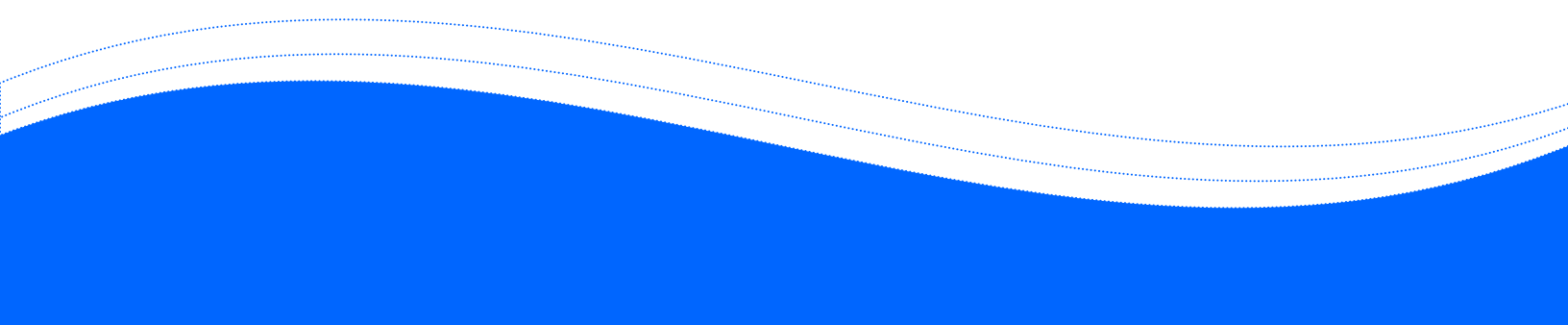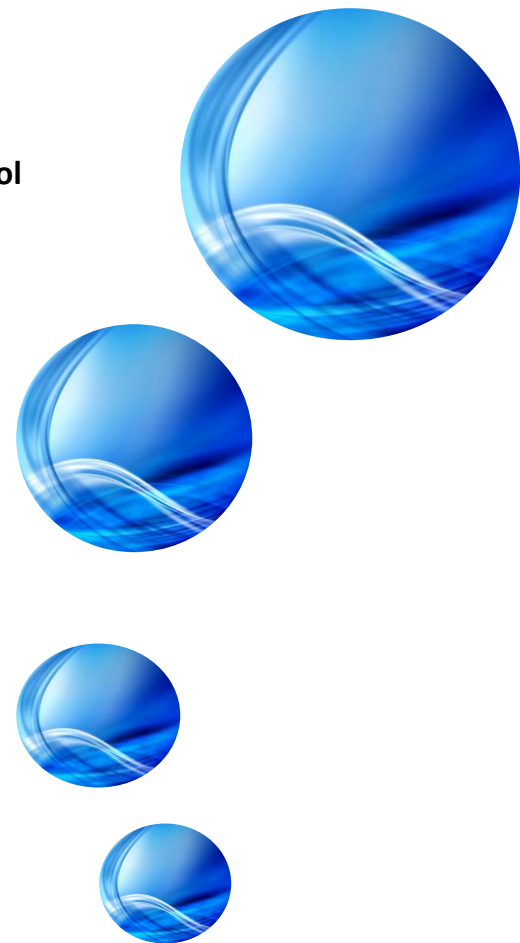# DATA GRID VIEW CONTROL

Gives you the details about Data Grid View Control

J

o   k o k J   kl   Jk k k   k   J

J   m   p J   J k   kl oJ   po   k J

Venkatesan Prabhu Jayakantham has more than 8 years experience in the Microsoft Technologies such as VB.Net, ASP.Net, C#.net, SSIS, SSAS, ADO.Net, etc. He is the Managing Director of KAASHIV INFOTECH (http://www.kaashivinfotech.com/), a software company in Chennai. Before that, he worked in HCL Technologies (India and Australia) for six years as Project Lead.

As a service motive, Venkat contributed more than 700 articles which is read by the developers in 170 countries (400 developers per day) (http://venkattechnicalblog.blogspot.com/). Aligned with KaaShiv InfoTech's mission, he met more than 20,000 young minds and spreaded Microsoft Technologies / Career guidance programs.

