Web Controls

Today You Will Learn

- Stepping Up to Web Controls
- Web Control Classes
- List Controls
- Table Controls

Stepping Up to Web Controls

Uses of Web Controls:

- **They provide a rich user interface:** A web control is programmed as an object but doesn't necessarily correspond to a single element in the final HTML page.
- **They provide a consistent object model:** HTML is full of quirks and idiosyncrasies. <input type="text"> <input type="password">
  <textarea> in HTML but in webcontrols it is textbox.
- **They tailor their output automatically:** ASP.NET server controls can detect the type of browser and automatically adjust the HTML code they write to take advantage of features such as support for JavaScript. You don't need to know about the client because ASP.NET handles that layer and automatically uses the best possible set of features. This feature is known as adaptive rendering.

Stepping Up to Web Controls

- **They provide high-level features:** You'll see that web controls allow you to access additional events, properties, and methods that don't correspond directly to typical HTML controls.

Basic Web Control Classes

Control Class	Underlying HTML Element
Label	
Button	<input type="submit"> or <input type="button">
TextBox	<input type="text">, <input type="password">, or <textarea>
CheckBox	<input type="checkbox">
RadioButton	<input type="radio">
Hyperlink	<a>
LinkButton	<a> with a contained <img> tag
ImageButton	<input type="image">

Stepping Up to Web Controls

Control Class	Underlying HTML Element
Image	<img>
ListBox	<select size="X"> where X is the number of rows that are visible at once
DropDownList	<select>
CheckBoxList	A list or <table> with multiple <input type="checkbox"> tags
RadioButtonList	A list or <table> with multiple <input type="radio"> tags
BulletedList	An ordered list (numbered) or unordered list (bulleted)
Panel	
Table, TableRow, and TableCell	<table>, <tr>, and <td> or <th>

The Web Control Tags

- ASP.NET tags always begin with the prefix asp: followed by the class name.

Stepping Up to Web Controls

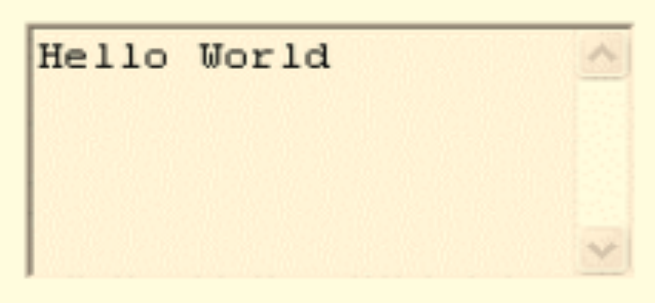
```
<asp:TextBox ID="txt" runat="server" />
```

- You could place some text in the TextBox, set its size, make it read-only, and change the background color. All these actions have defined properties.

```
<asp:TextBox ID="txt" BackColor="Yellow" Text="Hello World"
ReadOnly="True" TextMode="MultiLine" Rows="5" runat="server" />
```

