|  |
| --- |
|  |
| iFlexGrid - Walkthrough |
|  |
| Technology Group iInterchange Systems Pvt. Ltd. |

*Note: This is a privileged and confidential document of iInterchange Systems Pvt Ltd. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), without prior written consent from* ***iInterchange***

Table of Contents

[1. Introduction 3](#_Toc223232723)

[1.1. Purpose 3](#_Toc223232724)

[1.2. Scope 3](#_Toc223232725)

[1.3. Definitions, Acronyms and Abbreviations 3](#_Toc223232726)

Introduction

* 1. Purpose

This document gives an overview of how to make Use of iFlexGrid Step by Step.

* 1. Scope

The Main Scope of this Document is to know the Usefulness of the iFlexGrid which can be used for the Following Purposes:

1. Single Cell Editable.

2. Updating Records Row wise.

3. Deleting and Inserting Records.

4. All the Type of Validations can be done For each and Every Cell Based on the Controls used for columns inside of the iFlexGrid.

**1.3 Definitions, Acronyms and Abbreviations**

|  |  |
| --- | --- |
| iFlexGrid | iFlexGrid which is used for all the db manipulations |
| Product | Product also stands for Project |
| Tech Group | Technology Group |
| TL | Team Leader |
| VS | Visual Studio 2008 |
| Northwind | The Database Used in the code snippets. |

**Walkthrough To Just Bind The iFlexGrid With the Table**

**To Prepare for this walk****through:**

Install the web controls setup in the system and Include all the necessary dll’s as references in the website which you are going to use and include the controls in the VsToolbox by right clicking in the Toolbox and choosing required items.

1.Drag and Drop the iflexGrid Control From the Toolbox into the WebPage.

2. Select **Properties** of the iFlexGrid and inside that Click on the **Columns** Collection button.

3.A Dialog would appear Where we can add various Fields like TextBox,Lookup,Checkbox,Date,Container,HyperLink,Image,Expandable and Bound Fields.

4.Any Field Selected ,**DataField** Property (i.e., the column name From the DB) and **Header Text** Should be mentioned in that Field.

5.Extra Added Features like Header Image Url and other Formatting options which are similar to that of Gridview.

6.Based on the Field Selected all the features of that Field which when used individually does the same functionality inside the iFlexGrid.(For Eg:TextBox Field when used would have all those features of its iTextBox inside the iFlexGrid)

7.Based on the Requirements any number of columns can be included in that Dialog of the **columns collection** and clicked on **Ok.**

8.Then in the iFlexGrid There will be a property called **DataKeyNames** which will be the **Column Name(i.e., The Primary key or the ID)** of the Table which will be **binded** to the iFlexGrid.

9.Then there is a property called AddButtonText Which can be changed according to the Requirements.

10. The Properties like **AllowDelete,AllowEdit,AllowAdd** when set to True Will display the Add and Delete links at the footer of the **iflexgrid** will become editable.

11.A Single Cell will become Editable by just clicking the cell in the grid.

12. Updating will occur based on the row wise when moved onto different row by changing values in one row.

13. Delete record by selecting a particular record and clicking the delete button in the footer row.

14.All the validations would occur when the focus moves from the cell.

15. Paging is possible, which is same as that of Gridview.

16. A Grid with in a Grid(i.e., Nested Grid) is also possible.

17. A single Cell Editable feature in Nested Grid also possible.

18. The iflexgrid control **HeaderRows** property should be given based on the Design how the iflexgrid needs to be rendered number of rows needs to be given .

**The Following Code snippet can be used to bind the iFlexGrid:**

The Datakey Name🡪 EmployeeID

The columns under column collection🡪 LastName and FirstName.

Protected Sub Page\_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load

Dim con As New SqlConnection("data source=iic46\sql2005;user id=sa;password=sa;initial catalog=northwind;")

con.Open()

Dim dap As New SqlDataAdapter("SELECT EmployeeID,LastName,FirstName From Employees", con)

Dim ds As New DataSet

dap.Fill(ds)

iFlexGrid1.DataSource = ds

iFlexGrid1.DataBind()

End Sub

**Paging in iFlexGrid:**

1.To allow Paging in iFlexGrid **Allow Paging** Property should be set **True**.

