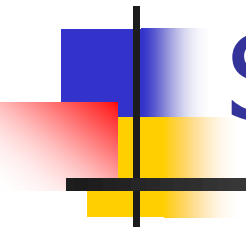
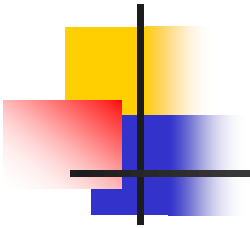


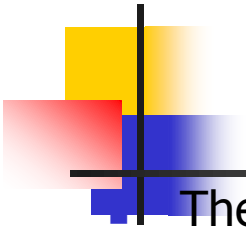
Session 7

HTML Controls and Validation Controls



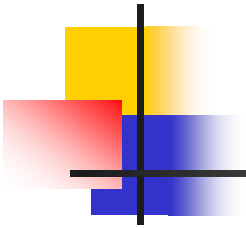
Review

- There are four sets of controls in ASP.NET:
  - Intrinsic Controls
  - List Controls
  - Rich Controls
  - Validation Controls
- Like objects, web controls possess methods and properties, and respond to events.
- ASP.NET provides three intrinsic controls for entering text. They are as follows:
  - Single Line Entry
  - Multi-Line Entry
  - Password Entry



Review Contd…

- There are four controls in ASP.NET that can be used to navigate between pages or transfer control to a specified page:
  - `Button`
  - `LinkButton`
  - `ImageButton`
  - `Hyperlink`
- ASP.NET provides the following set of selection controls:
  - `Checkbox`
  - `RadioButton`
  - `Listbox`
  - `DropDownList`
- ASP.NET provides container controls, that is, the controls that can contain other controls.
- ASP.NET provides two rich controls:
  - `AdRotator Control`
  - `Calendar Control`

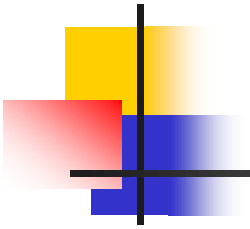


Objectives

- *Explain HTML Controls*
- *Explore the various Validation Controls*
- *Explain Code Behind*
- *Implement Code behind*

HTML Server Controls

- The HTML elements within an ASP.NET file are treated as literal text.
- These elements are programmatically inaccessible to the page developers. To make these elements programmatically accessible, we have to indicate that an HTML element should be parsed and treated as a server control.
- This can be done by adding a `runat="server"` attribute to the HTML element.
- The unique `id` attribute of the HTML element allows us to programmatically reference the control.
- HTML server controls must reside within a containing `<form>` tag with the `runat="server"` attribute.



HTML Server Controls Contd…

- `HtmlForm` Control – Used to create a container for elements in a web page.
- `HtmlImage` Control – Used to display an image.
- `HtmlInputFile` Control – Used to upload a file to the server.

HtmlInputFile Control - Example

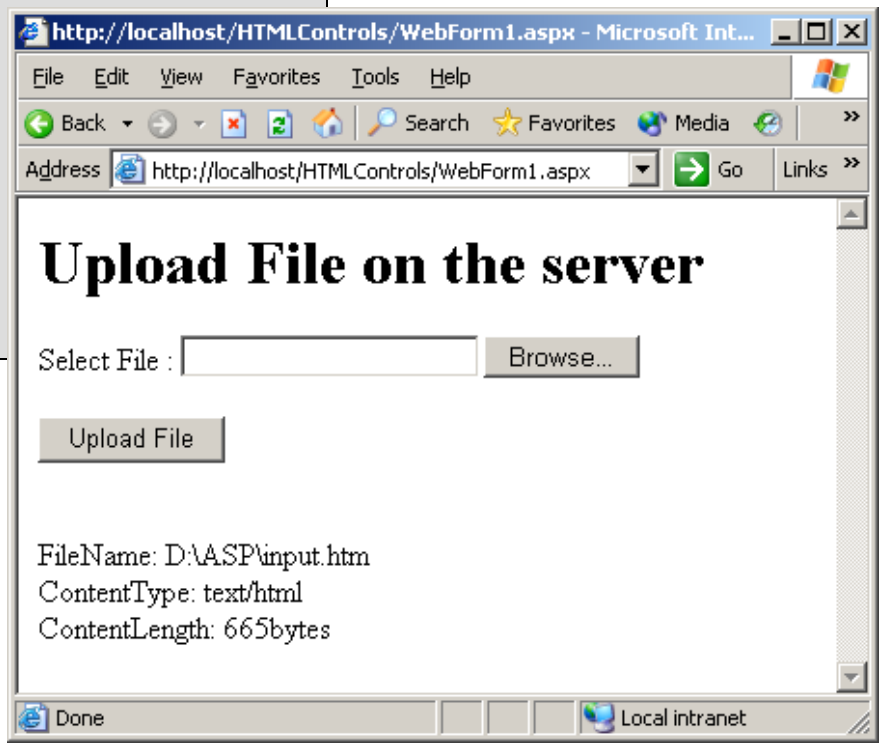
```
<%@ Page Language="C#" AutoEventWireup="True" %>
<html>
 <script runat="server">
   void BtnUpload_Click(Object sender, EventArgs e)
   { // Display information about posted file
     FileName.InnerHtml = MyFile.PostedFile.FileName;
     MyContentType.InnerHtml =
  MyFile.PostedFile.ContentType;
     ContentLength.InnerHtml =
  MyFile.PostedFile.ContentLength.ToString();
     FileDetails.Visible = true;
     // Save uploaded file to server
  MyFile.PostedFile.SaveAs("c:\\Inetpub\\uploadfile.doc
  ");
   }
 </script>
```

