Tutorial 09 – Introduction to ASP.NET

Parse HTTP Form Data with ASP.NET

When you use ASP.NET to process data that is entered on a Web page through a form, two different collections are used depending on which HTTP method is specified in the form declaration:

- GET uses the Request.QueryString collection.
- POST uses the Request.Form collection.

This article describes how to process form data regardless of which HTTP method is used on the form.

Use a Generic Collection to Parse HTTP Form Data

The following steps describe how to create a sample HTML page with two Web forms (one that uses the GET method and one that uses the POST method), and how to create a sample ASP.NET page that processes the form by using either method.

1. Save the following HTML code as ParseData.htm in the folder of a Web site project.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>Parse Data</title>
</head>
<body>
    <form action="showdata.aspx" method="GET">
        GET
        <input type="text" name="txtData" />
        <input type="submit" />
    </form>
    <form action="showdata.aspx" method="POST">
        POST
        <input type="text" name="txtData" />
        <input type="submit" />
    </form>
</body>
</html>
```

2. Save the following ASP.NET code as ShowData.aspx in the folder of a Web site Project.

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Show Data</title>
</head>
<body>
    <%
        Response.Write("Request Method = ");
        Response.Write(Request.ServerVariables["Request_Method"]);
        Response.Write("<br />");

        if (Request.ServerVariables["Request_Method"].Equals("GET"))
        {
            Response.Write("Request.QueryString[\"txtData\"] = ");
            Response.Write(Request.QueryString["txtData"]);
        }

        if (Request.ServerVariables["Request_Method"].Equals("POST"))
        {
            Response.Write("Request.Form[\"txtData\"] = ");
            Response.Write(Request.Form["txtData"]);
        }
    %>
</body>
</html>
```

3. When you open ParseData.htm by using HTTP, you may use either form to view the form results.

Use Session Variables to Store and Retrieve HTTP Form Data

ASP.NET session variables enable you to store and retrieve values for a user as the user navigates the different ASP.NET pages that make up a Web application. Session variables are created by simply referring to the session variable by name.

1. Save the following HTML code as NewParseData.aspx in the folder of a Web site project.

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">
    protected void Button1_Click(object sender, EventArgs e)
    {
        Session["txtData"] = this.TextBox1.Text;
        Response.Redirect("NewShowData.aspx");
    }
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Parse Data</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:Button  ID="Button1"  runat="server"  OnClick="Button1_Click"
Text="Submit Query" /></div>
    </form>
</body>
</html>
```

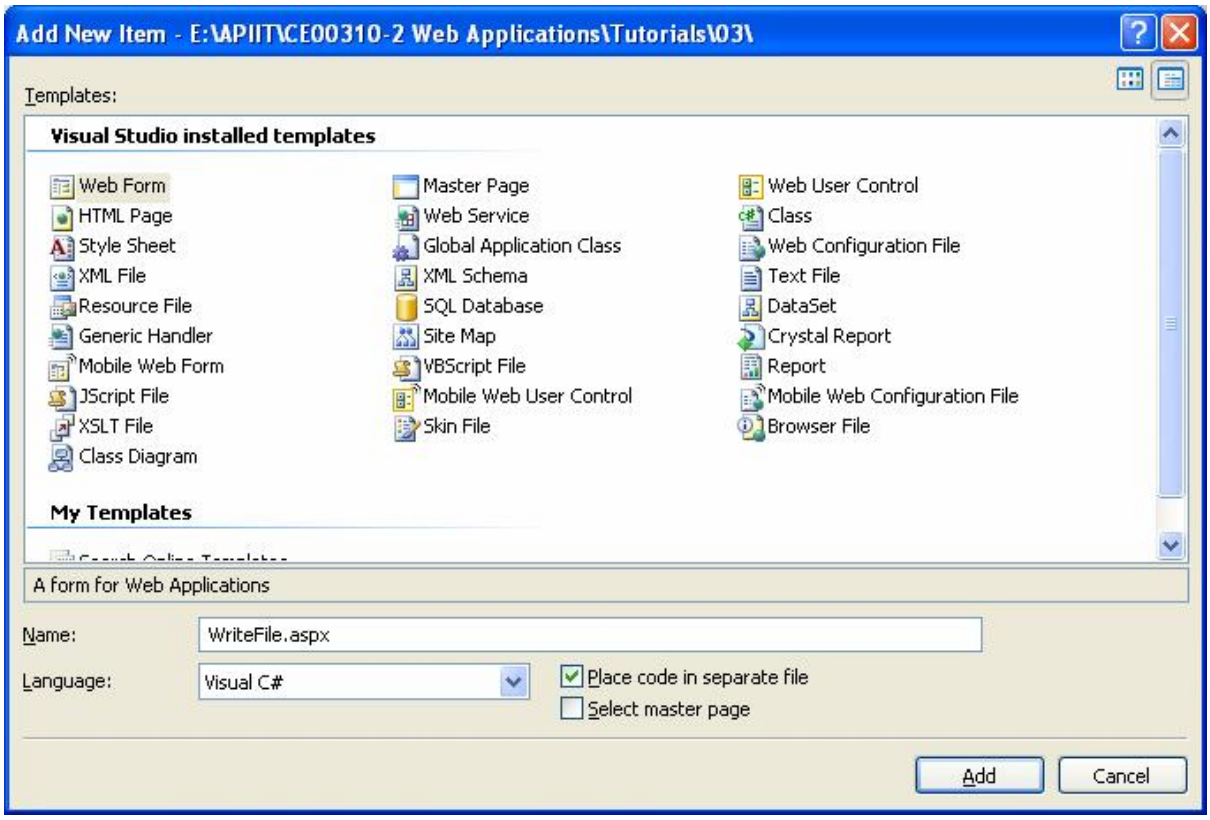
2. Save the following ASP.NET code as NewShowData.aspx in the folder of a Web site Project.

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Show Data</title>
</head>
<body>
<%
    Response.Write("Session[\"txtData\"] = ");
    Response.Write(Session["txtData"]);
%>
</body>
</html>
```

