UNIT II

Web Server Controls

Web Controls are very similar to the HTML Server controls such as Button, Textbox and Hyperlink except the web controls have a standardized set of property names.

Web controls provide the same functionality as their HTML server control. However, Web control includes additional methods, events, and properties.

Difference between HTML Control & ASP.NET Web Control

HTML Control	ASP.NET Web Control
• Runs at client side	• Runs at server side
• Can run on server side by adding runat="server"	• Cannot run on client side
• Does not provide state management	• Provide state management
• Executes fast	• Executes slow
• Do not have separate class for controls	• Have separate class for controls
• Does not support object oriented paradigm	• Support object oriented paradigm
• Limited set of properties and/or methods	• Rich set of properties and/or method
• Example <input type="text" ID="txtname">	• Example <asp:TextBox ID="txtname" runat="server"></asp:TextBox>

Following is the example of the simple web page which was developed using ASP.NET

```
<%@ Page Language="C#" AutoEventWireup="false"
CodeBehind="All_Control.aspx.cs" Inherits="Control_Example.All_Control"
%>
<head runat="server">
<title> Example of Control</title>
</head>
<body>
<form id="frmAllControl" runat="server">
<table border="1" cellpadding="5" align=center>
<tr>
<td>
<asp:Label ID="lblFirstName" runat="server" Text="Enter Your First
Name:"></asp:Label>
</td>
<td>
<asp:TextBox ID="txtFirstName" runat="server"
Width="180px"></asp:TextBox>
</td>
</tr>
</table>
</form>
</body>
</html>
```

WORKING WITH PROPERTIES, EVENTS & METHODS OF SERVER CONTROLS **

Asp.net provides many built in server controls that can be easily drag and drop to the web pages.

Common Properties for all controls

Property	Description
BackColor	Background color.
BorderColor	Border color.
CssClass	CSS class.
Height	Gets or sets the Height of the Web server control.

ID	Identifier for the control.
ToolTip	Gets or sets the text displayed when the mouse pointer hovers over the web server control.
Visible	It indicates whether a server control is visible.
Width	Gets or sets the width of the Web server control.
Runat	Server side Controls are useless without runat="server" Property.
AutoPostBack	AutoPostBack property is used to set or return whether or not an automatic post back occurs. If this property is set to TRUE the automatic post back is enabled, By default AutoPostBack is FALSE.
Text	Gets or sets the text caption displayed on the control.

Common Event for all controls

Event	Description
Click	The Button, HyperLink, ImageButton, and LinkButton controls send this event when users click them.
SelectedIndexChanged	The ListBox, DropDownList, CheckBoxList, and RadioButtonList controls send this event when the selected index is changed.(For Databound Controls)
CheckedChanged	The CheckBox, RadioButton control sends this event when the control becomes checked or unchecked. (For Databound Controls)
TextChanged	The TextBox control send this event when text in the TextBox Changed.
Init	Called when the control is initialized. This is the first event called for every control.
Load	Called when the Page object loads the control.
PreRender	Called right before the control is rendered into the HTML result stream.
Disposed	Called when the control is released from memory. This call can happen at any time after the page is fully rendered, when the .NET garbage collector runs.

Button

Displays a push button.

Buttons in an ASP.NET Web page allow users to send a command.
Buttons submit the page to the server and cause it to be processed along with any pending events.

ASP.NET provides three types of button control.

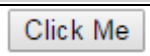
1. Button: It displays text within a rectangular area.
2. Link Button: It displays text that looks like a hyperlink.
3. Image Button: It displays an image.

Special Attributes:

- ImageUrl - For image button control only. The image to be displayed for the button.
- PostBackUrl - The URL of the page that is requested when the user clicks the button.
- OnClick - Attach a server side method that will fire when button will be clicked
- OnClientClick - Attach a client side (javascript) event that will fire when button will be clicked
- CauseValidation – This indicates whether validation will be performed when a button is clicked

Example

```
<asp:Button ID="Button1" runat="server" Text="Click Me" />
```



Textbox

The TextBox control is used to create a text box where the user can input text.
The TextBox Web server control provides a way for users to type information into an ASP.NET Web page, including text, numbers, and dates.
This control is used to take input from user into a default string format.