2. using the **Page Size** Property we can specify number of Records to be displayed in a single page.

3. **usecachedDatasource** Property should be set to **True** For paging to occur.

**Sorting in iFlexGrid:**

1. A property Mentioning Allow Sorting which should be set to True.
2. Inside the iFlexGrid Binded Field , There will be a Property SortExpression Which should be mentioned as the DataField Name.
3. Sort Expression Should be mentioned for those columns which we require sorting.

**Merge able Header Columns in iFlexGrid:**

1. Based on the Requirements and how the Grid is required to be rendered **Header Rows** Property should be mentioned(i.e.,1 or 2).
2. A rendered Method should be mentioned in the code behind page.
3. The Header Text should not be mentioned inside the columns of the iFlexGrid those which is going to be rendered by means of code behind.

**The code snippet is as follows:**

Header Rows 🡪 1

Sub NewRenderMethod(ByVal writer As HtmlTextWriter, ByVal ctl As Control)

writer.Write("<TD align=center border=0 colspan=2 >EmployeeID</TD>")

writer.Write("</TR>")

IFlexGrid1.HeaderStyle.AddAttributesToRender(writer)

writer.Write("<TR>")

Dim iLoop As Integer

For iLoop = 0 To ctl.Controls.Count - 1

If iLoop >= 0 And iLoop <= 2 Then

CType(ctl.Controls.Item(iLoop), WebControl).CssClass = "gHdr"

ctl.Controls.Item(iLoop).RenderControl(writer)

End If

Next

End Sub

1. In the **iFlexGrid\_RowCreated** Event that rendered method should be mentioned.

**The Following Code snippet:**

Protected Sub iFlexGrid1\_RowCreated(ByVal sender As Object, ByVal e As iInterchange.WebControls.v3.Data.iFlexGridRowEventArgs) Handles iFlexGrid1.RowCreated

If e.Row.RowType = DataControlRowType.Header Then

e.Row.SetRenderMethodDelegate(New RenderMethod(AddressOf NewRenderMethod))

End If

End Sub

**Navigation Through iFlexGrid:**

1. A Tab will move to the next cell which is an editable one.
2. If there is any readonly cell it will automatically go the next editable cell.
3. A SHIFT+TAB Moves to the Previous Cell From the Current Focus of the Cursor Cell.

**Adding a Record in a iFlexGrid:**

1. When the Allow Add Property of the iFlexGrid when set to True it would display a Footer with Addrow link.
2. When clicked on that Addrow , A new row is added in the same page index of the iflexgrid at the bottom.
3. The new row added will be moved to the last page of the iflexgrid only when we move to some other page or while saving ,deleting records.

**Updating a Record in a iFlexGrid:**

1. All the Updations of the Records would occur Row by row based on the focus which moves from one row to another row .
2. Based on the changes in the iFlexGrid the Data will be saved in the Datasource.

**Deleting a Record in a iFlexGrid:**

1. A Record in the Datasource would get deleted from the iflexgrid immediately After Selecting the Record and pressing the Delete Button.

**Validation in the iFlexGrid:**

1. All Validations would occur when the focus moves from one cell to another cell, if the validation fails Focus would not move to next cell.

Nested Grid or (Expandable Grid):

# A nested Grid is a one which is a iflexgrid with in another iflexgrid.

1. Two iflexgrid should be dragged and dropped on the Aspx page.
2. First all the columns required should be binded with the Second iflexGrid which is similar to as mentioned in **Just Bind The iFlexGrid With the Table** section.
3. The Column where we Require the Expandable Field + should be binded with the second iFlexGrid.
4. There will be property in the Expandable Field GridID which should be mentioned as ID of the First iFlexGrid or the Grid which you desire to come under it when expanded.
5. Then the following code snippet should be used to bind the Second iFlexGrid which was binded by using above 5 points.

