Posted on March 29, 2012 by Venu Gopal in ASP.NET, C#, Dotnet

GridView displays the values of a data source in a table where each column represents a field and each row represents a record. It enables you to select, sort, and edit these items. In article, I am going to explain how to insert, update or delete a row from a grid view control. If you are new to using a grid control, I would recommend you to read my blogs **"Using Gri ASP.NET & C# —PART 1 & PART 2".**

Database Connection

Create a new ASP.NET Web Application project. Open web.config and add following entry to connection strings element

```
<add name="Sql" connectionString="Data Source=<SERVERNAME>;
     Initial Catalog= TEST;User=testuser;Password=testuser;"
     providerName="System.Data.SqlClient"/>
```

Page Design

Create a new aspx page and name it as ProductsEditView.aspx. Open the aspx file in design mode and add the following code in div tag. I will explain each setting in detail later.

```
<asp:gridview id="gvProducts"
          autogeneratecolumns="False"
          emptydatatext="No data available."
          runat="server" DataKeyNames="ProductID"
          OnRowEditing="gvProducts_RowEditing"
          OnRowCancelingEdit="gvProducts_RowCancelingEdit"
          OnRowCommand="gvProducts_RowCommand"
          onrowupdating="gvProducts_RowUpdating"
          OnRowDeleting="gvProducts_RowDeleting">
  <Columns>
  <asp:BoundField DataField="ProductID"
HeaderText="Product ID"
ReadOnly="true"/>
      <asp:TemplateField headertext="Product Number">
          <ItemTemplate> <%#Eval("ProductNumber")%></ItemTemplate>
          <EditItemTemplate>
               <asp:TextBox id="txtProductNumber" runat="server"
text='<%#Eval("ProductNumber")%>'/>
          </EditItemTemplate>
          <FooterTemplate>
         <asp:TextBox ID="txtNewProductNumber" runat="Server"/>
    </FooterTemplate>
      </asp:TemplateField>
      <asp:TemplateField headertext="Product Name">
         <ItemTemplate> <%#Eval("Name")%></ItemTemplate>
         <EditItemTemplate>
                <asp:TextBox id="txtProductName" text='<%#Eval("Name")%>'
                         runat="server"/>
         </EditItemTemplate>
         <FooterTemplate>
                    <asp:TextBox ID="txtNewProductName" runat="Server"/>
         </FooterTemplate>
     </asp:TemplateField>
     <asp:TemplateField headertext="Price">
         <ItemTemplate> <%#Eval("ListPrice")%></ItemTemplate>
         <EditItemTemplate>
               <asp:TextBox id="txtListPrice" text='<%#Eval("ListPrice")%>'
                               runat="server"/>
         </EditItemTemplate>
         <FooterTemplate>
                <asp:TextBox ID="txtNewListPrice" runat="Server">
         </FooterTemplate>
     </asp:TemplateField>
     <asp:TemplateField>
         <ItemTemplate>
              <asp:LinkButton ID="btnedit" runat="server"
CommandName="Edit"
Text="Edit"/>
         </ItemTemplate>
          <EditItemTemplate>
              <asp:LinkButton ID="btnupdate" runat="server"
             CommandName="Update" Text="Update" />
              <asp:LinkButton ID="btncancel" runat="server"
             CommandName="Cancel" Text="Cancel"/>
   <asp:LinkButton ID="btnDelete" runat="server"
                     CommandName="Delete" Text="Delete"/>
          </EditItemTemplate>
          <FooterTemplate>
              <asp:Button ID="btnInsert" runat="Server" Text="Insert"
CommandName="Insert" UseSubmitBehavior="False" />
          </FooterTemplate>
      </asp:TemplateField>
  </Columns>
</asp:gridview>
<asp:Button ID="btnAdd" runat="server" Text="Add" OnClick="AddNewRecord" />
```

