

# Chapter -3 ASP.Net Controls

## 3.1 Web Server Controls

(Button, Check Box, Check Box List, Drop Down List, Hyperlink, Image, Image Button, Label, Link Button, List Box, List Item, Panel, Place Holder, Radio Button, Radio Button List, Textbox)

### **ASP.NET Server Controls:-**

**Server Controls are tags that are understood by the server.**

➤ There are three kinds of server controls:

- **HTML Server controls** – Traditional HTML tags.

These are classes that wrap the standard HTML elements. Apart from this attribute, the declaration for an HTML server control remains the same. Two examples include Html Anchor (for the <a> tag) and Html Select (for the <select> tag).

- **Web Server Controls** – New ASP.NET tags.

These classes duplicate the functionalities of the basic HTML elements but have a more consistent and meaningful set of properties and methods that make it easier for the developer to declare and access them. Example Hyperlink, List Box, and Button Controls. In visual studio, you find the basic web forms controls in the standard tab of the Toolbox.

- **Validation Server Controls** – For input validation

These set of controls allow you to easily validate an associated input control against several standard or user – defined rules.

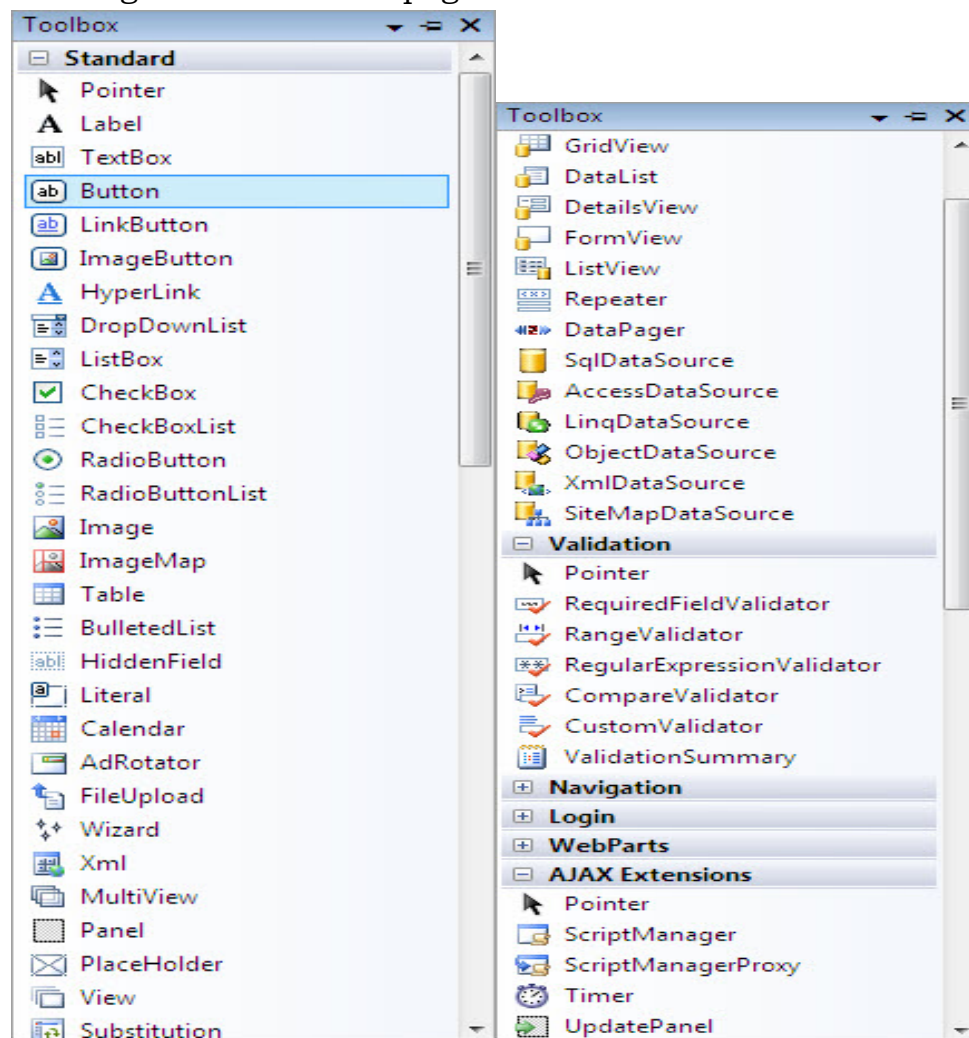
Example you can specify that the input can't be empty, that it must be greater than a certain value, and so on. If validation fails, you can prevent page processing or allow these controls to show in line error messages in the page. In visual studio, you find the basic web forms controls in the standard tab of the Toolbox.

### **Advantages of Web Server Controls**

➤ Automatic browser detection. The controls can detect browser

capabilities and render appropriate markup.

- HTML Server Controls are HTML elements containing attributes that make them programmable in server code.
- HTML Server Controls run at side. HTML Server Controls have property, methods, and even expose or raise server events during the processing of the ASP.NET page.



### Difference between HTML Server Control and Web server Control

HTML Server Control	Web Server Control
HTML Server Controls are derived from <b>System.Web.UI. HtmlControls base class</b>	Web Server Controls are derived from the <b>System.Web.UI.WebControls base class</b>
HTML elements in ASP.NET files are, by default, treated as text. To make these elements programmable, add a <code>runat="server"</code> attribute to the HTML element. This attribute indicates that the element should be treated as a server control.	Like HTML server controls, Web server controls are also created on the server and they require a <code>runat="server"</code> attribute to work. However, Web server controls do not necessarily map to any existing HTML elements and they may represent more complex elements.
The HTML Server Controls follow the HTML-centric object model. Model similar to HTML	ASP .NET Server Controls have an object model different from the traditional HTML
You can easily change the code of control in HTML.	The control of the code is inbuilt with the web server controls so you have no much of direct control on these controls

## 3.2 Working with Control Properties and Events

### 3.1.1 Button Control

ASP .Net provides three types of button controls: buttons, link buttons and image buttons. As the names suggest a button displays text within a rectangular area, a link button displays text that looks like a hyperlink. And an Image Button displays an image.

- The button control renders a push button that you can use to submit a form to the server. It is possible to write an event handler to control the actions performed when the submit button is clicked.
- A button can be set as default button (Default button is a button that is Clicked when the user presses Enter key while filling a form.) by specifying Default Button property of the `<form>` tag or to the `asp:Panel` tag in which it is

placed.

eg. If we have to set a button with id btnSubmit as default button we can add following properties to the <form> element.

Syntax
<asp:Button ID="Button1" runat="server" Text="Submit" />

Property	Description
Command Name	Specify a command name that is passed to the command event.
Enabled	Disable the Button control.
OnClientClick	Specify a client side script that executes when the button is clicked.
PostBack Url	Post a form to a particular page.
Text	The text displayed by the button. This is for button and link button controls only.
Validation Group	Specifies the group of controls a button causes validation, when it posts back to the server
Event (Click)	Occurs when we click on button.
Command Argument	A string value that's passed to the Command event when a user clicks the button.
PostBack Url	The URL of the page that should be requested when the user clicks the button.

#### Example:

#### Design Code : ButtonControl.aspx page

```
<table width="30%">
  <tr>
    <td align="center" colspan="2">
      <asp:Button ID="Button1" runat="server" Text="Submit" />
    </td>
  </tr>
</table>
```

```
<td align="center" colspan="2">  
    <asp:Label ID="lblDate" runat="server"></asp:Label>  
</td>  
</tr>  
</table>
```

**VB Code :**

```
Partial Class ButtonControl Inherits System.Web.UI.Page  
Protected Sub Button1_Click(ByVal sender Object)  
    lblDate.Text=DateTime.Now.ToString()  
End Sub  
End Class
```

**Output View****Button Controls and Validation**

If a page contains ASP.NET validator controls, by default, clicking a button control causes the validator control to perform its check. If client-side validation is enabled for a validator control, the page is not submitted if a validation check has failed.

The following table describes the properties supported by button controls that enable you to control the validation process more precisely.

Property	Description
Causes Validation	Specifies whether clicking the button also performs a validation check. Set this property to false to prevent a validation check.
Validation Group	Enables you to specify which validators on the page are called when the button is clicked. If no validation groups are established, a button click calls all of the validators that are on the page.

### 3.1.1 Checkbox Control

- The Checkbox control is used to display a check box.
- Checkboxes are those controls which gives us an option to select, say, Yes/No or True/False. A checkbox is clicked to select and clicked again to deselect some option. When a checkbox is selected a check (a tick mark) appears indicating a selection. The Checkbox control is based on the Textbox Base class which is based on the Control class.
- A check box displays a single option that the user can either check or uncheck
- We all know what checkboxes are-those square controls that toggle a checkmark when clicked and that can display caption text.
- Set there AutoPostBack property to True if you want to handle their events when they happen

Property	Description
Appearance	Default value is Normal. Set the value to Button if you want the CheckBox to be displayed as a Button.
<b>Syntax</b>	
<asp:CheckBox ID="CheckBox1" runat="server" Text="B.S Patel Polytechnic"/>	

AutoPostBack	Specifies whether the form should be posted immediately after the Checked property has changed or not. Default is false
InputAttributes	Attribute names and values used for the Input element for the CheckBox control
LabelAttributes	Attribute names and values used for the Label element for the CheckBox control
Text	The text displayed next to the check box or radio button.
TextAlign	On which side of the check box the text should appear (right or left)
OnCheckedChanged	The name of the function to be executed when the Checked property has changed
CheckAlign:	Used to set the alignment for the CheckBox from a predefined list.
Checked	Specifies whether it is selected or not, default is false.
CheckState	Default value is Unchecked. Set it to True if you want a check to appear. When set to Indeterminate it displays a check in gray background.

