GANPAT UNIVERSITY
U. V. PATEL COLLEGE OF ENGINEERING

2CEIT403
APPLICATION DEVELOPMENT TOOLS

UNIT 5

WORKING WITH ASP.NET

Prepared by: Prof. Y. J. Prajapati (Asst. Prof in IT Dept. , UVPCE)

Outline

- Introduction to three-tier Client Server systems
- Introduction of Web Application
- Introduction of ASP.NET
- ASP.Net Page Life Cycle
- ISPOSTBACK Property
- ASP.Net Web application EVENTS
- ASP.Net Server Controls
- ASP.Net Built in Objects
- State Management in ASP.Net
- Introduction To ASP.Net AJAX
- Introduction to LINQ
- ASP.NET Security

Introduction to three-tier Client Server systems

Basically three tier architecture means our project **divided into three main layers** or we can also say our project developed and maintained in to three separate layers.

1. Presentation Layer (UI – User Interface Layer)
2. Business Logic Layer (use for write logic code)
3. Data Access Layer (DAL use for connectivity with Database)

Cont..

1. Presentation Layers or User Interface Layer

- Presentation layer is a user interface layer where we can design our web page or windows page.
- .aspx page where we can make design with controls.

Cont..

2. **Business logic layer (BLL)**

- Business layer **is intermediate or middle** layer that communicate **presentation layer and Data access layer**.
- Business layer used to **validate user input** before calling method from the data layer.

Cont..

3. Data Access Layer (DAL)

- Data Access Layer used to make connection with database server.
- In data access layer we can write **database query, stored procedure for insert, update, delete, select operation on database.**
- This layer only communicate with **Business logic layer**.

2. Introducing Web Applications

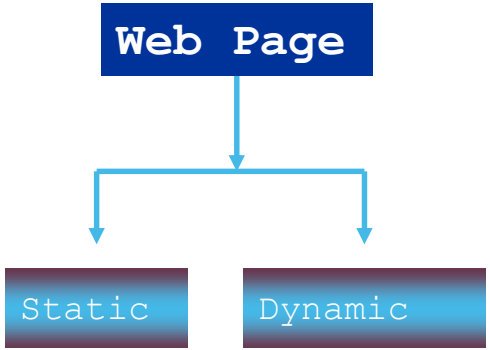
Key Points about Web Application.

1. **Web applications** are **executed on the server.**
2. Web applications are **stateless**:

HTTP is a Stateless protocol

- A **stateless server** is a server that treats **each request as an independent transaction** that is unrelated to any previous request.

2. Introducing Web Applications



Static web page

- A static website is the most basic type of website and contains web pages with fixed content

- Each page is coded in HTML and displays the same information to every user.

- **Examples of static web page** include about us page with a corporate website, mission, vision etc.

Contact Us

Indian Institute of Technology Gandhinagar
Palaj, Gandhinagar - 382355, Gujarat

Emails

Phones

Social

Offices	Email
Academics	academics@iitgn.ac.in
Accounts	accounts@iitgn.ac.in
Administration	genadmin@iitgn.ac.in
Director's Office	director@iitgn.ac.in
Faculty Affairs	ar.fa@iitgn.ac.in