Name	Description
Autogeneratecolumns ="False"	Set's a value indicating bound fields should not created for each field in the data source.
emptydatatext= "No data available."	It displays "No data available" in the grid, if resultset is emtpy
OnRowCommand= "ProductsView_RowCommand"	when a button is clicked in the GridView control, ProductsView_RowCommand is called. This enables you to provide an event-handling method that performs a custom routine whenever this event occurs.
OnRowEditing= "gvProducts_RowEditing"	when a row's Edit button is clicked, gvProducts_RowEditing is called before the control enters edit mode
asp:BoundField	Displays the value of a field in a data source.
BoundField -> DataField	Gets or sets the name of the data field to bind to the BoundField object.
BoundField -> HeaderText	Gets or sets the text that is displayed in the header of a data control
TemplateField	Used TemplateField to display custom content for each record displayed.
TemplateField -> ItemTemplate	Use the ItemTemplate property to specify the custom content displayed for the items in a TemplateField object. Define the content by creating a template that specifies how the items are rendered.

<%#Eval("ProductNumber")%>	Data-binding expressions are contained within <%# and %> delimiters. At run time, the **Eval** method calls the Eval method of the DataBinder object. The **Eval** method takes the name of a data field and returns a string containing the value of that field from the current record in the data source.
<ItemTemplate> <%#Eval ("ProductNumber")%> </ItemTemplate>	Specify the custom content displayed for the templatefield items. In our example, we are directly writing the product number text. We can also define a new control in the itemtemplate
<EditItemTemplate> <asp:TextBox id="txtProductNumber" runat="server" text='<%#Eval ("ProductNumber")%>'/> </EditItemTemplate>	When the control enters edit mode, content in edititemtemplate is displayed. When user clicks on "Edit" link button, a textbox with existing product number is displayed
<FooterTemplate>	We can define content to be displayed in the footer section of the control. In our example, we define content that needs to be displayed when user requests to enter data for new product

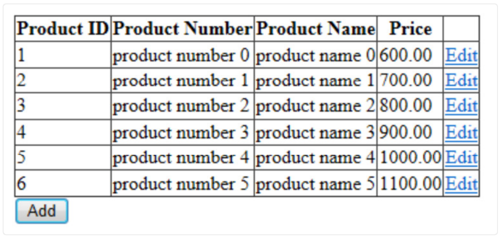
<asp:TextBox ID=”txtNewProductNumber” runat=”Server”/> </FooterTemplate>	
<asp:LinkButton ID=”btnedit” runat=”server” CommandName=”Edit” Text=”Edit”/>	Command name is set to “Edit”. When this button is clicked, control goes into edit mode
<asp:LinkButton ID=”btnupdate” runat=”server” CommandName=”Update” Text=”Update” />	Command name is set to “Update”. When this button is clicked, control would understand it is an update.

View Products

I am using ADO.NET API such as SQL Connection, SQL Command, SQL Adapters etc. to get the products from the database and to bind to gridview.

```
private void BindData()
{
    //Bind the grid view
    gvProducts.DataSource = RetrieveProducts();
    gvProducts.DataBind();
}
private DataSet RetrieveProducts()
{
    if (ViewState["Products"] != null)
        return (DataSet)ViewState["Products"];

    //fetch the connection string from web.config
    string connString =
ConfigurationManager.ConnectionStrings["Sql"].ConnectionString;

    //SQL statement to fetch entries from products
    string sql = @"Select top 10  P.ProductID, P.Name,
                P.ProductNumber, ListPrice  from tblProduct P";

    DataSet dsProducts = new DataSet();
    //Open SQL Connection
    using (SqlConnection conn = new SqlConnection(connString))
    {
        conn.Open();
        //Initialize command object
        using (SqlCommand cmd = new SqlCommand(sql, conn))
        {
            SqlDataAdapter adapter = new SqlDataAdapter(cmd);
            //Fill the result set
            adapter.Fill(dsProducts);

        }
    }
    return dsProducts;
}
```

If you run the application now, you would see the following page listing all the products.



