

.NET PROGRAMMING USING C#

UNIT II

Jalpa A. Poriya

Index

- [Controls in ASP](#)
- [Common Properties for asp controls](#)
- [TextBox Control](#)
- [Button Control](#)
- [LinkButton](#)
- [ImageButton Control](#)
- [CheckBox Control](#)
- [Radio Button](#)
- [DropDownList Control](#)
- [CheckBoxList Control](#)
- [ListBox](#)
- [FileUpload Control](#)

Controls in ASP

- **Client Side Controls**
 - Client Controls are bound to client side javascript data and create their Html dynamically on the client side.
- **Server Side Controls**
 - The Microsoft.NET Framework provides a rich set of server-side controls for developing Web applications. You can add these controls to WebForms pages just as you add Windows controls to a form. Server-side controls are often called server controls or Web Forms controls.
 - There are four types of Server controls: HTML server controls. Web server controls, validation control, and user controls.

Client Side Controls

[HTML Controls]

- HTML controls are client side control.
- Example

```
<input type="text" name="txt_name">
```

```
<input type="button" name="button1" value="Click">
```

- Note: If we add `runat="server"` attribute in HTML tag it will convert to server side control

ServerSide Control

[Standard ASP Control]

- **Button** : It displays text within a rectangular area.
- **Link Button** : It displays text that looks like a hyperlink.
- **Image Button** : It displays an image.
- **Text Boxes** : Text box controls are typically used to accept input from the user. A text box control can accept one or more lines of text depending upon the settings of the TextMode attribute.
- **Labels** : Label controls provide an easy way to display text which can be changed from one execution of a page to the next. If you want to display text that does not change, you use the literal text.

ServerSide Control

[Standard ASP Control]

- **Check Boxes and Radio Buttons**
- A check box displays a single option that the user can either check or uncheck and radio buttons present a group of options from which the user can select just one option.
- **HyperLink Control**
- The HyperLink control is like the HTML `<a>` element.
- **Image Control**
- The image control is used for displaying images on the web page, or some alternative text, if the image is not available.

ServerSide Control

[Standard ASP Control]

- **List Controls**
- ASP.NET provides the following controls
 - Drop-down list,
 - List box,
 - Radio button list,
 - Check box list,
 - Bulleted list.
- These control let a user choose from one or more items from the list. List boxes and drop-down lists contain one or more list items.

WebControl Properties

(Common Properties for asp controls)

- **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.
- **BackColor, ForeColor, BorderColor**
- Sets the colors used for the background, foreground, and border of the and BorderColor control. In most controls, the foreground color sets the text color.
-

WebControl Properties

(Common Properties for asp controls)

- **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.

- **Enabled**

- When set to false, the control will be visible, but it will not be able to receive user input or focus.

WebControl Properties

(Common Properties for asp controls)

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.

WebControl Properties

(Common Properties for asp controls)

- **Font**
- Specifies the font used to render any text in the control.
- **Height and Width**
- Specifies the width and height of the control. For some controls, these properties will be ignored when used with older browsers. Page Provides a reference to the web page that contains this control as a `System.Web.UI.Page` object.
-

WebControl Properties

(Common Properties for asp controls)

- **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.
- **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.

TextBox Control

- The TextBox control can be used to display three different types of input fields depending on the value of its TextMode property. The **TextMode** property accepts the following values:
- Value of TextMode Property:
 - SingleLine: Displays a single-line input field.
 - MultiLine: Displays a multi-line input field.
 - Password: Displays a single-line input field in which the text is hidden.
 - Number, Color, Date, Month, Week, Range etc.

Special Properties of TextBox control

- ***TextMode***
- ***AutoPostBack***
 - Enables you to post the form containing the TextBox back to the server automatically when the content of the TextBox is changed.
- ***MaxLength***
 - Enables you to specify the maximum length of data that a user can enter in a text box (does not work when TextMode is set to Multiline).