**Events:**

Events	Description
AppearanceChanged	Triggered when the Appearance property is changed.
CheckedChanged	Triggered when the Checked property is changed.
CheckStateChanged	Triggered when the CheckState property is changed.

**Example****Design Code :** CheckBox.aspx page

```
<table>
  <tr>
    <td> Select CheckBox : </td>
  </tr>
```

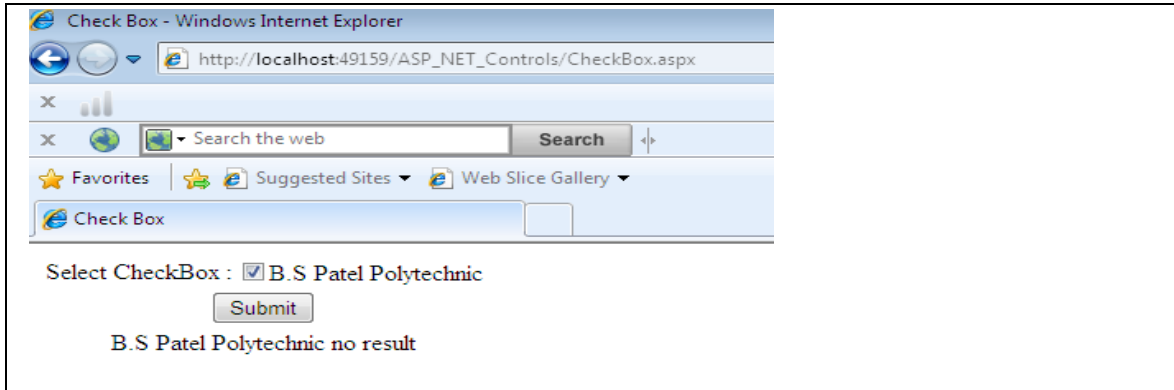
```
        <asp:CheckBox ID="CheckBox1" runat="server"
            Text="B.S Patel Polytechnic"/>
    </td>
</tr>
<tr>
    <td align="center" colspan="2">
        <asp:Button ID="Button1" runat="server"
            Text="Submit" />
    </td>
</tr>
<tr>
    <td align="center" colspan="2">
        <asp:Literal ID="ltlResult" runat="server">
            </asp:Literal>
        <asp:Literal ID="Literal1" runat="server"></asp:Literal>
    </td>
</tr>
</table>
```

**VB Code**

```
Partial Class ASPButton Inherits System.Web.UI.Page
    Protected Sub submit(ByVal sender As Object, ByVal e As System.EventArgs)
        Handles Button1.Click
        If CheckBox1.Checked Then
            ltlResult.Text = CheckBox1.Text
        Else
            Literal1.Text = "No Result"
        End If
    End Sub
End Class
```

**Output View**





### 3.1.2 Checkbox List Control

- User can select more than one choice from checkbox list.
- The Checkbox List control is used to create a multi-selection check box group. It can be used on many applications ranging from job applications to surveys, the Checkbox List control is a flexible tool that can be used on many web sites.
- The Checkbox List control is used to create a multi-selection check box group.
- A check box list presents a list of independent options. These controls contain a collection of List Item objects that could be referred to through the Items property of the control.

#### Syntax

```
<asp:CheckBoxList ID="CheckBoxList1" runat="server"> </asp:CheckBoxList>
```

Property	Description
Horizontal Scrollbar	Property used to Get or set if a checked list box should display a horizontal scroll bar.
Item	Create the list of choices of Checkbox List
Item Height	Property returns the height of an item.
BackColor	Represent the background color of Checkbox List
ForeColor	Represent the font color of Checkbox List
ID	Represent the identity of Checkbox List
DataTextField	Name of the source field to supply the text of the items.

DataValueField	Name of the source field to supply the Value of the items.
Selected Index	Property used to set or get the selected item in the control.
SelectedItem	Property used to get or set the selected item.

**Events:**

Events	Description
(SelectedIndex Changed)	When users select any check box in the list. You can specify this option by setting the AutoPostBack property to true.
ItemCheck	Triggered when the item checked state changes.

**Example****Design Code :** CheckBoxList.aspx Page

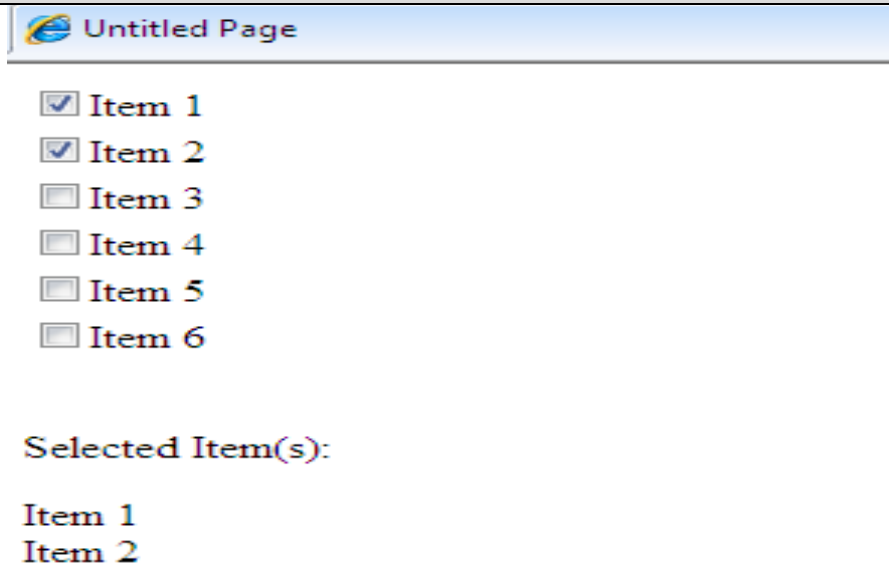
```

<html>
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:CheckBoxList ID="check1" AutoPostBack="True" TextAlign="Right"
      OnSelectedIndexChanged="check1_SelectedIndexChanged" runat="server">
      <asp:ListItem>Item 1</asp:ListItem>
      <asp:ListItem>Item 2</asp:ListItem>
      <asp:ListItem>Item 3</asp:ListItem>
      <asp:ListItem>Item 4</asp:ListItem>
      <asp:ListItem>Item 5</asp:ListItem>
      <asp:ListItem>Item 6</asp:ListItem>
    </asp:CheckBoxList><br />
    <asp:label id="lblmess" runat="server"/>
  </form>
</body>
</html>

```

**VB Code**

```
Protected Sub check1_SelectedIndexChanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles check1.SelectedIndexChanged
    Dim i As Integer
    lblmess.Text = "<p>Selected Item(s):</p>"
    For i = 0 To check1.Items.Count - 1
        If check1.Items(i).Selected Then
            lblmess.Text += check1.Items(i).Text + "<br />"
        End If
    Next
End Sub
```

**Output View****3.1.3 Dropdown List Control**

- The Dropdown List control is used to create a drop-down list.
- Drop-down list contain one or more list items. Dropdown List allow user to select one item from a list using dropdown menu.
- Dropdown List allow user to extract item from database.

- Dropdown List to create a single-selection Dropdown List control.
- You can use the control Selected Item and Selected Index properties to work with the selection.

**Syntax**

```
<asp:DropDownList ID="DropDownList1" runat="server">  
  
</asp:DropDownList>
```

Property	Description
SelectedIndex	The index of a selected item
SelectedValue	Get the value of the Selected item from the dropdown box.
SelectedItem	Gets the selected item from the list.
Items	Gets the collection of items from the dropdown box.
AutoPostBack	true or false. If true, the form is automatically posted back to the server when user changes the dropdown list selection. It will also fire OnSelectedIndexChanged method.
OnSelectedIndexChanged	The name of the function to be executed when the index of the selected item has changed
ForeColor	Represent the font color of DropDownList
ID	Represent the identity of DropDownList
DataTextField	Name of the source field to supply the text of the items.
DataValueField	Name of the source field to supply the Value of the items.

Events	Description
(SelectedIndex Changed)	Fires when the selected index has been changed. You can specify this option by setting the AutoPostBack property to true.