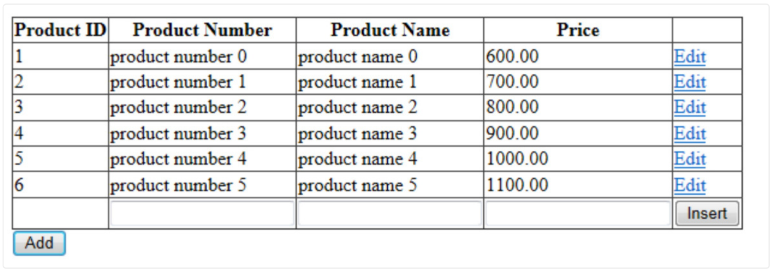
Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number 1	product name 1	700.00	Edit
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit

Create Product

Click on “Add” button and it would create a blank row. All we did in the button click event is to make the footer row visible. In the footer design itself, we designed it in such a way th looks like a blank product record.

```
protected void AddNewRecord(object sender, EventArgs e)
{
     gvProducts.ShowFooter = true;
```

```
        BindData();
}
```



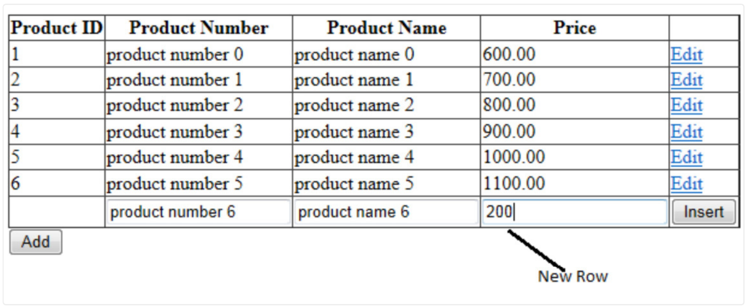
Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number 1	product name 1	700.00	Edit
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
				Insert

Add

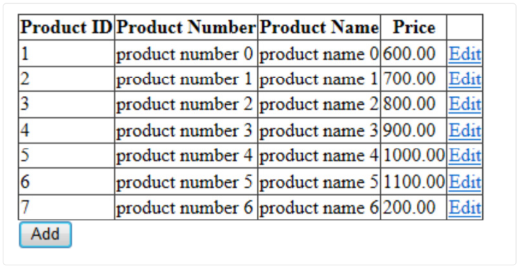
After you fill in the data, click on the insert button to add the product to the back-end database. When user clicks on the “insert” button, gvProducts_RowCommand event handler is ca
the event handler, we get the data from the controls and insert them in the database.

```
protected void gvProducts_RowCommand(object sender,
GridViewCommandEventArgs e)
{
    // If multiple ButtonField column fields are used, use the
    // CommandName property to determine if "Insert" button was clicked.
    if (e.CommandName.Equals("Insert"))
    {
        //fetch the values of the new product
        TextBox txtNewProductNumber =
    gvProducts.FooterRow.FindControl("txtNewProductNumber") as TextBox;
        TextBox txtNewProductName =
     gvProducts.FooterRow.FindControl("txtNewProductName") as TextBox;
        TextBox txtNewListPrice =
           gvProducts.FooterRow.FindControl("txtNewListPrice") as TextBox;
        //insert the new product into database
        InsertProduct(txtNewProductNumber.Text, txtNewProductName.Text,
                          txtNewListPrice.Text);
        gvProducts.ShowFooter = false;
        //clear the view state so that latest list will be retrieved from db
        ViewState["Products"] = null;
        BindData(); // rebind the data
    }
}

private void InsertProduct(string productNumber,
                            string productName,
                            string listPrice)
{
    //fetch the connection string from web.config
    string connString =
          ConfigurationManager.ConnectionStrings["Sql"].ConnectionString;
    //SQL statement to insert a product
    string sql = String.Format("Insert into tblProduct
values('{0}','{1}',{2})",
                                    productName,
                                    productNumber,
                                    listPrice);
    using (SqlConnection conn = new SqlConnection(connString))
    {
        conn.Open();
        //Initialize command object
        using (SqlCommand cmd = new SqlCommand(sql, conn))
        {
            cmd.ExecuteNonQuery();
        }
    }
}
```