Static Web Page

University | Engineering
॥ विद्यया समाजोत्कर्षः ॥

About + Admission + Academics + Department + Life at UVPCE +

Home / About / About College

About

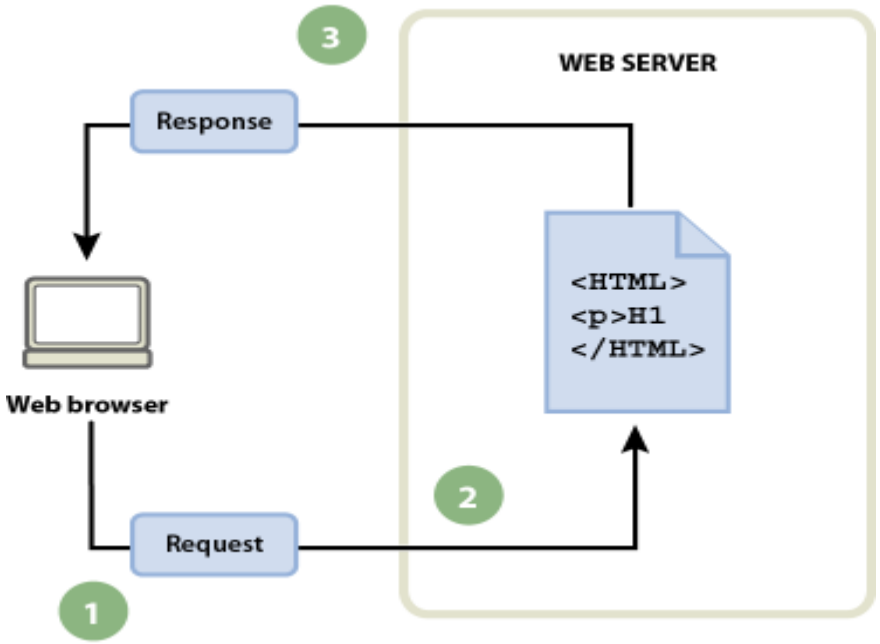
- About College
- About University
- Vision & Mission
- Principal & Dean Message
- People
  - Head Of Departments
  - PG Coordinators
  - Faculty Members
  - Staff
  - Directory
- The Management

About College

Our spirituous efforts are directed towards leading our student community to such an acme of technical excellence meeting the requirements of the industry, the nation and the globe at large. Nurturing an entirely

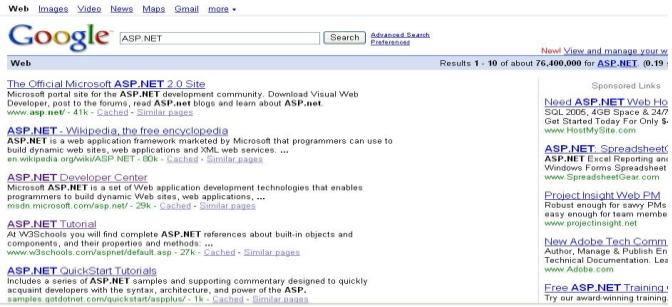
Example-

A. Web browser requests static page.
B. Web server finds page.
C. Web server sends page to requesting browser



Dynamic web page

- A dynamic website contains information that **changes, depending on the viewer, geographical location, time of the day and other factors**.
- They utilize **databases** and other mechanisms that enable to
  - identify **their visitors**
  - present them with customized **Greeting(Welcome) messages**
  - **restructure the content according** to user input etc..(user/admin dashboard)
- Examples:
  - Online shopping stores,
  - search engines
  - Email,
  - social media
  - etc.

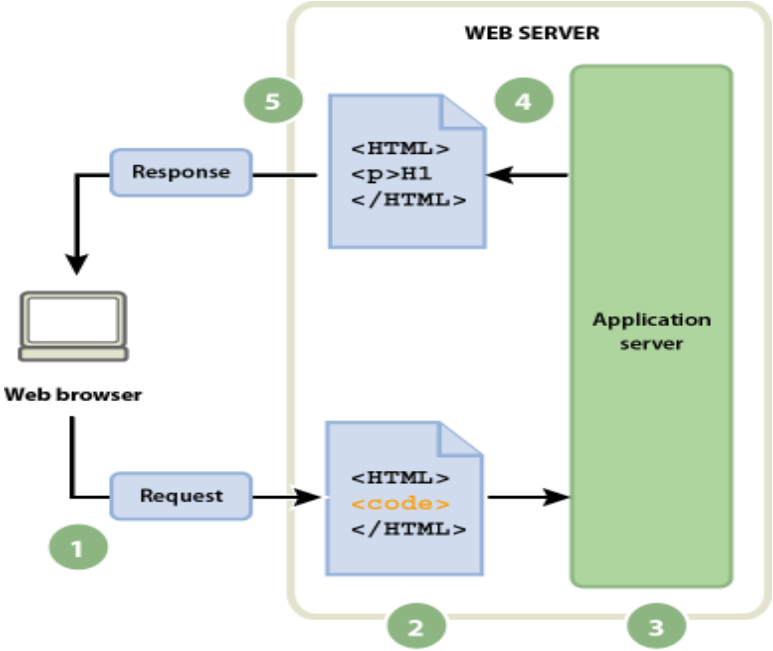


Dynamic web page

- Dynamic Web sites make use of "server-side technology".
- The web server will first
  - interpret the server-side code present in web pages,
  - generate the appropriate HTML and then
  - send the response to the web browser.