Write Text to a File

The following example shows how to write text to a text file.

1. Create a Web Form called WriteFile.aspx with the "Place code in separate file" option checked.



2. Add a TextBox control to the page.
3. Add a Button control to the page and set the following properties:

Property	Value
Text	Write to File

4. Double click the Button on the web form and type the following codes.

```
protected void Button1_Click(object sender, EventArgs e)
{
      System.IO.FileStream fs = new
            System.IO.FileStream(Page.Server.MapPath("App_Data//Data.txt"),
            System.IO.FileMode.Append, System.IO.FileAccess.Write);
            System.IO.StreamWriter sw = new System.IO.StreamWriter(fs);

      sw.WriteLine(this.TextBox1.Text);

      sw.Close();
fs.Close();
}
```

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

Read Text from a File

The following example shows how to read text from a text file.

1. Create a Web Form called ReadFile.aspx with the “Place code in separate file” option checked.

2. Add a Label control to the page.

3. Double click the web form and type the following codes.

```
protected void Page_Load(object sender, EventArgs e)
{
      System.IO.FileStream fs = new
      System.IO.FileStream(Page.Server.MapPath("App_Data//Data.txt"),
      System.IO.FileMode.Open, System.IO.FileAccess.Read);
      System.IO.StreamReader sr = new System.IO.StreamReader(fs);

      string oneLine = sr.ReadLine();
      this.Label1.Text = "";

      while (oneLine != null)
      {
            this.Label1.Text = this.Label1.Text + oneLine + "<br />";
            oneLine = sr.ReadLine();
      }
      sr.Close();
      fs.Close();
}
```

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

Parse Strings Using the Split Method

The following code example demonstrates how a string can be parsed using the System.String.Split method. This method works by returning an array of strings, where each element is a word. As input, Split takes an array of chars that indicate which characters are to be used as delimiters. In this example, spaces, commas, periods, colons, and tabs are used. An array containing these delimiters is passed to Split, and each word in the sentence is displayed separately using the resulting array of strings.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <%
        char[] delimiterChars = { ' ', ',', '.', ':', '\t' };

        string text = "one\ttwo three:four,five six.seven";
        Response.Write("Original text: " + text);
        Response.Write("<br />");

        string[] words = text.Split(delimiterChars);

        Response.Write("Words in text: " + words.Length);
        Response.Write("<br />");

        foreach (string s in words)
        {
            Response.Write(s + "<br />");
        }
    %>
</body>
</html>
```

Exercises

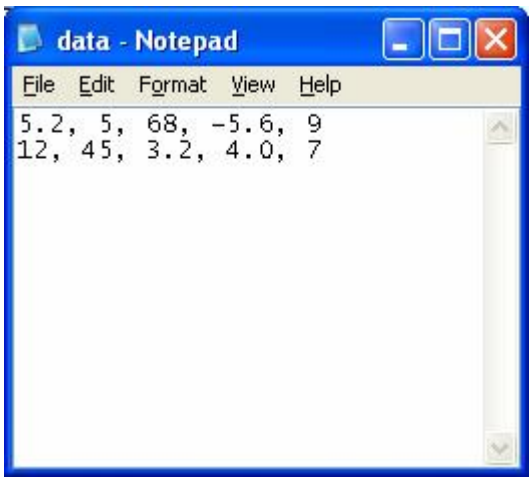
Create an ASP.NET web form for each of the following

1. Compute and output the total of three integer number entered from keyboard.

2. Compute and output the average of two floating-point numbers entered from keyboard.

3. Input person's name and year born to the web form and display the current age.

4. (i). Write 5 floating point numbers (separate by comma) to a text file.



(ii). Read the floating numbers from the text file and display the average for each line of numbers.