Agenda

1. *ASP.NET Introduction*
2. *First ASP.NET Application*
3. *AutoPostback Property*
4. *Event Handler Parameters*
5. *Dynamically initializing Controls*
6. *IsPostBack property of Page class*
7. *ListControls*
8. *Comparison between HtmlControls and WebControls*
9. *Control Properties and Methods*
10. *FileUpload Control*

ASP.NET

ASP.NET is a framework used for development and execution of web based applications using MS.NET framework.

To create a new ASP.NET application:

File → New Website

File System → It uses ASP.NET Development Server and for every web application a new instance of ASP.NET Development Server Starts

HTTP → It uses IIS server.

Points to be noted / observed:

1. Every Webform has an extension "**.aspx**"
2. A Webform doesn't have action attribute and it always submits to itself.
3. The default method rendered by the Webform in ASP.NET is "POST"
4. Here the UI (aspx) is separated from the Business Code/ Event Handlers (aspx.cs).
   <%@ Page AutoEventWireup="true" **CodeFile="Default.aspx.cs"** Inherits="_Default" %>.
5. When an ASP.NET web application is build in VS.NET we don't get any output file in the form any EXE or DLL.
6. ASP.NET controls uses the same programming model which we use for Windows based application
7. Every server side control renders HTML output to the web browser.

Example:

Handling events in ASP.NET

```
<asp:Literal ID="ltrMessage" runat="server"></asp:Literal><br />
First Name:<asp:TextBox ID="txtFirstName" runat="server"></asp:TextBox><br />
Last Name:<asp:TextBox ID="txtLastName" runat="server"></asp:TextBox><br />
<asp:Button ID="btnSayHello" runat="server" Text="Say Hello" OnClick="btnSayHello_Click" />
```

CodeBehind:

```
protected void btnSayHello_Click(object sender, EventArgs e)
{
   ltrMessage.Text = "Hello " + txtFirstName.Text + " " + txtLastName.Text;
}
```

Code: 4.1 | **C#**

Example 2:

1. Add a new WebForm: Solution Explorer → Right Click on Project → Add New Item → Default2.aspx
2. Right Click on Default2.aspx → Set as Start Page.
3. Add the following between <Form>.

Handling events in ASP.NET

```
<asp:Label runat="server" ID="lblDemo">This is Demo Text</asp:Label>
Please Select a Color:
```

```
<asp:DropDownList ID="ddlColor" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="ddlColor_SelectedIndexChanged">
   <asp:ListItem>Red</asp:ListItem>
   <asp:ListItem>Green</asp:ListItem>
   <asp:ListItem>Blue</asp:ListItem>
   <asp:ListItem>Orange</asp:ListItem>
</asp:DropDownList>
```

CodeBehind:

```
protected void ddlColor_SelectedIndexChanged(object sender, EventArgs e)
{
   lblDemo.ForeColor = System.Drawing.Color.FromName(ddlColor.SelectedValue);
}
```

Code: 4.2 | **C#**

Example 3:

Multiple RadioButtons mapping to same method / event handler on server and significance of sender in event handler.

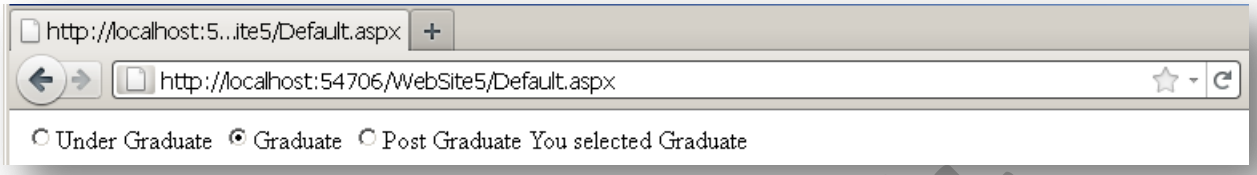
```
<asp:RadioButton ID="rbnUnderGraduate" runat="server" AutoPostBack="true" GroupName="qualification"
   Text="Under Graduate" OnCheckedChanged="rbnQualification_CheckedChanged" />
<asp:RadioButton ID="rbnGraduate" runat="server" AutoPostBack="true" GroupName="qualification"
   Text="Graduate" OnCheckedChanged="rbnQualification_CheckedChanged" />
<asp:RadioButton ID="rbnPostGraduate" runat="server" AutoPostBack="true" GroupName="qualification"
   Text="Post Graduate" OnCheckedChanged="rbnQualification_CheckedChanged" />
<asp:Literal ID="ltrComment" runat="server"></asp:Literal>
```