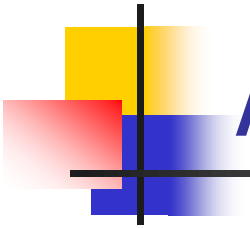
HtmlInputFile Control - Output

```
<body>
  <form action="Ch4Ex1.aspx" method="post"
    enctype="multipart/form-data"
    runat="server"><h1>Upload File on the server</h1>
  Select File : <input id="MyFile" type="file"
     runat="server"><br><br>
  <input type=submit value="Upload File"
     OnServerclick="BtnUpload_Click"
     runat="server">
  <br><br><br>
  <div id="FileDetails" Visible=false runat="server">
  FileName: <span id="FileName" runat="server"/>
  <br>
  ContentType: <span id="MyContentType" runat="server"/><br>
```

HtmlInputFile Control - Output

```
    ContentLength: <span
id="ContentLength" runat="server"/>bytes
    <br>
    </div>
   </form>
  </body>
</html>
```

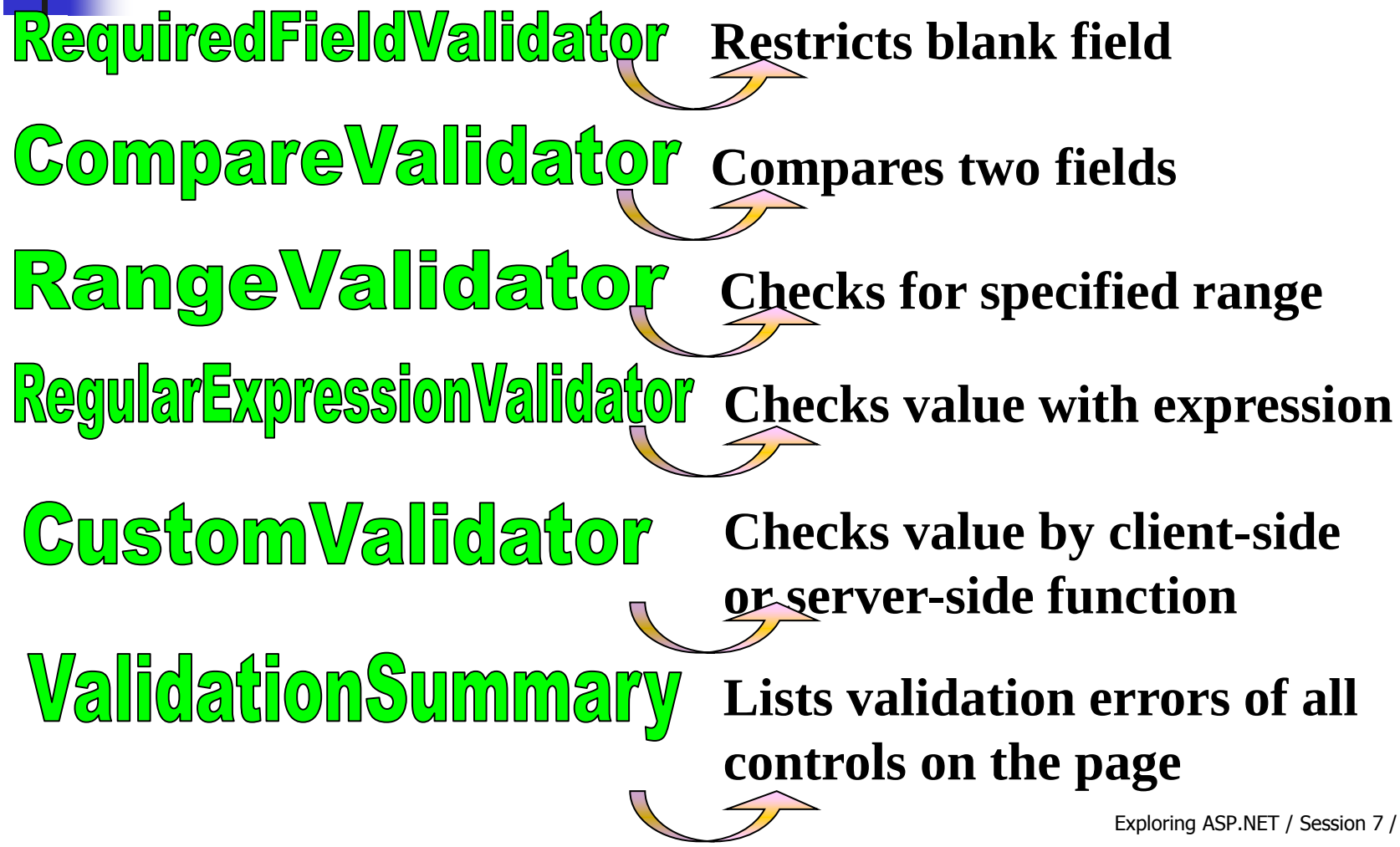




Additional HTML Server Controls

Control	Description
`HtmlAnchor`	Allows to link to another Web page