Special Attributes:

- TextMode - Specifies the type of text box. SingleLine creates a standard text box, MultiLIne creates a text box that accepts more than one line of text and the Password causes the characters that are entered to be masked. The default is SingleLine.
- MaxLength - The maximum number of characters that can be entered

into the text box.

- ReadOnly - Determines whether the user can change the text in the box; default is false, i.e., the user can change the text.

Example

```
<asp:TextBox ID="TextBox1" runat="server" MaxLength="10" TextMode="MultiLine"></asp:TextBox>
```

Label

The Label Web server control provides a way to display text programmatically control in an ASP.NET Web page.
This Control is used to display information on web page.

Example

```
<asp:Label ID=" Label1" runat="server" Visiable="true" Text="Hi this is darshan"></asp:Label>
```

Hi this is darshan

CheckBox

This control is used to select multiple values from the list.
Display a check box that allows the user to select value, if it is select then its value will becomes TRUE otherwise its value will becomes FALSE.

Special Attributes:

- Checked - Specifies whether it is selected or not, default is false.
- TextAlgn – Side of the checkbox where the text will be displayed

Example:-

```
<asp:CheckBox ID=" CheckBox1" runat="server" AutoPostBack="true" Visible="true" > </asp:CheckBox>
```

Radio Button

The RadioButton control is used to display a radio button.
RadioButton present a group of options from which the user can select just one option.

Special Attributes:

- GroupName - Name of the group the control belongs to.
- Checked - Specifies whether it is selected or not, default is false.
- TextAlgn – Side of the checkbox where the text will be displayed

Example

```
<asp:RadioButton ID="RadioButton1" runat="server" GroupName="Gender" Text="Male" />
<asp:RadioButton ID="RadioButton2" runat="server" GroupName="Gender" Text="Female" />
```

Male Female

Link Button

The LinkButton control is used to create a hyperlink button.
Displays a hyperlink-style button control on a Web page.
This control looks like a HyperLink control but has the same functionality as the Button control.

Example

```
<asp:LinkButton ID="LinkButton1" runat="server" PostBackUrl="~/Home.aspx">Click Here!</asp:LinkButton>
```

Image

The Image control is used to display an image on the web page.

Click Here!

Special Attributes:

- AlternateText - Alternate text to be displayed in absence of the image.
- ImageAlign - Alignment options for the control.
- ImageUrl - Path of the image to be displayed by the control.

Example

```
<asp:Image ID="Image1" runat="server" AlternateText="Image Not Available!" ImageAlign="Left" ImageUrl="~/App_Data/bird.jpg" />
```

Hyperlink

The HyperLink control is used to create a hyperlink.
This control provides easy navigation between various Pages.

Special Attributes:

- NavigateUrl - The target URL of the link.
- Target - The target frame of the URL.

Example

```
<asp:HyperLink ID="HyperLink1" runat="server"
NavigateUrl=https://www.google.co.in
Target="_blank">HyperLink</asp:HyperLink>
```

Panel

The Panel control works as a container for other controls on the page.
It controls the appearance and visibility of the controls it contains.

Special Attributes:

- BackImageUrl - URL of the background image of the panel.
- Direction - Text direction in the panel.
- HorizontalAlign - Horizontal alignment of the content in the panel.
- ScrollBars - Specifies visibility and location of scrollbars within the panel.
- Wrap - Allows text wrapping.

Example

```
<asp:Panel ID="panel1" runat="server" HorizontalAlign="Left"
ScrollBars="Auto" Direction="LeftToRight" Wrap="true" Visible="false">
Hey This is example of Panel Control <br />It can contains more than one
control <br /> </asp:Panel>
```

PlaceHolder

PlaceHolder is an asp.net web server control which used to store dynamically added web server controls on the web page. by using a PlaceHolder control we can dynamically add Label, TextBox, Button, RadioButton, Image and many more web server controls in an asp.net web page.
PlaceHolder server control act as a container control to store server controls that are dynamically added to the web page.
PlaceHolder control does not provide any visible output. We only can see the dynamically added server controls inside a PlaceHolder control as child controls. We can add, insert and remove server controls programmatically in the PlaceHolder control.