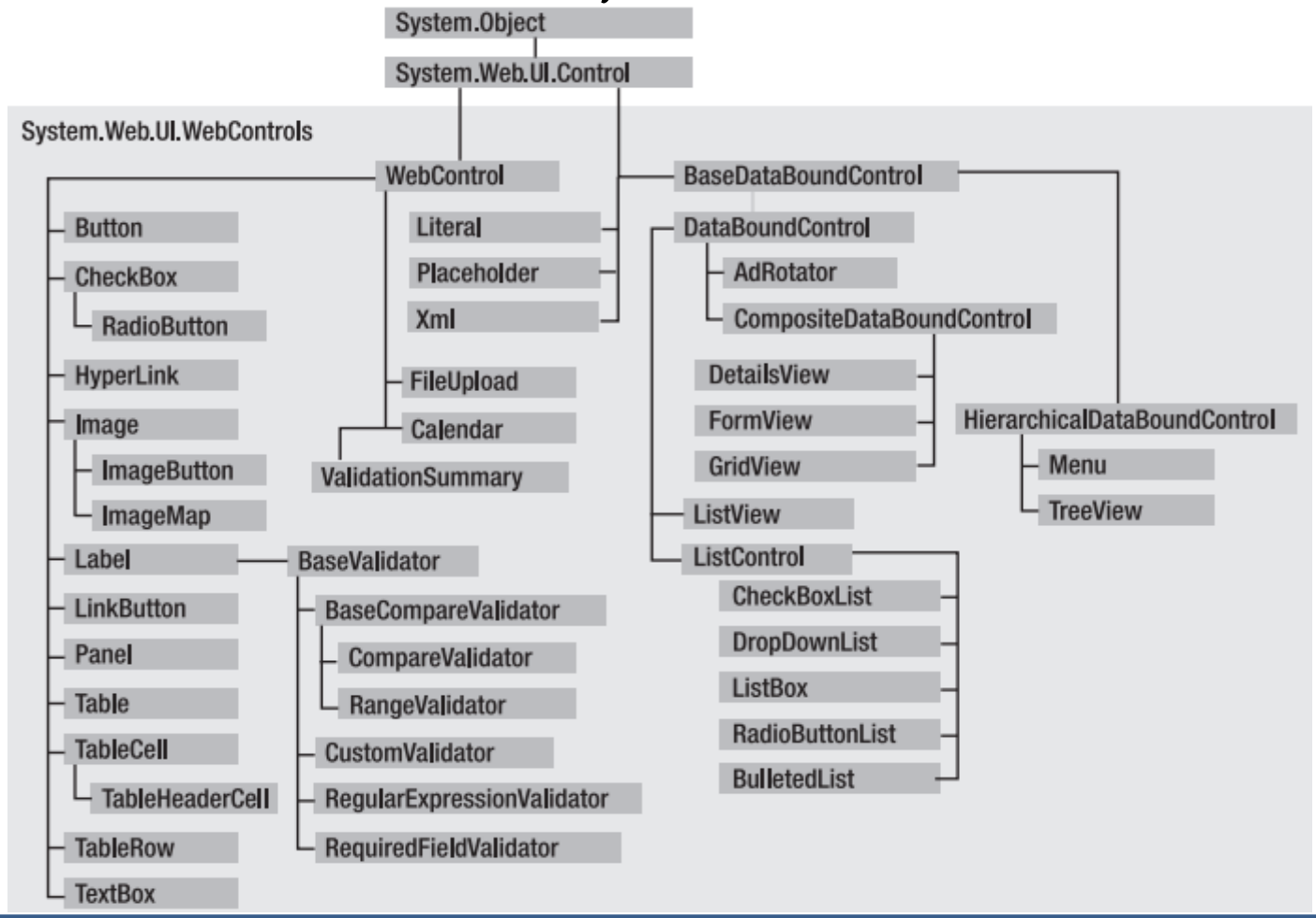
- TextArea uses properties like name, column, read-only and style.

```
<textarea name="txt" rows="5" cols="20" readonly="readonly" id="txt"
style="background-color:Yellow;">Hello World</textarea>
```



Web Control Classes

- Web control classes are defined in the System.Web.UI.WebControls namespace.



Web Control Classes

The WebControl Base Class

Property	Description
AccessKey	Specifies the keyboard shortcut as one letter. For example, if you set this to Y, the Alt+Y keyboard combination will automatically change focus to this web control. This feature is supported only on Internet Explorer 4.0 and higher.
BackColor, ForeColor, and BorderColor	Sets the colors used for the background, foreground, and border of the control. In most controls, the foreground color sets the text color.
BorderWidth	Specifies the size of the control border.
BorderStyle	One of the values from the BorderStyle enumeration, including Dashed, Dotted, Double, Groove, Ridge, Inset, Outset, Solid, and None.
Controls	Provides a collection of all the controls contained inside the current control. Each object is provided as a generic System.Web.UI.Control object, so you will need to cast the reference to access control-specific properties.
Enabled	When set to False, the control will be visible, but it will not be able to receive user input or focus.

