# **ASP Gridview Programming**

The GridView is composed of a set of fields that indicate what properties from the DataSource are to be included in the rendered output along with how the data will be displayed. The simplest field type is the BoundField, which displays a data value as text. Other field types display the data using alternate HTML elements. The CheckBoxField, for example, renders as a check box whose checked state depends on the value of a specified data field; the ImageField renders an image whose image source is based upon a specified data field. Hyperlinks and buttons whose state depends on an underlying data-field value can be rendered using the HyperLinkField and ButtonField field types, respectively.

While the CheckBoxField, ImageField, HyperLinkField, and ButtonField field types allow for an alternate view of the data, they still are fairly limited with respect to formatting. A CheckBoxField can display only a single check box, whereas an ImageField can display only a single image. What if a particular field must display some text, a check box, and an image, all based upon different data-field values? Or what if we wanted to display the data using a Web control other than the CheckBox, Image, HyperLink, or Button? Furthermore, the BoundField limits its display to a single data field. What if we wanted to show two or more data-field values in a single GridView column?

To accommodate this level of flexibility, the GridView offers the TemplateField, which renders using a template. A template can include a mix of static HTML, Web controls, and data-binding syntax. Furthermore, the TemplateField has a variety of templates that can be used to customize the rendering for different situations. For example, the ItemTemplate is used by default to render the cell for each row, but the EditItemTemplate template can be used to customize the interface when editing data.

For farther practical knowledge please visit :

<http://msdn.microsoft.com/en-us/library/bb288031.aspx>

# **How to access control inside TemplateField**

You can also access controls inside templatefield in your grid using the FindControl method like

DropDownList DDL =

(DropDownList)this.GridView1.Rows[index].Cells[0].FindControl("DropDownList1");

***OR***

DropDownList DDL =

(DropDownList)this.GridView1.Rows[index].FindControl("DropDownList1");

***you can also access specific control in all rows using :***

foreach(GridViewRow row in GridView1.Rows) {

if(row.RowType == DataControlRowType.DataRow) {

HyperLink myHyperLink = row.FindControl("myHyperLinkID") as HyperLink;

}

}

# Accessing Controls in TemplateFields on GridView Edit Mode

This example demonstrates how to access ASP.NET server controls that resides within the TemplateField Column of GridView on Edit Mode. In this example we are using the PreRender event of GridView instead of RowEditing event since we cannot directly access controls at RowEditing event of GridView.

***See below for example:***

protected void GridView1\_PreRender(object sender, EventArgs e){

if (this.GridView1.EditIndex != -1){

Button b = GridView1.Rows[GridView1.EditIndex].FindControl("Button1")

as Button;

if (b != null){

//do something

}

}

}

Or you can fool the ASP.Net Engine by passing -1 for the event in order to do inline update like in the following example

protected void gvPos\_RowUpdating(object sender, GridViewUpdateEventArgs e)

{

GridViewRow row = gvPos.Rows[e.RowIndex];

TextBox titleID = (TextBox)row.FindControl("txttitleid");

TextBox t1 = (TextBox)row.FindControl("txtTile");

cmd.CommandText = "UPDATE …" ;

con.Open();

cmd.ExecuteNonQuery();

con.Close();

e.Cancel = true;

}