Venkat won many awards in his career, which includes Prestigious **Microsoft award MVP (Most Valuable Professional)** award for the years 2008,2009,2010,2011 and won many awards.

List of other awards in his career,

- Microsoft certified Smart .Net Candidate in 2004
- Most valuable member for dotnetspider site in 2007
- HCL SQL Subject Matter expert (SME - SQL Server) for the year (2008,2009)
- HCL Special contribution award winner on Dotnet skills for year(2008)
- HCL SQL Knowledge Champion for the year 2009
- MindCracker MVP on SQLServer –2010 for the year 2010,2011
- INETA champion - Gold Member – 2010 for the year 2010
- HCL Service Contribution Award for the year 2010
- Leading Lights "Rising Star" award from Common Wealth Bank, Australia for the year 2010
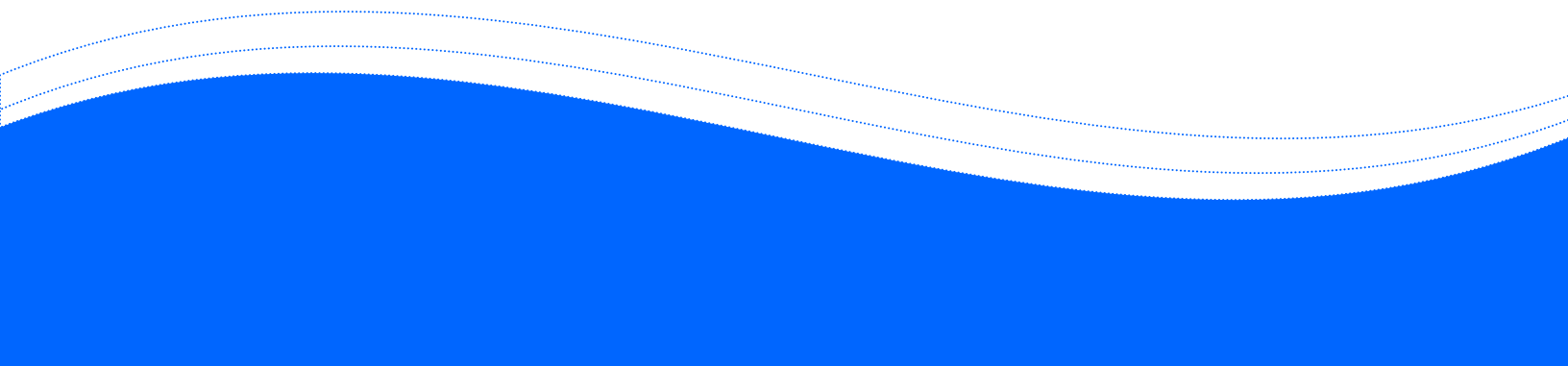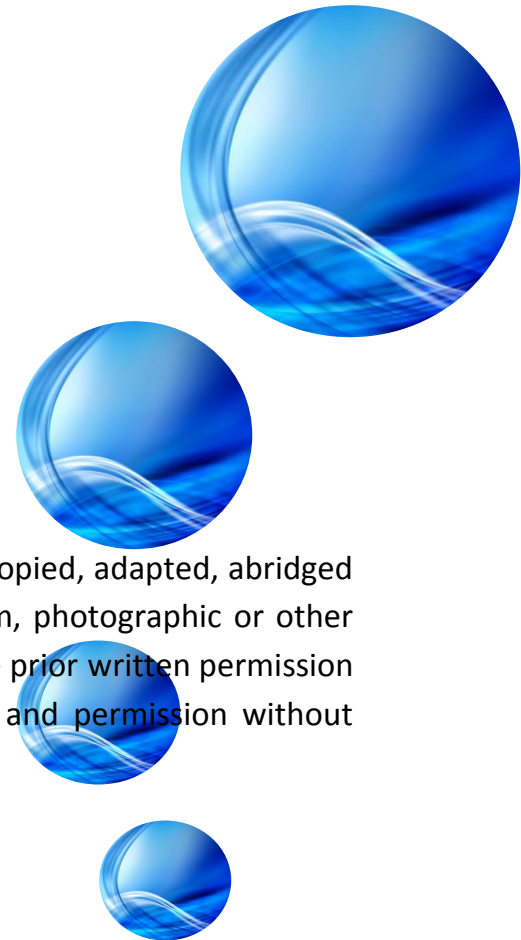
Technical certifications

- Cisco certified Network Associate (CCNA) – 2004
- Microsoft Certified Application Developer (MCAD) – 2005

I would like to thank my family members for their support and encouragement. Without them it would impossible for me to publish this e-book.

