

فصل دهم :

اتصال کنترل های سرور وب به پایگاه داده

مقدمه :

کنترل های متنوعی همراه ASP.NET با خاصیت اتصال به پایگاه داده ارائه شده اند که بازه ی وسیعی از توانایی ها و انعطاف پذیری را ارائه می دهند. در طی فصل جاری به صورت مشروحی توانایی های آنها را خصوصا پرکاربردترین آنها یعنی دیتا گرید را مورد بررسی قرار خواهیم داد.

نمایش داده های تکرار شونده :

می توان از کنترل Repeater برای نمایش داده ها به صورت لیست های ویژه و سفارشی استفاده کرد. برای فرمت کردن آنها می توان از Templates استفاده نمود که می تواند شامل Header و Footer و غیره باشد.

همانند دیتاگرید که در طی فصول پیشین طرز استفاده ی مقدماتی از آن را آموختید ، باید خاصیت DataSource را تنظیم و سپس متد DataBind آنرا فراخوانی کرد. این کنترل انواع مختلفی از Templates را پشتیبانی می کند. از ItemTemplates هنگامی استفاده می شود که بخواهیم فرمت خروجی را در هر ردیف داده ها مشخص نماییم که اجباری می باشد. از AlternatingItemTemplate



برای فرمت کردن خروجی ردیف ها به صورت یک در میان استفاده می شود. از HeaderTemplate و FooterTemplate برای خروجی که قبل و بعد از داده ها می آیند استفاده می کنیم. از SeperaterTemplate برای فرمت کردن بین ردیف های داده استفاده می شود برای مثال می توان یک خط افقی را بین داده ها مشخص کرد.

مثال ۱ :

می خواهیم از جدول Titles دیتابیس Pubs ستون Title را نمایش دهیم :

پروژه ای جدید را ایجاد کنید و سپس یک کنترل Repeater روی فرم قرار دهید. روی آن کلیک راست نمایید و گزینه ی View HTML Source را انتخاب نمایید و یا اینکه در پایین صفحه در کنار گزینه ی Design حالت HTML را انتخاب کنید تا بتوان تغییرات لازم را ایجاد نمود. ابتدا HeaderTemplate را اضافه می کنیم که با استفاده از تگ های استاندارد HTML ایجاد می شوند.

```
<asp:Repeater id="Repeater1" runat="server">
  <HeaderTemplate>
    <table border="1">
      <TR>
        <TD><B>Title table</B></TD>
      </TR>
    </HeaderTemplate>
```

سپس AlternatingTemplate را ایجاد می نماییم تا جدول Title را بتواند نمایش دهد :

```
<AlternatingItemTemplate>
  <tr>
    <td style="background-color: lightblue">
      <b>
        <%# DataBinder.Eval(Container.DataItem,"title") %>
      </b>
    </td>
  </tr>
</AlternatingItemTemplate>
```



نکته :

اگر فقط می خواستیم یک آرایه ساده را نمایش دهیم کافی بود که از `DataRow` بجای عبارت `DataRow` استفاده نماییم.

در ادامه قسمت اجباری کار یعنی ایجاد `ItemTemplate` را که مخلوطی از تگ های `HTML` و `XML` است را ایجاد می نماییم.

```
<ItemTemplate>
<tr>
<td style="background-color: lightgray">
<%# DataBinder.Eval(Container.DataItem,"title") %>
</td>
</tr>
</ItemTemplate>
```

و در پایین با ارائه `FooterTemplate` کار طراحی را خاتمه می دهیم :

```
<FooterTemplate>
</table>
</FooterTemplate>
</asp:Repeater>
```

اکنون به حالت `Design` برگردید و روی صفحه دوبار کلیک نموده و کد زیر را برای نمایش وارد نمایید ، همانطور که ملاحظه می نمایید باید `DataMember` این کنترل را هم برای نمایش جدول مشخص کرد:

```
private void Page_Load(object sender, EventArgs e)
{
    SqlConnection cnn = new
        SqlConnection("server=(local);uid=sa;pwd=;database=pubs" );

    SqlDataAdapter dtAdap = new
        SqlDataAdapter("select title from titles",cnn);

    DataSet datasetTitles = new DataSet();
    dtAdap.Fill(datasetTitles, "titles");

    Repeater1.DataMember ="titles";
    Repeater1.DataSource = datasetTitles.Tables["titles"].DefaultView;
    Repeater1.DataBind();
}
```

نمایش لیست های پیچیده :