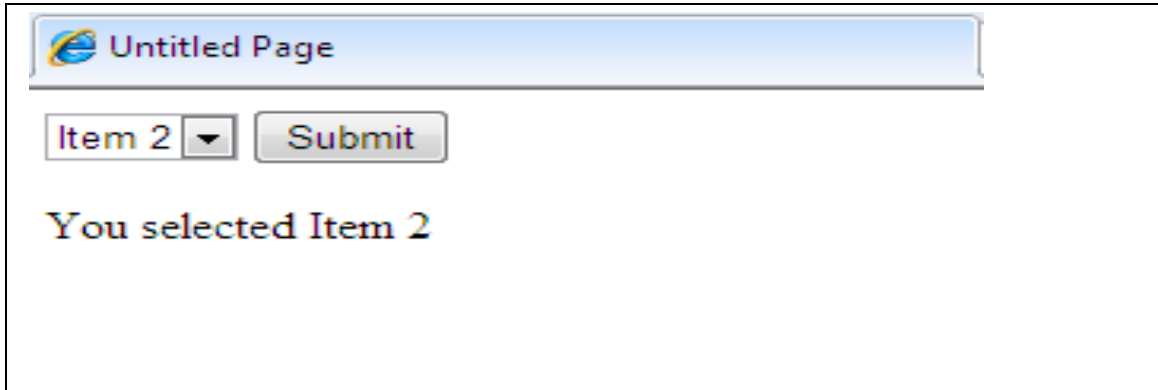
**Example****Design Code :** DropDownList.aspx Page

```
<html>
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:DropDownList ID="drop1" runat="server">
            <asp:ListItem>Item 1</asp:ListItem>
            <asp:ListItem>Item 2</asp:ListItem>
            <asp:ListItem>Item 3</asp:ListItem>
            <asp:ListItem>Item 4</asp:ListItem>
            <asp:ListItem>Item 5</asp:ListItem>
            <asp:ListItem>Item 6</asp:ListItem>
        </asp:DropDownList>
        <asp:Button ID="Button1" Text="Submit" OnClick="Button1_Click"
runat="server" />
        <p> <asp:Label ID="mess" runat="server" /></p>
    </form>
</body>
</html>
```

**VB Code**

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
    mess.Text = "You selected " & drop1.SelectedItem.Text
End Sub
```

**Output View**



### 3.1.4 Hyperlink Control

- The Hyperlink control is used to create a hyperlink.
- A Link Label control is a label control that can display a hyperlink. A Link Label control is inherited from the Label class so it has all the functionality provided by the Windows Forms Label control. Link Label control does not participate in user input or capture mouse or keyboard events.
- The Hyperlink control is like the HTML <a> element.
- The Hyperlink control to create a link to another web page, that can be a page in your web application, or a page anywhere on the world wide web.


#### Syntax

```
<asp:HyperLink ID="HyperLink1" runat="server">HyperLink</asp:HyperLink>
```

Property	Description
ImageUrl	The URL of the image to display for the link
NavigateUrl	The target URL of the link
Runat	Specifies that the control is a server control. Must be set to "server"
Target	The target frame of the URL
Background	This property is used for background color.
Text	The text to display for the link

**Example****Design Code :** Hyperlink.aspx Page

```
<html>
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:HyperLink ID="HyperLink1" ImageUrl="~/Images/smiley.gif"
      NavigateUrl="http://www.Bspp.com" Text="Visit Bspp!" Target="_blank"
      runat="server" />
  </form>
</body>
</html>
```

**Output View** Untitled Page

- The text in the HyperLink control is specified with the text property. You can

also display an image as specified by the ImageUrl property.

- You set the Url that the link navigates to with the NavigateUrl property.
- By default, when you click a HyperLink control, the linked to content appears in a new browser window. You can set the Target property to the name of a window or frame, or one of these values.
- **\_blank**-display the linked content in a new window without frames.
- **\_parent**-display the linked content in the immediate frameset parent.
- **\_self**-display the link content in the frame with focus.

### 3.1.5 Image Control

1. The Image control is used to display image in Web application.
2. The image control is used for displaying images on the web page, or some alternative text, if the image is not available.
3. You set the URL of the image with the ImageUrl property. The alignment of the image in relation to other elements on the web page is specified by setting Image Align property. You can specify the text to display in place of an image if the image is not available by setting the Alternate Text property.
4. Basic syntax for an image control

#### Syntax

```
<asp:Image ID="Image1" runat="server" />
```

Property	Description
AlternateText	Alternate text to be displayed
DescriptionUrl	a link to a page that contains a detailed description of the image
ImageAlign	Alignment options for the control
ImageUrl	Path of the image to be displayed by the control



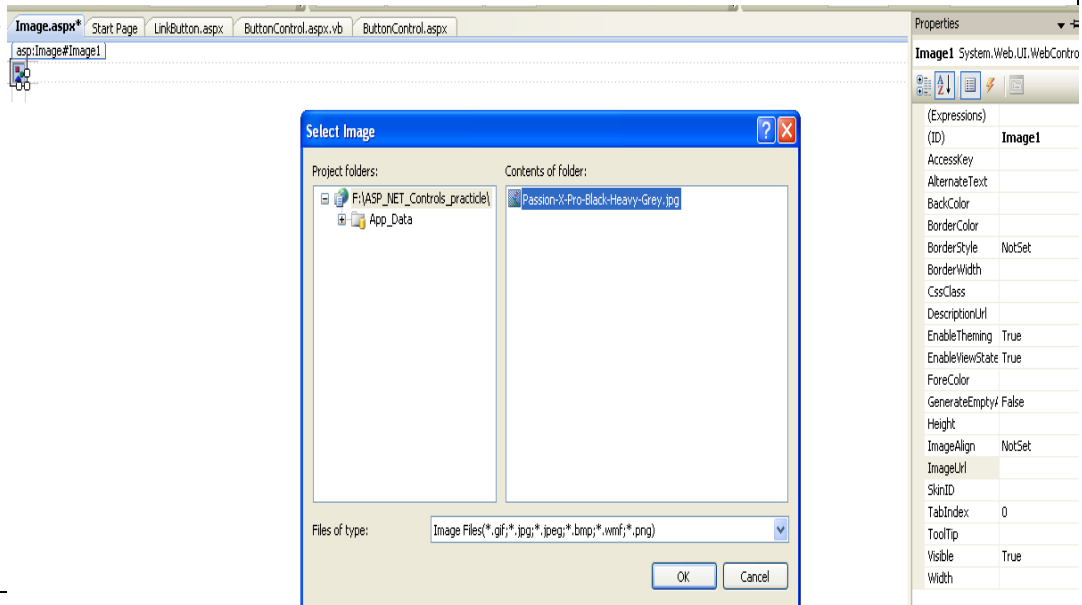
Font	Set the font property for the text associated with the control.
ImageAlign	Set the alignment of the Image control in relation to other element on the web page.

**Example****Design Code : Image.aspx Page**

```

<html>
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:Image ID="Image1" runat="server" ImageUrl="~/Penguins.jpg"
      Height="50%" Width="50%"/>
  </form>
</body>
</html>

```



**Output View**

URL: <http://localhost:49190/pr1/Default.aspx>

**3.1.6 Label**

- To display the static text on web page that user cannot change.
- Whenever you need to modify the text display in a page dynamically, you can use the label 1 control.
- Any text that you place before the opening and closing tags gets assigned to the text property.
- The label control supports several properties you can use to format the text display by the label.
- You can change the display text with the text property in code. Web server labels become `<span>` elements that enclose text.

**Syntax**

```
<span id="label1">label</span>
```

Property	Description
Text	Represent the caption of label.
BackColor	Represent the background color of label.
BorderColor	Enables you to set the color of a border rendered around the label.
BorderStyle	Enables you to display a border around the label. Possible values are NotSet, None, Dotted, Dashed, Solid, Double, Groove, Ridge, Inset, and Outset.
BorderWidth	Enables you to set the size of a border rendered around the label.
ForeColor	Represent the font color of label.
Visible	Indicate that label is visible or not.
ID	Represent the Indentify of label.
CssClass	Enables you to associate a cascading style sheet class with the label.
Style	Assign style attribute to the label.
ToolTip	Set a label's title attribute.
<b>Example</b>	
<b>Design Code :</b> Label.aspx Page	
<pre>&lt;html&gt; &lt;head runat="server"&gt;   &lt;title&gt;&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;form id="form1" runat="server"&gt;     &lt;asp:Label ID="Label1" runat="server" Text="Jatin P.Patel"&gt;&lt;/asp:Label&gt;   &lt;/form&gt; &lt;/body&gt; &lt;/html&gt;</pre>	
<b>Output View</b>	

URL: <http://localhost:49190/pr1/Default.aspx>

Jatin P. Patel

### 3.1.7 The Link Button Control

- The link button control, like the Button control, enables you to post a form to the server, unlike a button control, however, the Link Button control renders a link instead of a push button.
- Link buttons, which look just like hyperlinks, but act like buttons-with both Click and Command events. When the corresponding hyperlink is clicked, you can take some action in code, not just have browser navigate to anew page.
- Link button supported with HTML <a> elements, just as Hyperlink control is, but the code for link buttons is processed back at the server.
- The Link Button control in ASP.NET is a Hyperlink style Button. The Link Button looks like a Hyperlink control but it is a Button which works as a Hyperlink. It has events like Click and Command. You can set the PostBackUrl property of Link Button. You can give the Link name of the web page which is in your website or give the link of another website in PostBackUrl property.

#### Syntax