Web Control Classes

EnableViewState	Set this to False to disable the automatic state management for this control. In this case, the control will be reset to the properties and formatting specified in the control tag (in the .aspx page) every time the page is posted back. If this is set to True (the default), the control uses the hidden input field to store information about its properties, ensuring that any changes you make in code are remembered.
Font	Specifies the font used to render any text in the control as a System.Web.UI.WebControls.FontInfo object.
Height and Width	Specifies the width and height of the control. For some controls, these properties will be ignored when used with older browsers.
ID	Specifies the name that you use to interact with the control in your code (and also serves as the basis for the ID that's used to name the top-level element in the rendered HTML).
Page	Provides a reference to the web page that contains this control as a System.Web.UI.Page object.
Parent	Provides a reference to the control that contains this control. If the control is placed directly on the page (rather than inside another control), it will return a reference to the page object.

Web Control Classes

TabIndex	A number that allows you to control the tab order. The control with a TabIndex of 0 has the focus when the page first loads. Pressing Tab moves the user to the control with the next lowest TabIndex, provided it is enabled. This property is supported only in Internet Explorer 4.0 and higher.
ToolTip	Displays a text message when the user hovers the mouse above the control. Many older browsers don't support this property.
Visible	When set to False, the control will be hidden and will not be rendered to the final HTML page that is sent to the client.

Units

- All the properties that use measurements, including BorderWidth, Height, and Width, require the Unit structure, which combines a numeric value with a type of measurement (pixels, percentage, and so on).
- Unit types are indicated by px (pixel) or % (for percentage).

Web Control Classes

```
<asp:Panel Height="300px" Width="50%" ID="pnl" runat="server" />
```

- Use Pixel() to supply a value in pixels, and use Percentage() to supply a percentage value.

```
//Convert the number 300 to a Unit object representing pixels, and assign it.
        pnl.Height = Unit.Pixel(300)
// Convert the number 50 to a Unit object representing percent, and assign it.
        pnl.Width = Unit.Percentage(50)
```

- You could also manually create a Unit object and initialize it using one of the supplied constructors and the UnitType enumeration.

```
//Create a Unit object.
        Unit myUnit = new Unit(300, UnitType.Pixel);
// Assign the Unit object to several controls or properties.
        pnl.Height = myUnit
        pnl.Width = myUnit
```

Web Control Classes

Enumerations

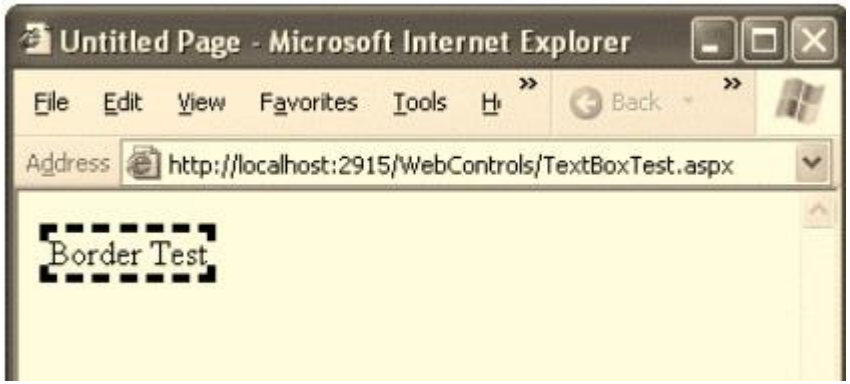
- Enumerations are used heavily in the .NET class library to group a set of related constants.

  Ex: You can choose one of several predefined values from the BorderStyle enumeration.

  ctrl.BorderStyle = BorderStyle.Dashed

  Code in .aspx file is as follows:

```
<asp:Label BorderStyle="Dashed" Text="Border Test" ID="lbl"  runat="server" />
```



Web Control Classes

Colors

- The Color property refers to a Color object from the System.Drawing namespace.
- Ways to create Color object:
  - **Using an ARGB (alpha, red, green, blue) color value:** You specify each value as an integer from 0 to 255. The alpha component represents the transparency of a color, and usually you'll use 255 to make the color completely opaque.
  - **Using a predefined .NET color name:** You choose the correspondingly named read-only property from the Color structure. These properties include the 140 HTML color names.
  - **Using an HTML color name:** You specify this value as a string using the ColorTranslator class.

