فصل دهم :

آشنایی با بانک های اطلاعاتی بر پایه ی XML و بکارگیری آنها در ASP.NET

مقدمه :

XML مخفف eXtensible Markup Language می باشد. این زبان برمبنـای اسـتاندارد ISO 8879 بنیانگذاری شده است و در سال ۱۹۹۶ توسط مجمع W3C ایجاد گشت.

XML راهی است عالی برای انتقـال داده هـا از سـایتی بـه سـایت دیگـر و یکـی از اجـزای متـشکله ی سیستم های B2B می باشد. مایکروسافت روی تکنولوژی XML به شدت سرمایه گذاری کرده و دات نت فریم ورک به همین منظور برای کار با XML از کلاس های بسیار متعددی تشکیل شده است.

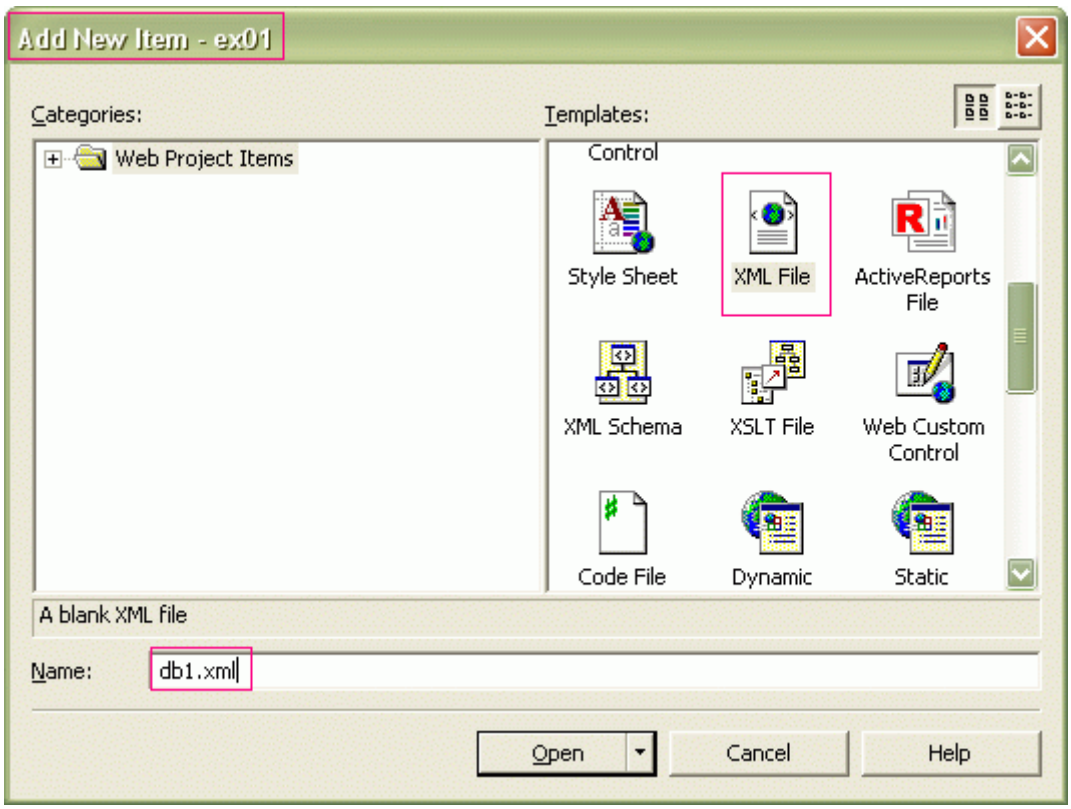
هدف اصلی ما در این فصل خواندن اطلاعات از یک فایل XML و نمایش آن در ASP.NET و همچنین تبدیل دیتابیس های SQL-Server به فایل XML می باشد.