```
<asp:LinkButton ID="LinkButton1"
runat="server">LinkButton</asp:LinkButton>
```

Property	Description
CausesValidation	Value can be set as true/false. This indicates whether validation will be performed when a button is clicked.
ValidationGroup	Gets or Sets the name of the validation group that the button belongs to. This is used to validate only a set of Form controls with a Button.
CommandName	Specify a command name that is passed to the command event.
Enabled	Disable the Button control.
OnClickClientClick	Specify a client side script that executes when the Linkbutton is clicked.
PostBackUrl	Post a form to a particular page.
Text	Label the button control.
Event (Click)	Occurs when we click on button.
<b>Example</b>	
<b>Design Code : Link.aspx Page</b>	
<pre>&lt;html&gt; &lt;head runat="server"&gt;   &lt;title&gt;Untitled Page&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;form id="form1" runat="server"&gt;     &lt;asp:LinkButton ID="LinkButton1" runat="server"       PostBackUrl="~/RedioButton.aspx"&gt;Redirect another page     &lt;/asp:LinkButton&gt;   &lt;/form&gt; &lt;/body&gt; &lt;/html&gt;</pre>	

**VB Code**

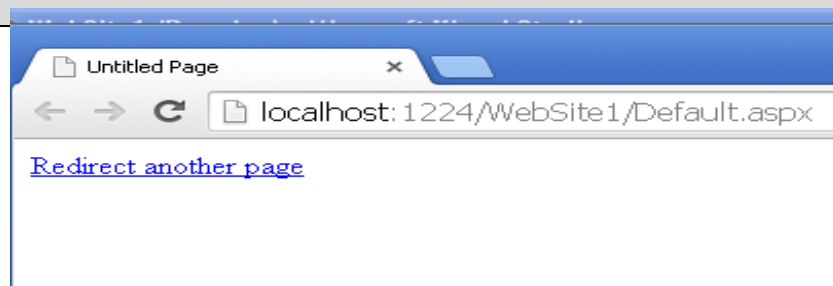
```
Partial Class ASPButton Inherits System.Web.UI.Page

    Protected Sub LinkButton1_Click(ByVal sender As Object, ByVal e As
    System.EventArgs) Handles LinkButton1.Click

        Response.Redirect("RedioButton.aspx")

    End Sub

End Class
```

**Output View :****3.1.8 ListBox Control**

- The ListBox control is used to create a single- or multi-selection drop-down list. List boxes and drop contain one or more list items. These control let a user choose from one or more items from the list.
- List boxes to create a control that single or multiple selection of item.
- You use the row property to specify the height of the control.
- The ListBox control is used to display a list of items to the user that the user can select by clicking. A ListBox control can provide single or multiple selections using the SelectionMode property.

**Syntax**

```
<asp:ListBox ID="ListBox1" runat="server"> </asp:ListBox>
```

Property	Description
Item	Create the list of choices of CheckBoxLayout
BackColor	Represent the background color of CheckBoxLayout
ForeColor	Represent the font color of CheckBoxLayout
ID	Represent the identity of CheckBoxLayout
DataSource	A data source that provides data for populating the list control.
DataMember	A string that specifies a particular table in the DataSource.
DataTextField	Name of the source field to supply the text of the items.
DataValueField	Name of the source field to supply the Value of the items.
Event (SelectedIndex Changed)	Fires when the selected index has been changed. You can specify this option by setting the AutoPostBack property to true.

**Example****Design Code :** Listbox.aspx Page

```
<table border="0">
  <tr>
    <td>
      <asp:ListBox ID="ListBox1" runat="server" Rows="10"
        Width="100" SelectionMode="Multiple">
        <asp:ListItem>James</asp:ListItem>
        <asp:ListItem>Kumar</asp:ListItem>
        <asp:ListItem>Ravi</asp:ListItem>
        <asp:ListItem>Reshmi</asp:ListItem>
        <asp:ListItem>Rachel</asp:ListItem>
```

```

                                <asp:ListItem>Arjun</asp:ListItem>
                            </asp:ListBox>
                        </td>
                        <td valign="middle">
                            <asp:Button ID="AddButton" runat="server" Text=">>" />
<br />
                            <asp:Button ID="RemoveButton" runat="server" Text="<<"
/>
                        </td>
                        <td>
                            <asp:ListBox ID="ListBox2" runat="server" Rows="10"
Width="100" SelectionMode="Multiple">
                            </asp:ListBox>
                        </td>
                    </tr>
                    <tr>
                        <td colspan="3" align="center">
                            <asp:Button ID="AddAllButton" runat="server" Text="Add All" />
                            <asp:Button ID="RemoveAllButton" runat="server" Text="Remove" />
                        </td>
                    </tr>
                </table>

```

**VB Code**

```

Imports System.Collections.Generic
Partial Class _Default Inherits System.Web.UI.Page
    Protected Sub AddButton_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles AddButton.Click
        For Each item As ListItem In ListBox1.Items
            If item.Selected And Not ListBox2.Items.Contains(item) Then
                Dim newItem As New ListItem(item.Text, item.Value)
                ListBox2.Items.Add(newItem)
            End If
        Next
    End Sub

```



```
End Sub
Protected Sub RemoveButton_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles RemoveButton.Click
    Dim itemsToRemove As New List(Of ListItem)
    For Each item As ListItem In ListBox2.Items
        If item.Selected Then
            itemsToRemove.Add(item)
        End If
    Next
    For Each item As ListItem In itemsToRemove
        ListBox2.Items.Remove(item)
    Next
End Sub
Protected Sub AddAllButton_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles AddAllButton.Click
    For Each item As ListItem In ListBox1.Items
        If Not ListBox2.Items.Contains(item) Then
            Dim newItem As New ListItem(item.Text, item.Value)
            ListBox2.Items.Add(newItem)
        End If
    Next
End Sub
Protected Sub RemoveAllButton_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles RemoveAllButton.Click
    ListBox2.Items.Clear()
End Sub
End Class
```

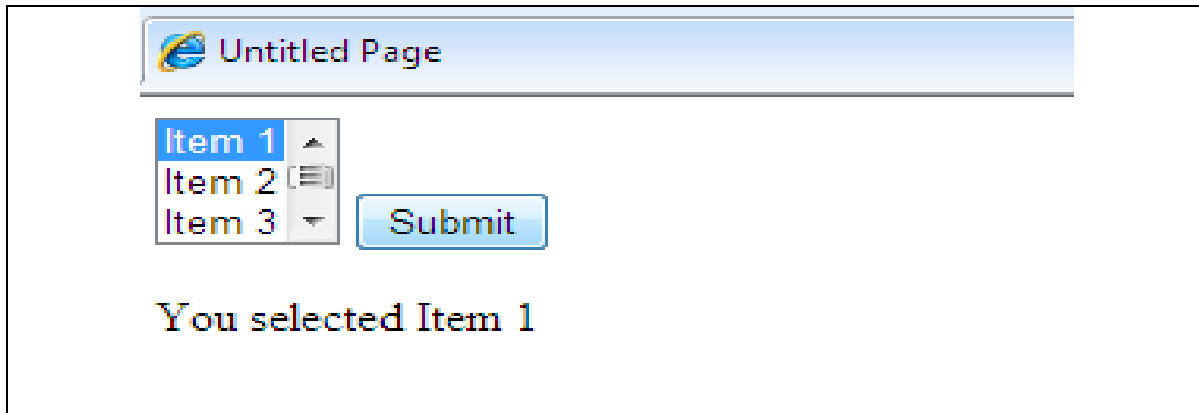
**Example****Design Code :** Link.aspx Page

```
<html>
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ListBox ID="drop1" Rows="3" runat="server">
      <asp:ListItem Selected="true">Item 1</asp:ListItem>
      <asp:ListItem>Item 2</asp:ListItem>
      <asp:ListItem>Item 3</asp:ListItem>
      <asp:ListItem>Item 4</asp:ListItem>
      <asp:ListItem>Item 5</asp:ListItem>
      <asp:ListItem>Item 6</asp:ListItem>
    </asp:ListBox>
    <asp:Button ID="Button1" Text="Submit" OnClick="Button1_Click"
      runat="server" />
    <p> <asp:Label ID="mess" runat="server" /></p>
  </form>
</body>
</html>
```

**VB Code**

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
    mess.Text = "You selected " & drop1.SelectedItem.Text
End Sub
```

**Output View**



### 3.1.9 Panel Control

- The Panel control is used as a container for other controls.
- The Panel control works as a container for other controls on the page. It controls the appearance and visibility of the controls it contains. It also allows generating controls programmatically.
- They're useful when you want to hide a group of control at once, or when you want you add controls to a web page in code.
- The Panel control is a container control that is used to host a group of similar child controls. One of the major uses I found for a Panel control when you have to show and hide a group of controls. Instead of show and hide individual controls, you can simply hide and show a single Panel and all child controls.

#### Syntax

```
<asp:Panel ID= "Panel1" runat = "server"></asp:Panel>
```

Property	Description
BackImageUrl	Specifies a URL to an image file to display as a background for this control
DefaultButton	Specifies the ID of the default button in the Panel
Direction	Specifies the content display direction of the Panel

GroupingText	Specifies the caption for the group of controls in the Panel
HorizontalAlign	Specifies the horizontal alignment of the content
Runat	Specifies that the control is a server control. Must be set to "server"
ScrollBars	Specifies the position and visibility of scroll bars in the Panel

**Example****Design Code :** Panel.aspx Page

```

<html>
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:Panel ID="panel1" runat="server" BackColor="#ff0000" Height="100px"
        Width="100px">
            Hello World!
        </asp:Panel>
        <asp:CheckBox ID="check1" Text="Hide Panel control" runat="server" />
        <br />      <br />
        <asp:Button ID="Button1" Text="Reload" runat="server" />
    </form>
</body>
</html>

```


**VB Code**

```

Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load

```


```
If check1.Checked Then  
    panel1.Visible = False  
Else  
    panel1.Visible = True  
End If  
End Sub
```

**Output View** Untitled Page

Hello World!