- ***ReadOnly***

- Enables you to prevent users from changing the text in a text box.

- ***Columns***

- Enables you to specify the number of columns to display.

- ***Rows***

- Enables you to specify the number of rows to display.

- ***Wrap***

- Enables you to specify whether text word-wraps when the TextMode is set to Multiline.

- ***Methods***

- *Focus*

- Enables you to set the initial form focus to the text box.

-

- ***Events***

- *TextChanged*

- Raised on the server when the contents of the text box are changed.

Button Control

- The Button control renders a push button that you can use to submit a form to the server.

Special Properties of Button

- *CommandArgument* Enables you to specify a command argument that is passed to the Command event.
- *CommandName* Enables you to specify a command name that is passed to the Command event.
- *OnClick* Enables you to specify a client-side script that executes when the button is clicked.
- *PostBackUrl* Enables you to post a form to a particular page.

- *Text* Enables you to label the Button control.
- *UseSubmitBehavior* Enables you to use JavaScript to post a form.

Button Control

- The Button control renders a push button that you can use to submit a form to the server.

Special Properties:

- *OnClick*
 - Enables you to specify a client-side script that executes when the button is clicked.
- *PostBackUrl*
 - Enables you to post a form to a particular page.
- *Text*
 - Enables you to label the Button control.

Special Properties of Button:

- *CommandArgument*
 - Enables you to specify a command argument that is passed to the Command event.
- *CommandName*
 - Enables you to specify a command name that is passed to the Command event.

Method and Events:

- *Focus()*
 - Enables you to set the initial form focus to the Button control.
-
- Events
- *Click*
 - Raised when the Button control is clicked.
- *Command*
 - Raised when the Button control is clicked. The `CommandName` and `CommandArgument` are passed to this event.

LinkButton

- The LinkButton control, like the Button control, enables you to post a form to the server.
- Unlike a Button control, however, the LinkButton control renders a link instead of a push button.

Properties

- *Text*

Example

```
<asp:LinkButton  
ID="lnk_home"  
runat="server"  
ForeColor="White"  
onclick="inbox.aspx"  
CausesValidation="False">  
    Home  
</asp:LinkButton>
```


ImageButton Control

- The ImageButton control, like the Button and LinkButton controls, enables you to post a form to the server.
- However, the ImageButton control always displays an image.

Properties

- *AlternateText*
 - Enables you to provide alternate text for the image (required for accessibility).
- *DescriptionUrl*
 - Enables you to provide a link to a page that contains a detailed description of the image
- *GenerateEmptyAlternateText*
 - Enables you to set the Alternate Text property to an empty string.
- *ImageAlign*
 - Possible values are AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop, and Top.
- *ImageUrl*
 - Enables you to specify the URL to the image.

Example

- `<asp:ImageButton ID="ImageButton1" runat="server" ImageUrl="image1.jpg" DescriptionUrl="inbox.aspx" />`

CheckBox Control

- ***Properties:***

- *Checked*

- Enables you to get or set whether the CheckBox is checked.

- *Text*

- Enables you to provide a label for the check box.

- *AutoPostBack*

- Enables you to provide event handler for check box by default value is false

- ***Events:***

- *CheckedChanged:*

- Raised on the server when the check box is checked or unchecked.

- ***Example***

- `<asp:CheckBox ID="chk_remember" runat="server"
Text="Remember Me" AutoPostBack="True"
Oncheckedchanged="chk_CheckedChanged" />`

RadioButton Control

- You always use the RadioButton control in a *group*.
- Only one radio button in a group of RadioButton controls can be checked at a time.

• ***Properties:***

• *Checked*

- Enables you to get or set whether the RadioButton control is checked.

• *GroupName*

- Enables you to group RadioButton controls.

• *Text*

- Enables you to label the RadioButton control.