Example-

1. Web browser requests dynamic page.
2. Web server finds page and passes it to application server.
3. Application server scans page for instructions and finishes page.
4. Application server passes finished page back to web server
5. Web server sends finished page to requesting browser



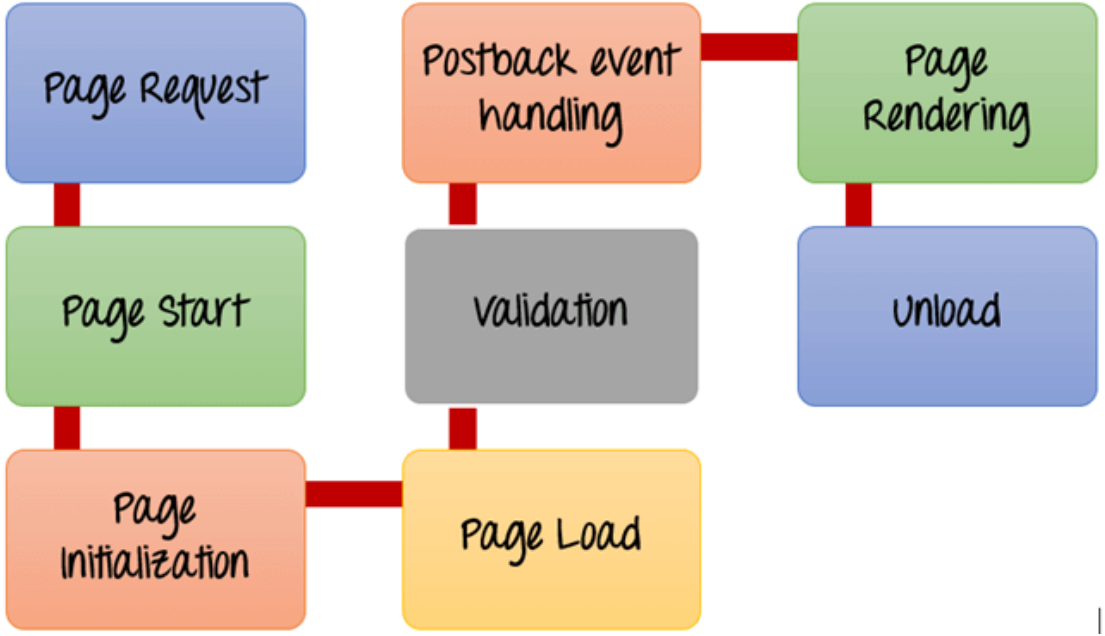
Introduction OF ASP.NET

- ASP.NET is part of the Microsoft .NET framework
- ASP.NET is an effective and flexible technology for creating **interactive and dynamic web Application.**
- It is a convergence of two major Microsoft technologies:
  - **Active Server Pages (ASP)**
    - Active Server Pages is Microsoft's server side scripting technology for building dynamic web pages.
  - **.NET Framework**
    - The .NET Framework is a suite of technologies designed by Microsoft where program development takes place.

Introduction OF ASP.NET

- It is built on .NET **Common Language Runtime**
- ASP.NET :
  - Provides better **user authentication**
  - Has better language support. (C#,Visual Basic.Net,Jscript,J#)
  - Has **a large set of new controls**
- The ASP.NET pages(web Form) are saved with the **.aspx extension.**

ASP.NET Page Life Cycle



ASP.NET Page Life Cycle

1) Page Request

This is when the page is first time requested **(browser-server)**.

When the page is requested, **the server checks if it is requested for the first time. If so, then it compiles the page and generate output.**

If it is not the first time the **page is requested, the cache is checked to see if the page output exists.** If so, then cached compiled output is taken.

2) Page Start

During this time, 2 objects, known as the Request and Response object are created.

When a browser asks for a page from a server, it is called a request. The Request object is used to get information from a user.

The ASP Response object is used to send output to the user from the server.

ASP.NET Page Life Cycle

3) Page Initialization

During this time, **all the controls on a web page is initialized.**

So if you have use any **label, textbox or any other controls on the web form**, they are all initialized.