☒ Hide Panel control

Reload

 Untitled Page

☒ Hide Panel control

Reload

**Example:****Design Code :** Panel.aspx Page

```
<form id="form1" runat="server">

    <asp:Panel ID="Panel1" runat="server">

        <asp:Label ID="Label1" runat="server" Text="Panel 1 Display...">

        </asp:Label>

        <asp:Button ID="Button1" runat="server" Text="Show Panel 2" />

    </asp:Panel>

    <asp:Panel ID="Panel2" runat="server">

        <asp:Label ID="Label2" runat="server" Text="Panel 2
Display..."></asp:Label>

        <asp:Button ID="Button2" runat="server" Text="Show Panel 1" />

    </asp:Panel>

</form>
```

**VB Code :**

```
Partial Class ASPButton Inherits System.Web.UI.Page

    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click

        Panel1.Visible = False

        Panel2.Visible = True

    End Sub
```

```
Protected Sub Button2_Click(ByVal sender As Object, ByVal e As  
System.EventArgs) Handles Button2.Click
```

```
    Panel1.Visible = True
```

```
    Panel2.Visible = False
```

```
End Sub
```

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As  
System.EventArgs) Handles Me.Load
```

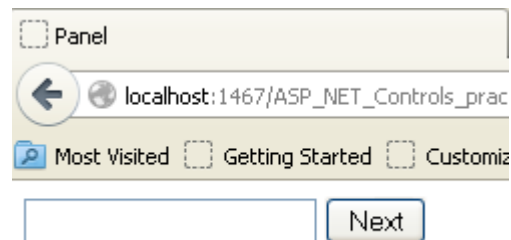
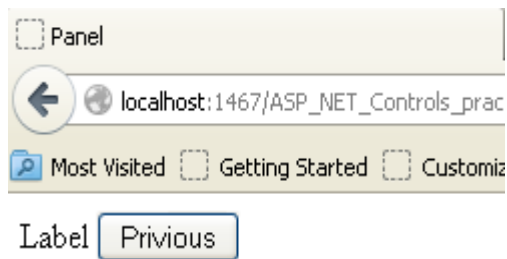
```
    Panel1.Visible = True
```

```
    Panel2.Visible = False
```

```
End Sub
```

```
End Class
```

### Output View :



### 3.1.10 The Place Holder Control

Reserves a location in the page control hierarchy for controls that are added programmatically. Use the Placeholder control as a container to store dynamically added server controls to the Web page. The Placeholder control does not produce any visible output and is only used as a container for other controls on the Web page. You can use the Control.Controls collection to add, insert, or remove a control from the Placeholder control.

**Syntax**

```
<asp:Placeholder id="Placeholder1" runat="server"/>
```

The following example demonstrates how to add Web server controls to the Placeholder control

**Example****Design Code :** Panel.aspx Page

```
<html>
<body>
  <form runat="Server">
    <h3>Placeholder Example</h3>

    <asp:Placeholder id="Placeholder1"
      runat="server"/>
  </form>
</body>
</html>
```

**VB Code**

```
<%@ Page Language="VB" AutoEventWireup="True" %>
<script runat="server">
  Sub Page_Load(Sender As Object, e As EventArgs)
    Dim myButton As HtmlButton = New HtmlButton()
```



```
myButton.InnerText = "Button 1"
Placeholder1.Controls.Add(myButton)
myButton = New HtmlButton()
myButton.InnerText = "Button 2"
Placeholder1.Controls.Add(myButton)
myButton = New HtmlButton()
myButton.InnerText = "Button 3"
Placeholder1.Controls.Add(myButton)
myButton = New HtmlButton()
myButton.InnerText = "Button 4"
Placeholder1.Controls.Add(myButton)
End Sub
</script>
```

### 3.1.11 Radio Button Control

- The Radio Button control is used to display a radio button. radio buttons present a group of options from which the user can select just one option.
- Radio button allows the user to choose only one of a predefined set of options. When a user clicks on a radio button, it becomes checked, and all other radio buttons with same group become unchecked. The buttons are grouped logically if they all share the same Group Name property.
- Radio Button web server control is used for select or deselect an item when all these Radio Buttons will be in a same group. It is more useful component in ASP.NET.
- Just like checkboxes, when only one radio button can be selected at a time.

Syntax
<asp:RadioButton ID="rdbBSPatel" runat="server" />

Property	Description
----------	-------------

AutoPostBack	A Boolean value that specifies whether the form should be posted immediately after the Checked property has changed or not. Default is false
Checked	A Boolean value that specifies whether the radio button is checked or not
Id	A unique id for the control
GroupName	The name of the group to which this radio button belongs
OnCheckedChanged	The name of the function to be executed when the Checked property has changed
CausesValidation	true/false. If true, Form is validated if Validation control has been used in the form.
ValidationGroup	Used to put a radio button under a particular validation group. It is used when you have many set of form controls and by clicking a particular button you want to validate a particular set of controls only.
Runat	Specifies that the control is a server control. Must be set to "server"
Text	The text displayed next to the check box or radio button.
TextAlign	On which side of the radio button the text should appear (right or left)

**Example**

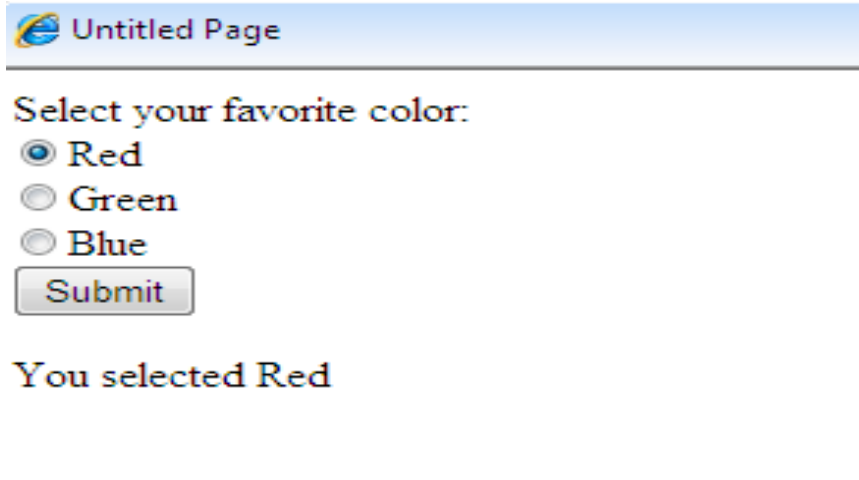
**Design Code :** Radiobutton.aspx Page

```
<html>
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        Select your favorite color:
        <br />
        <asp:RadioButton ID="red" Text="Red" Checked="True"
GroupName="colors"
runat="server" />
        <br />
        <asp:RadioButton ID="green" Text="Green" GroupName="colors"
runat="server" />        <br />
        <asp:RadioButton ID="blue" Text="Blue" GroupName="colors"
runat="server" />        <br />
        <asp:Button ID="Button1" Text="Submit" OnClick="submit"
runat="server" />
        <p> <asp:Label ID="Label1" runat="server" /> </p>
    </form>
</body>
</html>
```

**VB Code :**

```
Protected Sub submit(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Button1.Click
    If red.Checked Then
        Label1.Text = "You selected " & red.Text
    ElseIf green.Checked Then
        Label1.Text = "You selected " & green.Text
    ElseIf blue.Checked Then
        Label1.Text = "You selected " & blue.Text
    End If
```

End Sub

**Output View :**

Untitled Page

Select your favorite color:

☒ Red

☐ Green

☐ Blue

Submit

You selected Red

**Radio Button list**

- A radio button list presents a list of mutually exclusive options.
  - Radio button list control gives you an easy way to display a single-selection radio button group.
- RadioButtonList control is a single control that groups a collection of radio buttons, all are rendered through an individual `<input type=radio></input>`

**Syntax**

```
<asp:RadioButtonList ID="RadioButtonList1" runat="server">  
</asp:RadioButtonList>
```

Property	Description
AutoPostBack	true/false. If true, the form is automatically posted back to the server when user click any of the radiobutton. It will also

	fireOnSelectedIndexChanged method.
SelectedValue	Get the value first selected item.
SelectedIndex	Gets or Sets the index of the first selected item.
SelectedItem	Gets the first selected item
TextAlign	Gets or Sets the alignment of the radiobutton text.
RepeatLayout	This attribute specifies whether the table tags or the normal html flow to use while formatting the list when it is rendered. The default is Table
RepeatDirection	It specifies the direction in which the controls to be repeated. The values available are Horizontal and Vertical. Default is Vertical
RepeatColumns	It specifies the number of columns to use when repeating the controls; default is 0.
DataTextField	Name of the source field to supply the text of the items.
DataValueField	Name of the source field to supply the Value of the items.
Event (SelectedIndex Changed)	When users select any RedioButton in the list. You can specify this option by setting the AutoPostBack property to true.

**Example**

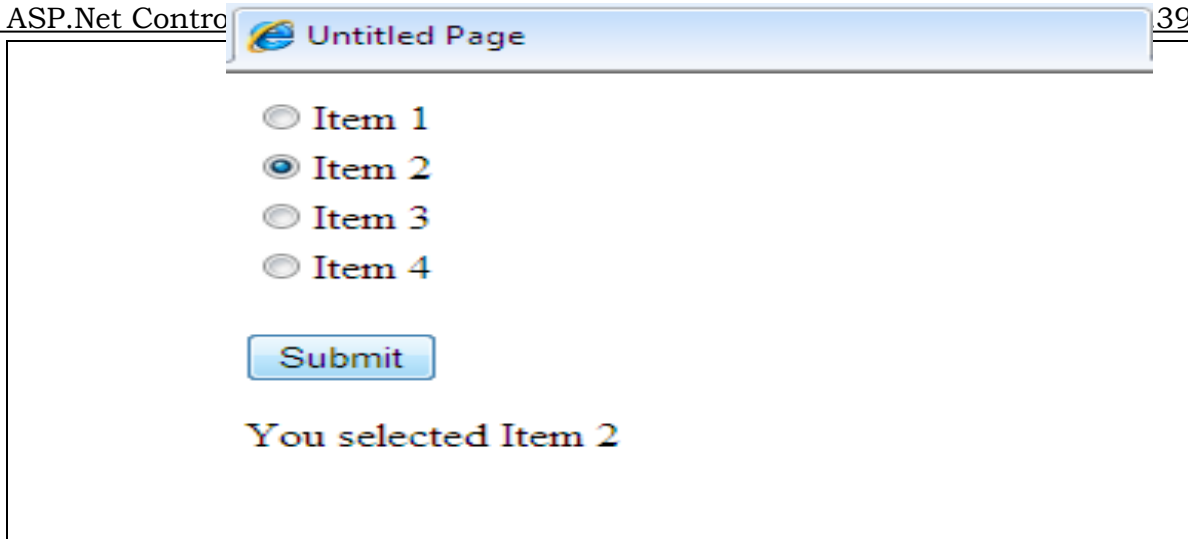
**Design Code :** Rediobuttonlist.aspx Page