**Code snippet:**

Private Sub bindgrid()

Try

Dim str As String = "SELECT EmployeeID,LastName,FirstName FROM Employees"

Dim adopt As New SqlDataAdapter(str, con)

Dim dt As New DataTable()

adopt.Fill(dt)

IFlexGrid2.DataSource = dt

IFlexGrid2.DataBind()

Catch ex As Exception

Response.Write("Error " & ex.Message)

End Try

End Sub

The above code should be called in the page load Event.

1. Then the First iFlexGrid should be binded with all the columns of the corresponding dependent table. For Example:

Employees Table and EmployeeTerritories Table of the Northwind Database(where EmployeeID would be available in both Employees and EmployeeTerritories)

1. Then in the First iFlexGrid (i.e., iFlexGrid1\_Expanded Event the following code snippet should be entered)

**Code snippet:**

Protected Sub iFlexGrid1\_Expanded(ByVal sender As Object, ByVal e As iInterchange.WebControls.v3.Data.iFlexGridExpandedEventArgs) Handles iFlexGrid1.Expanded

'Dim strQuery As String = "SELECT TerritoryID,(select EmployeeID From Employees where EmployeeID=EmployeeTerritories.EmployeeID) EmployeeID From EmployeeTerritories"

Dim strQuery As String = "SELECT TerritoryID,EmployeeID From EmployeeTerritories"

Dim adopt As New SqlDataAdapter(strQuery, con)

Dim dt As New DataTable()

Dim dtp As New DataTable()

dtp = CType(iFlexGrid2.DataSource, DataTable)

adopt.Fill(dt)

Dim dv As DataView = dt.DefaultView

dv.RowFilter = "EmployeeID=" & dtp.Rows(e.RowIndex).Item("EmployeeID")

e.DataSource = dv

End Sub

1. The Datakey Names for both the iFlexGrid’s Should be mentioned.

The useCachedDatasource Property Should be set True for both the iFlexGrid’s.

1. The Type of the Parent Grid should be Mergeable.

**Search in Grid:**

1. For Search to happen if we just set the **Allow Search** Property to **True,** Automatically a Search Button would appear in the Footer of the iflexGrid.
2. When Search Button is clicked it would Display a blank html input type textboxes in the first row of the iflexgrid with a filter icon on its side.
3. When filter icon is clicked it would display a menu for filtering which would have Like,Not Like,In,Not In and equal.
4. After selecting any one in that menu , we need to click on search.
5. A Cancel button would also get generated at the bottom , if you click on cancel button the search row generated on the top would go off.

**Other Properties of iFlexGrid:**

1. Various other properties of iFlexGrid like AllowEdit,AllowDelete,Allowpaging,AllowStaticHeader,AllowSearch,staticHeaderHeight.
2. The Properties AllowEdit,AllowDelete,AllowPaging we need to set True or False based on the Requirements in the iFlexGrid.
3. The Property AllowStaticHeader when set to True , The Header would Remain Static (i.e., Even if we scroll the rows would move but the header would remain static),if set to false the header would not stay static ,Static Header Height should also be set which should be mentioned in pixels .
4. The Properties like **usecachedDatasource** which should be set to True in order to maintain datasource in the session at the control level itself by default it would be false.
5. The Properties which should be Mentioned **Mandatory** are Datakeynames and **Header Rows**.
6. Validation Group is a property which can be used when we want to validate only some controls on the page not for all. Validation would occur based on a particular group.
7. **onAfterCallback:**
8. In this we need to specify the Function Name which we need to Fire after the call back operation is over.
9. Parameters can be passed from the server side by setting the **e.outputparameters** in the server side events like **RowInserting**, **RowUpdating** and **RowDeleting** Events, which will be accessible in the client side function.
10. Specify the function name that will be called **after the callback** operation is **over**.
11. **onBeforeCallback:**
12. This is similar to onAfterCallback only difference the function would fire **before Callback** **operation** takes place.
13. In this Parameters can be passed from the client side which will be accessible in the server side events like RowInserting,RowUpdating and RowDeleting Events
14. Which can be done by using **e.inputparameters**
15. **onClientExpand:**
16. This Function will be called before the **callback operation** for expanded grid takes place.
17. This Function can be used to pass parameters from the client side to the server side.
18. This can be accessed in the Expanded Event of the Grid by using **e.parameters** arguments.