Following are the screen shots of products page before and after clicking on “Insert” button



Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number 1	product name 1	700.00	Edit
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
	product number 6	product name 6	200	Insert

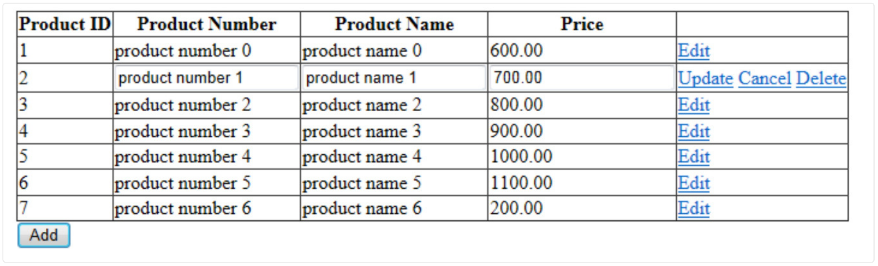


Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number 1	product name 1	700.00	Edit
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
7	product number 6	product name 6	200.00	Edit

Update Product

Click on "Edit" and the grid would go into edit mode. If you look at event handling function, you would notice that we are setting edit index of the grid to the row you trying to edit.
a post back, we have to re-bind the grid control.

```
protected void gvProducts_RowEditing(object sender,
                                      GridViewEditEventArgs e)
{
gvProducts.EditIndex = e.NewEditIndex;
BindData();
}
```

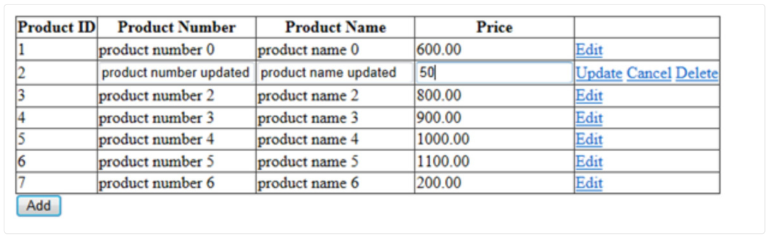


Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number 1	product name 1	700.00	Update Cancel Delete
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
7	product number 6	product name 6	200.00	Edit

Change product number to "product number updated" and product name to "product name updated" and price to 50. Clicking on "Update" would save your changes to DB.

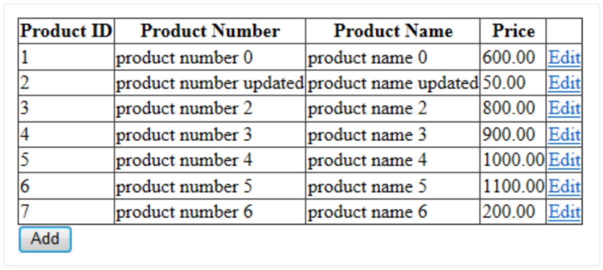
When you click on "Update" button, gvProducts_RowUpdating event is fired. Based on the selected row, get the gridviewrow. Once you get the gridviewrow, you could search for the c
controls and get the updated values. Once we have the latest values, we could simply persist them to the database using ADO.NET API.

```
protected void gvProducts_RowUpdating(Object sender,
                                       GridViewUpdateEventArgs e)
{

    // Get the product id of the selected product
    string ID = gvProducts.DataKeys[e.RowIndex].Value.ToString();

    // Get the GridViewRow object that represents the row being edited
    // from the Rows collection of the GridView control.
    GridViewRow row = gvProducts.Rows[e.RowIndex];

    // Get the controls that contain the updated values. In this
    // example, the updated values are contained in the TextBox
    // controls declared in the edit item templates of each TemplateField
    // column fields in the GridView control.
    TextBox txtProductNumber = (TextBox)row.FindControl("txtProductNumber");
    TextBox txtProductName = (TextBox)row.FindControl("txtProductName");
    TextBox txtListPrice = (TextBox)row.FindControl("txtListPrice");
    //update the product
    UpdateProduct(ID, txtProductNumber.Text, txtProductName.Text, txtListPrice.Text);
    gvProducts.EditIndex = -1;
    BindData();
}

private void UpdateProduct(string productID, string productNumber,
                    string productName, string listPrice)
{
    //fetch the connection string from web.config
    string connString =
           ConfigurationManager.ConnectionStrings["Sql"].ConnectionString;
    //SQL statement to update a product
    string sql = String.Format(@"Update tblProduct set Name='{0}'
                                ,ProductNumber='{1}'
                                ,ListPrice ={2}
                                where ProductID = {3}",
                                productName,
                                productNumber,
                                listPrice, productID);
    using (SqlConnection conn = new SqlConnection(connString))
    {
        conn.Open();
        //Initialize command object
        using (SqlCommand cmd = new SqlCommand(sql, conn))
        {
            cmd.ExecuteNonQuery();
        }
    }
}
```



Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number updated	product name updated	50	Update Cancel Delete
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
7	product number 6	product name 6	200.00	Edit

Add



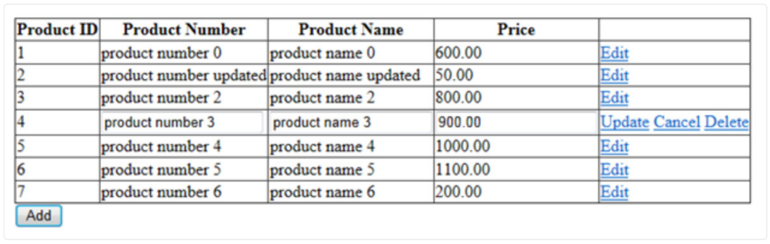
Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number updated	product name updated	50.00	Edit
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
7	product number 6	product name 6	200.00	Edit

Add

Delete Product

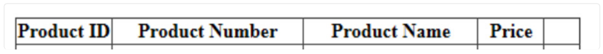
Clicking on "Delete" button would delete that record from the table. When you click on "Delete" button, **gvProducts_RowDeleting** event would be fired. In the event handler, we fetc product ID from the row you were trying to delete. Based on the product ID, which is primary key for tblProduct table, that record is deleted using ADO.NET API.

```
protected void gvProducts_RowDeleting(Object sender,
GridViewDeleteEventArgs e)
{
    // Get the product id of the selected product
    string ProductID = gvProducts.DataKeys[e.RowIndex].Value.ToString();
    //delete the product
    DeleteProduct(ProductID);
    gvProducts.EditIndex = -1;
    BindData();
}
private void DeleteProduct(string productID)
{
    //fetch the connection string from web.config
    string connString =
         ConfigurationManager.ConnectionStrings["Sql"].ConnectionString;
    //SQL statement to delete from products
    string sql = String.Format(@"Delete from tblProduct
                                where ProductID = {0}",
                                productID);
    using (SqlConnection conn = new SqlConnection(connString))
    {
        conn.Open();
        //Initialize command object
        using (SqlCommand cmd = new SqlCommand(sql, conn))
        {
            cmd.ExecuteNonQuery();
        }
    }
}
```

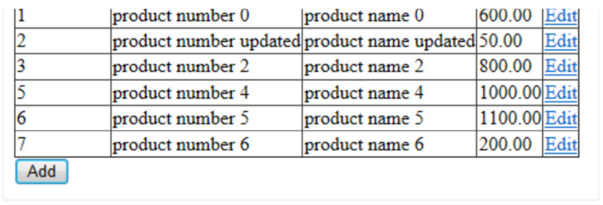


Product ID	Product Number	Product Name	Price	
1	product number 0	product name 0	600.00	Edit
2	product number updated	product name updated	50.00	Edit
3	product number 2	product name 2	800.00	Edit
4	product number 3	product name 3	900.00	Update Cancel Delete
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
7	product number 6	product name 6	200.00	Edit