Web Control Classes

```
Using System.Drawing;
```

The following code shows several ways to specify a color in code:

```
//Create a color from an ARGB value
        int alpha=255, red=0, green=255, blue=0;
        ctrl.ForeColor = Color.FromArgb(alpha, red, green, blue);
//Create a color using a .NET name
        ctrl.ForeColor = Color.Crimson ;
// Create a color from an HTML code
        ctrl.ForeColor = ColorTranslator.FromHtml("Blue")
```

Defining a color in the .aspx file, using one of the known color names:

```
<asp:TextBox  ForeColor="Red" Text="Test" ID="txt" runat="server" />
```

Use a hexadecimal color number:

```
<asp:TextBox  ForeColor="#ff50ff" Text="Test"  ID="txt" runat="server" />
```

Web Control Classes

Font

- The Font property actually references a full FontInfo object, which is defined in the System.Web.UI.WebControls namespace.

Property	Description
Name	A string indicating the font name (such as Verdana).
Names	An array of strings with font names, in the order of preference. The browser will use the first matching font that's installed on the user's computer.
Size	The size of the font as a FontUnit object. This can represent an absolute or relative size.
Bold, Italic, Strikeout, Underline, and Overline	Boolean properties that apply the given style attribute.

```
ctrl.Font.Name = "Verdana"
```
(In code file .cs)

```
<asp:TextBox Font-Name="Tahoma" Font-Size="40" Text="Size Test" ID="txt" runat="server" />
```
(In .aspx file)

Web Control Classes

Focus

- The Focus() method affects only input controls (controls that can accept keystrokes from the user). When the page is rendered in the client browser, the user starts in the focused control.

<form DefaultFocus="TextBox2" runat="server"> (The cursor focus on the TextBox2)

- **Manage focus is using access keys:** If you set the AccessKey property of a TextBox to A, pressing Alt+A focus will switch to the TextBox.

The trick is to set the Label.AssociatedControlID property to specify a linked input control. That way, the label transfers focus to the associated control.

<asp:Label AccessKey="2" AssociatedControlID="TextBox2" runat="server" Text="TextBox2:"/>

<asp:TextBox runat="server" ID="TextBox2" />

List Controls

- The list controls include the ListBox, DropDownList, CheckBoxList, RadioButtonList, and BulletedList.
- All the selectable list controls provide a SelectedIndex property that indicates the selected row as a zero-based index.
- The ListItem object provides three important properties:
  - Text (the displayed content)
  - Value (the hidden value from the HTML markup)
  - Selected (True or False depending on whether the item is selected).

```
ListItem item;
item = Currency.SelectedItem;
```

List Controls

Multiple-Select List Controls

- The ListBox, provides you a set of SelectionMode property to the enumerated value ListSelectionMode.Multiple.
- The .aspx file for this page defines CheckBoxList, Button, and Label controls:

```
<form runat="server">
    <div>
        Choose your favorite programming languages:<br /><br />
        <asp:CheckBoxList  ID="chklst" runat="server" /><br /><br />
        <asp:Button ID="cmdOK" Text="OK" OnClick =“cmdOK_Click” runat="server" />
        <br /><br />
        <asp:Label ID="lblResult" runat="server" />
    </div>
</form>
```

List Controls

- The code adds items to the CheckListBox at startup:

```
public partial class CheckListTest : System.Web.UI.Page
{
    protected void Page_OnLoad(Object sender, EventArgs e)
    {
        if(!this.IsPostBack)
        {
            chklist.Items.Add("C");
            chklist.Items.Add("C++");
            chklist.Items.Add("C#");
            chklist.Items.Add("VB");
            chklist.Items.Add("Java");
            chklist.Items.Add("Pascal");
        }
    }
}
```

List Controls

The BulletedList Control

- The BulletedList control is a server-side equivalent of the <ul> (unordered list) and <ol> (ordered list) elements.