CodeBehind:

```
protected void rbnQualification_CheckedChanged(object sender, EventArgs e)
{
   /*  if (rbnUnderGraduate.Checked)
        ltrComment.Text = "Please graduate atleast";
     else if (rbnGraduate.Checked)
        ltrComment.Text = "Try for PG";
     else
        ltrComment.Text = "Enough of studies now do some work";*/
   RadioButton rbn = (RadioButton)sender; //sender is reference to the control because of which the event is raised.
   ltrComment.Text = "You selected " + rbn.Text;
```

```
}
```

Code: 4.3 | C#



Example 4: Understanding the second parameter of the event handler.

1. Add calendar control to the webform.

Note: For every day/cell of a calendar control, DayRender event is raised on server when that day is rendered.

2. To handle the event: Select Calendar in design view → Properties → Events Tab → double click on DayRender

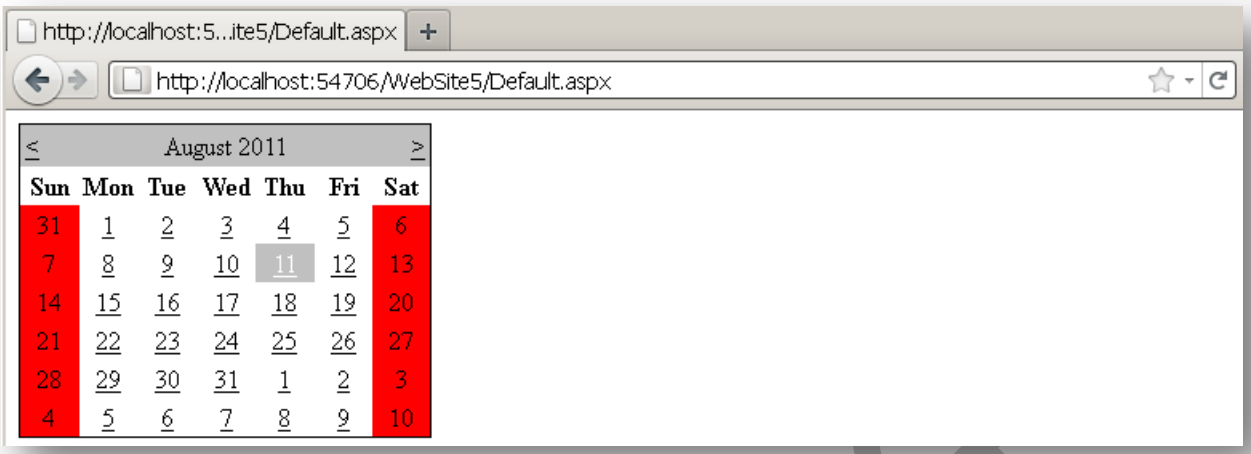
Calender control

```
<asp:Calendar ID="Calendar1" runat="server" ondayrender="Calendar1_DayRender1"></asp:Calendar>
```

CodeBehind:

```
protected void Calendar1_DayRender(object sender, DayRenderEventArgs e)
{
   if (e.Day.IsWeekend)
   {
      e.Day.IsSelectable = false;
      e.Cell.BackColor = System.Drawing.Color.Red;
   }
}
```

Code: 4.4 | C#



Example 5: Example with ListBox

Listbox control

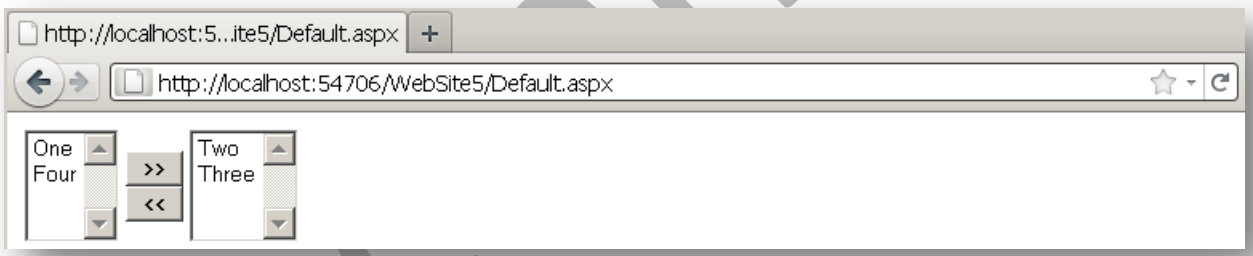
```
<table>
  <tr>
    <td>
      <asp:ListBox ID="lst1" runat="server">
        <asp:ListItem>One</asp:ListItem>
        <asp:ListItem>Two</asp:ListItem>
        <asp:ListItem>Three</asp:ListItem>
        <asp:ListItem>Four</asp:ListItem>
      </asp:ListBox>
    </td>
    <td>
      <asp:Button ID="btnMoveRight" runat="server" Text="&gt;&gt;" OnClick="btnMoveRight_Click" /><br />
      <asp:Button ID="btnMoveLeft" runat="server" Text="&lt;&lt;" OnClick="btnMoveLeft_Click" />
    </td>
    <td>
      <asp:ListBox ID="lst2" runat="server"></asp:ListBox>
    </td>
  </tr>
</table>
```