4) Page Load

This is **when the page is actually loaded** with all **the default values.**

So **if a textbox is supposed to have a default value, that value is loaded during the page load time.**

5) Validation

Sometimes there can be some validation set on the form.

For example, mobile no,email id, password

ASP.NET Page Life Cycle

6) Postback event handling

- This event is triggered **if the same page is being loaded again.**
- Sometimes there can be a situation that a user clicks on a **submit button on the page.** In this case, the **same page is displayed again**. In such a case, the **Postback event handle is called.**

7) Page Rendering

- This happens **just before all the response information is sent to the user from server.**
- Rendering is the phase **where the response from server(in form of HTML)sent to the browser**.

ASP.NET Page Life Cycle

8) Unload

- **Once the page output is sent to the user,** there is no need to keep the **ASP.net web form objects** in memory.
- So the **unloading process involves removing all unwanted objects from memory.**

IsPostBack Property

- IsPostBack is **a Page level Property.**
- It returns **bool value.**
- It can be used to **determine whether the page is being loaded in response to some event generated by some control** or Page is loaded **first time.**

- Postback is actually sending all the information from client to web server, then web server process all those contents and returns back to the client. Most of the time ASP control will cause a post back (e. g. buttonclick) but some don't unless you tell them to do In certain events (Listbox Index Changed,RadioButton Checked etc..) in an ASP.NET page upon which a PostBack might be needed.

ASP.Net Web application Events

Events can occurs at 3 levels.

1. At the application level
2. At the Page level
3. At the Control level.

1.Application level Events(Global.asax)

The Global.asax is also known as the ASP.NET application file and is used to serve **application-level and session-level events**.

It allows us to write code that response to **application events raised by ASP.NET or by HttpModules.**

The Global.asax file (also known as the ASP.NET application file) is an optional file

Major Events()

1. Application_Start()
2. Application_End()
3. Session_Start()
4. Session_End()

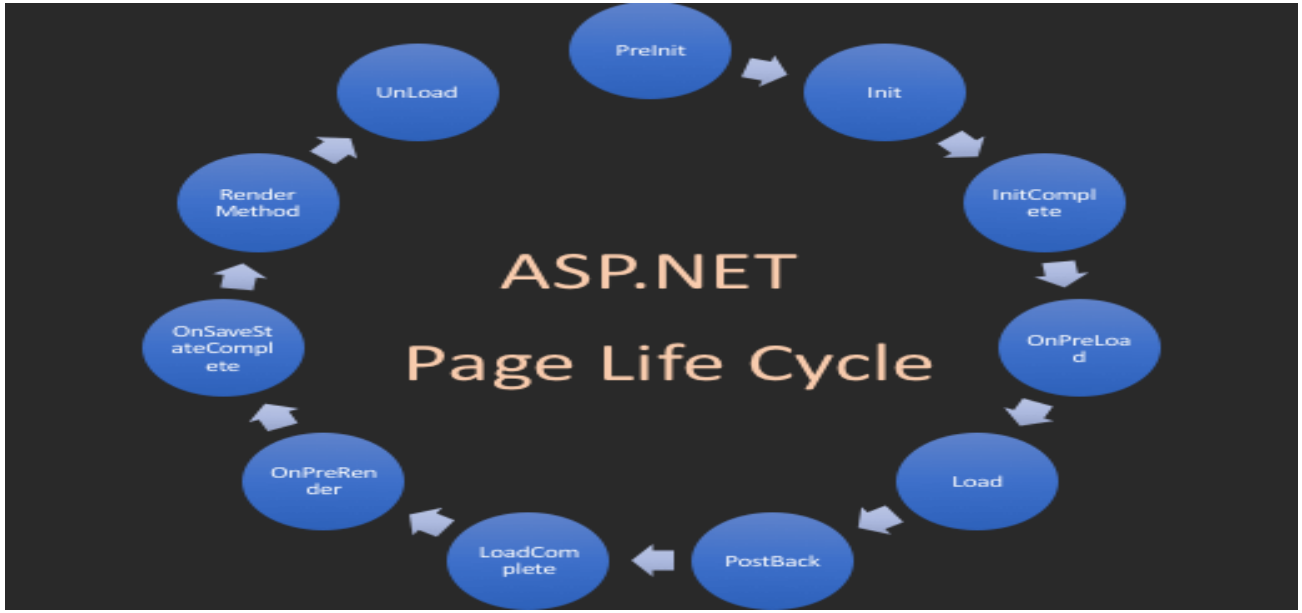
1. Application level Events(Global.asax)

```
void Application_Start(object sender, EventArgs e)
{
    // Code that runs on application startup

}

void Application_End(object sender, EventArgs e)
{
    //  Code that runs on application shutdown

}

void Application_Error(object sender, EventArgs e)
{
    // Code that runs when an unhandled error occurs

}

void Session_Start(object sender, EventArgs e)
{
    // Code that runs when a new session is started

}

void Session_End(object sender, EventArgs e)
{
    // Code that runs when a session ends.
    // Note: The Session_End event is raised only when the sessionstate mode
    // is set to InProc in the Web.config file. If session mode is set to StateSer
    // or SQLServer, the event is not raised.
```