Property	Description
BulletStyle	Determines the type of list. Choose from Numbered (1, 2, 3, . . .), LowerAlpha (a, b, c, . . .) and UpperAlpha (A, B, C, . . .), LowerRoman (i, ii, iii, ; . . .) and UpperRoman (I, II, III, . . .), and the bullet symbols Disc, Circle, Square, or CustomImage (in which case you must set the BulletImageUrl property).
BulletImageUrl	If the BulletStyle is set to CustomImage, this points to the image that is placed to the left of each item as a bullet.
FirstBulletNumber	In an ordered list (using the Numbered, LowerAlpha, UpperAlpha, LowerRoman, and UpperRoman styles), this sets the first value. For example, if you set FirstBulletNumber to 3, the list might read 3, 4, 5 (for Numbered) or C, D, E (for UpperAlpha).
DisplayMode	Determines whether the text of each item is rendered as text (use Text, the default) or a hyperlink (use LinkButton or HyperLink). The difference between LinkButton and HyperLink is how they treat clicks. When you use LinkButton, the BulletedList fires a Click event that you can react to on the server to perform the navigation. When you use HyperLink, the BulletedList doesn't fire the Click event—instead, it treats the text of each list item as a relative or absolute URL, and renders them as ordinary HTML hyperlinks. When the user clicks an item, the browser attempts to navigate to that URL.

Table Controls

- The Table control is built out of a hierarchy of objects.