`HtmlButton`	Allows to create push buttons
`HtmlInputImage`	Allows to create an button that displays an image
`HtmlInputText`	Allows to create a single line text box to receive user input.
`HtmlInputRadioButton`	Allows to create a radio button
`HtmlSelect`	Allows to create a list control
`HtmlTextArea`	Allows to create a multiline text box
`HtmlTable`	Allows to create a table

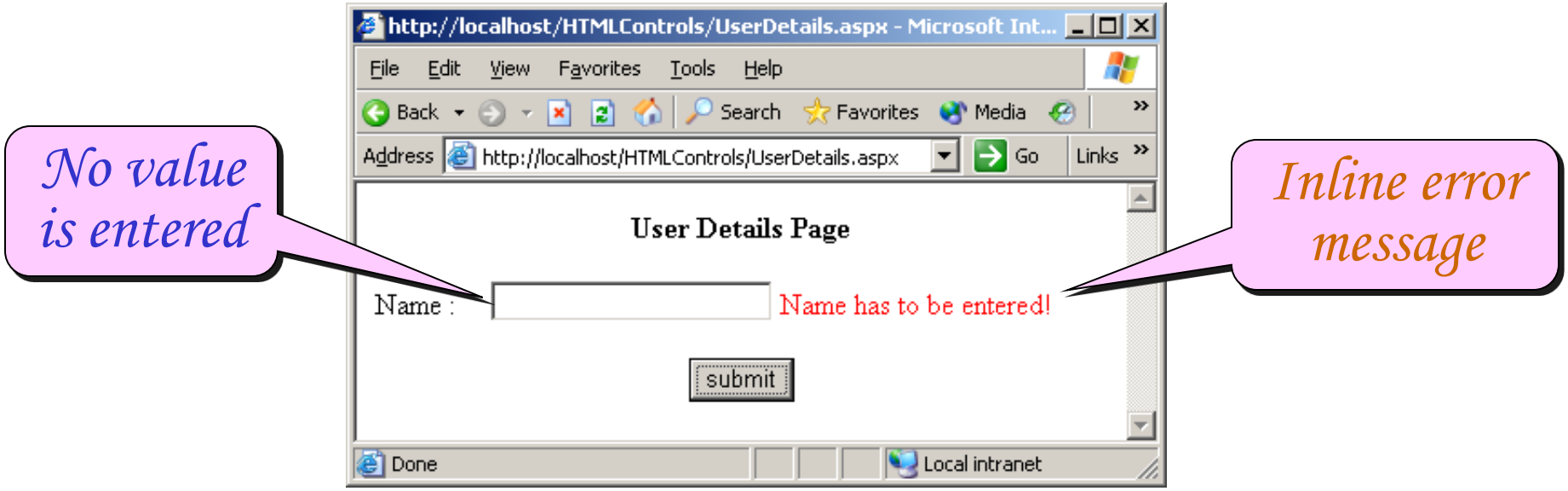
Validation Controls



RequiredFieldValidator - Example

```
<html>
    <form runat="server">
        <center><b>User Details Page</b>
        </center>
        <br>
        Name :    
        <asp:textbox id="txtName" runat=server />
        <asp:requiredfieldvalidator
controltovalidate="txtName" display="static"
errormessage="Name has to be entered!"  runat=server/>
        <br><br>
        <center><asp:button id= btnSubmit text="submit"
runat=server /> </center>
     </form>
</html>
```