خواندن اطلاعات از بانک های اطلاعاتی XML و نمایش آنها

مفهوم دیتاست که توسط مایکروسافت ارائه شده است ، داده ها را از منبع آن جدا مـی سـازد. بنـابراین هنگام کار با دیتاست ها آنچنان فرقی نمی کند که این داده درون دیتابیس SQL-Server ذخیره شده باشد و یا درون یک فایل XML ، زیرا ذات دیتاست ها از منبع داده ی آن مستقل است.

مثال ۱ :

در این مثال قصد داریم داده هایی را از یک فایل XML بخوانیم.
یک پروژه ی جدید ASP.NET را باز نموده و سپس از منوی پروژه گزینه ی Add new item را برگزیده و یک فایل XML با نام db1.xml به پروژه اضافه نمایید (شکل ۱).



شکل ۱- اضافه کردن یک فایل XML جدید به پروژه.

محتویات این فایل را به صورت زیر ایجاد نمایید (سعی کنید حتما یکبار آنرا تایپ نمایید تا توانایی های ویژوال استودیو و هوشمندی آنرا در این زمینه ملاحظه نمایید):

Start Page | WebForm1.aspx | **db1.xml** | WebForm1.aspx.cs

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
<schema id="DocumentElement"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xdo="urn:schemas-microsoft-com:xml-xdo"
xdo:DataSetName="DocumentElement">
<element name="Products" >
  <complexType>
   <all>
     <element name="ProductID" type="int"></element>
     <element name="CategoryID" minOccurs="0" type="int" ></element>
     <element name="ProductName" type="string"></element>
   </all>
  </complexType>
</element>
</schema>
<DocumentElement>
<Products>
  <ProductID>1001</ProductID>
  <CategoryID>1</CategoryID>
  <ProductName>Chocolate city milk</ProductName>
</Products>
<Products>
  <ProductID>3004</ProductID>
  <CategoryID>3</CategoryID>
  <ProductName>Fizzy Fizzing Drink</ProductName>
</Products>
</DocumentElement>
</root>
```

توضیحاتی در مورد محتوای فایل فوق:

یک فایل XML تنها از یک سری تگ تشکیل شده است. با یک تگ به نام ’root‘ آغاز و سپس با تگ دیگری به نام ’/root‘ خاتمه می یابد. XML به قوانین تگ ها به شدت حساس است. بنابراین باید یک سری اصول اولیه را هنگام کار با آنها رعایت نمود.

در مثال فوق تگ <element> دارای ویژگی به نام ’name‘ می باشد که با نام جدول دیتابیس مقدار دهی شده است. در این حالت نام آن Products است.

در بین تگ های <all> و </all> که مشاهده می نمایید ، باردیگر از تگ های دیگری استفاده گردیده است ، اما اینبار نام فیلدها و نوع های داده ای آنها تعریف شده است و این موارد با ویژگی های name و

type در تگ های مثال فوق مشخص گردیده اند. تمام این موارد یک schema را تعریف می کنند بنابراین با یک </schema> خاتمه می یابند.

تگ schema دارای ویژگی به نام DataSetName است که تگی به نام DocumantElement را مشخص می نماید، جایی که تمام داده های جدول قرارگرفته اند. هر رکورد بین تگ های products قرار می گیرد و سپس مقادیر هر فیلد در یک رکورد ، بین تگ هایی با نام فیلدها محصور می شوند. بنابراین نمایش و استفاده از داده هایی اینچنین خوش فرم در دیتاگرید ساده می باشد.

ابتدا فضای نام زیر به برنامه ملحق می شود:

```
using System.IO;
```

با استفاده از کد زیر در فرمی که روی آن یک Label و یک DataGrid قرار گرفته است می توان داده های فوق را نمایش داد:

