to the HTML view of Default.aspx to set the properties of the GrieView:  
  
<asp:GridView ID="GridView1"   
    runat="server"   
    AutoGenerateColumns="False"  
    AutoGenerateSelectButton="True"   
    AllowPaging="True"   
    AllowSorting="True"  
    DataKeyNames="Title\_Id"   
    onselectedindexchanged="GridView1\_SelectedIndexChanged"  
    onpageindexchanging="GridView1\_PageIndexChanging">  
    <Columns>                          
        <asp:BoundField HeaderText="Book Title" DataField="title" />  
    </Columns>  
</asp:GridView>

* AutoGenerateColumns is used to enable disabling automatic generation of the columns in the GridView. It set to False because we want to display only the "Title" column in the GridView, not both columns.
* AutoGenerateSelectButton is set to add a Select column to select the rows.
* AllowPaging and AllowSorting are used to enable paging and sorting.
* DataKeyNames is used to identify a row in controls like GridView. It is just like a primary key of a table. More than one column can also be set to DataKeyNames separated by commas. Here it is set to the "Title\_Id" column so that we can get Title\_Id of the selected row of the GridView.
* The SelectedIndexChanged event is fired after the Select button of the row is clicked.
* The PageIndexChanging event is fired when one of the pager buttons is clicked.
* A bound column is added to display book title.
* Write the following code in the GridView SelectedIndexChanged event:  
    
  protected void GridView1\_SelectedIndexChanged(object sender, EventArgs e)  
  {  
      TitleID = GridView1.SelectedValue.ToString();  
      BindDetailsView(TitleID);  
  }

Here, the title id of the selected row is fetched in the TitleID and it is passed to the BindDetailsView method to fill the DetailsView with the author details of the selected title.  
  
private void BindDetailsView(string TitleID)  
{  
    CmdString = "SELECT a.au\_fname AS 'First Name', a.au\_lname AS 'Last Name', a.Phone AS 'Phone', a.Address AS 'Address', a.city AS 'City', a.state AS 'State', a.Zip AS 'Zip' FROM authors a JOIN titleauthor t ON a.au\_id=t.au\_id WHERE t.title\_id=@TitleID";  
    con = new SqlConnection(ConString);  
    cmd = new SqlCommand(CmdString, con);  
    cmd.Parameters.AddWithValue("@TitleID", TitleID);  
    sda = new SqlDataAdapter(cmd);  
    dt = new DataTable();  
    sda.Fill(dt);  
    DetailsView1.DataSource = dt.DefaultView;  
    DetailsView1.DataBind();  
}  
  
Here, the author details of the given title id is retrieved from joining the "Authors" and "TitleAuthor" table and is set as the data source of the DetailsView.

* Add the following code in the HTML view of Default.aspx to set the properties of the DetailsView:  
    
  <asp:DetailsView ID="DetailsView1"   
      runat="server"   
      HeaderText="Author's Details"   
      AllowPaging="True"   
      onpageindexchanging="DetailsView1\_PageIndexChanging">  
  </asp:DetailsView>

HeaderText property is used to set header text of the DetailsView.  
  
Paging is enabled also in the DetailsView as one book can have multiple authors that can be paged.

* We also have to write code for PageIndexChanging events of both controls   
    
  protected void GridView1\_PageIndexChanging(object sender, GridViewPageEventArgs e)  
  {  
      GridView1.PageIndex = e.NewPageIndex;  
      BindGridView();  
  }  
     
  protected void DetailsView1\_PageIndexChanging(object sender, DetailsViewPageEventArgs e)  
  {  
      DetailsView1.PageIndex = e.NewPageIndex;  
      TitleID=GridView1.SelectedValue.ToString();  
      BindDetailsView(TitleID);  
  }
* Finally, format the GridView and DetailsView using the built-in Auto Format tool.  
    
  Click on little task button on the top-right corner. And select "Auto Format". Now select a scheme from the left and click OK. The Controls are then formated for you.