

Unit 4

Asp.Net Server Controls

Server Controls are the tags that are understood by the server. There are basically three types of server controls.

- **HTML Server Controls** - Traditional HTML tags
- **Web Server Controls** - New ASP. NET tags
- **Validation Server Controls** - For input validation

HTML Server Controls

- ASP.NET provides a way to work with HTML Server controls on the server side; programming with a set of controls collectively is called HTML Controls.
- HTML elements in ASP. NET files are, by default, treated as text. To make these elements programmable, add a `runat="server"` attribute to the HTML element. This attribute indicates that the element should be treated as a server control.

Control Name	HTML tag
HtmlHead	<head>element
HtmlInputButton	<input type=button submit reset>
HtmlInputCheckbox	<input type=checkbox>
HtmlInputFile	<input type = file>
HtmlInputHidden	<input type = hidden>
HtmlInputImage	<input type = image>
HtmlInputPassword	<input type = password>
HtmlInputRadioButton	<input type = radio>
HtmlInputReset	<input type = reset>
HtmlText	<input type = text password>
HtmlImage	 element
HtmlLink	<link> element
HtmlAnchor	<a> element
HtmlButton	<button> element
HtmlButton	<button> element
HtmlForm	<form> element
HtmlTable	<table> element
HtmlTableCell	<td> and <th>
HtmlTableRow	<tr> element
HtmlTitle	<title> element
HtmlSelect	<select> element

Web Server Controls

- Web server controls are special ASP. NET tags understood by the server.
- Like HTML server controls, Web server controls are also created on the server and they require a runat="server" attribute to work.

Web Server Control	Description
<u>AdRotator</u>	Displays a sequence of images
<u>Button</u>	Displays a push button
<u>Calendar</u>	Displays a calendar
<u>CalendarDay</u>	A day in a calendar control
<u>CheckBox</u>	Displays a check box
<u>CheckBoxList</u>	Creates a multi-selection check box group
DataGrid	Displays fields of a data source in a grid
DataList	Displays items from a data source by using templates
<u>DropDownList</u>	Creates a drop-down list
<u>HyperLink</u>	Creates a hyperlink
<u>Image</u>	Displays an image
<u>ImageButton</u>	Displays a clickable image
<u>Label</u>	Displays static content which is programmable (lets you apply styles to its content)
<u>LinkButton</u>	Creates a hyperlink button
<u>ListBox</u>	Creates a single- or multi-selection drop-down list
<u>ListItem</u>	Creates an item in a list
<u>Literal</u>	Displays static content which is programmable(does not let you apply styles to its content)
<u>Panel</u>	Provides a container for other controls
<u>Placeholder</u>	Reserves space for controls added by code
<u>RadioButton</u>	Creates a radio button
<u>RadioButtonList</u>	Creates a group of radio buttons
<u>BulletedList</u>	Creates a list in bullet format
Repeater	Displays a repeated list of items bound to the control
<u>Style</u>	Sets the style of controls
<u>Table</u>	Creates a table

Validation Server Controls

- After you create a web form, you should make sure that mandatory fields of the form elements such as login name and password are not left blank; data inserted is correct and is within the specified range. Validation is the method of scrutinizing (observing) that the user has entered the correct values in input fields.
- A Validation server control is used to validate the data of an input control. If the data does not pass validation, it will display an error message to the user.

Validation Server Control	Description
<u>CompareValidator</u>	Compares the value of one input control to the value of another input control or to a fixed value
<u>CustomValidator</u>	Allows you to write a method to handle the validation of the value entered
<u>RangeValidator</u>	Checks that the user enters a value that falls between two values
<u>RegularExpressionValidator</u>	Ensures that the value of an input control matches a specified pattern
<u>RequiredFieldValidator</u>	Makes an input control a required field
<u>ValidationSummary</u>	Displays a report of all validation errors occurred in a Web page

How to work with Button Controls

- The Button control is used to display a push button. The push button may be a submit button or a command button. By default, this control is a submit button.
- A submit button does not have a command name and it posts the Web page back to the server when it is clicked. It is possible to write an event handler to control the actions performed when the submit button is clicked.
- A command button has a command name and allows you to create multiple Button controls on a page. It is possible to write an event handler to control the actions performed when the command button is clicked.