Add



Product ID	Product Number	Product Name	Price	



1	product number 0	product name 0	600.00	Edit
2	product number updated	product name updated	50.00	Edit
3	product number 2	product name 2	800.00	Edit
5	product number 4	product name 4	1000.00	Edit
6	product number 5	product name 5	1100.00	Edit
7	product number 6	product name 6	200.00	Edit

Add

Cancel Edit:

If you ever want to get out the edit mode, click on “Cancel” button and you would be back in view mode.

```
protected void gvProducts_RowCancelingEdit
        (object sender, GridViewCancelEditEventArgs e)
{
    gvProducts.EditIndex = -1;
    BindData();
}
```

SQL Statements:

I have created a tblProduct table and loaded it with some test data. If you would like to use the same table, you could use below scripts.

```
CREATE TABLE [dbo].[tblProduct](
 [ProductID] [int] IDENTITY(1,1) NOT NULL,
 [Name] [varchar](255) NOT NULL,
 [ProductNumber] [varchar](255) NOT NULL,
 [ListPrice] [numeric](9, 2) NOT NULL,
 CONSTRAINT [PK_tblProduct_ProductID] PRIMARY KEY CLUSTERED
(
 [ProductID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO

DECLARE @COUNTER INT, @productname VARCHAR(255),
 @productnumber VARCHAR(255), @listprice DECIMAL
SET @COUNTER = 0
SET @productname = 'product name '
SET @productnumber = 'product number '
SET @listprice = 500
WHILE(@COUNTER <= 5)
BEGIN
 SET @listprice= @listprice + 100
 INSERT INTO [dbo].[tblProduct]
     VALUES
           (@productname + convert(VARCHAR, @COUNTER)
           ,@productnumber + convert(VARCHAR, @COUNTER)
           ,@listprice)
 SET @COUNTER = @COUNTER + 1
END
```

Be Sociable, Share!

MORE

Post Tagged with ASP.NET, C#, GridView

← Previous Post Ne

Software architect with over 10 years of proven experience in designing & developing n-tier and web based software applications, for Finance, Telecommunicat
Manufacturing, Internet and other Commercial industries. He believes that success depends on one's ability to integrate multiple technologies to solve a simple
well as complicated problem.

View all articles by Venu Gopal

Email : gopal@qnownow.com

Export data from ASP.NET GridView Control to ExcelEverything Technical says:

April 6, 2012 at 4:51 pm

[…] Using GridView in ASP.NET & C# –PART 1 Using GridView in ASP.NET & C# –PART 2 How to Insert, Update & Delete rows in ASP.NET GridView Control […]

Reply

tuba al says:

May 24, 2012 at 8:19 am

thanks, very useful.

Reply

sunil says:

May 28, 2012 at 11:22 am

if database table have no record, I mean grid not showing any record but I want to insert a new record, but insert is not enable if record count is less than 1.

Reply

Vanamali says:

June 10, 2012 at 7:05 pm

Hi Sunil

The reason you are not seeing any record is because the table layout is not drawn. When there are no records in the gridview, nothing is getting displayed. So w
to fix the gridview. Once you get atleast the header drawn, clicking on Add would show the footer row.