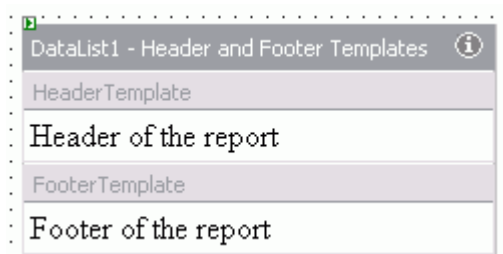
می توان با استفاده از کنترل DataList برای نمایش و فرمت لیست های پیچیده استفاده کرد. بسیاری از ویژگی های آن با کنترل repeater یکسان است بعلاوه یک سری گزینه های فرمت کردن Template های اضافی و تعیین جهت نمایش داده ها. برای اتصال پایگاه داده به آن از خاصیت DataSource و سپس فراخوانی متد DataBind استفاده می شود. همانطور که ذکر شد این کنترل Templates زیر را پشتیبانی می کند :

ItemTemplate, AlternatingTemplate, HeaderTemplate, FooterTemplate, SeparatorTemplate

بعلاوه از SelectedItemTemplate برای هنگامیکه کاربر آیتمی را انتخاب و EditItemTemplate هنگامیکه کاربر آیتمی را ادیت می کند، بکار می روند.

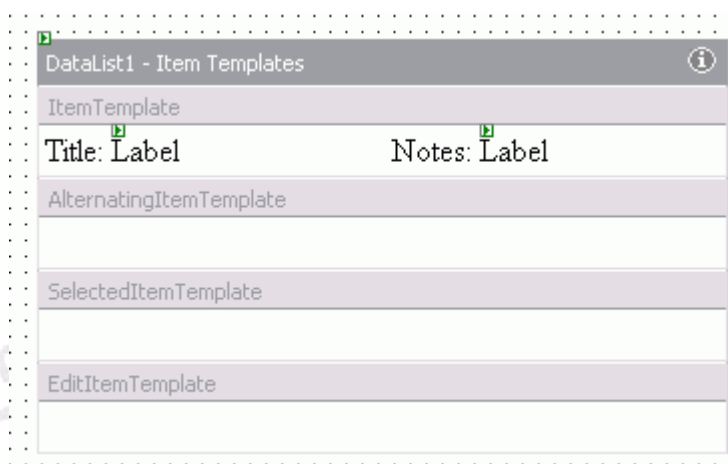
مثال ۲ :

یک دیتالست را روی صفحه قرار دهید و سپس روی آن کلیک راست نمایید. از منوی ظاهر شده گزینه ی Edit Templates و سپس Header and footer templates را انتخاب نمایید. در این حالت Templates کنترل قابل ادیت کردن می شود و می توان متنی را به آن اضافه کرد و یا کنترل های دلخواهی را. در قسمت سفید رنگی که زیر Header و یا Footer قرار دارد می توان این عملیات را انجام داد. برای مثال آنرا به شکل زیر تغییر دهید :



شکل ۱- ادیت کردن Header and footer templates کنترل دیتا لیست.

دوباره روی کنترل کلیک راست نموده و گزینه ی خاتمه ی ویرایش Template را انتخاب نمایید. این بار می خواهیم ItemTemplates را ویرایش نماییم. بدین منظور عملیات ذکر شده مربوط به کلیک راست روی کنترل و انتخاب گزینه ی Edit Templates و سپس انتخاب Item Templates را انجام دهید تا بتوان این قسمت را ویرایش کرد (شکل ۲).



شکل ۲- ادیت کردن ItemTemplates کنترل دیتا لیست .

دو کنترل Label روی یک قسمت خالی فرم قرار دهید و سپس آنها را به قسمت سفید رنگ Item Templates کات و Drag نمایید (شکل ۲). در اینجا می توان مخلوطی از متن کنترل ها را ارائه داد. نام این لیبل ها را به lblTitles و lblNotes تغییر دهید.