• ***Events:***

• *CheckedChanged*

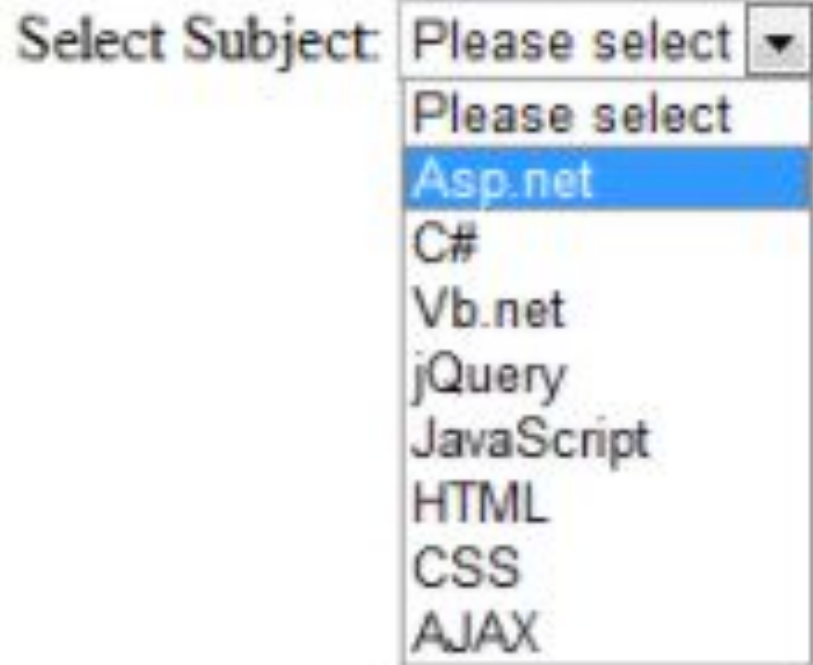
- Raised on the server when the Radio Button is checked or unchecked.

- ***Example***

- `<asp:RadioButton ID="rdoMale" runat="server" Text="Male" GroupName="Gender" />`
- `<asp:RadioButton ID="rdoFemale" runat="server" Text="Female" GroupName="Gender" />`

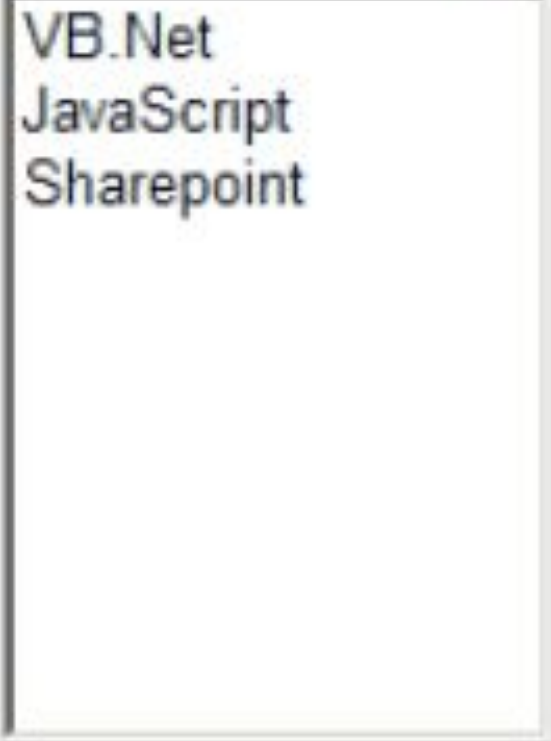
DropDown List Control

- Items
 - Add(), Insert(), Remove(), RemoveAt(), Contains(), IndexOf(), Clear()
 - Count, Selected, Text, Value
- *AutoPostBack*
- SelectedIndex
- SelectedValue
- Text
- DataSource
- DataTextField
- DataValueField



ListBox Control

- SelectionMode - Single, Multiple
- Items
 - Add(), Insert(), Remove(), RemoveAt(), Contains(), IndexOf(), Clear()
 - Count, Selected, Text, Value
- AutoPostBack
- Rows
- Selectedtem
- SelectedIndex
- SelectedValue
- DataSource
- DataTextField
- DataValueField



VB.Net
JavaScript
Sharepoint

CheckBoxList control

- CheckBoxList control is a single control that groups a collection of checkable list items, all are rendered through an individual.