Application_Start()	This method is invoked when the application first starts up and the application domain is created. This event handler is a useful place to provide application-wide initialization code. For example, at this point you might load and cache data that will not change throughout the lifetime of an application, such as navigation trees, static product catalogs, and so on.
Application_End()	This method is invoked just before an application ends. The end of an application can occur because IIS is being restarted or because the application is transitioning to a new application domain in response to updated files or the process recycling settings.
Application_Error()	This method is invoked whenever an unhandled exception occurs in the application.

Session_Start()	This method is invoked each time a new session begins. This is often used to initialize user-specific information.
Session_End()	This method is invoked whenever the user's session ends. A session ends when your code explicitly releases it or when it times out after there have been no more requests received within a given timeout period (typically 20 minutes).

2. Page level Events



1.Preinit

1. Raised when **client request for a page** and **before the initialization stage begins.**
2. Create **Dynamic Controls.**
3. **Set Master Page and Theme Dynamically.**

```
protected void Page_PreInit(object sender, EventArgs e)
{

}
```

2.Init

1. Raised **when Controls are initialized**.
2. **Init event of Control** occure before **init event of Page**.
3. Used to **get or set Control Properties**.

```
protected void Page_Init(object sender, EventArgs e)
{

}
```

3. InitComplete

1. Raised at the **end of Page initialization.**
2. **ViewState** is not loaded yet.
3. This event can be used for processing task that **required all initialization to be completed.**

```
protected void Page_InitComplete(object sender, EventArgs e)
{

}
```

4.PreLoad

1. Use this event if you need to perform processing on your **page or control before the Load event.**
2. After the Page raises this event, **it loads view state for itself and all controls.**

```
Protected void Page_PreLoad(object sender, EventArgs e)
{
}
```

5.Load

1.First Page calls load method **on page object**.

2. Then recursively **load is called for all controls.**

```
Protected void Page_Load(object sender, EventArgs e)
{
}
```

6. Control Events

Use this event handle specific control events.

7. LoadComplete

1. Raised at the **end of the event-handling stage**.
2. Use this event for tasks that require that **all other controls on the page be loaded**

8. OnPreRender

1. This is the final stop in the page load cycle **where you can make changes to page contents or controls.**
2. The Page object raises the **PreRender event on the Page object,** and then **recursively does the same for each child control.**
3. The PreRender event of individual controls occurs after the **PreRender event of the page.**
4. Allows final changes to the page or its control.
5. **For example:** After this event, **you cannot change any property of a button**

9. OnSaveStateComplete

1. Raised after **view state and control state have been saved for the page and for all controls.**
2. Before this event occurs, **ViewState has been saved for the page and for all controls.**

10. Render

1. This is a **method of the page object and its controls (and not an event).**
2. The Render method **generates the client-side HTML, Dynamic Hypertext Markup Language** (DHTML), and **script that are necessary to properly display** a control at the browser.

11. UnLoad

1. This event is **used for clean up code**.
2. This event is raised for **each control** and **then for page.**

3. Events Occurred at Control Level

- ASP.Net Server Controls such as Texxtbox,button ,dropdownList,
- All controls have their own events. We also have validation controls.

3 typed of Control level events.

1.Postback Events.

2.Cached Events.

3.Validation Events.

3. Events Occurred at Control Level

1.Postback Events.