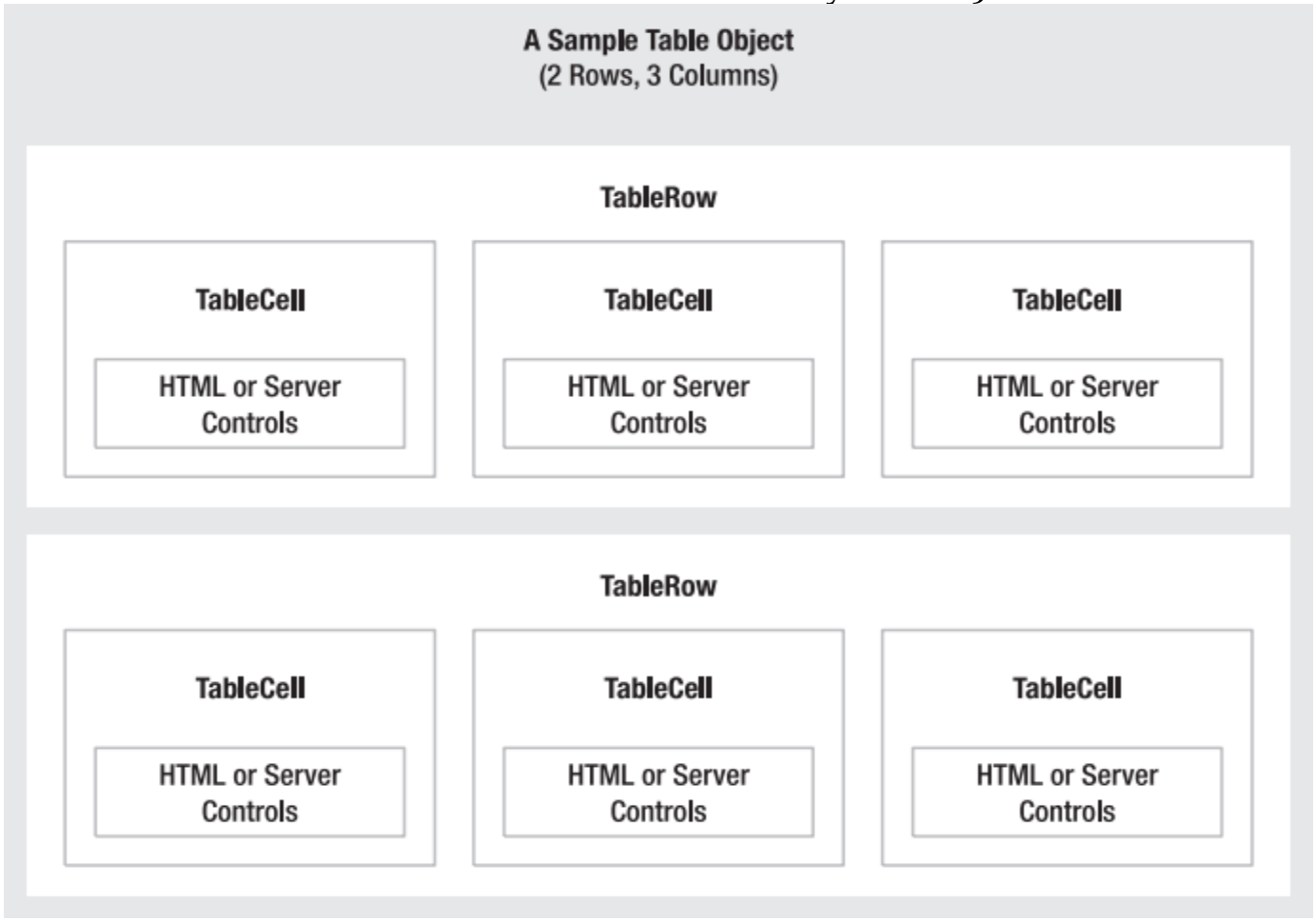


Table Controls

- The .aspx code creates the TextBox, CheckBox, Button, and Table controls:

```
<form runat="server">
 <div>
  Rows:
  <asp:TextBox ID="txtRows" runat="server" />
   Cols:
  <asp:TextBox ID="txtCols" runat="server" />
  <br /><br />
  <asp:CheckBox ID="chkBorder" runat="server"
  Text="Put Border Around Cells" />
  <br /><br />
  <asp:Button ID="cmdCreate" runat="server"
  Text="Create" OnClick =“cmdCreate_Click” />
  <br /><br />
  <asp:Table ID="tbl" runat="server" />
 </div>
</form>
```

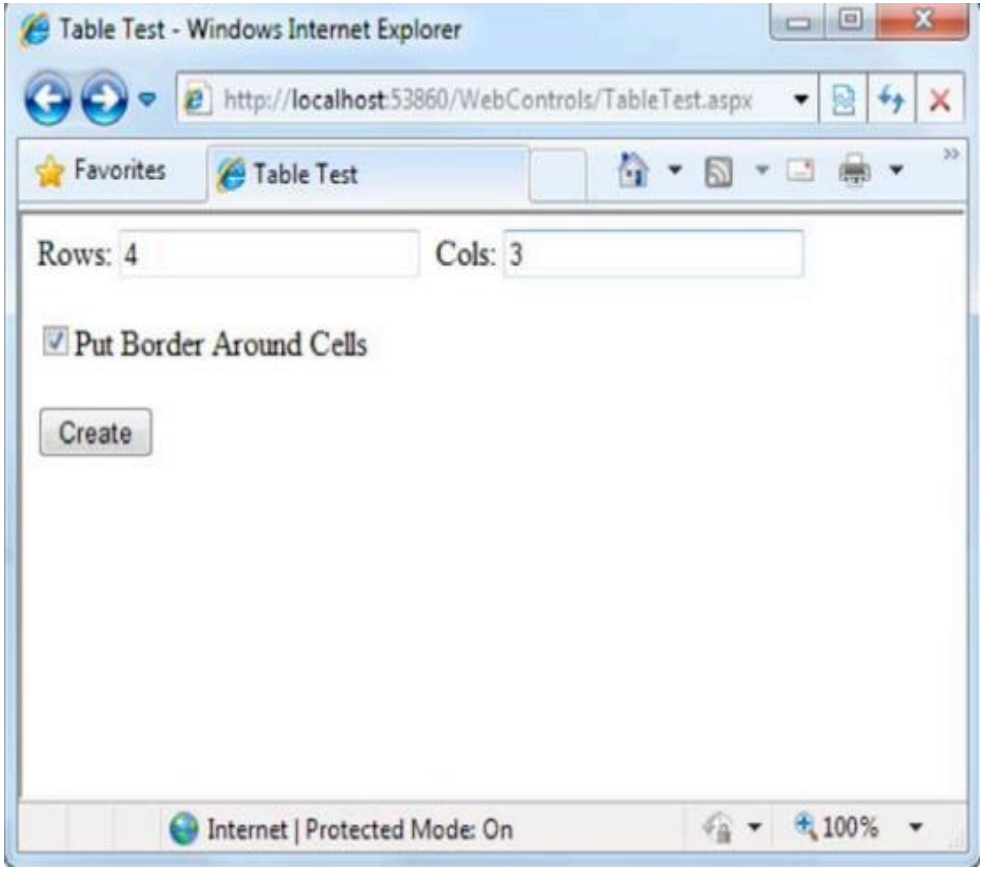


Table Controls

- The table will be created as a server-side control object.
- The table uses two events Page_Load and Button_Click.

```
public partial class TableTest : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        //Configure the table's appearance. This could be also performed in the .aspx file
        // or in the cmdCreate_Click event handler.
        tbl.BorderStyle = Border.Style.Inset;
        tbl.BorderWidth = Unit.Pixel(1);
    }
}
```