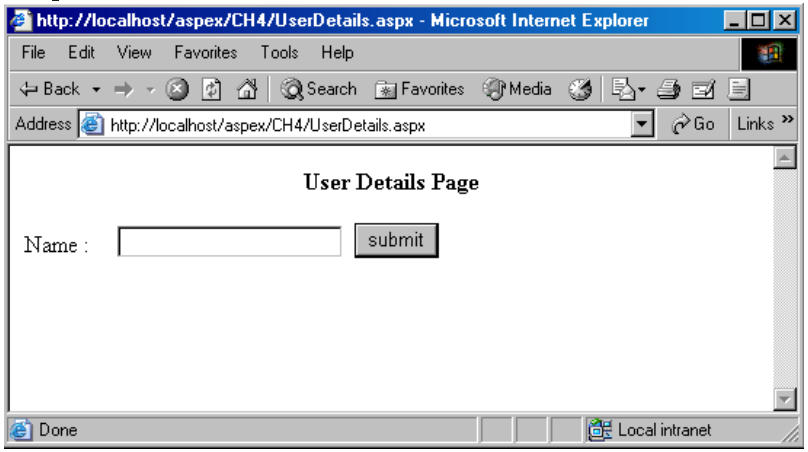
RequiredFieldValidator - Output



RequiredFieldValidator - Properties

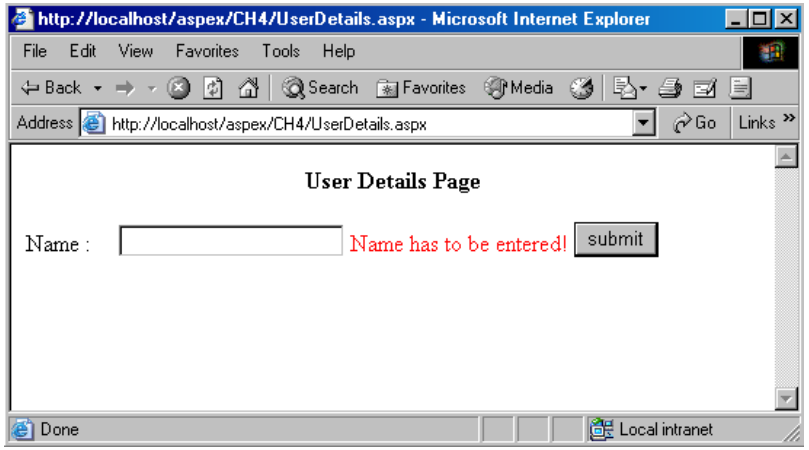
- `Controltovalidate` property specifies the control that needs to be validated
- `Errormessage` property is used to specify the message that has to be displayed when using the `RequiredFieldValidator` control in the page
- When the Display is set to static, certain amount of space is reserved on the page for the error message
- If Display is set to dynamic, no space is reserved on the page for displaying the error message
- Display is set to none, when only the summary of all the error messages of the page needs to be displayed

Validation Error Message



Before dynamic display of errors

Dynamic Display



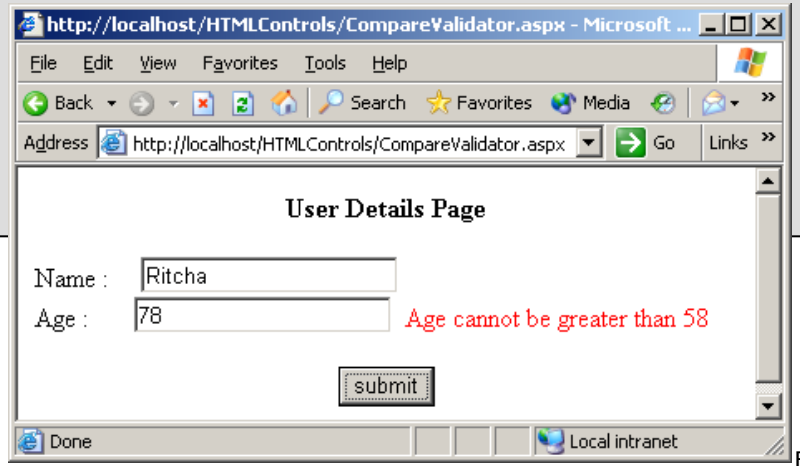
After dynamic display of errors

CompareValidator – Example 1

```
<html>
   <form runat="server">
        <center><b>User Details Page</b>
        </center>
        <br>
        Name :    
        <asp:textbox id="txtName" runat=server />
        <asp:requiredfieldvalidator id="reqvaln"
controltovalidate="txtName" errormessage="Name has to be
entered" display="dynamic" runat=server/>
        <br>
        Age :      
        <asp:textbox id="txtAge" runat=server />
        <asp:requiredfieldvalidator id="reqvala"
controltovalidate="txtAge" errormessage="Age has to be
entered" display="dynamic" runat=server/>
```

CompareValidator – Example 1

```
      <asp:comparevalidator controltovalidate="txtAge"
display="dynamic" errormessage="Age cannot be greater than
58"
valuetocompare=58 type="Integer" operator="LessThanEqual"
 runat=server>
      </asp:comparevalidator>
      <br><br>
      <center><asp:button id=btnSubmit text="submit"
runat=server/>
      </center>
       </form>
</html>
```