I would also like to thank my KaaShiv InfoTech team for their support to publish this e-book.

Data Grid View Control

DataGridView Control:

DataGridView control provides the best way to bind data from database tables to UI. The data is displayed in a tabular format. The Header Style, Row Style, Alternating RowStyle properties of the grid eases in formatting the grid display. Various types of columns can be added in gridview such as checkbox, radiobutton, textbox, dropdown etc.

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

Grid View Control

The main advantage of gridview control is easy data modification. That is, gridview control allows the user to add rows, modify existing row data, delete a row and update a row data. Also data from more than one table can be displayed in gridview.

	Account Key	Account Description	Account Type	Value Type
Edit Delete	Databound	Databound	Databound	Databound
Edit Delete	Databound	Databound	Databound	Databound
Edit Delete	Databound	Databound	Databound	Databound
Edit Delete	Databound	Databound	Databound	Databound
Edit Delete	Databound	Databound	Databound	Databound

Editable gridview control

The above gridview is an editable grid with databound columns. Columns in gridview can be auto generated when data binds to grid or added using "asp:TemplateField" or "asp:BoundField". Below is the TemplateField code for "Account Description" Column:

4

```
<asp:TemplateField HeaderText="Account Key" ItemStyle-HorizontalAlign="Center">
    <ItemTemplate>
        <asp:Label ID="lblAccKey" runat="server" Text='<%# Eval("AccountKey") %>' />
    </ItemTemplate>
</asp:TemplateField>
<asp:TemplateField HeaderText="Account Description">
    <ItemTemplate>
        <%# Eval("AccountDescription") %>
    </ItemTemplate>
    <EditItemTemplate>
        <asp:TextBox ID="txtAccDesc" runat="server" Text='<%# Eval("AccountDescription") %>'></asp:TextBox>
    </EditItemTemplate>
    <FooterTemplate>
        <asp:TextBox ID="add_AccDesc" runat="Server" />
    </FooterTemplate>
</asp:TemplateField>
```

GridView Template Fields

In the above code, the binding data is displayed with <ItemTemplate> tag and while editing, the data is displayed in a text box defined within <EditItemTemplate> tag.

The <FooterTemplate> tag defines a text box which is displayed when user inserts a new record using gridview control.

Below is code for Edit, Update, Delete and Cancel events in gridview:

```
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
{
    GridView1.EditIndex = e.NewEditIndex;
}
protected void GridView1_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)
{
    GridView1.EditIndex = -1;
}
protected void GridView1_RowUpdating(object sender, GridViewUpdateEventArgs e)
{
    GridViewRow row = (GridViewRow)GridView1.Rows[e.RowIndex];
    Label lblAK = (Label)row.FindControl("lblAccKey");
    TextBox textDesc = (TextBox)row.FindControl("txtAccDesc");

    GridView1.EditIndex = -1;

        // form the update query
    lblAK.Text = gridview.gridQuery = "update DimAccount set AccountDescription = txtDesc.Text = "where AccountKey =

        conn.Close();
    }
```

Edit, Update and Cancel

The Edit link in gridview changes as Update and Cancel, when the user clicks on edit button in a particular row. Hence the above code helps to handle edit, update and cancel events. The Update event updates the record in database table.

The delete button, when clicked, deletes the selected row from the database.

```
protected void GridView1_RowDeleting(object sender, GridViewDeleteEventArgs e)
{
    GridViewRow row = (GridViewRow)GridView1.Rows[e.RowIndex];
    Label lblDeleteKey = (Label)row.FindControl("lblAccKey");
    conn.Open();
    SqlCommand cmd = new SqlCommand("delete DimAccount1 where AccountKey=" + lblDeleteKey.Text + "", conn);
    cmd.ExecuteNonQuery();
    conn.Close();
    BindData();
}
```

Delete Event in GridView

s

Coming to adding records, there are two types of record addition events. One is adding a new row to a fresh table and the other one is adding a row to an existing set of rows in a table.