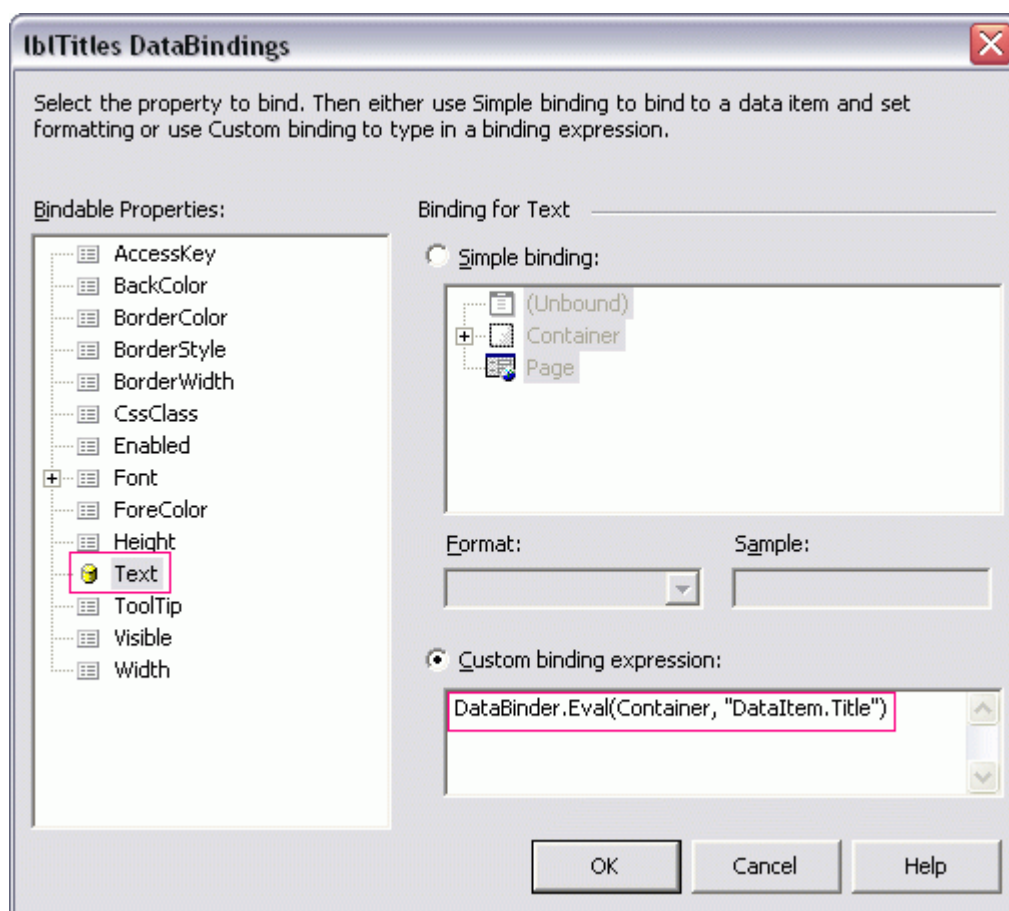
برای بایند کردن داده ها به این لیبل ها باید در برگه ی خواص هر کدام خاصیت DataBindings را انتخاب نمود و سپس در صفحه ی ظاهر شده در قسمت Binable properties خاصیت Text و از پنل سمت راست قسمت Custom binding expression را انتخاب نمود و سپس از عبارت DataBinder.Eval بهره جست (شکل ۳).

برای لیبل که جهت نمایش دادن عنوان ها قرار داده ایم عبارت زیر را بنویسید :

```
DataBinder.Eval(Container, "DataItem.Title")
```

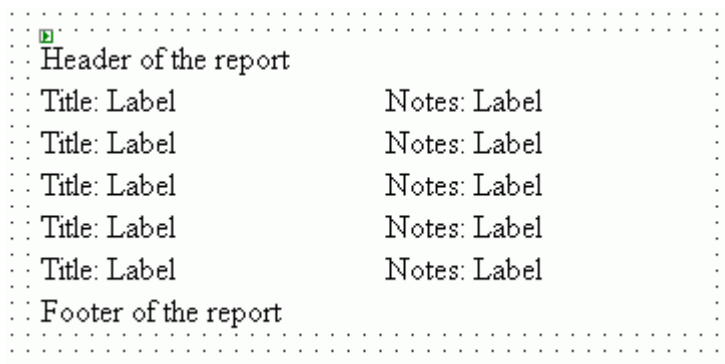
و برای لیبل مربوط به Notes عبارت زیر را بنویسید :

```
DataBinder.Eval(Container, "DataItem.Notes")
```



شکل ۳- مشخص کردن نحوه ی اتصال لیبل ها در کنترل دیتا لیست.

اکنون با استفاده از کلیک راست روی صفحه می توان به ادیت آن خاتمه بخشید. شکل نهایی در شکل زیر نمایش داده شده است.



شکل ۴- شکل نهایی دیتا لیست پس از خاتمه طراحی آن .



اکنون نوبت کد نویسی این مجموعه است. روی فرم برنامه دوبار کلیک نموده و کد زیر را داخل آن بنویسید :

```
private void Page_Load(object sender, System.EventArgs e)
{
    SqlConnection cnn = new
        SqlConnection("server=(local);uid=sa;pwd=;database=pubs" );

    SqlDataAdapter dtAdap = new
        SqlDataAdapter("select title,notes from titles",cnn);

    DataSet datasetTitles = new DataSet();
    dtAdap.Fill(datasetTitles, "titles");

    DataList1.DataSource = datasetTitles.Tables["titles"].DefaultView;
    DataList1.DataBind();
}
```

استفاده از DropDownList :

از آن برای نمایش یک ستون از دیتابیس با اعمال فیلترهای مختلف می توان استفاده کرد .

مثال ۳ :

می خواهیم ستون Title را در جدول Titles دیتابیس Pubs را در یک DropDownList نمایش دهیم و پس از انتخاب هر آیتم توسط کاربر نام آن در یک Label نمایش داده شود.