```
private void Page_Load(object sender, System.EventArgs e)
{
      if(!Page.IsPostBack)
      {
            DataSet ds = new DataSet();
            FileStream fs = new FileStream(
                  Server.MapPath("db1.xml"),
                  FileMode.Open,
                  FileAccess.Read);
            StreamReader r = new StreamReader(fs);
            ds.ReadXml(r);
            fs.Close();
            DataView Sv=  new DataView(ds.Tables[0]);
            Label1.Text = Sv.Table.TableName;
            DataGrid1.DataSource = Sv;
            DataGrid1.DataBind();
      }
}
```

در کد فوق توسط شیء FileStream که fs نام دارد فایل XML را خوانده و نهایتا به DataGrid بایند می شود (شکل ۳).

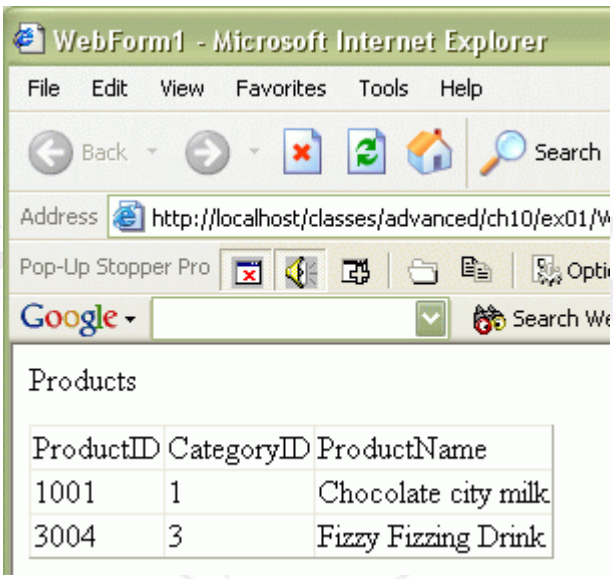
هنگام کار با شیء FileStream سه آرگومان آن به صورت زیر هستند:

- نام فایل XML
- حالت باز کردن فایل
- نحوه ی دسترسی به فایل

تابع MapPath بکارگرفته شده ، مسیر را روی سرور بر می گرداند. بنابراین نگرانی در مورد محل دقیق فایل روی سرور وجود نخواهد داشت.

سپس یگ شیء StreamReader تعریف شده و شیء FileStream بعنوان آرگومان ورودی به آن فرستاده شده است.

در ادامه تابع ReadXml مربوط به شیء دیتاست ، این استریم ورودی را می خواند. در خاتمه فایل بسته خواهد شد و یک DataView برای بایند کردن به دیتاگرید خلق می شود.



شکل ۳- نمایی از اجرای مثال یک.

مثال ۲ :

این مثال با مثال قبلی تفاوت آنچنانی ندارد و تنها در آن schema و داده ها در فایلهای مجزای xml ذخیره شده اند. بنابراین با استفاده از تابع ReadXmlSchema می توان برای خواندن آن اقدام کرد. انتخاب میان این دو حالت مطرح شده در مثالهای یک و ۲ تنها به سلیقه ی شما بستگی پیدا می کند و از لحاظ کارآیی تفاوتی ندارند.

یک پروژه ی جدید ASP.NET را آغاز نمایید و سپس دو فایل xml را از منوی پروژه به آن اضافه کنید (به نامهای s.xml و d.xml).

محتویات این دو فایل به صورت زیر است:

Start Page | WebForm1.aspx | **s.xml** | d.xml | WebForm1.aspx.cs

```
<?xml version="1.0" encoding="utf-8" ?>
<schema id="DocumentElement"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xdo="urn:schemas-microsoft-com:xml-xdo"
xdo:DataSetName="DocumentElement">
<element name="Products" >
  <complexType>
   <all>
     <element name="ProductID" type="int"></element>
     <element name="CategoryID" minOccurs="0" type="int" ></element>
     <element name="ProductName" type="string"></element>
   </all>
  </complexType>
</element>
</schema>
```

Start Page | WebForm1.aspx | s.xml | **d.xml** | WebForm1.aspx.cs