Modify BindData function in the above example as shown below and I have also created a new function FixGridHeader() that would show the header when recor
is zero.

```
private void BindData()
{
    //execute the select statement
    DataSet dsProducts = RetrieveProducts();
    DataTable dtProducts = dsProducts.Tables[0];

    //if it is an empty dataset, add a dummy row to the header
    if (dtProducts.Rows.Count == 0)
    {
        FixGridHeader(dtProducts);
    }
    else
    {
        //if the query returned results, just add them to the grid
        gvProducts.DataSource = dtProducts;
        gvProducts.DataBind();
    }
}

private void FixGridHeader(DataTable dataSource)
{
    //add blank row to the the resultset
    dataSource.Rows.Add(dataSource.NewRow());

    gvProducts.DataSource = dataSource;
    gvProducts.DataBind();

    //hide empty row
    gvProducts.Rows[0].Visible = false;
    gvProducts.Rows[0].Controls.Clear();
}
```

Also for more information, take a look at the below article that would explain to fix the header when there are no records in the gridview.
http://technico.qnownow.com/2012/06/08/show-gridview-header-when-there-is-no-data/

Reply

rabbil says:

June 4, 2012 at 11:21 am

excellent example, with all the concept….thanks a ton…!!

Reply

pviningston says:

July 2, 2012 at 5:31 am

UpdateProduct(ID, txtProductNumber.Text, txtProductName.Text, txtListPrice.Text);

this line takes the default values from the textboxes and not the updated values…
any reason why?

Reply

Vanamali says:

July 2, 2012 at 11:14 am

In the Page_Load event may be you are you rebinding the grid all the time? We should only bind it on initial load.

```
protected void Page_Load(object sender, EventArgs e)
{
    if(!IsPostBack)
        BindData();
}
```

Reply

pviningston says:

July 3, 2012 at 12:02 am

thanks. It worked….

Reply

HelpfulGuy says:

July 6, 2012 at 2:00 pm

Good article, but like everybody who posts, neglects the required using statements:

using System.Data;
using System.Data.SqlClient;
using System.Configuration;

Reply

NotACoder says:

July 17, 2012 at 8:40 am

The function for FixGridHeader does not work when recordcount is 0. Get 'System.Data.DataRowView' does not contain a property with the name 'ID.' I am also using an id
field as my ID field.

Reply

Vanamali says:

July 18, 2012 at 11:01 am

I do not think it has to do anything with record count. Is your code working fine if there is existing data? Please Check your SELECT statement and see if you are included "ID" in it.

Reply

NotACoder says:

July 18, 2012 at 5:58 pm

I have included the ID field in my SELECT statement. I havent tried adding any rows to the table yet. It's blank. Brand new application.

Reply

Jack says:

July 18, 2012 at 8:36 am

First of all Thanks for tutorial.
I am trying to insert sth I have re-write this code step by step for my data. But texbox' text is always null after writing some values to them.I am talking about textboxes lik

Reply

Vanamali says:

July 18, 2012 at 8:57 am

code is missing after textboxes like this.. please try to put the between

```
[Code][/Code]
```

tags , with a small c instead of capital

Reply

Jack says:

July 18, 2012 at 9:17 am

<asp:Label ID="projectNameLbl" runat="server" Text='' />

I couldn't implement your advise can you help me on code?

Reply

Jack says:

July 18, 2012 at 9:19 am

Sorry I wrote wrong code, right one is below.

<asp:TextBox ID="txtUserName" runat="server" Text='' />

Reply

Vanamali says:

July 18, 2012 at 9:33 am

Yes. Please email me your code.

Reply

Jack says:

July 18, 2012 at 9:35 am

```
<asp:TemplateField HeaderText="User Name">
                        <ItemTemplate>
```

```
        <%#Eval("userName")%></ItemTemplate>
    <EditItemTemplate>
        <asp:TextBox ID="txtUserName" runat="server" Text='<%#Eval("userName")%>' />
    </EditItemTemplate>
    <FooterTemplate>
        <asp:TextBox ID="txtNewUserName" runat="Server" />
    </FooterTemplate>
</asp:TemplateField>
```

Vanamali says:

July 18, 2012 at 10:34 am

The Edit Template looks correct, are you seeings values in the grid?

When are you getting text box value as null, is it while inserting or while updating?

Reply

Jack says:

July 18, 2012 at 10:59 am

I have working on inserting and get nulls. in footer textboxes texts are null. but I can display my data.