Record addition is done with the help of footer template in gridview.

	Account Key	**Account Description**	
Edit Delete	Databound	Databound	
Edit Delete	Databound	Databound	
Edit Delete	Databound	Databound	
Edit Delete	Databound	Databound	
Edit Delete	Databound	Databound	

Bind Data   Add   Cancel

On clicking add button, the footer row is displayed as below:

	Account Key	**Account Description**	**Account Type**
Edit Delete	22	Current	Current
Edit Delete	45	Savings	Savings
Edit Delete	52	New Acc	Savings
Edit Delete	53	Savings 1	Savings 1
			Insert

Bind Data   Add   Cancel

When the user types the values and click Insert, the record is inserted into the table in db and displayed in grid after binding as follows:

	Account Key	Account Description	Account Type
Edit Delete	22	Current	Current
Edit Delete	45	Savings	Savings
Edit Delete	52	New Acc	Savings
Edit Delete	53	Savings 1	Savings 1
Edit Delete	54	Savings 2	Savings 2

`Bind Data` `Add` `Cancel`

New record is added to grid with the help of <EmptyDataTemplate> tag as follows:

```
<EmptyDataTemplate>
    Account Description
    <asp:TextBox ID="txtADesc" runat="Server" /><br />
    AccountType
    <asp:TextBox ID="txtAType" runat="Server" /><br />
    <asp:Button ID="btnInsert" runat="Server" Text="Insert"
    CommandName="EmptyInsert"
        UseSubmitBehavior="False" />
</EmptyDataTemplate>
```

When an empty table is bind to gridview, then the gridview paves way to add a new record with the help of above code. It appears in the design view as follows:

```
Account Description [        ]
AccountType         [        ]
[ Insert ]
```

`Bind Data` `Add` `Cancel`

When a new record data is typed and Insert button is clicked, the grid is displayed as follows:

```
Account Description [New1    ]
AccountType         [New1    ]
[ Insert ]
```

`Bind Data` `Add` `Cancel`

	Account Key	Account Description	Account Type
Edit Delete	56	New1	New1

`Bind Data` `Add` `Cancel`

```csharp
protected void GridView1_RowCommand(object sender, GridViewCommandEventArgs e)
{
    if (e.CommandName == "Insert")
    {
        TextBox txtAccDesc = (TextBox)GridView1.FooterRow.FindControl("add_AccDesc");
        TextBox txtAccTyp = (TextBox)GridView1.FooterRow.FindControl("add_AccType");

        string strInsert = "Insert into DimAccount1(AccountDescription, AccountType) values('" +
            txtAccDesc.Text + "','" + txtAccTyp.Text + "')";

        conn.Open();
        SqlCommand cmd = new SqlCommand(strInsert, conn);
        cmd.ExecuteNonQuery();
        conn.Close();
    }
    else if (e.CommandName == "EmptyInsert")
    {
        TextBox txtAccDesc = (TextBox)GridView1.Controls[0].Controls[0].FindControl("txtADesc");
        TextBox txtAccTyp = (TextBox)GridView1.Controls[0].Controls[0].FindControl("txtAType");

        string strInsert = "Insert into DimAccount1(AccountDescription, AccountType) values('" +
            txtAccDesc.Text + "','" + txtAccTyp.Text + "')";

        conn.Open();
        SqlCommand cmd = new SqlCommand(strInsert, conn);
        cmd.ExecuteNonQuery();
        conn.Close();
    }
}
```

Paging in gridview, helps to restrict the data displayed in a single page thus, giving a neat appearance for the grid data. When the `AllowPaging="True"` is set, the grid appears as follows:

	Account Key	Account Description	Account Type
Edit Delete	56	New1	New1
Edit Delete	57	Savings1	Savings1
Edit Delete	58	Savings2	Savings2

1 2

Bind Data | Add | Cancel

Paging event has to be handled as below:

```csharp
protected void GridView1_PageIndexChanging(object sender, GridViewPageEventArgs e)
{
    GridView1.PageIndex = e.NewPageIndex;
}
```