CompareValidator - Properties

- The Type property of the CompareValidator is used to specify the data type of the two values being compared. Type can take one of the following values-
  - String
  - Integer
  - Double
  - Date
  - Currency
- Operator property is used to specify the type of comparison to be made. Operator can take one of the following values:
  - Equal
  - NotEqual
  - GreaterThan
  - GreaterThanEqual
  - LessThan
  - LessThanEqual

CompareValidator – Example 2

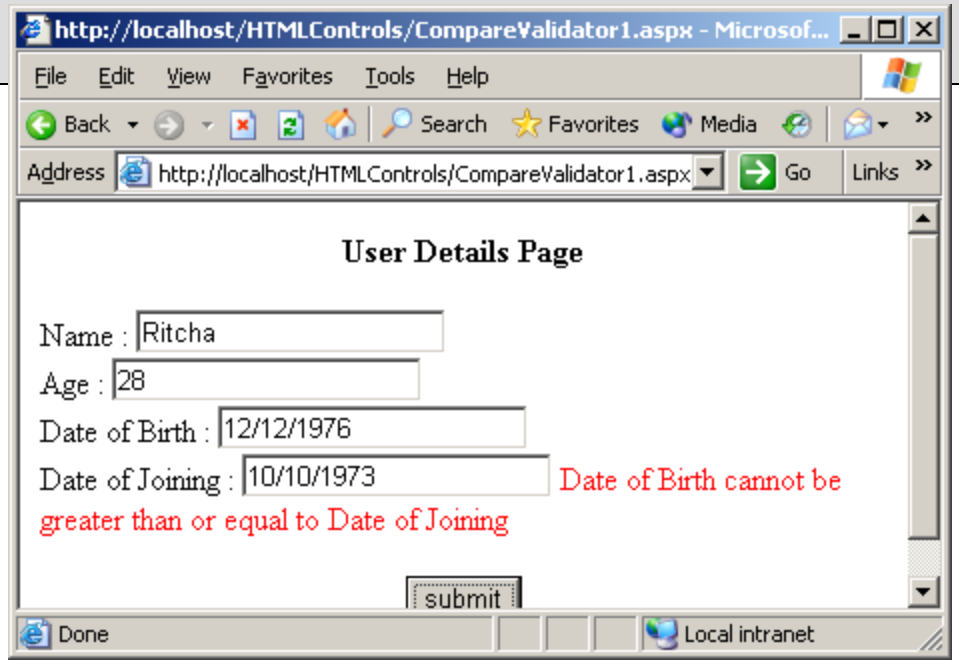
```
<html>
    <form runat="server">
       <center><b>User Details Page</b>
       </center>
       <br>
       Name :
       <asp:textbox id="txtName" runat=server />
       <asp:requiredfieldvalidator id="reqvaln"
controltovalidate="txtName" errormessage="Name has to be
entered" display="dynamic" runat=server/>
       <br>
       Age :
       <asp:textbox id="txtAge" runat=server />
       <asp:requiredfieldvalidator id="reqvala"
controltovalidate="txtAge" errormessage="Age has to be
entered" display="dynamic" runat=server/>
```

CompareValidator – Example 2

```
        <asp:comparevalidator controltovalidate="txtAge"
display="dynamic" errormessage="Age cannot be greater than
58" valuetocompare=58 type="Integer" operator="LessThanEqual"
runat=server/>
        <br>
        Date of Birth :
        <asp:textbox id="txtDob" runat=server />
        <br>
        Date of Joining :
        <asp:textbox id="txtDoj" runat=server />
        <asp:comparevalidator controltovalidate="txtDoj"
display="dynamic" errormessage="Date of Birth cannot be
greater than or equal to Date of Joining"
Controltocompare="txtDob" type="Date" operator="GreaterThan"
runat=server/>
```

CompareValidator – Example 2

```
        <br>
        <br>
        <center><asp:button id=btnSubmit text="submit"
runat=server /></center>
     </form>
</html>
```

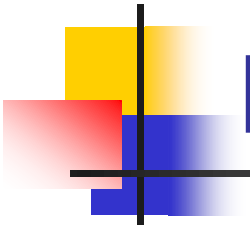


RangeValidator

- The RangeValidator control is used to check if the value of a control lies within a range.
- The range can be specified with the help of two controls or by specifying static values

Code Snippet -

```
<asp:textbox id="txtAge" runat=server/>
<asp:rangevalidator controltovalidate="txtAge"
type="Integer" minimumvalue="18" maximumvalue="58"
errormessage="Your age must be in the range of 18-58 yrs"
display="dynamic" runat="server" >
</asp:rangevalidator>
```



RegularExpressionValidator

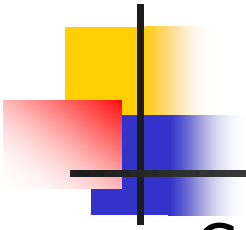
- Many times, the values entered into a control have to be in a pre–specified format (for example – telephone numbers, email addresses)
- Comparison of the values entered into a control with a specified pattern is made possible by the RegularExpressionValidator control
- The pattern to which the value must match is specified in the ValidationExpression property