CodeBehind:

```
protected void btnMoveRight_Click(object sender, EventArgs e)
{
   if (lst1.SelectedItem != null)
   {
```

```
        ListItem li = lst1.SelectedItem;
        lst1.Items.Remove(li);
        li.Selected = false;
        lst2.Items.Add(li);
    }
}
protected void btnMoveLeft_Click(object sender, EventArgs e)
{
    if (lst2.SelectedItem != null)
    {
        ListItem li = lst2.SelectedItem;
        lst2.Items.Remove(li);
        li.Selected = false;
        lst1.Items.Add(li);
    }
}
```

Code: 4.5 | C#



Example 6: To demonstrate **IsPostBack** property of page.

Usage of IsPostBack property

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!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></title>
</head>
<body>
    <form id="form1" runat="server">
    <table>
        <tr>
```

```
    <td>
      <asp:Label ID="lblDemo" runat="server" />
    </td>
  </tr>
  <tr>
    <td>
      <asp:TextBox runat="server" ID="txtDemo" />
    </td>
  </tr>
  <tr>
    <td>
      <asp:CheckBox Text="Bold" ID="chkBold" runat="server" />
    </td>
    <td>
      <asp:CheckBox Text="Italics" ID="chkItalics" runat="server" />
    </td>
  </tr>
  <tr>
    <td>
      Size:
    </td>
    <td>
      <asp:DropDownList ID="ddlSize" runat="server">
      </asp:DropDownList>
    </td>
  </tr>
  <tr>
  <td>
    <asp:RadioButton Text="Red" ID="rbnRed" GroupName="rbn" runat="server" />
    <asp:RadioButton Text="Green" ID="rbnGreen" GroupName="rbn" runat="server" />
    <asp:RadioButton Text="Blue" ID="rbnBlue" GroupName="rbn" runat="server" />
  </td>
  </tr>
  <tr>
  <td>
    <asp:Button Text="Set" ID="btnSet" runat="server" onclick="btnSet_Click" />
  </td>
  </tr>
```

```
  </table>
  </form>
</body>
</html>
```

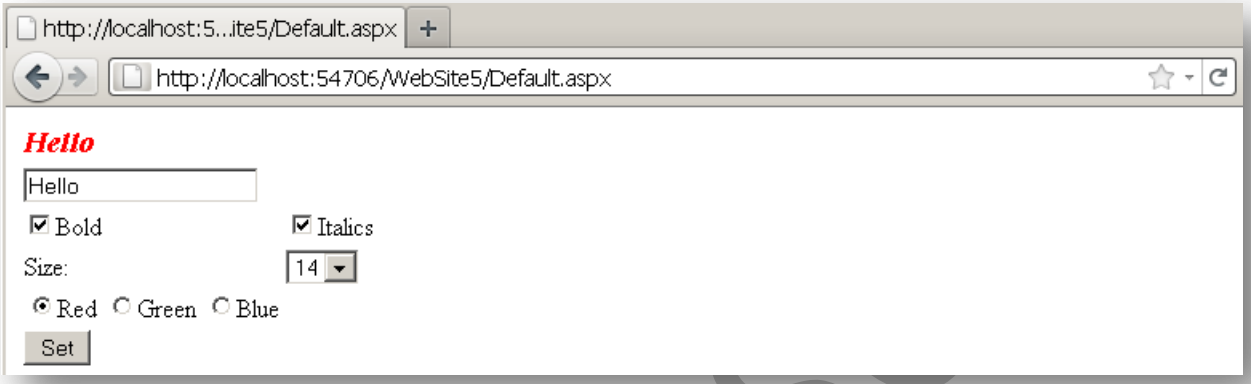
CodeBehind:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class _Default : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      if (!IsPostBack)
      {
         //To Add ListItems to ddlSize
         for (int i = 8; i <= 24; i += 2)
         {
            ListItem li = new ListItem(i.ToString());
            ddlSize.Items.Add(li);
         }
      }
   }
   protected void btnSet_Click(object sender, EventArgs e)
   {
      lblDemo.Text = txtDemo.Text;
      lblDemo.Font.Bold = chkBold.Checked;
      lblDemo.Font.Italic = chkItalics.Checked;
      lblDemo.Font.Size = new FontUnit(int.Parse(ddlSize.SelectedValue));
      string col;
      if (rbnRed.Checked)
         col = "red";
      else if (rbnGreen.Checked)
         col = "Green";
```

```
        else
            col = "blue";
        lblDemo.ForeColor = System.Drawing.Color.FromName(col);
    }
}
```