```
<html>
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:RadioButtonList ID="radiolist1" runat="server">
      <asp:ListItem Selected="true">Item 1</asp:ListItem>
      <asp:ListItem>Item 2</asp:ListItem>
      <asp:ListItem>Item 3</asp:ListItem>
      <asp:ListItem>Item 4</asp:ListItem>
    </asp:RadioButtonList>
    <br />
    <asp:Button ID="Button1" Text="Submit" OnClick="Button1_Click"
      runat="server" />
    <p> <asp:Label ID="Label1" runat="server" /> </p>
  </form>
</body>
</html>
```

**VB Code :**

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
    Label1.Text = "You selected " & radiolist1.SelectedItem.Text
End Sub
```

**Output View :**



Untitled Page

☐ Item 1

☒ Item 2

☐ Item 3

☐ Item 4

Submit

You selected Item 2

### 3.1.12 Textbox

- To create a box on web form in which user can type or read standard text.
- This control is used get information like text, numbers, and dates from users in a web form.
- Text box controls are typically used to accept input from the user. A text box control can accept one or more lines to text depending upon the setting of the TextMode attribute.
- You can set the style of the text box using the TextMode property. By default, the TextMode property is set to Single Line to create a single-line HTML text field, but also can be set to multiline for a multiline text box, or password to create a password control.
- You set the display width of a text box width its Columns property.

#### Syntax

```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

Property	Description
Text	Represent the caption of label.

TextMode	Represent the mode of text entered into Textbox.3 mode: <b>SingleLine:</b> Display a single – line input field. <b>MultiLine:</b> Display a multi – line input field. <b>Password:</b> Display a single – line input field in which the text is hidden.
BackColor	Represent the background color of TextBox.
ForeColor	Represent the font color of TextBox.
ReadOnly	Set “true” if you want to set the text of textbox is can no be edit.
ID	Represent the Indentify of TextBox.
AutoPostBack	Enables you to post the form containing the textbox back to the server automatically when the content of the textbox is changed.
MaxLength	The maximum number of characters that can be entered into the text box.

**Example:****Design Code :** Textbox.aspx Page

```

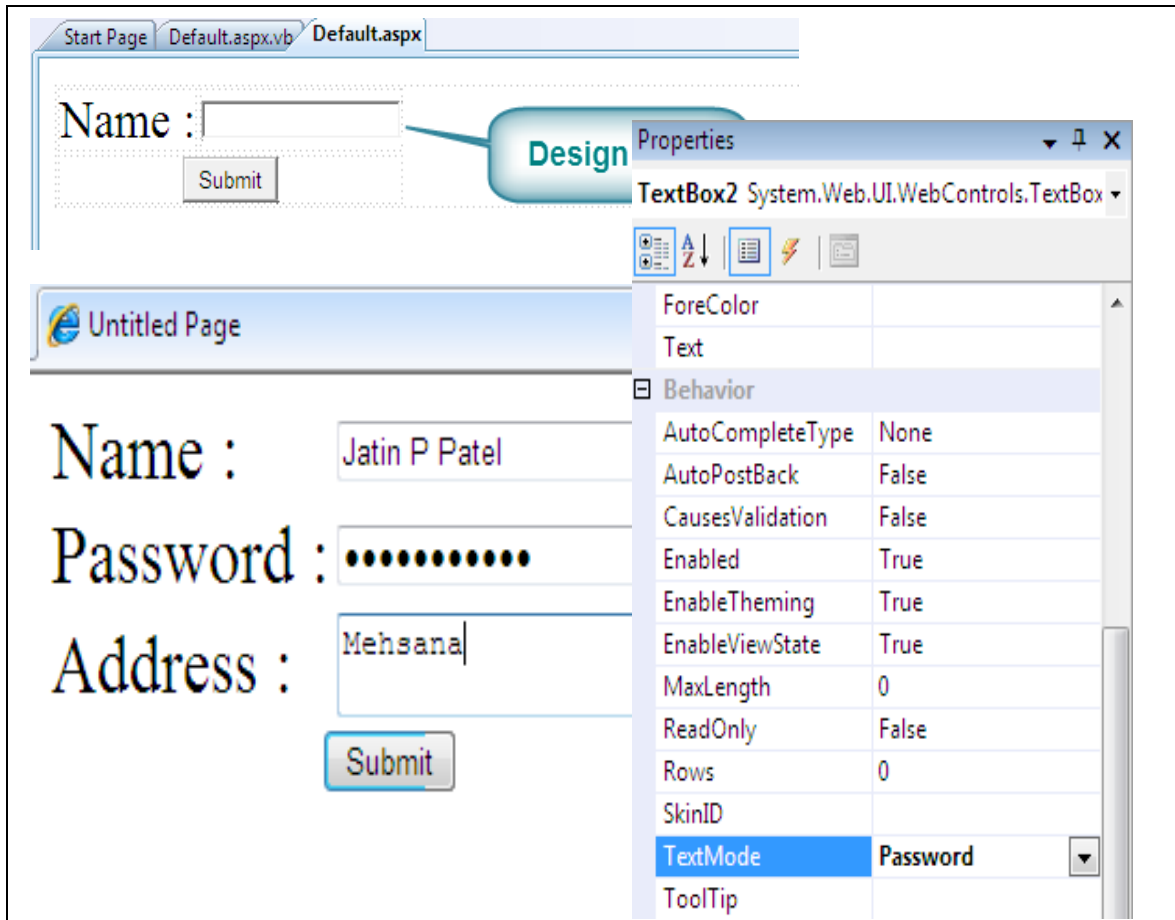
<table>
<tr>
<td>
<asp:Label ID="Label1" runat="server" Text="Name : "
Font-Size="30px"></asp:Label>
</td>
<td>
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td>
<asp:Label ID="Label2" runat="server" Text="Password : "
Font-Size="30px"></asp:Label>

```



```
</td>
<td>
    <asp:TextBox ID="TextBox2" runat="server"
    TextMode="Password"></asp:TextBox>
</td>
</tr>
<tr>
<td>
    <asp:Label ID="Label3" runat="server" Text="Address : "
    Font-Size="30px"></asp:Label>
</td>
<td>
    <asp:TextBox ID="TextBox3" runat="server"
    TextMode="MultiLine"></asp:TextBox>
</td>
</tr>
<tr>
<td align="center" colspan="2">
    <asp:Button ID="Button1" runat="server" Text="Submit" />
</td>
</tr>
</table>
```

**Output View :**



### 3.3 Validation Controls

The Web Forms framework includes a set of validation server controls that provide an easy-to-use but powerful way to check input forms for errors and, if necessary, display messages to the user. Validation controls are added to a Web Forms page like other server controls.

There are controls for specific types of validation, such as range checking or pattern matching, plus a Required Field Validator that ensures that a user does

not skip an entry field. You can attach more than one validation control to an input control. Validation should require for any web page.

Validation is process to check whether user has entered the data into control or not. And to check data entered by user are valid or not. In ASP.NET validation is performed using different validation controls. Validation controls are performing Client-Side and Server-Side validation.

**ASP.Net provides the following validation controls:**

1. Required Field Validator
2. Range Validator
3. Compare Validator
4. Regular Expression Validator
5. Custom Validator
6. Validation Summary

## **1. The RequiredFieldValidator**

- The RequiredFieldValidator control ensures that the required field is not empty. It is generally tied to a text box to force input into the text box.
- Ensures that the user does not skip an entry.
- Checks that the validated control contains a value. It cannot be empty. Can be used in conjunction with other validators on a control to trap empty values. Simply, we can use it to check if the input control has any value.
- Perform the validation of a required field of a page. It insure that field is filled up or not by user. It will not validate the entered data.

<b>Syntax</b>
<pre>&lt;asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server" ErrorMessage="Error message"&gt;&lt;/asp:RequiredFieldValidator&gt;</pre>

Property	Description
ControlToValidate	Represent the control which should be validate
ErrorMessage	Represent the Error Message to be displayed if filed is not fill up.
IsValid	Indicates whether the value of the control is valid.
ValidationGroup	Represent the name of group of validation. It should be unique for all validation control.
Text	Error text to be shown if validation fails.
ForeColor	Represent the font color of text of validation control.

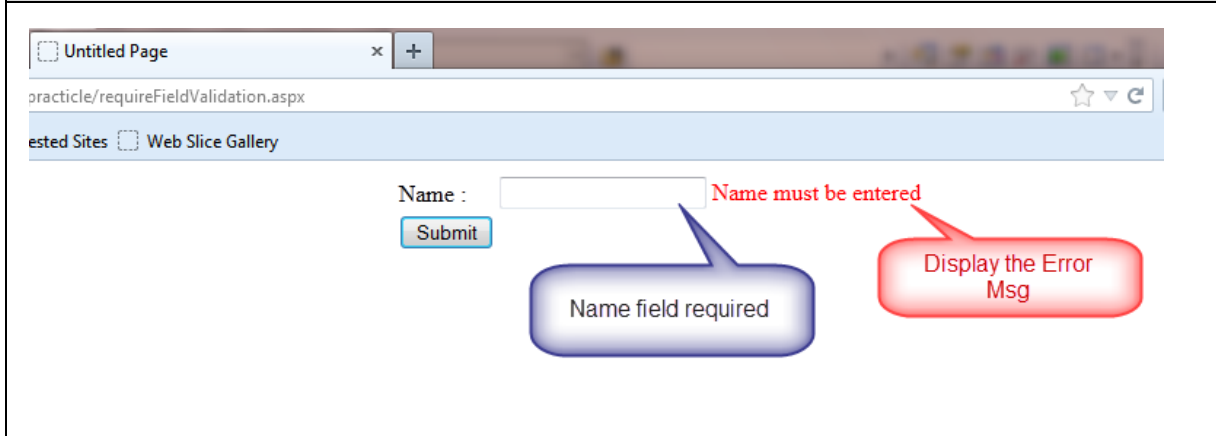
### Example

#### Design Code : RequiredFieldValidator.aspx Page