- This event **submit the page immediately to Server** for processing .
- For Example **Click Event** of Button

2.Cached Events.

- This Events are saved in the Pages's View State and Processed when Postback Event occurs.
- Example ---TextChange—>TextBox,
- SelectedindexChange—>DropdownList

3. Events Occurred at Control Level

3. Validation Events.

- **Validation events occur on the client before the form/page is postback to server.**

ASP.Net Server Controls

System.Object

System.Web.UI.Control

System.Web.UI.HtmlControls

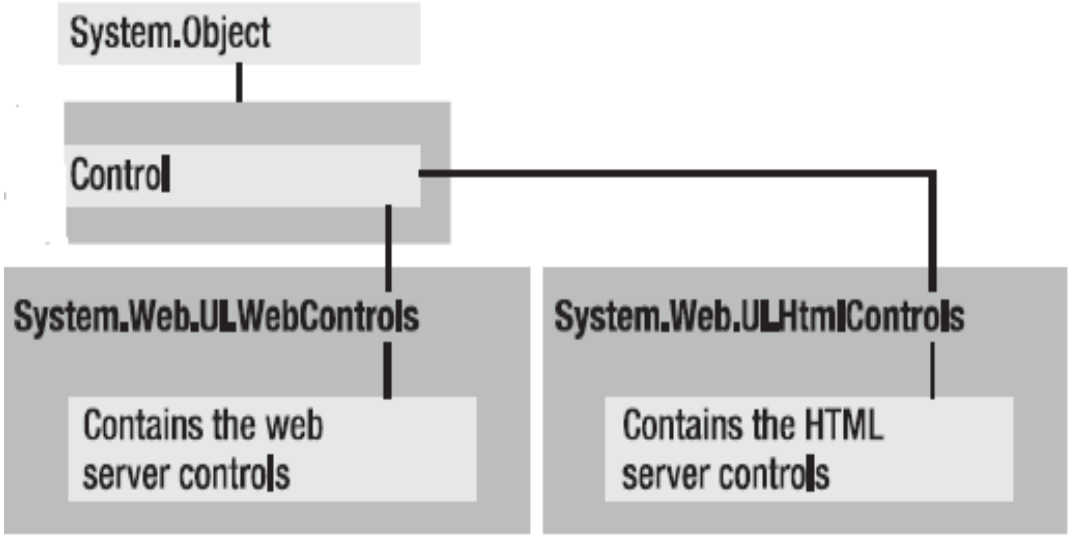
System.Web.UI.WebControls

ASP.Net Server Controls

There are three kinds of server controls:

1. HTML Server Controls - Traditional HTML tags
2. Web Server Controls - New ASP.NET tags
3. Validation Server Controls - For input validation

ASP.Net Server Controls



Properties of the Control Class

Property	Description
ClientID	Gets the ID assigned to the control in the HTML page. The string is a slightly different version of the UniqueID property. *UniqueID* can contain the colon symbol (:), but this symbol is not allowed in *ClientID* and is replaced with the underscore (_).
Controls	Gets a collection filled with references to all the child controls.
EnableViewState	Gets or sets whether the control should persist its view state—and the view state of any child controls across multiple requests—to the configured location (for example, HTML hidden field, Web server memory, server-side databases or files).
ID	Gets or sets the name that will be used to programmatically identify the control in the ASP.NET page.
NamingContainer	Gets a reference to the control's naming container. A naming container is the namespace to which the control belongs. If the control doesn't define its own naming space, a reference to the parent control (or the page) is returned.
Page	Gets a reference to the *Page* instance that contains the control.
Parent	Gets a reference to the parent of the control in the page hierarchy.
TemplateSourceDirectory	Gets the virtual directory of the host page.
UniqueID	Gets a hierarchically qualified ID for the control.
Visible	Gets or sets whether ASP.NET has to render the control.