How to work with Text Boxes

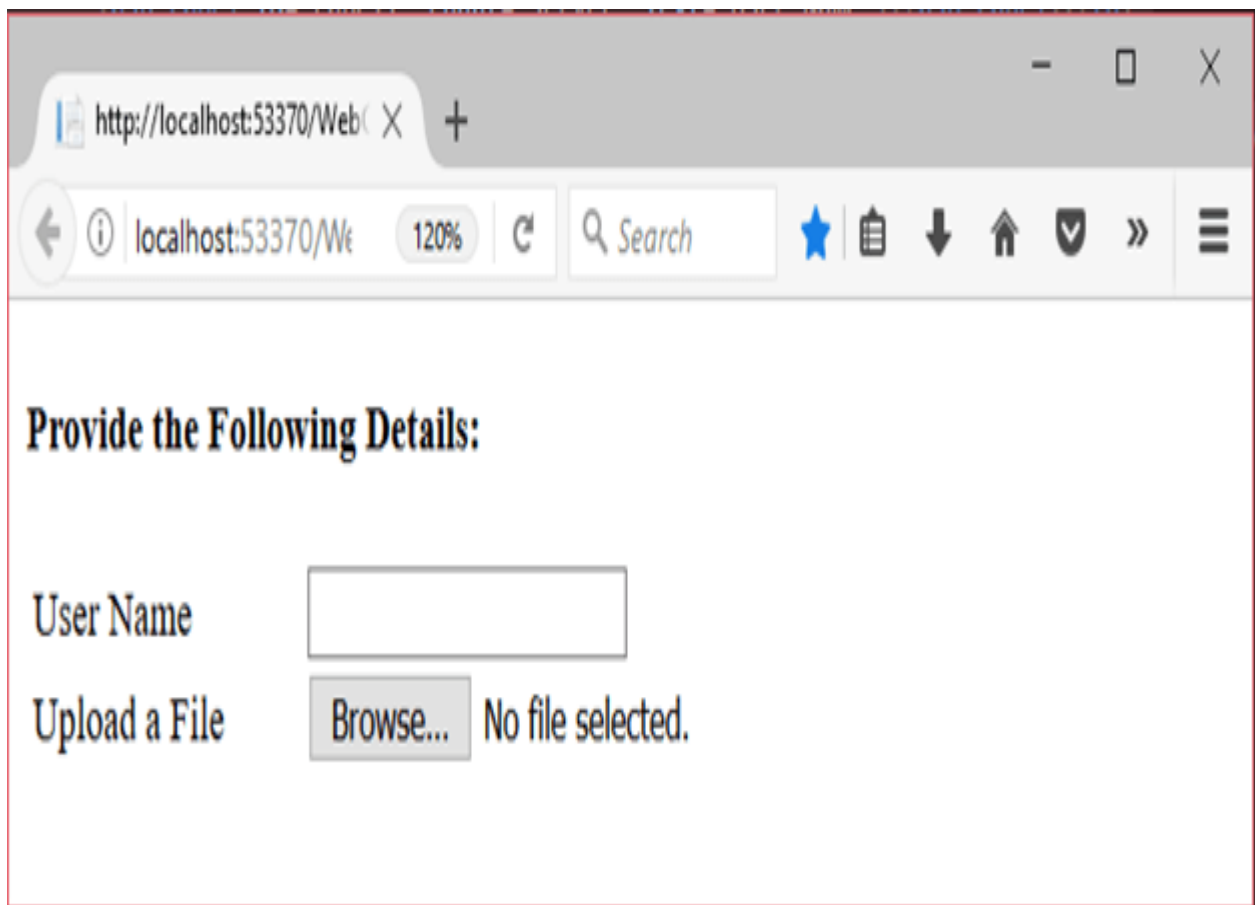
- The Text Box control is used to create a text box where the user can input text.
- **Auto Post Back:** Boolean value signaling whether the Auto Post Back feature is enabled for the control. This determines if the form will automatically be posted back to the Web server when the contents of the field change.
- **Back Color:** The background color of the control. This may be controlled with CSS as well.
- **Border Color:** The border color for the border (if used) displayed around the control.
- **Border Width:** The width of the border around the control on the page.
- **Causes Validation:** Boolean value signaling whether validation is performed when post back of form is executed.
- **Columns:** The display width of the Text Box control.
- **Css Class:** Allows you to assign a CSS class to the control.
- **Font:** The various font attributes associated with the control. The preferred approach is to use CSS, but the Font class contains properties

