Tutorial 10 – Reading and Writing XML Documents

- XML (Extensible Markup Language) is a simple, portable, and flexible way to represent data in a structured format.
- XML is used in a myriad of ways, from acting as the foundation of web-based messaging protocols like SOAP, to being one of the more popular ways to store configuration data (such as the `web.config`, `machine.config`, or `security.config` files in the .NET Framework).
- Microsoft provides classes like `XmlTextReader` and `XmlTextWriter` for lighter access and `XmlDocument` for full DOM (Document Object Model) processing support.
- It is likely that if you use .NET you will be dealing with XML to one degree or another, and in this tutorial we are going to read and write XML documents programmatically.

Creating a New XML Document

The following example creates a new XML document called `Products.xml` programmatically:

1. Create a new Web Form called WriteNewXml.aspx
2. Add a Button control to the page and set the button’s Text property value to “Create a new XML document”.
3. Double click the button and add the following code.

```
protected void Button1_Click(object sender, EventArgs e)
{
    // Create a new, empty document
    System.Xml.XmlDocument doc = new System.Xml.XmlDocument();
    System.Xml.XmlNode docNode = doc.CreateXmlDeclaration("1.0", "UTF-8",
null);
    doc.AppendChild(docNode);

    // Create and insert a new element
    System.Xml.XmlNode productsNode = doc.CreateElement("products");
    doc.AppendChild(productsNode);

    // Create a nested element (with an attribute)
    System.Xml.XmlNode productNode = doc.CreateElement("product");
    System.Xml.XmlAttribute productAttribute = doc.CreateAttribute("id");
    productAttribute.Value = "1001";
    productNode.Attributes.Append(productAttribute);
    productsNode.AppendChild(productNode);

    // Create and add the subelements for this product node
    // (with contained text data)
    System.Xml.XmlNode nameNode = doc.CreateElement("productName");
    nameNode.AppendChild(doc.CreateTextNode("3 in 1 Coffee"));
    productNode.AppendChild(nameNode);
    System.Xml.XmlNode priceNode = doc.CreateElement("productPrice");
    priceNode.AppendChild(doc.CreateTextNode("6.99"));
    productNode.AppendChild(priceNode);
```

```
     // Create and add another product node
     productNode = doc.CreateElement("product");
     productAttribute = doc.CreateAttribute("id");
     productAttribute.Value = "1002";
     productNode.Attributes.Append(productAttribute);
     productsNode.AppendChild(productNode);

     nameNode = doc.CreateElement("productName");
     nameNode.AppendChild(doc.CreateTextNode("3 in 1 Tea"));
     productNode.AppendChild(nameNode);
     priceNode = doc.CreateElement("productPrice");
     priceNode.AppendChild(doc.CreateTextNode("5.99"));
     productNode.AppendChild(priceNode);

     // Save the document
     doc.Save(Server.MapPath("products.xml"));
}
```