Reply

Vanamali says:

July 19, 2012 at 6:35 pm

Please check if you are rebinding the grid in the page_load all the time. You could should only bind it during page_load and not on subsequent postbacks.
protected void Page_Load(object sender, EventArgs e)
{
if(!IsPostBack)
BindData();
}

Reply

Anuj Kumar Rai says:

July 19, 2012 at 6:03 am

dear sir when i m using to edit,update and cancelEdit template in gridview source code…and after that i write all .cs code then it is giving error

Error 1 'ASP.default_aspx' does not contain a definition for 'GridView1_Editing' and no extension method 'GridView1_Editing' accepting a first argument of type 'ASP.defau could be found (are you missing a using directive or an assembly reference?)

Error 2 'ASP.default_aspx' does not contain a definition for 'GridView1_update' and no extension method 'GridView1_update' accepting a first argument of type 'ASP.default_aspx' could be found (are you missing a using directive or an assembly reference?)

Reply

Vanamali says:

July 19, 2012 at 9:32 am

You might have written **OnRowEditing="GridView1_Editing"** in ASPX page but in the code behind you might have still kept the **gvProducts_RowEditing** as nam the event. Please make sure both of them are the same.

Reply

sobhan says:

July 19, 2012 at 5:44 pm

How do I refer txtProductName from the code behind? It is not identified since it is inside the gridview control. Any help on this would be appreciated.

Reply

Vanamali says:

July 19, 2012 at 6:39 pm

Hi Sobhan,

That is why we use findcontrol method of the gridview row to get reference to the control. something like this …

TextBox txtProductName = (TextBox)row.FindControl("txtProductName");

Reply

sobhan says:

July 19, 2012 at 9:52 pm

Hi
I asked you a wrong question. I would like to identify the control on client side in javascript.

Reply

Manish Joshi says:

July 21, 2012 at 2:53 pm

nice article…

Reply

Chetan says:

July 24, 2012 at 9:04 am

Amazing solution

Thanks…

Reply

Chetan says:

July 25, 2012 at 6:30 am

Please explain , how can I add dropdownlist in this code.

Thanks

Reply

Vanamali says:

July 25, 2012 at 5:47 pm

you will find that solution in these articles. It was explained how to use dropdownlist in GridView.

http://technico.qnownow.com/2012/04/05/use-dropdownlist-in-asp-net-gridview-control/

http://technico.qnownow.com/2012/06/26/using-radiobuttonlist-dropdownlist-gridview-row/

Reply

jlwill3 says:

July 24, 2012 at 11:53 am

Thank you for a very useful article.

It's a shame the gridview does not have insert functionality built in, and that we have to result to a hack. One thing that gets lost using this method is the ability to use buil field validation capability of asp.net. Validation doesn't work on controls that are placed in the footer. Probably for the same reason as why the footer controls are not avai the code-behind. I'm sure that's by design as the footer was never intended to contain these types of controls anyway.

I just ended up adding some validation code to the method for creating a new record, and sending a javascript alert to the user if validation fails, and the reason why it fail

Reply

Name

Email

Website

Comment

You may use these HTML tags and attributes: <a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <cite> <code> <del datetime=""> <em> <i> <q cite=""> <st <strong>

☐ Notify me of follow-up comments by email.

☐ Notify me of new posts by email.

Search

Qnow Now Home

Qnow Now Technical

Qnow Now Funzone

Qnow Now Finance

Qnow Now Politics

The All New-GNIIT
Cloud Centric Curriculum
NIIT
THE TURNING POINT
• Blend of Core IT Skills, Professional Skills & Cloud Technology
Enroll Now
For Class 12th pass-outs/appeared and college students
NEW GNIIT
India's First Cloud Campus

Implementing the output clause in SQL Server

FAQ 5

FAQ 4

FAQ 3

FAQ 2

July 2012

June 2012

May 2012

April 2012

March 2012

Qnownow | About | Contact | Privacy Policy |

Colorway Wordpress theme by InkThemes.com