پروژه ای جدید را آغاز نموده و سپس یک DropDownList و یک Label روی فرم قرار دهید . از کد زیر برای بایند کردن اطلاعات می توان استفاده کرد.



```
private void Page_Load(object sender, System.EventArgs e)
{
    if (!Page.IsPostBack) // run 1 time
    {
        SqlConnection cnn = new
        SqlConnection("server=(local);uid=sa;pwd=;database=pubs" );

        SqlDataAdapter dtAdap = new
        SqlDataAdapter("select title from titles",cnn);

        DataSet datasetTitles = new DataSet();
        dtAdap.Fill(datasetTitles, "titles");

        DropDownList1.DataTextField = "title";
        DropDownList1.DataSource = datasetTitles.Tables["titles"].DefaultView;
        DropDownList1.DataBind();
    }
}
```

همانطور که ملاحظه می فرمایید خاصیت DataTextField این کنترل برای مشخص کردن فیلدی که قرار است نمایش داده شود بکار برده می شود.

برای اینکه با انتخاب یک آیتم توسط کاربر لیبل آنرا نمایش دهد ابتدا باید خاصیت AutoPostBack کنترل DropDownList را true کنیم. سپس در برگه ی خواص کنترل در قسمت مربوط به رخدادهای آن در قسمت SelectedIndexChanged دوبار کلیک نمایید تا روال رخداد آن ایجاد شود. سپس به سادگی از کد زیر برای این قسمت می توان استفاده کرد.

```
private void DropDownList1_SelectedIndexChanged(object sender, System.EventArgs e)
{
    Label1.Text = DropDownList1.SelectedItem.Text;
}
```

اگر آیتم زیر را به Page_Load اضافه نمی کردیم با هر بار PostBack شدن صفحه ، یکبار کل کنترل دوباره بایند می شد.

```
if (!Page.IsPostBack) // run 1 time
```



چند کنترل دیگر هم مانند TreeView و ToolBar و TabControl و MultiPageControl در سایت مایکروسافت یافت می شوند که از آنها نیز می توان استفاده کرد :

http://msdn.microsoft.com/downloads/samples/Internet/ASP_DOT_NET_ServerControls/WebControls/

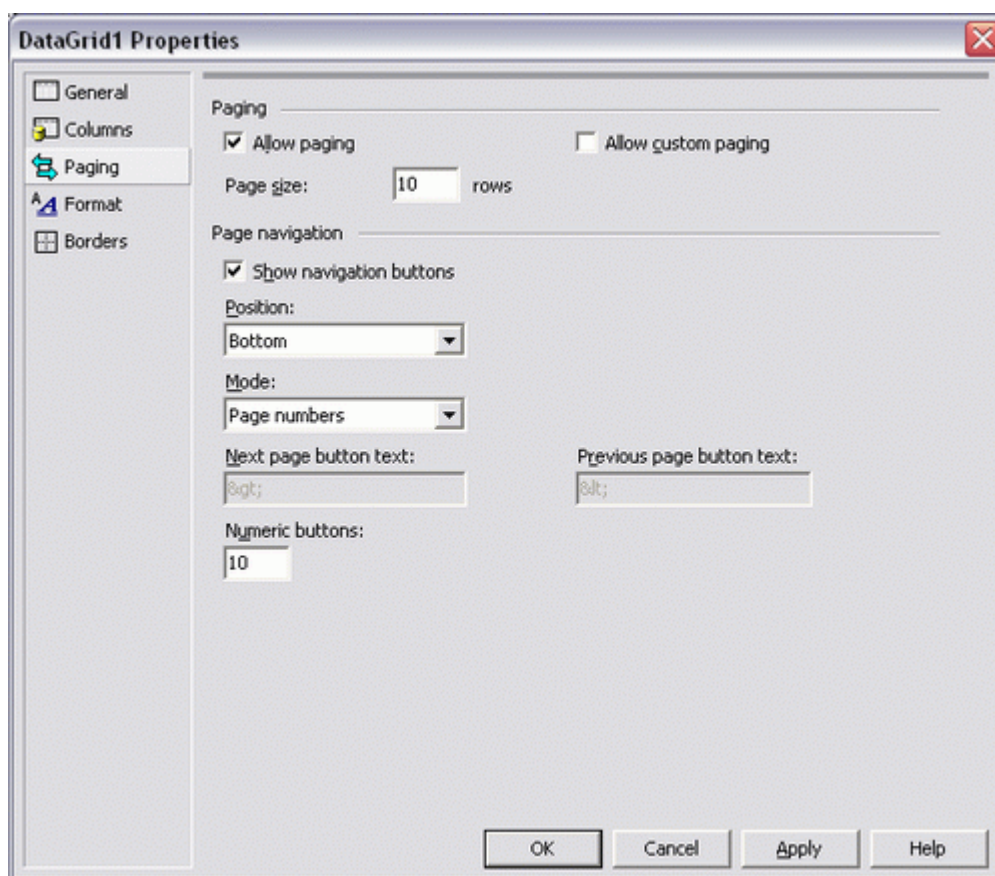
استفاده از توانایی های دیتاگرید :