```

<table align="center">
  <tr>
    <td>
      Name :
    </td>
    <td>
      <asp:TextBox ID="txtname" runat="server"></asp:TextBox>
      <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server"
      ErrorMessage="Name must be entered"
ControlToValidate="txtname">
        </asp:RequiredFieldValidator>
      </td>
    </tr>
    <tr>
      <td align="center">
        <asp:Button ID="Button1" runat="server" Text="Submit" />
      </td>
    </tr>
  </table>

```

**Output View :****2. The CompareValidator**

- The CompareValidator control compares a value in one control with a fixed value, or, a value in another control.
- Compares a user's entry with a constant value or a property value of another control using a comparison operator (less than, equal to, greater than, and so on).
- This validation is used to compare the value of two controls, whether they are same or not.
- The CompareValidator control compares a value in one control with a fixed value, or, a value in another control.
- Can be used to compare two input values like comparing password & re-entered password in registration forms.

**Syntax**

```
<asp:CompareValidator ID="CompareValidator1" runat="server"
ErrorMessage=" ErrorMessage ">
</asp:CompareValidator>
```

It has the following specific properties:

Properties	Description
Type	it specifies the data type
Control To Compare	it specifies the value of the input control to compare with
Control To Validate	Represent the Control which should be validated.
Value To Compare	it specifies the constant value to compare with
Text	The message to display when validation fails Specifies the data type of the values to compare.
ErrorMessage	The text to display in the Validation Summary control when validation fails. Note: This text will also be displayed in the validation control if the Text property is not set
ForeColor	Represent the font color of text of validation control.

### Example

#### Design Code : CompareValidator.aspx Page

```
<table>
  <tr>
    <td>      Password :      </td>
    <td>
      <asp:TextBox ID="txtpassword" runat="server"
      TextMode="Password">
```

```
        </asp:TextBox>
    </td>
</tr>
<tr>
    <td>        Conform Password :        </td>
    <td>
        <asp:TextBox ID="txtConformPass"
runat="server" TextMode="Password">
        </asp:TextBox>
        <asp:CompareValidator ID="CompareValidator1" runat="server"
        ErrorMessage="Plese enter valid password"
        ControlToValidate="txtpassword"
ControlToCompare="txtConformPass">
        </asp:CompareValidator>
    </td>
</tr>
<tr>
    <td> <asp:Button ID="Button1" runat="server" Text="Submit" />
</td>
</tr>
</table>
```

**Output View :**

Compare Validation

Password : .....

Conform Password : ... Plese enter valid password

Submit

Password do not match !!

### 3. The Range Validator

- The RangeValidator does exactly what the name implies; it makes sure that the user input is within a specified range. You can use it to validate both numbers, strings and dates, which can make it useful in a bunch of cases.
- Checks that a user's entry is between specified lower and upper boundaries. You can check ranges within pairs of numbers, alphabetic characters, or dates. Boundaries can be expressed as constants.
- This validation control is used ensure that data entered by the user are within a specified range or not.
- The RangeValidator control verifies that the input value falls within a predetermined range.

#### Syntax

```
<asp:RangeValidator ID="RangeValidator1" runat="server" ErrorMessage="ErrorMessage"></asp:RangeValidator>
```

Properties	Description
------------	-------------



MinimumValue	it specifies the minimum value of the range
MaximumValue	it specifies the maximum value of the range
ErrorMessage	Represent the error message to be display if value is out of given range.
ForeColor	The foreground color of the control
Id	A unique id for the control

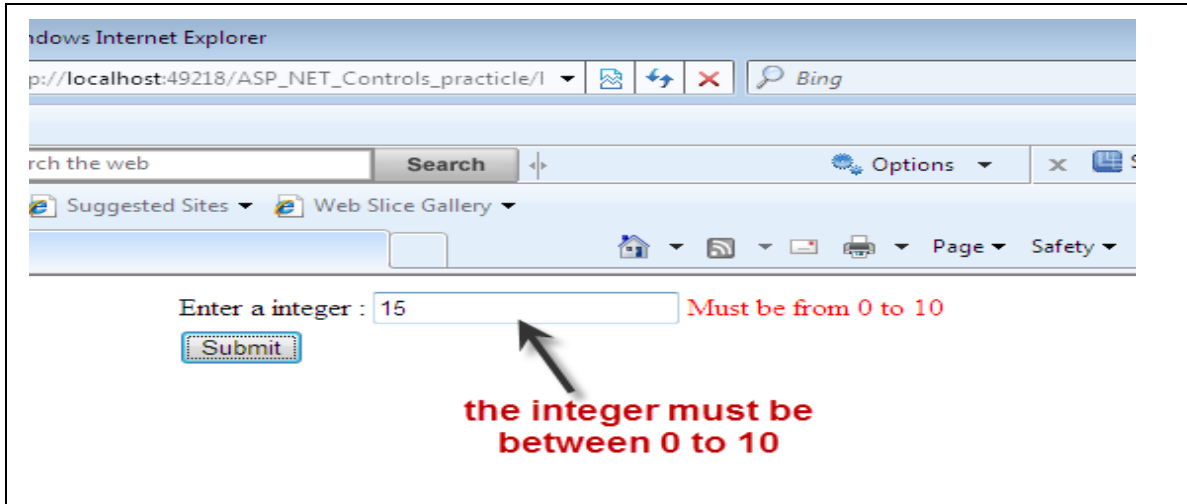
**Example****Design Code :** Range.aspx Page

```

<table align="center">
  <tr>
    <td>
      Enter a integer :
    </td>
    <td>
      <asp:TextBox ID="txtname" runat="server"></asp:TextBox>
      <asp:RangeValidator ID="RangeValidator1" runat="server"
        ControlToValidate="txtname" ErrorMessage="Must be from
          0 to 10" MinimumValue="0" MaximumValue="10"
        Type="Integer">
        </asp:RangeValidator>
      </td>
    </tr>
    <tr>
      <td>
        <asp:Button ID="Button1" runat="server" Text="Submit" />
      </td>
    </tr>
  </table>

```

**Output View :**



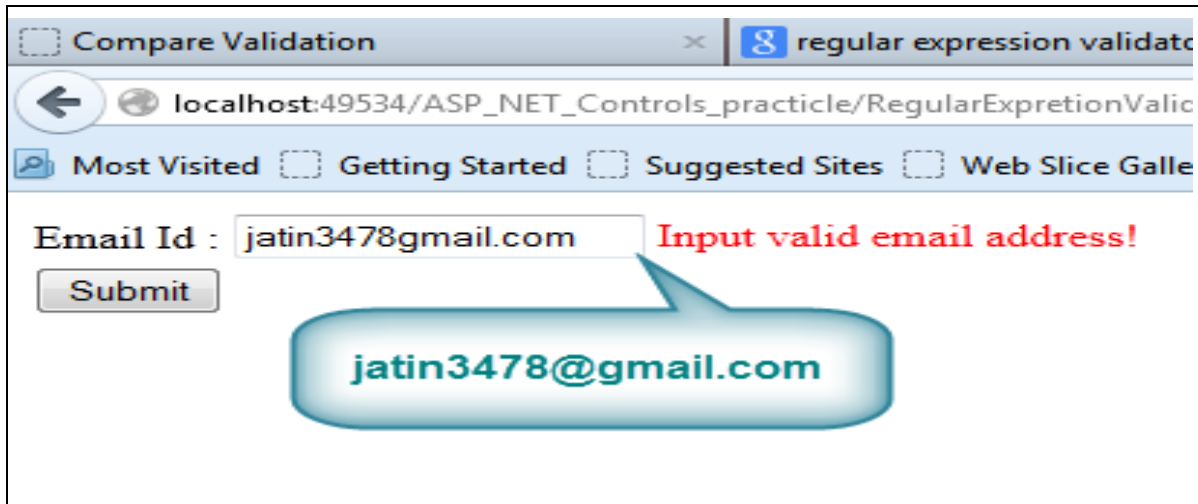
#### 4. The RegularExpressionValidator

- The RegularExpressionValidator allows validating the input text by matching against a pattern against a regular expression. The regular expression is set in the Validation Expression property.
- Checks that the entry matches a pattern defined by a regular expression. This type of validation allows you to check for predictable sequences of characters, such as those in social security numbers, e-mail addresses, telephone numbers, postal codes, and so on.
- The RegularExpressionValidator allows validating the input text by matching against a pattern against a regular expression. The regular expression is set in the Validation Expression property.
- Checks if the input matches a pattern defined by a regular expression.

##### Syntax