- When the button is clicked, it dynamically creates the required TableRow and TableCell objects in a loop.

```
protected void  cmdCreate_Click(object sender, EventArgs e)
{
    //Remove all the current rows and cells
    tbl.Controls.Clear();
```

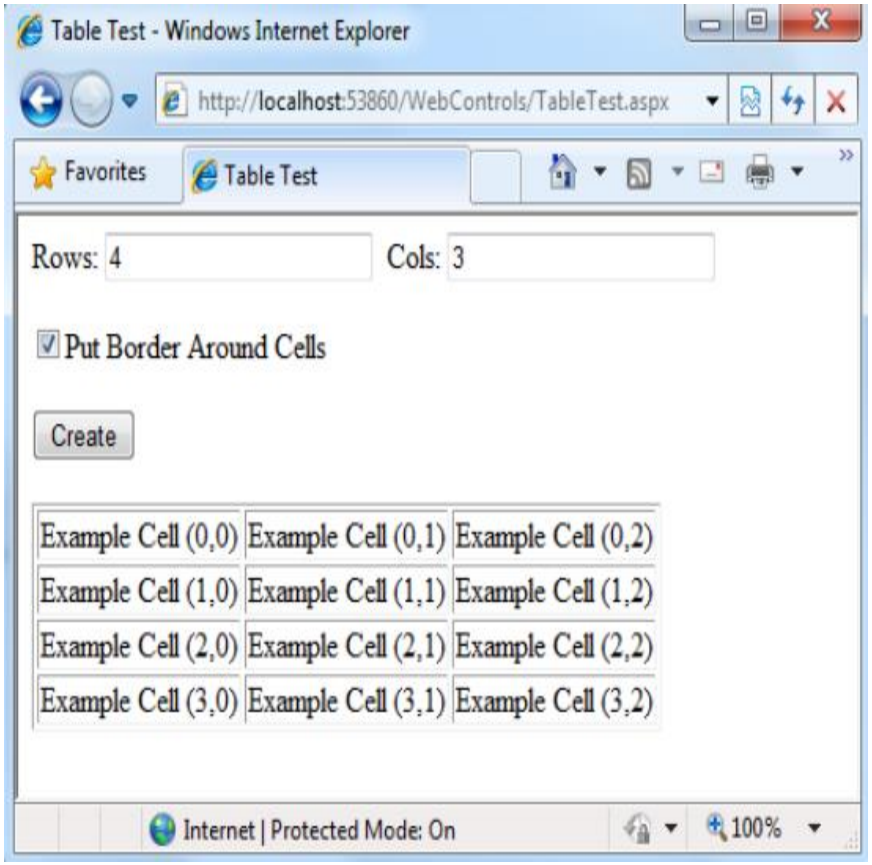
Table Controls

```
int rows = Int32.Parse(txtRows.Text);
int cols = Int32.Parse(txtCols.Text);
for(int row=0; row< rows; row++)
{
    //Create a new TableRow object
    TableRow rowNew = new TableRow();

    //Put the TableRow in the Table
    tbl.Controls.Add(rowNew);
    for(int col=0; col<cols; col++)
    {
        //Create a new TableCell object
        TableCell cellNew = new TableCell();
        cellNew.Text = "Example Cell (" + row.ToString() + ",";
        cellNew.Text += col.ToString() + ")";

        if(chkBorder.Checked)
        {
            cellNew.BorderStyle = BorderStyle.Inset;
            cellNew.BorderWidth = Unit.Pixel(1);
        }

        //Put TableCell in the TableRow.
        rowNew.Control.Add(cellNew);
    }
}
```



END OF LECTURE