for signaling whether the text appears in bold, italic, strikeout, or underline, as well as font size, name, and more.

- **Fore Color:** The color of the text displayed in the field. The preferred technique is CSS.
- **Height:** Get or set the height of the control.

How to work with Labels

- This control is used to display textual information on the web forms. It is mainly used to create caption for the other controls like: textbox.
- To create **label** either we can write code or use the drag and drop facility
- This is server side control, asp provides own tag to create label. The example is given below.
- `<asp:LabelID="Label1" runat="server" Text="Label" ></asp:Label>`



The screenshot shows a web browser window with the address bar displaying 'http://localhost:53370/Web'. The browser's address bar also shows 'localhost:53370/Web' and '120%'. The page content includes a heading 'Provide the Following Details:' followed by two form elements. The first element is a label 'User Name' next to a text input field. The second element is a label 'Upload a File' next to a 'Browse...' button and the text 'No file selected.'

Check Box Controls:

Check Box control is an asp.net web server control. Check Box control visually as square on web forms. The Checkbox control allow user to **check square or uncheck square**.

In Check Box control check and uncheck checkbox specify by the checked property of check box true or false.

If checkbox control square **ticked then checked = true** and **unchecked then checked=false**.

We can drag the checkbox control from toolbox and drop it to on web forms shows like below. Checkbox control allow user to do either checked (tick) or unchecked (untick) the checkbox control in asp.net.

Radio Buttons Control:

Radio Button control is used, when we want the user to select only one option from the available choices.

For example, the gender of a person. A person can be "Male" or "Female". He/she cannot be both.

So, if the user has first selected "Male" and if he/she tries to select "Female", the initial "Male" selection he made should automatically get de-selected.

Important properties of the radio Button Control

Checked - This is a Boolean property, that is used to check if the button is checked or not.

Text - This is string property used to get or set the text associated with the radio button control.

Text Align - Right or Left. On which side of the radio button the text should appear.

Auto Post back – set this property to true, if you want the web form to be posted immediately when the checked status of the radio button changes.

Group Name – By default, the individual radio button selections are not mutually exclusive. If you have a group of radio buttons, and if you want the selections among the group to be mutually exclusive, then use the same group name for all the radio button controls.

Events:

Checked Changed - This event is fired when the checked status of the radio button control is changed.

List Controls:

ASP.NET provides the following list controls.

- Drop-down list
- List box
- Radio button list
- Check box list
- Bulleted list

These controls display list of options to select. You can select one or more options, the choice depends upon control. They all derive from the `System.Web.UI.WebControls.ListControl` class

some of the important common properties of list controls are as follows:

- **Selected Value:** Get the value of the selected item from the dropdown list.
- **Selected Index:** Gets the index of the selected item from the dropdown box.
- **Selected Item:** Gets the text of selected item from the list.
- **Items:** Gets the collection of items from the dropdown list.
- **Data Text Field:** Name of the data source field to supply the text of the items. Generally this field came from the data source.
- **Data Value Field:** Name of the data source field to supply the value of the items. This is not visible field to list controls, but you can use it in the code.
- **Data Source ID:** ID of the data source control to provide data.

There are several ways through which you can populate these controls such as:

- By using data from database.
- Directly write code to add items.
- Add items through the items collection from property window.
- Write HTML to populate these controls.
- Use inbuilt data source controls.