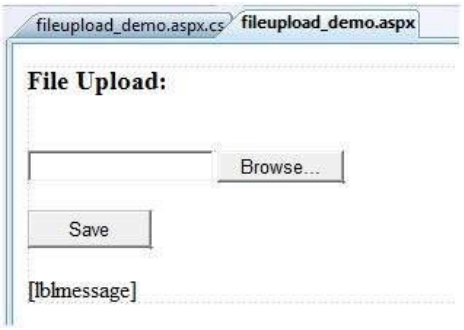
File Upload

The FileUpload control allows the user to browse for and select the file to be uploaded, providing a browse button and a text box for entering the filename. Once, the user has entered the filename in the text box by typing the name or browsing, the SaveAs method of the FileUpload control can be called to save the file to the disk.

Example

The following example demonstrates the FileUpload control and its properties. The form has a FileUpload control along with a save button and a label control for displaying the file name, file type, and file length.

In the design view, the form looks as follows:



The content file code is as given:

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

    <div>
      <h3> File Upload:</h3>
      <br />
      <asp:FileUpload ID="FileUpload1" runat="server" />
      <br /><br />
      <asp:Button ID="btnsave" runat="server" onclick="btnsave_Click"  Text="Save"
style="width:85px" />
      <br /><br />
      <asp:Label ID="lblmessage" runat="server" />
    </div>

  </form>
</body>
```

The code behind the save button is as given:

```
protected void btnsave_Click(object sender, EventArgs e)
{
  StringBuilder sb = new StringBuilder();

  if (FileUpload1.HasFile)
  {
    try
    {
      sb.AppendFormat(" Uploading file: {0}", FileUpload1.FileName);

      //saving the file
      FileUpload1.SaveAs("<c:\\SaveDirectory>" + FileUpload1.FileName);

      //Showing the file information
      sb.AppendFormat("<br/> Save As: {0}",  FileUpload1.PostedFile.FileName);
      sb.AppendFormat("<br/> File type: {0}",    FileUpload1.PostedFile.ContentType);
      sb.AppendFormat("<br/> File length: {0}",  FileUpload1.PostedFile.ContentLength);
      sb.AppendFormat("<br/> File name: {0}",  FileUpload1.PostedFile.FileName);

    }catch (Exception ex)
    {
      sb.Append("<br/> Error <br/>");
      sb.AppendFormat("Unable to save file <br/> {0}", ex.Message);
    }
  }
  else
  {
    lblmessage.Text = "NO FILE IS SELECTED……..."
  }
}
```

LIST CONTROLS

Asp.Net provides many built in List controls to show the data in the form of list.
User can select one or more value from the list.
List controls are many and they are used as per user's requirement

CheckBoxList

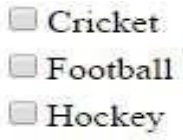
Provides a checkbox with each data and user can select more than one item.
A check box list presents a list of independent options.
These controls contain a collection of ListItem objects that could be referred to through the Items property of the control.

Special Attributes:

- 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 - Specifies the number of columns to use when repeating the controls; default is 0.

Example

```
<asp:CheckBoxList ID="CheckBoxList1" runat="server">
<asp:ListItem>Cricket</asp:ListItem>
<asp:ListItem>Football</asp:ListItem>
<asp:ListItem>Hockey</asp:ListItem>
</asp:CheckBoxList>
```



ListBox

Provides all the data in a box format and user can select one and more data.
List boxes contain one or more list items.
These lists can be loaded either by code or by the ListItem objects.

Special Attributes:

- Items - The collection of ListItem objects that represents the items in the control.
- Rows - Specifies the number of items displayed in the box. If actual list contains more rows than displayed then a scroll bar is added.
- SelectionMode - Indicates whether a list box allows single selections or multiple selections.
- SelectedIndex - The index of the currently selected item. If more than one item is selected, then the index of the first selected item. If no item is selected, the value of this property is -1.
- SelectedValue - The value of the currently selected item. If more than one item is selected, then the value of the first selected item. If no item is selected, the value of this property is an empty string ("").

Example