RegularExpressionValidator Characters

Sign	Meaning
^	Specifies that checking starts from here
$	Specifies that the checking ends here
[]	Checks whether the value entered matches with any of the characters that are in the square brackets
\w	Allows any value to be entered
\d{}	"\d" specifies that the value entered is a digit, and {} specifies the number of occurrences of the specified data type
+	Indicates that one or more elements to be added to the expression being checked

RegularExpressionValidator Code Snippet

```
<asp:textbox id="txtEmailid" runat=server/>
<asp:regularexpressionvalidator
controltovalidate="txtEmailid" display="static"
validationexpression="^[\w-]+@[\w-
]+\.(com|net|org|edu)$" runat=server>
E-mail Id is not in the correct format
</asp:RegularExpressionValidator>
```

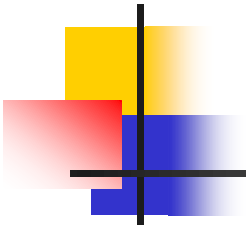


CustomValidator

- CustomValidator controls can be used to provide validation in case the provided controls are not enough to validate the inputs
- These controls call a client-side or server-side function that performs the required validation

Code Snippet

```
<asp:customvalidator runat="server"
controltovalidate="txtGrade"
clientvalidationfunction="clientval"
onservervalidate="serverval" display="static">
Wrong value
</asp:customvalidator>
```

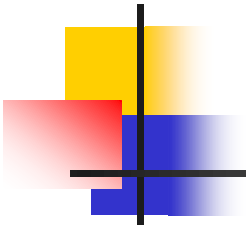


ValidationSummary

- Validation error messages on a web page can be displayed in two ways:
  - The errors can be displayed as and when the focus is lost from the control
  - A summary of all the error messages in a page can be displayed together
- The `ValidationSummary` control can be used to produce such a summary.
- The errors can be viewed in the form of a list, bullets, or a single paragraph by setting the displaymode to `list`, `bulletlist`, or `singleparagraph` respectively

Code Snippet -

```
<asp:validationsummary id="ValSum"  headertext="The errors
found are: " displaymode="singleparagraph" runat="server"/>
```



Page.IsValid Property

- The Page object has a property called `IsValid`, that returns true if all the validation tests are successful, and returns false if even a single validation test is unsuccessful
- `IsValid` property can be used to know if all the validations tests have been successful.
- The user can then be redirected to another page, or shown an appropriate message

Page.IsValid Property-Example

```
<html>
    <form runat="server">
       <script language="C#" runat="server" >
       void subbtn(Object Src, EventArgs E)
       {
          if (Page.IsValid == true)
          {
              lblMessage.Text = "Page is Valid!";
          }
       }
       </script>
       <center><b>User Details Page</b></center>
       <br><br>
       <asp:label id="lblMessage" runat="server"/><br><br>
       Name :
       <asp:textbox id="txtName" runat=server />
```

Page.IsValid Property-Example

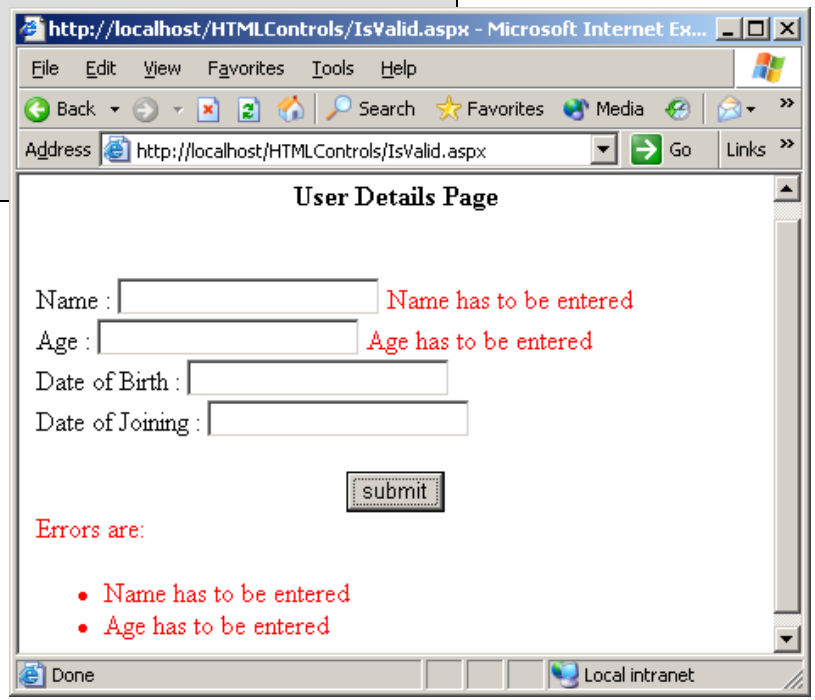
```
      <asp:requiredfieldvalidator id="reqvaln"
controltovalidate="txtName" errormessage="Name has to be
entered" display="dynamic" runat=server/>
      <br>
      Age :
            <asp:textbox id="txtAge" runat=server />
            <asp:requiredfieldvalidator id="reqvala"
      controltovalidate="txtAge" errormessage="Age has to
      be entered" display="dynamic" runat=server/>
            <asp:comparevalidator
      controltovalidate="txtAge" display="dynamic"
      errormessage="Age cannot be greater than 58"
      valuetocompare=58 type="Integer"
      operator="LessThanEqual" runat=server/>
            <br>
```