Web.Config Files:

The behavior of an ASP.NET application is affected by different settings in the configuration files:

- machine.config
- web.config

The machine.config file contains default and the machine-specific value for all supported settings. The machine settings are controlled by the system administrator and applications are generally not given access to this file.

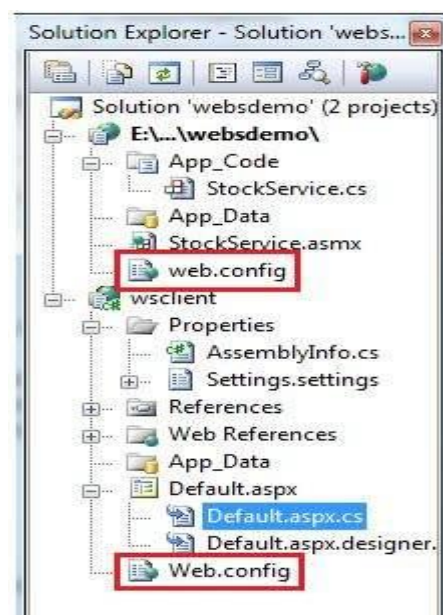
An application however, can override the default values by creating web.config files in its roots folder. The web.config file is a subset of the machine.config file.

If the application contains child directories, it can define a web.config file for each folder. Scope of each configuration file is determined in a hierarchical top-down manner.

Any web.config file can locally extend, restrict, or override any settings defined on the upper level.

Visual Studio generates a default web.config file for each project. An application can execute without a web.config file, however, you cannot debug an application without a web.config file.

The following figure shows the Solution Explorer for the sample example used in the web services tutorial:



Global.asax File:

Global.asax is an optional file which is used to handling higher level application events such as Application_Start, Application_End, Session_Start, Session_End etc. It is also popularly known as ASP.NET Application File. This file resides in the root directory of an ASP.NET-based application.

Global.asax contains a Class representing your application as a whole. At run time, this file is parsed and compiled into a dynamically generated .NET Framework class derived from the HttpApplication base class. You can deploy this file as an assembly in the \bin directory of an ASP.NET application. The Global.asax file itself is configured so that if a user requests the file, the request is rejected. External users cannot download or view the code written within it.

How to create a Global.asax file

Global.asax file don't create normally; you need to add it by yourself.

```
Open Visual Studio
  Create a new website
    Go to the Solution Explorer
      Add New Item
        Global Application Class
          Add
```

Rich Web Controls

In ASP.NET, rich web controls, also known as server-side controls or web server controls, are a set of pre-built, customizable user interface elements that simplify the development of interactive and feature-rich web applications. These controls are part of the ASP.NET framework and provide a high level of abstraction, allowing developers to create dynamic web applications without extensive knowledge of HTML, CSS, or JavaScript. Here are some commonly used rich web controls in ASP.NET:

GridView Control: The GridView control is used to display tabular data in a grid format. It supports features like sorting, paging, editing, and data binding, making it a powerful tool for working with data-driven web applications.

csharpCopy code

```
<asp:GridView ID="GridView1" runat="server"> <!-- Define columns and data binding here --> </asp:GridView>
```

DetailsView Control: The DetailsView control is used to display and edit a single record at a time. It is commonly used for displaying the details of a selected item from a GridView or other data source.

csharpCopy code

```
<asp:DetailsView ID="DetailsView1" runat="server"> <!-- Define fields and data binding here --> </asp:DetailsView>
```

FormView Control: The FormView control is similar to the DetailsView but provides more flexibility in terms of layout and customization. It is often used for data entry and editing.

csharpCopy code

```
<asp:FormView ID="FormView1" runat="server"> <!-- Define fields and data binding here --> </asp:FormView>
```

DataList Control: The DataList control is used for displaying repeated data items in a customizable template. It provides greater flexibility in layout compared to the GridView.

csharpCopy code

```
<asp:DataList ID="DataList1" runat="server"> <!-- Define templates for item rendering and data binding here --> </asp:DataList>
```

Repeater Control: The Repeater control is the most flexible data-bound control in ASP.NET. It allows developers to define custom templates for rendering data items and is often used for complex data presentation scenarios.