A screenshot of a CheckBoxList control. It consists of a vertical container with three items. Each item has a checkbox on the left and a text label on the right. The first item is 'Red' with a checked checkbox. The second item is 'Blue' with a checked checkbox. The third item is 'Green' with an unchecked checkbox.

<input checked="" type="checkbox"/>	Red
<input checked="" type="checkbox"/>	Blue
<input type="checkbox"/>	Green

CheckBoxList Properties

- Items
- RepeatDirection [**Vertical, Horizontal**]
- RepeatLayout [**Table, Flow**]
- AutoPostBack
- RepeatColumns
- TextAlign
- SelectedItem
- SelectedValue
- SelectedIndex

FileUpload Control

- The FileUpload control allows the user to browse for and select the file to be uploaded, providing a browse button and a text box for entering the filename.

Properties

- HasFile
- FileName
- FileBytes
- FileContent - Length, CanRead, CanWrite
- PostedFile - ContentLength, FileName

- **Methods**

- SaveAs(string fileinfo)
- Focus()

SDI and MDI

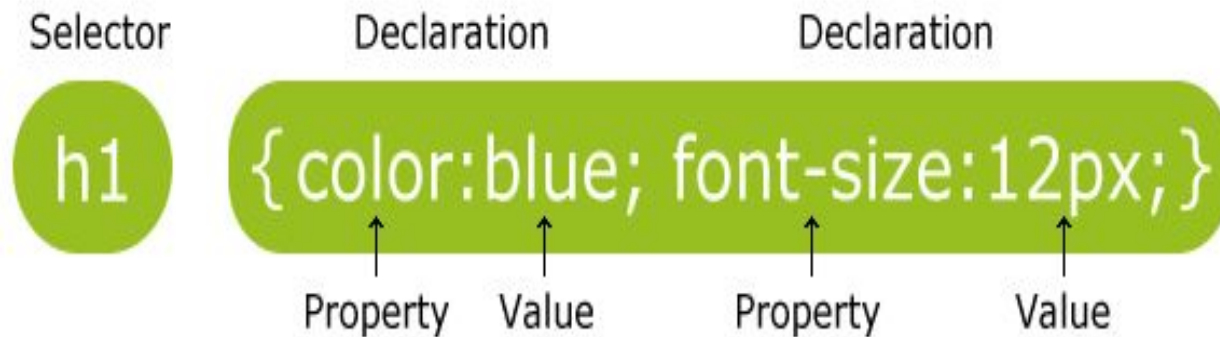
- MDI and SDI are interface designs for handling documents within a single application.
- MDI stands for “Multiple Document Interface” while SDI stands for “Single Document Interface”.
- One document per window is enforced in SDI while child windows per document are allowed in MDI.
- SDI contains one window only at a time but MDI contain multiple document at a time appeared as child window.

- You can create Multiple Document Interface in Visual Studio by following properties
- MdiParent
- MdiChildren

- Form Property – IsMDIContainer must be true if we want to create our form as MdiParent.

Introduction to CSS

- CSS Syntax and Selectors
- CSS Syntax



CSS Selectors

- CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more.
- The element Selector

```
p {  
  text-align: center;  
  color: red;  
}
```

```
h1, h2, p {  
  text-align: center;  
  color: red;  
}
```

- The id Selector
- The style rule below will be applied to the HTML element with id="para1":

```
#para1 {  
    text-align: center;  
    color: red;  
}
```

- The class Selector
- In the example below, all HTML elements with `class="center"` will be red and center-aligned:

```
.center {  
    text-align: center;  
    color: red;  
}
```

- In the example below, only <p> elements with class="center" will be center-aligned:

```
p.center {  
    text-align: center;  
    color: red;  
}
```

Master Pages

- Creating a set of controls that are common across all the web pages and attaching them to all the web pages.
- A centralized way to change the above created set of controls which will effectively change all the web pages.