```
<?xml version="1.0" encoding="utf-8" ?>
<DocumentElement>
<Products>
  <ProductID>1001</ProductID>
  <CategoryID>1</CategoryID>
  <ProductName>Chocolate city milk</ProductName>
</Products>
<Products>
  <ProductID>3004</ProductID>
  <CategoryID>3</CategoryID>
  <ProductName>Fizzy Fizzing Drink</ProductName>
</Products>
</DocumentElement>
```

و در آخر از کد زیر برای نمایش داده ها استفاده می نماییم:

```
private void Page_Load(object sender, System.EventArgs e)
{
      if(!Page.IsPostBack)
      {
            DataSet ds = new DataSet();

            FileStream fs = new FileStream(
                  Server.MapPath("s.xml"),
                  FileMode.Open,
                  FileAccess.Read);
            StreamReader sc = new StreamReader(fs);
            ds.ReadXmlSchema(sc);
            fs.Close();

            fs = new FileStream(
                  Server.MapPath("d.xml"),
                  FileMode.Open,
                  FileAccess.Read);
            StreamReader r = new StreamReader(fs);
            ds.ReadXml(r);
            fs.Close();

            DataView Sv=  new DataView(ds.Tables[0]);
            Label1.Text = Sv.Table.TableName;
            DataGrid1.DataSource = Sv;
            DataGrid1.DataBind();
      }

}
```

تبدیل بانک های اطلاعاتی SQL-Server به فایل XML :

در این قسمت برخلاف مثالهای پیشین ، داده ها را از دیتابیس SQL-Server خوانده و سپس آنها را به یک فایل با فرمت XML تبدیل می نماییم. تابع WriteXml مربوط به شیء دیتاست پر شده با داده ها به راحتی آنها را به یک خروجی XML تبدیل می نماید.

مثال ۳ :

ابتدا دو فضای نام زیر را به برنامه اضافه نمایید:

```
using System.Data.SqlClient ;
using System.IO;
```

سپس از کد زیر برای نمایش داده ها در یک دیتاگرید و ذخیره کردن آنها در یک فایل xml کمک می گیریم:

```
private void Page_Load(object sender, System.EventArgs e)
{
      if(!Page.IsPostBack)
      {
            SqlConnection c =
new SqlConnection("server=localhost;uid=sa;pwd=;database=pubs");
SqlDataAdapter co = new SqlDataAdapter("select * from authors",c);
            DataSet ds = new DataSet();
            co.Fill(ds,"table");
            DataView dv = new DataView(ds.Tables[0]);
            DataGrid1.DataSource = dv;
            DataGrid1.DataBind();

            FileStream myFs =
             new FileStream( Server.MapPath("myXmlData.xml"),
                                    FileMode.OpenOrCreate,
                                    FileAccess.Write);


            ds.WriteXml( myFs );
            myFs.Close();
      }

}
```

بعد از اجرای برنامه ، یک رکورد از محتویات فایل پس از ایجاد به صورت زیر می باشد:

```
<NewDataSet>
<table>
<au_id>172-32-1176</au_id>
<au_lname>White</au_lname>
<au_fname>Johnson</au_fname>
<phone>408 496-7223</phone>
<address>10932 Bigge Rd.</address>
<city>Menlo Park</city>
<state>CA</state>
<zip>94025</zip>
<contract>true</contract>
<table/>
<NewDataSet/>
```

تمرین :

۱- فایل ایجاد شده در مثال ۳ را در یک برنامه ی جدید ASP.NET خوانده و در یک دیتاگرید نمـایش دهید.

۲- برای تهیه ی Backup از یک دیتابیس SQL-Server دلخواه ، صفحه ای را ایجاد نمایید کـه تمـام پارامترهای کانکشن استرینگ را دریافت نموده و سپس با فشردن یک دکمه ، کل دیتابیس را به یک فایل xml تبدیل نموده و آدرس zip شده ی آنرا بر روی فرم برای دانلود نمایش دهد.

- راهنمایی: برای دریافت کامپوننت zip نوشته شده برای دات نت می توانید بـه آدرس هـای زیر مراجعه نمایید:

http://www.aspheute.com/english/20011115.asp

http://www.icsharpcode.net/**OpenSource**/NZipLib/default.asp