4. Click the Start Debugging icon or press F5 to debug the codes and preview the web form in browser.
5. When you click the button, a new XML document called `products.xml` will be created in the same directory that the Web Form is stored. The generated XML document looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<products>
  <product id="1001">
    <productName>3 in 1 Coffee</productName>
    <productPrice>6.99</productPrice>
  </product>
  <product id="1002">
    <productName>3 in 1 Tea</productName>
    <productPrice>5.99</productPrice>
  </product>
</products>
```

`Producsts.xml`

Reading XML documents

The following example reads the `products.xml` document programmatically:

1. Create a new Web Form called ReadXml.aspx
2. Double click the Web Form and add the following code.

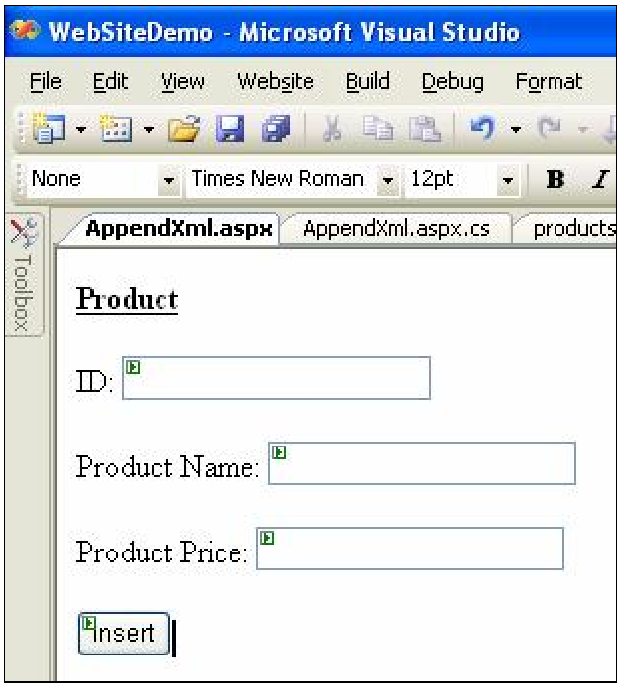
```
protected void Page_Load(object sender, EventArgs e)
{
    if (!System.IO.File.Exists(Server.MapPath("products.xml")))
    {
        Response.Write("File not exists!");
        return;
    }

    System.Xml.XmlTextReader reader = new
System.Xml.XmlTextReader(Server.MapPath("products.xml"));

    while (reader.Read())
    {
        switch (reader.NodeType)
        {
            // The node is a comment
            case System.Xml.XmlNodeType.Comment:
                Response.Write("Comment: " + reader.Value + "<br />");
                break;
            // The node is an element
            case System.Xml.XmlNodeType.Element:
                Response.Write("<" + reader.Name);
                Response.Write(">");
                break;
            //Display the text in each element
            case System.Xml.XmlNodeType.Text:
                Response.Write(reader.Value);
                break;
            //Display the end of the element
            case System.Xml.XmlNodeType.EndElement:
                Response.Write("</" + reader.Name);
                Response.Write(">");
                break;
            default: break;
        }
        // Newline for source document
        Response.Write("\n");
    }
    reader.Close();
}
```

3. Click the Start Debugging icon or press F5 to debug the codes and preview the web form in browser.

Insert Nodes in an XML Document

The following example insert a node into the `products.xml` document:

1. Create a new Web Form called AppendXml.aspx
2. Drag 3 Textbox and 1 Button controls to the form and set the properties accordingly.


WebSiteDemo - Microsoft Visual Studio
File Edit View Website Build Debug Format
None Times New Roman 12pt B I
Toolbox
AppendXml.aspx AppendXml.aspx.cs products
Insert

Product

ID:

Product Name:

Product Price:

3. Double click the button and add the following code.

```
protected void Button1_Click(object sender, EventArgs e)
{
    // Load XML document
    System.Xml.XmlDocument doc = new System.Xml.XmlDocument();
    doc.Load(Server.MapPath("products.xml"));

    // Create a nested element (with an attribute).
    System.Xml.XmlNode productNode = doc.CreateElement("product");
    System.Xml.XmlAttribute productAttribute = doc.CreateAttribute("id");
    productAttribute.Value = this.TextBox1.Text;
    productNode.Attributes.Append(productAttribute);
    doc.DocumentElement.AppendChild(productNode);

    // Create and add the subelements for this product node
    // (with contained text data).
    System.Xml.XmlNode nameNode = doc.CreateElement("productName");
    nameNode.AppendChild(doc.CreateTextNode(this.TextBox2.Text));
    productNode.AppendChild(nameNode);

    // Create and add the subelements for this product node
    // (with contained text data).
    System.Xml.XmlNode priceNode = doc.CreateElement("price");
    priceNode.AppendChild(doc.CreateTextNode(this.TextBox3.Text));
    productNode.AppendChild(priceNode);
    // Save the document
    doc.Save(Server.MapPath("products.xml"));
}
```

3. Click the Start Debugging icon or press F5 to debug the codes and preview the web form in browser.

Exercise

Create an online guest book application that retrieves and stores data using an XML document.