در ادامه ی فصل به توانایی های کنترلی که بیش از همه در صفحات aspx کاربرد دارد با ذکر مثالهایی کاربردی ، می پردازیم :

مثال ۴:

یکی از قابلیت های مهم دیتاگرید ، Paging می باشد. اگر به کنترل هایی که تابحال معرفی شدند دقت کرده باشید ، کل اطلاعات را حتی اگر ۱۰۰۰ رکورد هم باشند در یک صفحه ارائه می دهند که اینکار اصلا ظاهر خوشایندی ندارد. برای رفع این مشکل قابلیت ذکر شده به دیتا گرید اضافه گردیده است که از آن در طی یک مثال استفاده خواهیم کرد :

پروژه ای جدید را آغاز کرده و یک دیتا گرید روی صفحه قرار دهید. سپس روی آن کلیک راست کرده و Property builder را انتخاب نمایید. سپس قسمت Paging آنرا مطابق شکل زیر تنظیم نمایید (مطابق این تنظیمات در هر بار نمایش دیتاگرید فقط ۱۰ ردیف نشان داده خواهد شد و بقیه ردیف ها به صفحات بعدی منتقل خواهند شد) سپس Apply و Ok را فراموش نکنید:



شکل ۵- بوجود آوردن امکان Paging در کنترل دیتا گرید.

روی صفحه دوبار کلیک نموده و کد زیر را در آن وارد کنید تا بتوان تعدادی رکورد را به دیتاگرید بایند کرد.

```
private void Page_Load(object sender, System.EventArgs e)
{
    if(!Page.IsPostBack)
        bindToDatGrid();
}
void bindToDatGrid()
{
    SqlConnection cnn = new
    SqlConnection("server=(local);uid=sa;pwd=;database=pubs" );
    SqlDataAdapter dtAdap = new
    SqlDataAdapter("select * from sales",cnn);
    DataSet dsetSales = new DataSet();
    dtAdap.Fill(dsetSales, "sales");
    DataGrid1.DataSource = dsetSales.Tables["sales"].DefaultView;
    DataGrid1.DataBind();
}
```



اگر برنامه را در این حالت اجرا نمایید فقط ۱۰ رکورد در صفحه ی اول نمایش داده می شوند و بقیه صفحات کار نمی کنند. برای اینکه صفحات دیگر را هم بکار بیاندازیم باید برای آنها کد نوشت. پنجره ی خواص دیتاگرید را انتخاب کرده و سپس روی آیکون رعد و برق کلیک نمایید تا رخدادهای دیتاگرید ظاهر شوند. سپس **PageIndexChanged** را انتخاب نموده و دوبار روی تکست باکس روبروی آن کلیک نمایید. به صورت خودکار روال رخداد آن به برنامه اضافه می شود. با استفاده از کد زیر مبحث Paging تکمیل می شود.

```
private void DataGrid1_PageIndexChanged(object source,
    System.Web.UI.WebControls.DataGridPageChangedEventArgs e)
{
    DataGrid1.CurrentPageIndex = e.NewPageIndex ;
    bindToDatGrid();
}
```

هر بار که روی شماره ی یک صفحه ی جدید کلیک می شود ، روال رخداد فوق فراخوانی شده و عملیات بایند کردن با توجه به شماره ی صفحه ی جدید انجام می گیرد.

مثال ۵:

در این مثال می خواهیم Sort کردن یک فیلد را با استفاده از ابزار موجود در دیتا گرید انجام دهیم.

پروژه ای جدید را ایجاد نمایید و سپس یک دیتاگرید روی فرم قرار دهید. از کد زیر برای بایند کردن اطلاعات به این گرید استفاده می کنیم.

```
private void Page_Load(object sender, System.EventArgs e)
{
    if(!Page.IsPostBack)
        bindToDatGrid("qty");
}

void bindToDatGrid(string sort_field)
{
    SqlConnection cnn = new
        SqlConnection("server=(local);uid=sa;pwd=;database=pubs" );
```



```
SqlDataAdapter dtAdap = new  
    SqlDataAdapter("select * from sales",cnn);  
  
DataSet dsetSales = new DataSet();  
dtAdap.Fill(dsetSales, "sales");  
  
DataView dataviewSales =  
    dsetSales.Tables["sales"].DefaultView;  
dataviewSales.Sort = sort_field;  
  
DataGrid1.DataSource = dataviewSales;  
DataGrid1.DataBind();  
}
```

برای اینکه بتوان با کلیک کردن روی Header مربوط به هر ستون ، آن ستون مرتب شود باید اعمال زیر را انجام داد :

روی دیتاگرید کلیک راست کنید و سپس Property builder را انتخاب نمایید. در صفحه General آن تیک مربوط Allow sorting آنرا بگذارید. بدین صورت با کلیک کردن روی هدر هر ستون روال رخداده مرتب کردن ستون صدا زده خواهد شد.