**Code Snippet:**

function onexpan() {

alert('onexpand');

}

**The above function needs to be called in the onclientExpand Property of the iflexgrid where we have the expandable field binded into it.**

1. **onClientCollapse:**
2. This Function will be called before the callback operation of collapsing the Grid takes place.
3. This Function can be used to set the values of the column in the parent Grid using the values of the Child Grid at the time of collapsing.

The below code snippet is just a function to call onclientcollapse when expandable field of the child iflexgrid is collapsed.

**Code Snippet:**

function oncollapse() {

alert('collapsing');

}

The above function should be called on the onclientcollapse property of the iflexgrid where we have expandable field binded to it.

For **OnclientExpand** and **OnClientCollapse** Property to Funtion we need to set the type of the Grid which has expandable field as **Merge able**

**iFlexGrid Field Collections:**

In the iFlexGrid the Following Fields would be available

1. iTextBoxField

2. iLookupField

3. iDateField

4. iHyperLinkField

5. iContainerField

6. iCheckBoxField

7. iExpandedField

8. Bound Field.

**iTextBoxField:**

1. Its a TextBox Field , Which would have all the Validations which can be done using a iTextBox Field (i.e., Required Field,Regular Expression etc.,).
2. There would be properties for iTextbox under which we can set all validations.
3. If we just specify the DataField its enough.

**iLookupField:**

1. After selecting a LookupField inside that there will be a field **DataField**,

We need to specify the columnname which we have specified in the **datasource**.

1. Inside that there would be a Property called Lookup inside that we need to specify the **Datakey** (i.e., The tables datakey value From which we are populating Lookup).
2. Then we need to specify the Column Names under the Lookup Column Collection.
3. Similarly all the Dependent Child Controls properties etc., can also be done the same way as it can be done in the individual ilookup control.
4. There will be Two Properties **Foreign Data Field** and **Primary Data Field.**

**A Primary Data Field** is the one Column Name From the Table which we are going to configure Lookup.(For Eg: in State Table st\_id will be Primary Key and if we are using state table for lookup Primary key will st\_id)

**A Foreign Data Field** is the column name which is referred in the Parent Table.

(For Eg: in City Table The Stateid (i.e., the st\_id of the State table) is referred in the city Table as stt\_id therefore it will come in Foreign Data Field)

**iDateField :**

1. The **HTMLENCODE** by default this value would be true, it should be specified as false, in order to set the date in a particular format.
2. The Date Format String which gets or sets the date format value should be {0:dd-MMM-YYYY}

**iExpandedField:**

1. The Expanded Field is used For the Purposes of Nested Grid .
2. Which will Get Grid within another Grid.

**iHyperLinkField:**

1. The column Name From the datasource should be specified which would be displayed in the cell.

**Static Header Grid:**

Static Header Grid is a one whose header would be static.

For Static Header we need to Specify the Following Properties

1. Static Header Height (which should be in pixels).
2. Header Rows.
3. Allow Static Header based on the (Requirement as True or False).

**Some of the Very Important Properties of the iFlexGrid:**

* UseCachedDataSource
* AutoGenerateColumns(which should be false when binding from codebehind)
* AutoGenerateColumns When Set to True, Add,delete,editing,search rowFunctionalities would not work.
* HeaderRows
* Type (in case of nested.)
* If you set the Autogeneratecolumns property to true in iFlexGrid – Add,Edit,Delete Row functionality would not work.
* Allow Search,Allow Refresh,Allow AutoSearch for Search Operations to occur.
* Datakey
* StaticHeaderHeight(in case of Static Header in pixels)
* AllowStaticHeader(True or False).