- **Masterpage**

- Gives us a way to create common set of UI elements that are required on multiple pages of our website.

- **ContentPlaceholder**

- A control that should be added on the MasterPage which will reserve the area for the content pages to render their contents.

- **ContentPage**

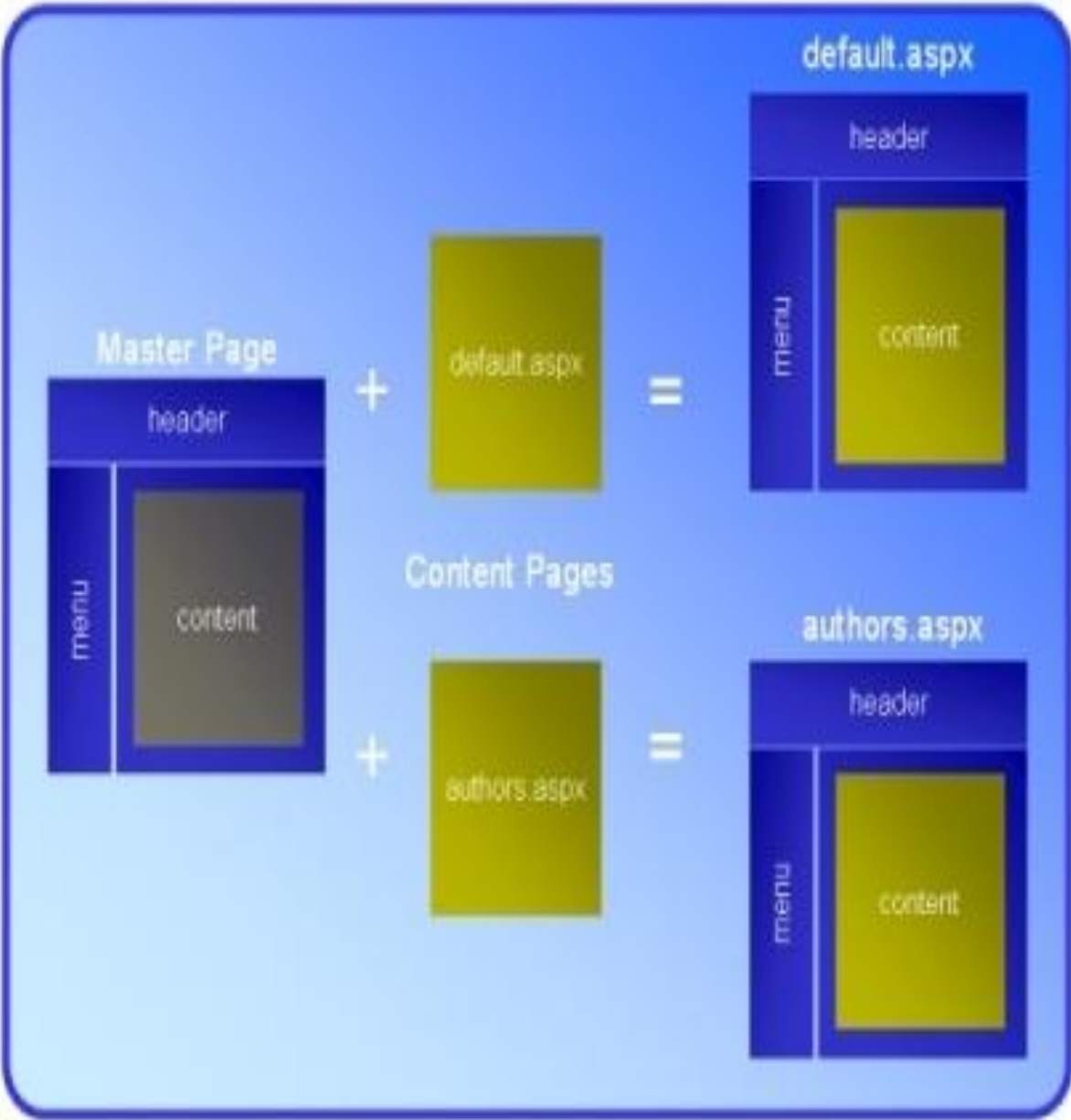
- The ASP.NET web page that will use master page to have the common UI elements displayed on rendering itself.