برای تکمیل کار به برگه خواص دیتاگرید مراجعه نموده و قسمت رخدادهای آنرا انتخاب نمایید. اکنون روی SortCommand دوبار کلیک نمایید تا روال رخداد آن به صورت اتوماتیک ایجاد شود. سپس از کد زیر می توان استفاده نمود :

```
private void DataGrid1_SortCommand(object source,  
    System.Web.UI.WebControls.DataGridSortCommandEventArgs e)  
{  
    string SortField = (string)e.SortExpression;  
    bindToDatGrid(SortField);  
}
```

با هر بار کلیک کردن کاربر روی نام یک ستون ، نام آن ستون دریافت شده و داده ها بر حسب آن مرتب می شوند.

مثال ۶:

در این مثال می خواهیم داده های نمایش داده شده در یک گرید را ویرایش کنیم. یکی از قابلیت های جالب آن اضافه کردن لینک های Edit و Update به ازای هر ردیف می باشد که با کلیک کردن بر روی Edit تمام سلولهای یک ردیف به شکل تکست باکس قابل ویرایش در آمده و پس از اتمام ویرایش با کلیک بر روی Update و البته کد نویسی برای آن ، داده ها در دیتابیس ذخیره خواهند شد .

پروژه ای جدید را ایجاد کرده و سپس یک دیتاگرید روی فرم قرار دهید. سپس روی آن کلیک راست کرده و Property builder را انتخاب نمایید. در آن در قسمت Columns ، ستون های Edit, Update, Cancel را اضافه کنید (شکل ۶) و سپس Apply و Ok فراموش نشود.

در این مثال از دیتابیس Northwind که همراه SQL-Server ارائه می شود به صورت زیر برای بایند کردن اطلاعات استفاده می کنیم.

```
private void Page_Load(object sender, EventArgs e)
{
    if(!Page.IsPostBack)
        bindToDatGrid();
}

void bindToDatGrid()
{
    SqlConnection cnn = new
    SqlConnection("DATABASE=Northwind;SERVER=localhost;UID=sa;PWD=;" );

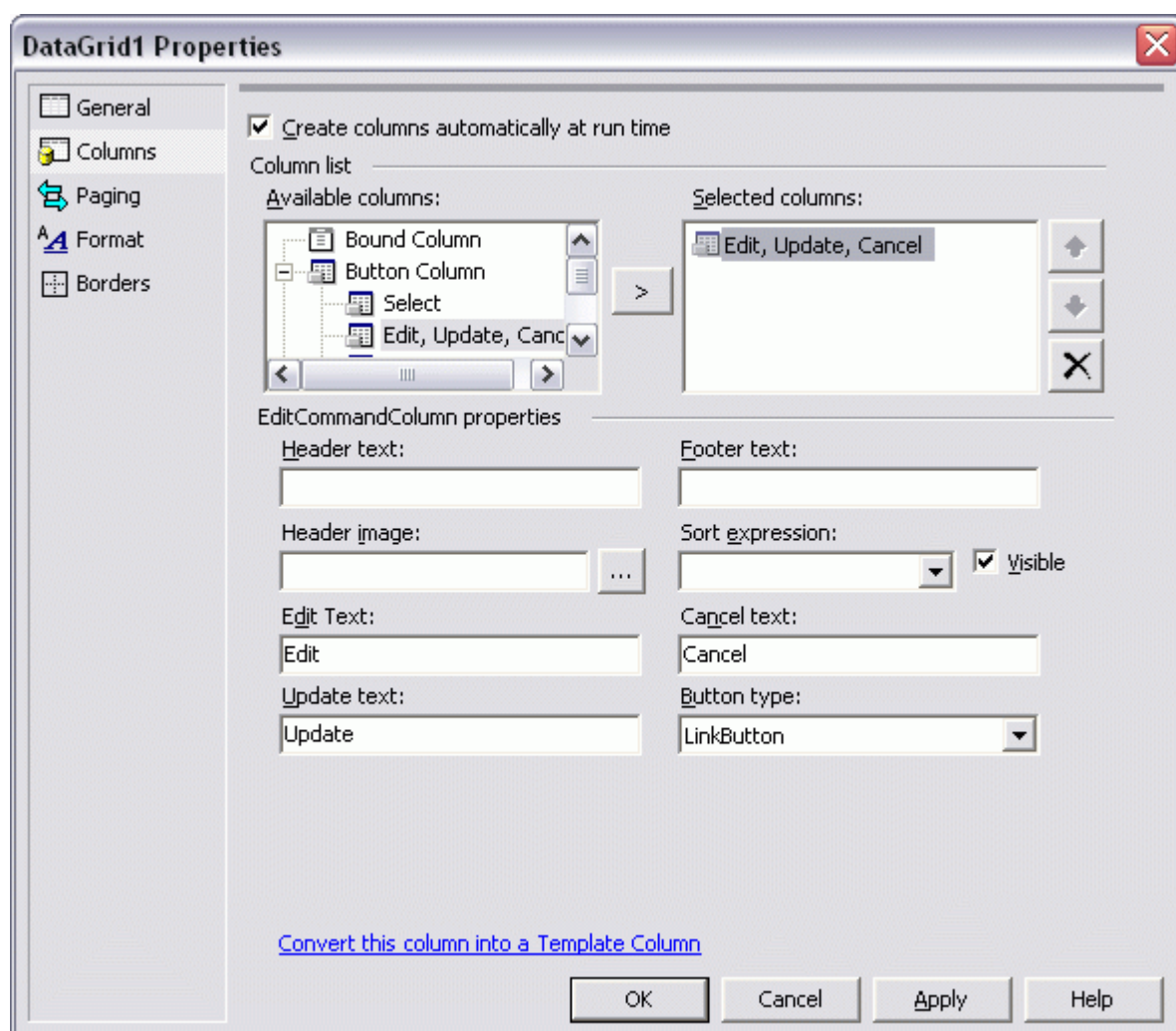
    SqlDataAdapter dtAdap = new
    SqlDataAdapter("SELECT employeeid, titleofcourtesy, "+
    " firstname, lastname, title, country FROM Employees",cnn);

    DataSet dsetSales = new DataSet();
    dtAdap.Fill(dsetSales, "sales");

    DataGrid1.DataSource = dsetSales.Tables["sales"].DefaultView;
    DataGrid1.DataBind();
}
```