Page.IsValid Property-Example

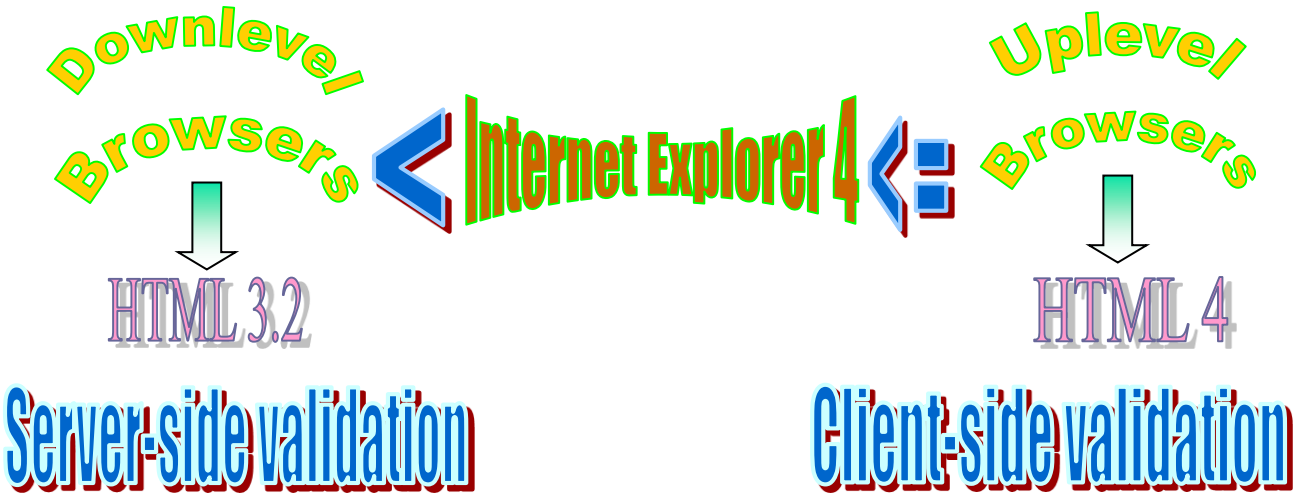
```
        Date of Birth :
        <asp:textbox id="txtDob" runat=server />
        <br>
        Date of Joining :
        <asp:textbox id="txtDoj" runat=server />
        <asp:comparevalidator
controltovalidate="txtDoj" display="dynamic"
errormessage="Date of Birth cannot be greater than or
equal to Date of Joining" Controltocompare="txtDob"
type="Date" operator="GreaterThan" runat=server/>
        <br><br>
        <center><asp:button id=btnSubmit text="submit"
onclick="subbtn" runat=server />
        </center>
```

Page.IsValid Property - Output

```
            <asp:validationsummary id="ValSum"
headertext="Errors are: " displaymode="bulletlist"
runat="server"/>
        </form>
</html>
```

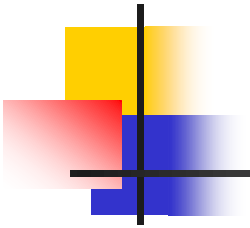


Uplevel and Downlevel Browsers




```
<%@ Page ClientTarget= DownLevel %>
```

disable client-side validation



Code Behind

- To avoid the increasing complexity involved in creating web pages that need high-quality graphics as well as complex programming, ASP.NET provides the technique of “Code Behind”.
- It is possible to write the code to provide the required functionality in a separate file, than that of the code to create the graphics for the web page
- The class file containing the functionality has to be created in any of the .NET supported languages from which the .aspx file can inherit.