Code: 4.6 | **C#**



Important:

- Page_Load event handler will always execute with every trip to server (irrespective of a direct request or the form is submitted)
- When the browser request for the page directly then IsPostBack is False, but when the Form is Posted / Submitted, IsPostBack is True.

Types of Server Controls

1. **HTML Server Controls** - Traditional HTML tags - **System.Web.UI.HtmlControls**
2. **Web Server Controls**: Web server controls are special ASP.NET tags understood by the ASP.NET server – **System.Web.UI.WebControls**

- HTML Controls are by default treated as text in ASP.NET. To make these elements programmable add runat="server" attribute to the HTML elements.
- If the requirement is to program the control on client side in javascript and has no code to execute on the server then only we should go for HTML controls otherwise always WebServer controls must be used because they are very dynamic, powerful and also has enhanced functionality.

- **Label**: Renders SPAN tag to the client. Because the span tag rendered to the browser had "Id" it can programmed using javascript on client.
  Properties: Text,
- **Literal Control:** Renders just text without any tag and hence cannot be formatted or programmed in browser.
  Properties: Text, Mode (Passthrough / Encode)
- **Textbox**: Can render either <input type="text" or <input type="password" or <textarea> tags
  Properties: Text, TextMode (SingleLine/MultiLine/Password), ReadOnly, MaxLength, AutoPostBack,
  Event: TextChanged
- **Button:** renders as <input type="**submit" …./>**
  Properties: Text, OnClientClick.
  Event: Click
  <asp:Button **OnClientClick**="return confirm('Are you sure')"…./> is rendered as <input type="submit" OnClick="return confirm('Are you sure')" ….>

  Note: In Javascript if onclick of Submit Button return false, the form is not submitted to server.

- **LinkButton**: renders <A href="__doPostBack(…)" …> -Used for posting the form to server.
  Properties: Text, OnClientClick
  Event: Click
- **ImageButton**: renders <input type="image" …> Right Click on Project → New Folder → (Images) → Right Click → Add Exisiting Item → Images
  Properties: ImageUrl, OnClientClick.
  Event: Click
- **HyperLink**: <A href="Url"… - Used for linking to another page and not posting the form.
  Properties: Text, ImageUrl, NavigateUrl, Target
- **CheckBox:** renders <input type="Checkbox" **…**
  Properties: Text, TextAlign, Checked, AutoPostBack
  Event: CheckedChanged
- **RadioButton:** renders <input type="radio"**. Its inherited from CheckBox class.**

Properties: Text, Checked, AutoPostBack, **GroupName** (must be same for grouping the radio buttons)

Event: CheckedChanged

- **ListControl:** It's a common parent for DropDownList, ListBox, CheckBoxList, RadioButtonList, BulletedList

  Properties: Items, SelectedIndex, SelectedItem, SelectedValue, Text, AutoPostBack

  Methods: ClearSelection

  Event: SelectedIndexChanged

  - **ListItem:** renders based on container tag in which it is used.

    Properties: Text, Value, Selected

- **DropDownList:** renders <select tag
- **ListBox:** renders <select **size**="4" …

  Properties: Rows (maps to size attribute), SelectionMode (Single / Multiple)

  Method: GetSelectedIndices() as Integer()

- **CheckboxList**: renders <input type="checkbox"… for every ListItem

  Properties: RepeatColumns, RepeatDirection, RepeatLayout

- **RadioButtonList**: renders <input type="radio" … for every ListItem

  Properties: RepeatColumns, RepeatDirection, RepeatLayout

Example:

Adding checkboxes dynamically

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!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></title>
</head>
<body>
   <form id="form1" runat="server">
   <asp:CheckBoxList  ID="lst" runat="server" RepeatColumns="3" RepeatDirection="Horizontal"
AutoPostBack="True" onselectedindexchanged="lst_SelectedIndexChanged"/>
   </form>
</body>
</html>
```

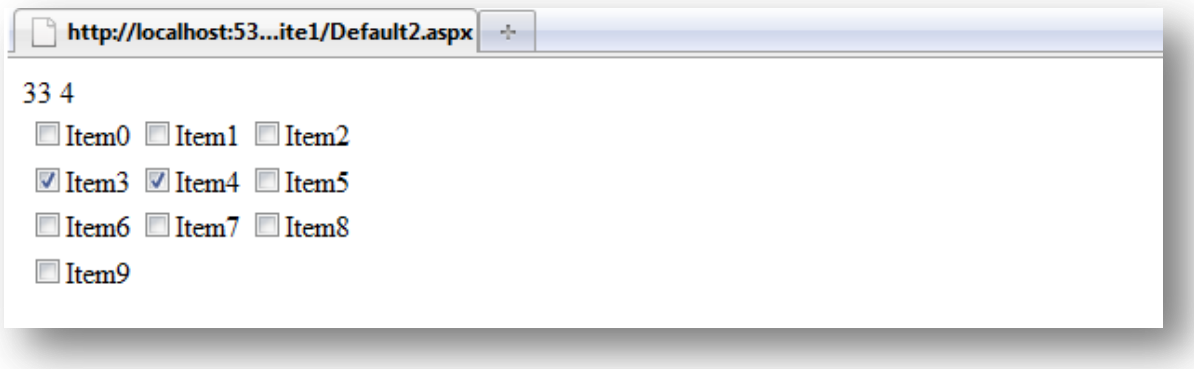
CodeBehind:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class _Default : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      if (!IsPostBack)
      {
         for (int i = 0; i < 10; i++)
         {
            ListItem li = new ListItem();
            li.Text = "Item" + i; //Displayed Text
            li.Value = i.ToString(); //Submitted Value
            lst.Items.Add(li);
         }
      }
   }
   protected void lst_SelectedIndexChanged(object sender, EventArgs e)
   {
      // For Single Selected
      Response.Write(lst.SelectedValue);
      //For multiple items selected
      string str = "";
      foreach (ListItem li in lst.Items)
      {
         if (li.Selected)
            str += li.Value + " ";
      }
      Response.Write(str);
   }
}
```

Code: 4.7	C#



Try the above program replacing "<asp:CheckBoxList…" with

1. <asp:DropDownList
2. <asp:ListBox SelectionMode="Multiple"…
3. <asp:RadioButtonList….

Note: With every request Load event of Page is raised i.e. even if the form is posted / submitted. Using *Page.IsPostBack* we can find if the request is for the first time or is because of submit button clicked.

FileUpload Control Demo

Using FileUpload control

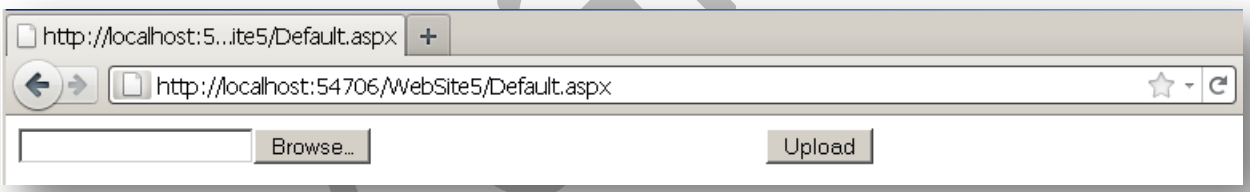
```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:FileUpload ID="FileUpload1" runat="server" Width="463px" />
        <asp:Button ID="btnUpload" runat="server" OnClick="btnUpload_Click" Text="Upload" />
    </form>
</body>
</html>
```

CodeBehind:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
using System.Web.UI;
using System.Web.UI.WebControls;
public partial class _Default : System.Web.UI.Page
{
   protected void btnUpload_Click(object sender, EventArgs e)
   {
      if (FileUpload1.HasFile)
      {
         string fn = FileUpload1.FileName;
         string virtualPath = "~/UPloadedFiles/" + fn;
         string physicalPath = MapPath(virtualPath);
         FileUpload1.SaveAs(physicalPath);
         //byte[] b=FileUpload1.Filebytes;
         //System.IO.Stream s=FileUpload1.FileContent;
      }
   }
}
```

Code: 4.8 | ***C#***



Note: File upload control doesn't retain its state on roundtrip.