```
<asp:RegularExpressionValidator ID="RegularExpressionValidator1"
runat="server" ErrorMessage="Input valid email address!" >
</asp:RegularExpressionValidator>
```

Properties	Description
ErrorMessage	Represent the error message to be displayed if expression is not match.
ForeColor	The foreground color of the control
Id	A unique id for the control
BackColor	The background color of the RegularExpressionValidator control
Text	The message to display when validation fails
ValidationExpression	Represent the regular expression to be validated.
<b>Example</b>	
<b>Design Code :</b> RegularExpressionvalidator.aspx Page	
<pre> &lt;table&gt;   &lt;tr&gt;     &lt;td&gt;      Email Id :    &lt;/td&gt;     &lt;td&gt;       &lt;asp:TextBox ID="txtId" runat="server"&gt;&lt;/asp:TextBox&gt;       &lt;asp:RegularExpressionValidator ID="RegularExpressionValidator1"       runat="server" ErrorMessage="Input valid email address!"       ControlToValidate="txtId" ValidationExpression=         "\w+([-+.']\w+)*@\w+([-.\w+)*\.\w+([-.\w+)*"&gt;       &lt;/asp:RegularExpressionValidator&gt;     &lt;/td&gt;   &lt;/tr&gt;   &lt;tr&gt;     &lt;td&gt;       &lt;asp:Button ID="Button1" runat="server" Text="Submit" /&gt;     &lt;/td&gt;   &lt;/tr&gt; &lt;/table&gt; </pre>	
<b>Output View :</b>	



## 5. The CustomValidator:

- The CustomValidator control allows writing application specific custom validation routines for both the client side and the server side validation.
- The client side validation is accomplished through the Client Validation Function property. The client side validation routine should be written in a scripting language, like JavaScript or VBScript, which the browser can understand.
- Allows you to develop custom validation. Performs user-defined validation on an input control using a specified function (client-side, server-side, or both). If the validated control is empty, no validation takes place.
- The CustomValidator control allows writing application specific custom validation routines for both the client side and the server side validation.

### Example

**Design Code :** Customvalidator.aspx Page

```
<body>
```

```
<form id="form1" runat="server">
  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
  <asp:CustomValidator ID="CustomValidator1" runat="server"
    ErrorMessage="enter the abc" ControlToValidate="TextBox1">
  </asp:CustomValidator>
  <asp:Button ID="Button1" runat="server" Text="Button" />
</form>
</body>
```

**Code**

Partial Class ASPButton Inherits System.Web.UI.Page

Protected Sub CustomValidator1\_ServerValidate (ByVal source As  
Object, ByVal args As  
System.Web.UI.WebControls.ServerValidateEventArgs)  
Handles CustomValidator1.ServerValidate

    If Len(args.Value) < 8 Or Len(args.Value) > 16 Then

        args.IsValid = False

    Else

        args.IsValid = True

    End If

End Sub

End Class

**Output View :**

Enter a User Name :

A Username must be between 8 and 16 characters!!

## 6. The Validation Summary Control

- The Validation Summary control does not perform any validation but shows a summary of all errors in the page. The summary displays the values of the ErrorMessage property of all validation controls that failed validation.
- The following two mutually inclusive properties list out the error message:
  - ShowSummary:** shows the error messages in specified format.
  - ShowMessageBox:** shows the error messages in a separate window.
- The Validation Summary control does not perform any validation but shows a summary of all errors in the page. The summary displays the values of the ErrorMessage property of all validation controls that failed validation. The following two mutually inclusive properties list out the error message:

Properties	Description
ForeColor	The fore color of the control
HeaderText	A header in the ValidationSummary control
Id	A unique id for the control
DisplayMode	How to display the summary. Legal values are:BulletList, List, SingleParagraph
ShowSummary	shows the error messages in specified format
ShowMessageBox	shows the error messages in a separate window

**Example****Design Code :** ValidationSummaryvalidator.aspx Page

```
<html>
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:Label ID="Label1" runat="server" Style="top: 239px; left: 75px;
    position: absolute; height: 22px; width: 128px"
    Text="Enter your Age."></asp:Label> &nbsp;
    <asp:Label ID="Label2" runat="server" Style="top: 94px; left: 81px;
    position: absolute; height: 22px; width: 128px" Text="Enter your name:">
    </asp:Label>
    <asp:TextBox ID="TextBox1" runat="server" Style="top: 95px; left: 250px;
    position: absolute; height: 22px; width: 128px"></asp:TextBox>
    <asp:TextBox ID="TextBox4" runat="server" Style="top: 195px; left: 249px;
    position: absolute; height: 22px; width: 127px"></asp:TextBox>
    <asp:Label ID="Label3" runat="server" Style="top: 148px; left: 76px;
    position: absolute; height: 22px; width: 128px" Text="Enter Password:">
    </asp:Label>
  </p>
  <p>
    <asp:TextBox ID="TextBox3" runat="server" Style="top: 146px; left: 249px;
    position: absolute; height: 22px; width: 127px" TextMode="Password">
    </asp:TextBox>
  </p>
  <p>
    <asp:Label ID="Label4" runat="server" Style="top: 197px; left: 75px;
    position: absolute; height: 22px; width: 128px" Text="Confirm Password:">
    </asp:Label>
  </p>
  <asp:TextBox ID="TextBox2" runat="server" Style="top: 236px; left: 250px;
```

```

position: absolute; height: 22px; width: 127px" TextMode="Password">
</asp:TextBox>
<asp:CompareValidator ID="CompareValidator1" runat="server"
Style="top: 197px; left: 522px; position: absolute; height: 22px; width: 17px"
ErrorMessage="CompareValidator" ControlToCompare="TextBox2"
ControlToValidate="TextBox3"> *</asp:CompareValidator>
<p>
  <asp:Button ID="Button1" runat="server" Style="top: 333px; left: 248px;
position: absolute; height: 26px; width: 56px" Text="Submit" />
  <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
Style="top: 196px; left: 393px; position: absolute; height: 22px; width: 22px"
ErrorMessage="password Not match" ControlToValidate="TextBox3">
  *</asp:RequiredFieldValidator>
  <asp:RangeValidator ID="RangeValidator1" runat="server" Style="top: 235p
x
left: 388px; position: absolute; height: 22px; width: 156px; bottom: 288px;"
ErrorMessage="age between 18-100" ControlToValidate="TextBox4"
MaximumValue="100" MinimumValue="18" Type="Integer">
  *</asp:RangeValidator>
  <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
Style="top: 92px; left: 393px; position: absolute; height: 22px;
width: 156px"
ErrorMessage="Name is required" ControlToValidate="TextBox1">
  *</asp:RequiredFieldValidator>
  <asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server"
Style="top: 146px; left: 391px; position: absolute; height: 22px;
width: 156px" ErrorMessage="Password mandatory"
ControlToValidate="TextBox2">*</asp:RequiredFieldValidator>
</p>
<asp:ValidationSummary ID="ValidationSummary1" runat="server"
Style="top: 390px; left: 44px; position: absolute; height: 38px;
width: 625px" />
</form>

```



```
</body>  
</html>
```

**Output View :****Output of ValidationSummary program**

Enter your name:

\*

Enter Password:

\*

Confirm Password:

\*

Enter your Age:

- Confirm Password mandatory & should match password
- Name is required
- Password mandatory

**Exercise**

1. Write down the advantage of ASP Web server control & HTML Server Controls.
2. Explain following ASP Controls with its properties :
  - a. Label
  - b. Textbox
  - c. Button (all 3 types)
  - d. Dropdown list
  - e. Radio button
  - f. Checkbox
3. What is the difference between radiobuttonlist & radiobutton with example?
4. Explain what is Postback in ASP.Net.
5. What is validation? Why we need validations in a web page?
6. List & explain all validation controls available in ASP.Net with its properties.
7. What is validation summary? Explain why it's needed along with its properties.
8. Create a web page for each of the following. [make use of ASP Controls]
  - a. Create a simple calculator in ASP.Net web page that performs the basic mathematical operations like +, -, \*, / on a data entered by user.
  - b. Create a web page that checks whether the number entered by the user is odd or even.
  - c. Create a web page that checks whether the year entered by the user is a leap year or not.
9. Create a web page that registers the user in a web site with the following details:
  - a. Name, address, D.O.B, e-mail id, age, password, confirm password, salary.
  - b. User will only be registered if and only if his age is between 18 to 50 & salary between 20,000 to 50,000 INR.
  - c. Apply appropriate validation controls.

\*\*\*\*\***END OF CHAPTER**\*\*\*\*\*