Code Behind - Example

.aspx file

```
<%@ Page language="C#" Inherits="codebehind" %>
<html>
        <form runat="server">
        <center><b>User Details Page</b></center>
        <asp:label id="lblMessage" runat="server"/><br>
        Name :              
        <asp:textbox id="txtName" runat=server />
        <asp:requiredfieldvalidator id="reqvaln"
controltovalidate="txtName" errormessage="Name has to
be entered" display="dynamic" runat=server/>
        <br>
        Age :              
        <asp:textbox id="txtAge" runat=server />
        <asp:requiredfieldvalidator id="reqvala"
controltovalidate="txtAge" errormessage="Age has to be
entered" display="dynamic" runat=server/>
```

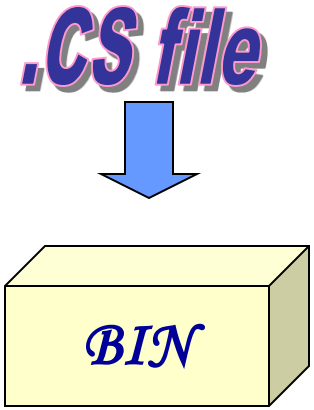
Provides the functionality

A

Code Behind Example Contd…

```
<asp:comparevalidator controltovalidate="txtAge"
display="dynamic" errormessage="Age cannot be
greater than 58" valuetocompare=58 type="Integer"
operator="LessThanEqual" runat=server/><br>
Date of Birth :    
<asp:textbox id="txtDob" runat=server /><br>
Date of Joining :
<asp:textbox id="txtDoj" runat=server />
<asp:comparevalidator controltovalidate="txtDoj"
display="dynamic" errormessage="Date of Joining
cannot be greater than or equal to date of Birth"
Controltocompare="txtDob" type="Date"
operator="GreaterThan" runat=server/><br> <br>
<center><asp:button id=btnSubmit text="submit"
onclick="subbtn" runat=server /></center>
<asp:validationsummary id="ValSum"
headertext="Errors are: " displaymode="bulletlist"
runat="server"/>
</form></html>
```

Code Behind Example Contd…

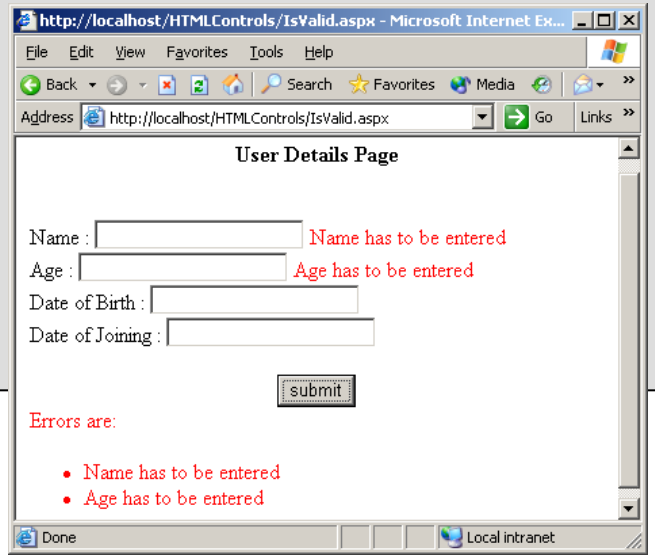


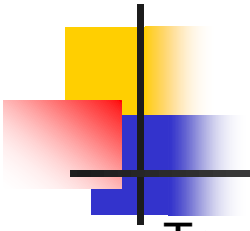

```
using System;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public class codebehind: Page
{
    public System.Web.UI.WebControls.Label
lblMessage;
    public System.Web.UI.WebControls.Button
btnSubmit;
```

Separated programming code

Code Behind Example Contd…

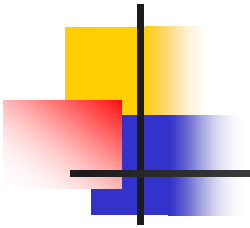
```
    protected void subbtn(Object Src, EventArgs E)
    {
        if (Page.IsValid == true)
        {
            lblMessage.Text = "Page is Valid!";
        }
    }

    void Main()
    {
    }
}
```





Summary

- To make HTML elements programmatically accessible, it is necessary to indicate that an HTML element be parsed and treated as a server control. This can be done by adding a runat="server" attribute to the HTML element.
- The process of checking whether the user has filled up a form in the right format, and has not left any fields blank is called validation.
- The validation controls available are as follows:
  - RequiredFieldValidator**:** Helps in ensuring that a value is entered for a field
  - CompareValidator**:** Checks if the value of a control is similar to the value of another control
  - RangeValidator**:** Checks if the value entered in a control is in the specified range of values
  - RegularExpressionValidator**:** Checks if the value entered fits the regular expression that is specified
  - CustomValidator**:** The value entered is checked by a client-side or server-side function written by the programmer
  - ValidationSummary**:** A list of all the validation errors occurring in all the controls is created, and can be displayed on the page.



Summary Contd…

- The Page object has a property called IsValid, that returns true if all the validation tests are successful, and vice-versa.
- To disable client-side validation, the ClientTarget property can be set to downlevel.
- Code Behind is a feature that enables the developer to write the code to provide the required functionality in a separate file than that of the code to create the graphics for the web page