csharpCopy code

```
<asp:Repeater ID="Repeater1" runat="server"> <!-- Define templates for item rendering and data binding here --> </asp:Repeater>
```

Calendar Control: The Calendar control is used for date selection and display. It provides features like date selection, event handling, and customizable appearance.

csharpCopy code

```
<asp:Calendar ID="Calendar1" runat="server"> <!-- Define event handlers and appearance properties here --> </asp:Calendar>
```

Menu Control: The Menu control provides a structured way to create navigation menus. It supports hierarchical menus and can be data-bound for dynamic menu generation.

csharpCopy code

```
<asp:Menu ID="Menu1" runat="server"> <!-- Define menu items and data binding here --> </asp:Menu>
```

TreeView Control: The TreeView control is used to display hierarchical data in a tree structure. It is often used for navigation menus, site maps, and directory structures.

csharpCopy code

```
<asp:TreeView ID="TreeView1" runat="server"> <!-- Define nodes and data binding here --> </asp:TreeView>
```

These rich web controls in ASP.NET simplify the development of web applications by providing a high-level abstraction for common UI elements. Developers can customize these controls by defining templates, handling events, and applying CSS styles to meet the specific requirements of their applications. Additionally, ASP.NET Web Forms provides server-side event handling and ViewState management, making it a robust platform for building interactive web applications.

Ad Rotator Control

The AdRotator control randomly selects banner graphics from a list, which is specified in an external XML schedule file. This external XML schedule file is called the advertisement file.

The AdRotator control allows you to specify the advertisement file and the type of window that the link should follow in the AdvertisementFile and the Target property respectively.

The basic syntax of adding an AdRotator is as follows:

```
<asp:AdRotator runat = "server" AdvertisementFile = "adfile.xml" Target = "_blank" />
```

Before going into the details of the AdRotator control and its properties, let us look into the construction of the advertisement file.

The Advertisement File

The advertisement file is an XML file, which contains the information about the advertisements to be displayed.

Extensible Markup Language (XML) is a W3C standard for text document markup. It is a text-based markup language that enables you to store data in a structured format by using meaningful tags. The term 'extensible' implies that you can extend your ability to describe a document by defining meaningful tags for the application.

XML is not a language in itself, like HTML, but a set of rules for creating new markup languages. It is a meta-markup language. It allows developers to create custom tag sets for special uses. It structures, stores, and transports the information.

Following is an example of XML file:

```
<BOOK>
  <NAME> Learn XML </NAME>
  <AUTHOR> Samuel Peterson </AUTHOR>
  <PUBLISHER> NSS Publications </PUBLISHER>
  <PRICE> $30.00</PRICE>
</BOOK>
```

Like all XML files, the advertisement file needs to be a structured text file with well-defined tags delineating the data. There are the following standard XML elements that are commonly used in the advertisement file:

Element	Description
Advertisements	Encloses the advertisement file.
Ad	Delineates separate ad.
ImageUrl	The path of image that will be displayed.
NavigateUrl	The link that will be followed when the user clicks the ad.
AlternateText	The text that will be displayed instead of the picture if it cannot be displayed.

Keyword	Keyword identifying a group of advertisements. This is used for filtering.
Impressions	The number indicating how often an advertisement will appear.
Height	Height of the image to be displayed.
Width	Width of the image to be displayed.

Apart from these tags, customs tags with custom attributes could also be included. The following code illustrates an advertisement file ads.xml:

```
<Advertisements>
  <Ad>
    <ImageUrl>rose1.jpg</ImageUrl>
    <NavigateUrl>http://www.1800flowers.com</NavigateUrl>
    <AlternateText>
      Order flowers, roses, gifts and more
    </AlternateText>
    <Impressions>20</Impressions>
    <Keyword>flowers</Keyword>
  </Ad>

  <Ad>
    <ImageUrl>rose2.jpg</ImageUrl>
    <NavigateUrl>http://www.babybouquets.com.au</NavigateUrl>
    <AlternateText>Order roses and flowers</AlternateText>
    <Impressions>20</Impressions>
    <Keyword>gifts</Keyword>
  </Ad>

  <Ad>
    <ImageUrl>rose3.jpg</ImageUrl>
    <NavigateUrl>http://www.flowers2moscow.com</NavigateUrl>
    <AlternateText>Send flowers to Russia</AlternateText>
    <Impressions>20</Impressions>
    <Keyword>russia</Keyword>
  </Ad>

  <Ad>
```

```
<ImageUrl>rose4.jpg</ImageUrl>
<NavigateUrl>http://www.edibleblooms.com</NavigateUrl>
<AlternateText>Edible Blooms</AlternateText>
<Impressions>20</Impressions>
<Keyword>gifts</Keyword>
</Ad>
</Advertisements>
```