برای اینکه وقتی کاربر روی لینک edit کلیک می کند سلولها به حالت ویرایش درآیند ، در برگه ی خواص دیتاگرید در قسمت رخدادهای آن ، روی EditCommand دوبار کلیک کنید تا روال رخداد آن به صورت خودکار اضافه شود ، سپس از کد زیر برای آن استفاده می کنیم:

```
private void DataGrid1_EditCommand(object source,
    System.Web.UI.WebControls.DataGridCommandEventArgs e)
{
    // Set the current item to edit mode
    DataGrid1.EditItemIndex = e.Item.ItemIndex;
    bindToDatGrid();
}
```



شکل ۶- اضافه کردن لینک های Edit, Update, Cancel به هر ردیف دیتاگرید.

از رخدادهای CancelCommand هم برای بازگرداندن به حالت عادی استفاده می شود که نحوه ی اضافه کردن این روال رخداد نیز مانند بقیه است :



```
private void DataGrid1_CancelCommand(object source,
                                     System.Web.UI.WebControls.DataGridCommandEventArgs e)
{
    // Reset the edit mode for the current item
    DataGrid1.EditItemIndex = -1;
    bindToDatGrid();
}
```

قسمت اساسی این مثال از این لحظه آغاز می شود! از کد زیر برای هنگامی استفاده می کنیم که کاربر روی لینک Update کلیک نموده است و باید داده های تغییر یافته در دیتابیس ذخیره شوند. برای اینکار ابتدا باید روال رخداد UpdateCommand را به برنامه اضافه نمود و سپس از کد زیر استفاده کرد .

تکست باکس ها در گرید در حالت ویرایش اندیس اشان از صفر شروع می شود.

```
private void DataGrid1_UpdateCommand(object source,
                                     System.Web.UI.WebControls.DataGridCommandEventArgs e)
{
    string employeeid = ((TextBox)e.Item.Cells[1].Controls[0]).Text;
    string titleofcourtesy = ((TextBox)e.Item.Cells[2].Controls[0]).Text;
    string firstname = ((TextBox)e.Item.Cells[3].Controls[0]).Text;
    string lastname = ((TextBox)e.Item.Cells[4].Controls[0]).Text;
    string title = ((TextBox)e.Item.Cells[5].Controls[0]).Text;
    string country = ((TextBox)e.Item.Cells[6].Controls[0]).Text;

    SqlConnection conn = new
        SqlConnection("DATABASE=Northwind;SERVER=localhost;UID=sa;PWD=;");

    // Prepare the command text
    String strCmd = "UPDATE Employees SET title='" +
        title + "'," + "country='" + country + "'" +
        " WHERE employeeid='" + employeeid + "'";

    SqlCommand cmd = new SqlCommand(strCmd, conn);

    conn.Open();
    cmd.ExecuteNonQuery();
    conn.Close();

    // Reset the edit mode for the current item
    DataGrid1.EditItemIndex = -1;

    // create data source again.....
    bindToDatGrid();
}
```

مثال ۷:

در گزارشات عددی عموماً اضافه کردن جمع به آخر گزارش نیاز می باشد. در این مثال قصد داریم این مورد را با استفاده از امکانات دیتاگرید مورد بررسی قرار دهیم.