Advantages of master pages include the following:

- They allow you to centralize the common functionality of your pages so that you can make updates in just one place.
- They make it easy to create one set of controls and code and apply the results to a set of pages. For example, you can use controls on the master page to create a menu that applies to all pages.
- They give you fine-grained control over the layout of the final page by allowing you to control how the placeholder controls are rendered.
- They provide an object model that allows you to customize the master page from individual content pages.

Run-time Behavior of Master Pages

1. Users request a page by typing the URL of the content page.
2. When the page is fetched, the `@ Page` directive is read. If the directive references a master page, the master page is read as well. If this is the first time the pages have been requested, both pages are compiled.
3. The master page with the updated content is merged into the control tree of the content page.
4. The content of individual Content controls is merged into the corresponding ContentPlaceHolder control in the master page.
5. The resulting merged page is rendered to the browser.



Requirement of a Master Page in an Asp.NET application

- The master pages can be used to accomplish the following:
- Creating a set of controls that are common across all the web pages and attaching them to all the web pages.
- A centralized way to change the above created set of controls which will effectively change all the web pages.
- Dynamically changing the common UI elements on master page from content pages based on user preferences.

Themes and Skins

- Themes
 - Themes are like CSS styles - they both define a set of common attributes that can be applied to controls and elements on any page
- Skins
 - Themes allow the customization of any property of an Asp.net control, such as text values, icons, template layouts and so on.



Themes

- A theme decides the look and feel of the website. It is a collection of files that define the looks of a page. It can include skin files, CSS files & images.

We define themes in a special App_Themes folder.

- Inside this folder is one or more subfolders named Theme1, Theme2 etc. that define the actual themes.
- The theme property is applied late in the page's life cycle, effectively overriding any customization you may have for individual controls on your page.

There are 3 different options to apply themes to our website:

- Setting the theme at the page level
- Setting the theme at the site level
- Setting the theme programmatically at runtime

Setting the theme at the page level

- The Theme attribute is added to the page directive of the page.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="Default" Theme="Theme1"%>
```

- Setting the theme at the site level: to set the theme for the entire website you can set the theme in the web.config of the website. Open the web.config file and locate the <pages> element and add the theme attribute to it:

```
<pages theme="Theme1">
```

```
....
```

```
</pages>
```


- Setting the theme programmatically at runtime: here the theme is set at runtime through coding. It should be applied earlier in the page's life cycle ie. Page_PreInit event should be handled for setting the theme. The better option is to apply this to the Base page class of the site as every page in the site inherits from this class.

```
Page.Theme = Theme1;
```

Skin

- ASP.Net skins can only be used to apply the styles to the ASP.Net controls.so in this article let us see the procedure for using ASP.Net Skins.
- Example
- In SkinFile.Skin

- In demo.aspx file

```
<asp:Label runat="server" ForeColor="Green" SkinID="lbltxt" Text="Label"></asp:Label>  
<asp:TextBox runat="server" ForeColor="DarkBlue" SkinID="txt"></asp:TextBox>  
<asp:Button runat="server" Text="Button" ForeColor="Chocolate" SkinID="btn" />
```

```
<asp:Label ID="Label1" runat="server" SkinID="lbltxt" Text="Label"></asp:Label>  
<asp:TextBox ID="TextBox1" runat="server" SkinID="txt"></asp:TextBox>  
<asp:Button ID="Button1" runat="server" SkinID="btn" Text="Button"/>
```