Properties and Events of the AdRotator Class

The AdRotator class is derived from the WebControl class and inherits its properties. Apart from those, the AdRotator class has the following properties:

Properties	Description
AdvertisementFile	The path to the advertisement file.
AlternateTextFeild	The element name of the field where alternate text is provided. The default value is AlternateText.
DataMember	The name of the specific list of data to be bound when advertisement file is not used.
DataSource	Control from where it would retrieve data.
DataSourceID	Id of the control from where it would retrieve data.
Font	Specifies the font properties associated with the advertisement banner control.
ImageUrlField	The element name of the field where the URL for the image is provided. The default value is ImageUrl.
KeywordFilter	For displaying the keyword based ads only.
NavigateUrlField	The element name of the field where the URL to navigate to is provided. The default value is NavigateUrl.
Target	The browser window or frame that displays the content of the page linked.
UniqueID	Obtains the unique, hierarchically qualified identifier for the AdRotator control.

Following are the important events of the AdRotator class:

Events	Description
AdCreated	It is raised once per round trip to the server after creation of the control, but before the page is rendered
DataBinding	Occurs when the server control binds to a data source.
DataBound	Occurs after the server control binds to a data source.
Disposed	Occurs when a server control is released from memory, which is the last stage of the server control lifecycle when an ASP.NET page is requested
Init	Occurs when the server control is initialized, which is the first step in its lifecycle.
Load	Occurs when the server control is loaded into the Page object.
PreRender	Occurs after the Control object is loaded but prior to rendering.
Unload	Occurs when the server control is unloaded from memory.

Working with AdRotator Control

Create a new web page and place an AdRotator control on it.

```
<form id="form1" runat="server">
  <div>
    <asp:AdRotator ID="AdRotator1" runat="server" AdvertisementFile
    = "~/ads.xml" onadcreated="AdRotator1_AdCreated" />
  </div>
</form>
```

The ads.xml file and the image files should be located in the root directory of the web site.

Try to execute the above application and observe that each time the page is reloaded, the ad is changed.

Validation Control in Asp.Net

ASP.NET validation controls validate the user input data to ensure that useless, unauthenticated, or contradictory data don't get stored.

ASP.NET provides the following validation controls:

- RequiredFieldValidator
- RangeValidator
- CompareValidator
- RegularExpressionValidator
- CustomValidator
- ValidationSummary

BaseValidator Class

The validation control classes are inherited from the BaseValidator class hence they inherit its properties and methods. Therefore, it would help to take a look at the properties and the methods of this base class, which are common for all the validation controls:

Members	Description
ControlToValidate	Indicates the input control to validate.
Display	Indicates how the error message is shown.
EnableClientScript	Indicates whether client side validation will take.
Enabled	Enables or disables the validator.
ErrorMessage	Indicates error string.
Text	Error text to be shown if validation fails.
IsValid	Indicates whether the value of the control is valid.

SetFocusOnError	It indicates whether in case of an invalid control, the focus should switch to the related input control.
ValidationGroup	The logical group of multiple validators, where this control belongs.
Validate()	This method revalidates the control and updates the IsValid property.

RequiredFieldValidator Control

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.

The syntax of the control is as given:

```
<asp:RequiredFieldValidator ID="rfvcandidate"
  runat="server" ControlToValidate ="ddlcandidate"
  ErrorMessage="Please choose a candidate"
  InitialValue="Please choose a candidate">

</asp:RequiredFieldValidator>
```

RangeValidator Control

The RangeValidator control verifies that the input value falls within a predetermined range.