پروژه ای جدید را ایجاد و یک دیتاگرید روی فرم قرار دهید. بر روی آن کلیک راست کرده و از منوی ظاهر شده Property builder را انتخاب نموده و سپس در برگه ی General مربوطه ، تیک مربوط به Show footer را قرار دهید. سپس Apply و بعد هم Ok فراموش نشود!

از کد زیر برای بایند کردن یک سری از اطلاعات به گرید استفاده خواهیم کرد:

```
private void Page_Load(object sender, System.EventArgs e)
{
    if(!Page.IsPostBack)
        bindToDatGrid();
}

void bindToDatGrid()
{
    SqlConnection cnn = new
    SqlConnection("DATABASE=Northwind;SERVER=localhost;UID=sa;PWD=;" );

    SqlDataAdapter dtAdap = new
    SqlDataAdapter("SELECT * FROM [Order Details]",cnn);

    DataSet dsetSales = new DataSet();
    dtAdap.Fill(dsetSales, "[Order Details]");

    DataGrid1.DataSource = dsetSales.Tables["[Order Details]"].DefaultView;
    DataGrid1.DataBind();
}
```

در جدول Order Details دیتابیس Northwind فیلدهای OrderID, ProductID, UnitPrice, Quantity, Discount وجود دارند. می خواهیم جمع کل دوفیلد آخر را در Footer دیتاگرید نمایش دهیم.



هنگامیکه هر ردیف به دیتاگرید اضافه می شود رخداد ItemDataBound آن به ازای هر ردیف ، فراخوانی می شود. از این مورد برای خواندن اطلاعات و جمع آن با ردیف های قبلی به سادگی می توان استفاده کرد که نحوه ی انجام آن در عمل در مثال زیر آورده شده است:

روال رخداد ItemDataBound دیتاگرید را با استفاده از برگه ی خواص و رخدادهای آن ، به صورت خودکار به برنامه اضافه نماییم .

ابتدا یک متغیر عمومی برای نگهداری جمع هر ستون اضافه می کنیم :

```
// calc total  
private double[] runningTotal = new double[16] ;
```

سپس از تابع زیر برای جمع زدن استفاده خواهیم کرد :

```
private void CalcTotal(string _price, int i)  
{  
    try  
    {  
        runningTotal[i] += Double.Parse(_price);  
    }  
    catch  
    {  
        //A value was null  
    }  
}
```

و در روال رخداد زیر کار نهایی انجام خواهد شد :

```
private void DataGrid1_ItemDataBound(object sender,  
    System.Web.UI.WebControls.DataGridItemEventArgs e)  
{  
    //is called as each row in the data source is bound to the DataGrid  
    if ( e.Item.ItemType == ListItemType.Item  
        || e.Item.ItemType == ListItemType.AlternatingItem)  
    {  
        CalcTotal( e.Item.Cells[3].Text ,3);  
        CalcTotal( e.Item.Cells[4].Text ,4);  
    }  
    else if(e.Item.ItemType == ListItemType.Footer )  
    {  
        e.Item.Cells[2].Text="Total: ";  
        e.Item.Cells[3].Text = runningTotal[3].ToString();  
        e.Item.Cells[4].Text = runningTotal[4].ToString();  
    }  
}
```



تمرین :

- ۱- می توان به هر ردیف دیتا گرید ستون لینک Delete را هم اضافه کرد. با توجه به اینکه این قسمت بسیار شبیه مثال ۶ فصل جاری می باشد و در هر حقیقت نمونه ی ساده شده ی آن است ، آنرا به مثال ذکر شده اضافه نمایید (به رخداده DeleteCommand نیاز خواهید داشت).
- ۲- در مثال ۷ فصل جاری ، یک رویه ذخیره شده برای محاسبه حاصلضرب تعداد در واحد قیمت (قیمت کل) و همچنین نمایش دادن بقیه فیلدها ایجاد نمایید و سپس در یک دیتا گرید از آن برای نمایش اطلاعات جدول Order Details به همراه مقدار محاسبه شده در رویه ذخیره شده با ته جمع فیلدهای تعداد ، تخفیف و قیمت کل ارائه دهید ، بدیهی است که در مثال ایجاد شده خاصیت paging هم باید لحاظ شده باشد.