```
<asp:ListBox ID="ListBox1" runat="server" SelectionMode="Multiple">
<asp:ListItem>Apple</asp:ListItem>
<asp:ListItem>Banana</asp:ListItem>
<asp:ListItem>Mango</asp:ListItem>
</asp:ListBox>
```

DropDownList

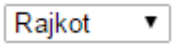
Provides a list of items with only one item visible at a time and user can select single or multiple items.

Special Attributes:

- Items - The collection of ListItem objects that represents the items in the control.

Example

```
<asp:DropDownList ID="DropDownList1" runat="server">
<asp:ListItem>Rajkot</asp:ListItem>
<asp:ListItem>Morbi</asp:ListItem><asp:ListItem>Jamnagar</asp:ListItem>
```




```
</asp:DropDownList>
```

ASP.Net AdRotator Control

AdRotator control is used to display a sequence of advertisement images on webpage

Adrotator control displays the sequence of images, which is specified in the external XML file.

In a xml file we indicate the images to display with some other attributes, like image impressions, NavigateUrl, ImageUrl, AlternateText.

In a Adrotator control images will be changed each time while refreshing the web page.

The basic syntax of adding an AdRotator is as follows:

```
<asp:AdRotator ID=”AdRotator1” runat = "server" AdvertisementFile = "adfile.xml" />
```

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.

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>
    <Keyword>flowers</Keyword>
  </Ad>

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

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

  <Ad>
    <ImageUrl>rose4.jpg</ImageUrl>
    <NavigateUrl>http://www.edibleblooms.com</NavigateUrl>
    <AlternateText>Edible Blooms</AlternateText>
    <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.

MultiView and View controls

MultiView and View controls allow you to divide the content of a page into different groups, displaying only one group at a time.

Each View control manages one group of content and all the View controls are held together in a MultiView control.

The MultiView control is responsible for displaying one View control at a time. The View displayed is called the **active view.**

The syntax of MultiView control is:

```
<asp:MultView ID= "MultiView1" runat= "server">
</asp:MultiView>
```

The syntax of View control is:

```
<asp:View ID= "View1" runat= "server">
</asp:View>
```

IMP NOTE- the View control cannot exist on its own. It would render error if you try to use it stand-alone. It is always used with a Multiview control as:

```
<asp:MultView ID= "MultiView1" runat= "server">
  <asp:View ID= "View1" runat= "server"> </asp:View>
</asp:MultiView>
```

Properties of View and MultiView Controls

Both View and MultiView controls are derived from Control class and inherit all its properties, methods, and events. The most important property of the View control is Visible property of type Boolean, which sets the visibility of a view.

The MultiView control has the following important properties:

Properties	Description
Views	Collection of View controls within the MultiView.
ActiveViewIndex	A zero based index that denotes the active view. If no view is active, then the index is -1.

The CommandName attribute of the button control associated with the navigation of the MultiView control are associated with some related field of the MultiView control.

For example, if a button control with CommandName value as NextView is associated with the navigation of the multiview, it automatically navigates to the next view when the button is clicked.

The following table shows the default command names of the above properties:

Properties	Description
NextViewCommandName	NextView
PreviousViewCommandName	PrevView
SwitchViewByIDCommandName	SwitchViewByID
SwitchViewByIndexCommandName	SwitchViewByIndex

The important methods of the multiview control are:

Methods	Description
SetActiveview	Sets the active view
GetActiveview	Retrieves the active view

Every time a view is changed, the page is posted back to the server and a number of events are raised. Some important events are:

Events	Description
ActiveViewChanged	Raised when a view is changed
Activate	Raised by the active view
Deactivate	Raised by the inactive view

Apart from the above mentioned properties, methods and events, multiview control inherits the members of the control and object class.

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

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.

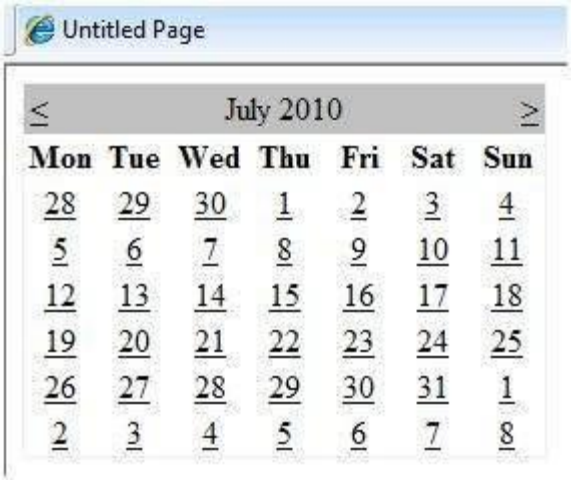
Titlestyle	Get the style properties of the title heading for the Calendar control.
TodayDayStyle	Gets the style properties for today's date on the Calendar control.
TodaysDate	Gets or sets the value for today's date.
VisibleDate	Gets or sets the date that specifies the month to display.
WeekendDayStyle	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.



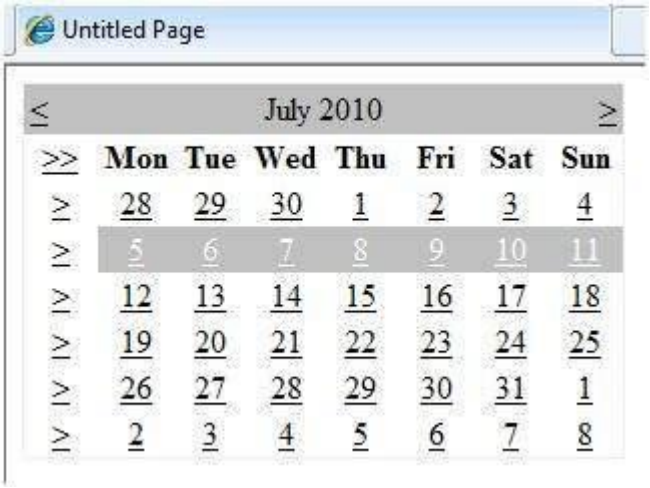
Calendar controls allow the users to select a single day, a week, or an entire month. This is done by using the SelectionMode property. This property has the following values:

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.



Example

The following example demonstrates selecting a date and displays the date in a label:

The content file code is as follows:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="calendardemo._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

  <head runat="server">
    <title>
      Untitled Page
    </title>
  </head>

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

      <div>
        <h3> Your Birthday:</h3>
        <asp:Calendar ID="Calendar1" runat="server  SelectionMode="DayWeekMonth"
onselectionchanged="Calendar1_SelectionChanged">
        </asp:Calendar>
      </div>
```

```
        <p>Todays date is:
          <asp:Label ID="lblday" runat="server"></asp:Label>
        </p>

        <p>Your Birday is:
          <asp:Label ID="lblbday" runat="server"></asp:Label>
        </p>

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

The event handler for the event SelectionChanged:

```
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
  lblday.Text = Calendar1.TodaysDate.ToShortDateString();
  lblbday.Text = Calendar1.SelectedDate.ToShortDateString();
}
```

When the file is run, it should produce the following output:

Your Birthday:

≤	December 2010						≥
>>	Mon	Tue	Wed	Thu	Fri	Sat	Sun
≥	29	30	1	2	3	4	5
≥	6	7	8	9	10	11	12
≥	13	14	15	16	17	18	19
≥	20	21	22	23	24	25	26
≥	27	28	29	30	31	1	2
≥	3	4	5	6	7	8	9

Todays date is: 11-07-2010

Your Birday is: 16-12-2010

Create Web User Control-

In addition to using Web server controls in your ASP.NET Web pages, you can create your own custom, reusable controls using the same techniques you use for creating ASP.NET Web pages. These controls are called user controls.

A user control is a kind of composite control that works much like an ASP.NET Web page—you can add existing Web server controls and markup to a user control, and define properties and methods for the control. You can then embed them in ASP.NET Web pages, where they act as a unit.

User controls are containers into which you can put markup and Web server controls. You can then treat the user control as a unit and define properties and methods for it.

User Control Structure

An ASP.NET Web user control is similar to a complete ASP.NET Web page (.aspx file), with both a user interface page and code.

You create the user control in much the same way you create an ASP.NET page and then add the markup and child controls that you need.

A user control can include code to manipulate its contents like a page can, including performing tasks such as data binding.

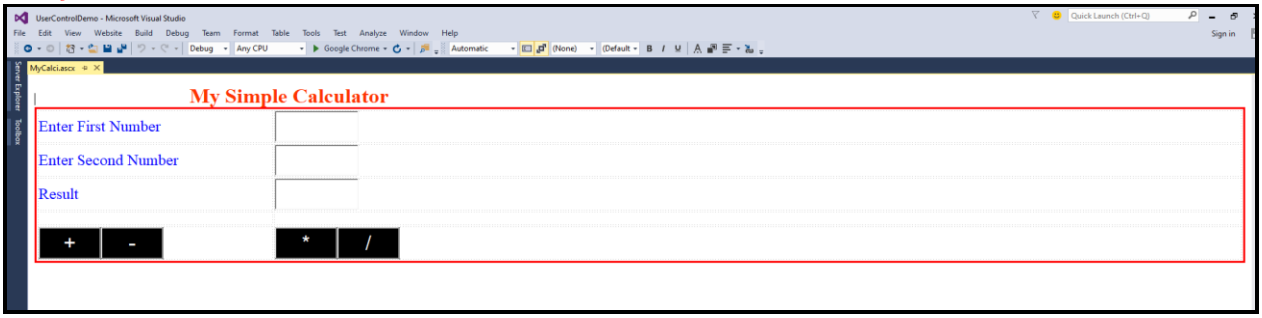
A user controls differs from an ASP.NET Web page in these ways:

- The file name extension for the user control is **.ascx**.
- Instead of an @ Page directive, the user control contains an **@ Control** directive that defines configuration and other properties.
- User controls cannot run as stand-alone files. Instead, you must add them to ASP.NET pages, as you would any control.
- The user control does not have html, body, or form elements in it. These elements must be in the hosting page.

You can use the same HTML elements (except the html, body, or form elements) and Web controls on a user control that you do on an ASP.NET Web page.

For example, if you are creating a user control to use as a toolbar, you can put a series of Button Web server controls onto the control and create event handlers for the buttons.

MyCalci.ascx



MyCalci.ascx.cs-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class MyCalci : System.Web.UI.UserControl
{   protected void Page_Load(object sender, EventArgs e)
    {
    }
   protected void Button1_Click(object sender, EventArgs e)
    {
        int rs = Convert.ToInt32(TextBox1.Text) +
Convert.ToInt32(TextBox2.Text);
        TextBox3.Text = rs.ToString();
    }
  protected void Button2_Click(object sender, EventArgs e)
    {
        int rs = Convert.ToInt32(TextBox1.Text) -
Convert.ToInt32(TextBox2.Text);
        TextBox3.Text = rs.ToString();
    }
  protected void Button3_Click(object sender, EventArgs e)
    {
```

```
        int rs = Convert.ToInt32(TextBox1.Text) *
Convert.ToInt32(TextBox2.Text);
        TextBox3.Text = rs.ToString();
    }

    protected void Button4_Click(object sender, EventArgs e)
    {
        int rs = Convert.ToInt32(TextBox1.Text) /
Convert.ToInt32(TextBox2.Text);
        TextBox3.Text = rs.ToString();
    }
}
```

Following Web Page uses User Control MyCalci

One.aspx

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

<%@ Register src="MyCalci.ascx" tagname="MyCalci" tagprefix="uc1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

          <uc1:MyCalci ID="MyCalci1" runat="server" />

    </div>
    </form>
</body>
</html>
```