It has three specific properties:

Properties	Description
Type	It defines the type of the data. The available values are: Currency, Date, Double, Integer, and String.
MinimumValue	It specifies the minimum value of the range.

MaximumValue	It specifies the maximum value of the range.
--------------	--

The syntax of the control is as given:

```
<asp:RangeValidator ID="rvclass" runat="server" ControlToValidate="txtclass"
  ErrorMessage="Enter your class (6 - 12)" MaximumValue="12"
  MinimumValue="6" Type="Integer">

</asp:RangeValidator>
```

CompareValidator Control

The CompareValidator control compares a value in one control with a fixed value or a value in another control.

It has the following specific properties:

Properties	Description
Type	It specifies the data type.
ControlToCompare	It specifies the value of the input control to compare with.
ValueToCompare	It specifies the constant value to compare with.
Operator	It specifies the comparison operator, the available values are: Equal, NotEqual, GreaterThan, GreaterThanEqual, LessThan, LessThanEqual, and DataTypeCheck.

The basic syntax of the control is as follows:

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

</asp:CompareValidator>
```

RegularExpressionValidator

The RegularExpressionValidator allows validating the input text by matching against a pattern of a regular expression. The regular expression is set in the ValidationExpression property.

The following table summarizes the commonly used syntax constructs for regular expressions:

Character Escapes	Description
\b	Matches a backspace.
\t	Matches a tab.
\r	Matches a carriage return.
\v	Matches a vertical tab.
\f	Matches a form feed.
\n	Matches a new line.
\	Escape character.

Apart from single character match, a class of characters could be specified that can be matched, called the metacharacters.

Metacharacters	Description
.	Matches any character except \n.
[abcd]	Matches any character in the set.
[^abcd]	Excludes any character in the set.

[2-7a-mA-M]	Matches any character specified in the range.
\w	Matches any alphanumeric character and underscore.
\W	Matches any non-word character.
\s	Matches whitespace characters like, space, tab, new line etc.
\S	Matches any non-whitespace character.
\d	Matches any decimal character.
\D	Matches any non-decimal character.

Quantifiers could be added to specify number of times a character could appear.

Quantifier	Description
*	Zero or more matches.
+	One or more matches.
?	Zero or one matches.
{N}	N matches.
{N,}	N or more matches.
{N,M}	Between N and M matches.

The syntax of the control is as given:

```
<asp:RegularExpressionValidator ID="string" runat="server"
ErrorMessage="string"
ValidationExpression="string" ValidationGroup="string">

</asp:RegularExpressionValidator>
```

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 ClientValidationFunction property. The client side validation routine should be written in a scripting language, such as JavaScript or VBScript, which the browser can understand.

The server side validation routine must be called from the control's ServerValidate event handler. The server side validation routine should be written in any .Net language, like C# or VB.Net.

The basic syntax for the control is as given:

```
<asp:CustomValidator ID="CustomValidator1" runat="server"
ClientValidationFunction=.cvf_func. ErrorMessage="CustomValidator">

</asp:CustomValidator>
```

ValidationSummary

The ValidationSummary 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 syntax for the control is as given:

```
<asp:ValidationSummary ID="ValidationSummary1" runat="server"
DisplayMode = "BulletList" ShowSummary = "true" HeaderText="Errors:" />
```

Validation Groups

Complex pages have different groups of information provided in different panels. In such situation, a need might arise for performing validation separately for separate group. This kind of situation is handled using validation groups.

To create a validation group, you should put the input controls and the validation controls into the same logical group by setting their *ValidationGroup* property.

Calendar Control

The calendar control is a functionally rich web control, which provides the following capabilities:

- Displaying one month at a time
- Selecting a day, a week or a month
- Selecting a range of days
- Moving from month to month
- Controlling the display of the days programmatically

The basic syntax of a calendar control is:

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

Properties and Events of the Calendar Control

The calendar control has many properties and events, using which you can customize the actions and display of the control. The following table provides some important properties of the Calendar control:

Properties	Description
Caption	Gets or sets the caption for the calendar control.
CaptionAlign	Gets or sets the alignment for the caption.

CellPadding	Gets or sets the number of spaces between the data and the cell border.
CellSpacing	Gets or sets the space between cells.
DayHeaderStyle	Gets the style properties for the section that displays the day of the week.
DayNameFormat	Gets or sets format of days of the week.
DayStyle	Gets the style properties for the days in the displayed month.
FirstDayOfWeek	Gets or sets the day of week to display in the first column.
NextMonthText	Gets or sets the text for next month navigation control. The default value is >.
NextPrevFormat	Gets or sets the format of the next and previous month navigation control.
OtherMonthDayStyle	Gets the style properties for the days on the Calendar control that are not in the displayed month.
PrevMonthText	Gets or sets the text for previous month navigation control. The default value is <.
SelectedDate	Gets or sets the selected date.
SelectedDates	Gets a collection of DateTime objects representing the selected dates.
SelectedDayStyle	Gets the style properties for the selected dates.

SelectionMode	Gets or sets the selection mode that specifies whether the user can select a single day, a week or an entire month.
SelectMonthText	Gets or sets the text for the month selection element in the selector column.
SelectorStyle	Gets the style properties for the week and month selector column.
SelectWeekText	Gets or sets the text displayed for the week selection element in the selector column.
ShowDayHeader	Gets or sets the value indicating whether the heading for the days of the week is displayed.
ShowGridLines	Gets or sets the value indicating whether the gridlines would be shown.
ShowNextPrevMonth	Gets or sets a value indicating whether next and previous month navigation elements are shown in the title section.
ShowTitle	Gets or sets a value indicating whether the title section is displayed.
TitleFormat	Gets or sets the format for the title section.
Titlestyle	Get the style properties of the title heading for the Calendar control.
TodayDayStyle	Gets the style properties for today's date on the Calendar control.
TodaysDate	Gets or sets the value for today's date.

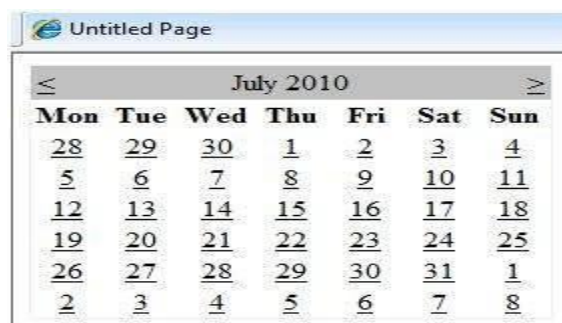
UseAccessibleHeader	Gets or sets a value that indicates whether to render the table header <th> HTML element for the day headers instead of the table data <td> HTML element.
VisibleDate	Gets or sets the date that specifies the month to display.
WeekendDayStyle	Gets the style properties for the weekend dates on the Calendar control.

The Calendar control has the following three most important events that allow the developers to program the calendar control. They are:

Events	Description
SelectionChanged	It is raised when a day, a week or an entire month is selected.
DayRender	It is raised when each data cell of the calendar control is rendered.
VisibleMonthChanged	It is raised when user changes a month.

Working with the Calendar Control

Putting a bare-bone calendar control without any code behind file provides a workable calendar to a site, which shows the months and days of the year. It also allows navigation to next and previous months.



Calendar controls allow the users to select a single day, a week, or an entire month. This is done by using the SelectionMode property. This property has the following values:

Properties	Description
Day	To select a single day.
DayWeek	To select a single day or an entire week.
DayWeekMonth	To select a single day, a week, or an entire month.
None	Nothing can be selected.

The syntax for selecting days:

```
<asp:Calender    ID    =    "Calendar1"    runat    =    "server"
SelectionMode="DayWeekMonth">
</asp:Calender>
```

When the selection mode is set to the value DayWeekMonth, an extra column with the > symbol appears for selecting the week, and a >> symbol appears to the left of the days name for selecting the month.