1. HTML Server Controls - Traditional HTML tags

- **HTML server controls are HTML tags understood** by the server.
- **HTML elements** in ASP.NET files are, by **default, treated as text.**
- **To Process control on server** , add a **runat="server"** attribute to the HTML element.
- This attribute indicates that the element should be treated as a **server control.**
- **All HTML server controls must be within a <form> tag with the runat="server" attribute.**
- <form id="form1" runat="server"></form>

Hierarchy of HTML Controls

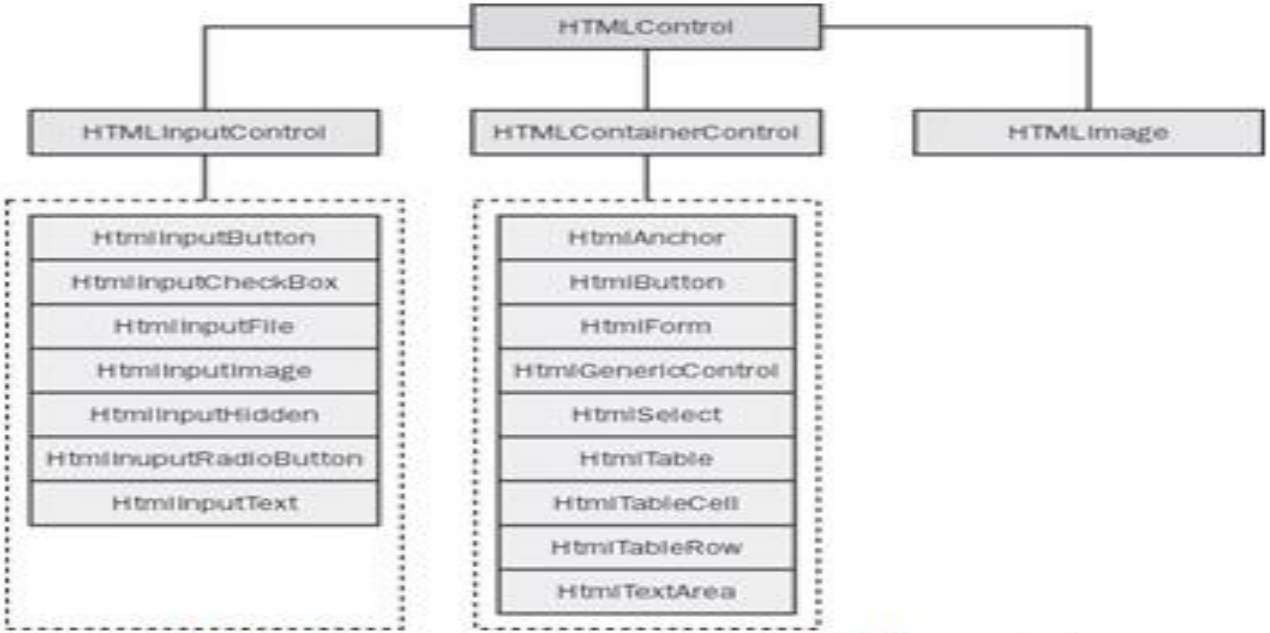


Figure 3-1: A diagram that groups all HTML controls.

The HtmlControl Base Class

Table 3-4: Specific Properties of an HTML Control

Property	Description
Attributes	Gets a collection object representing all the attributes set on the control with the corresponding value
Disabled	Gets or sets a Boolean value, which indicates whether the HTML control is disabled
Style	Gets a collection object representing all CSS properties applied to the control
TagName	Gets the name of the HTML tag behind the control

HTML Container Control Class

- The base class for container controls is the *HtmlContainerControl* class, which descends directly from *HtmlControl*.
- The HTML elements addressed by this tag are elements that must have a closing tag—that is, forms, selection boxes, and tables, as well as anchors and text areas.
- **Compared to the *HtmlControl* class, a container control features a couple of additional string properties—*InnerHtml* and *InnerText*.**

HTML INPUT Control Class

The HTML input controls allow for user interaction. These include the familiar graphical widgets, including check boxes, text boxes, buttons, and list boxes. All of these controls are generated with the <input> tag. The type attribute indicates the type of input control, as in <input type="text"> (a text box), <input type="submit"> (a submit button), and <input type="file"> (controls for uploading a file).

Table 4-5. *HtmlInputControl Properties*

Property	Description
Type	Gets the type of an HtmlInputControl. For example, if this property is set to text, the HtmlInputControl is a text box for data entry.
Value	Gets or sets the value associated with an input control. The value associated with a control depends on the type of control. For example, in a text box this property contains the text entered in the control. For buttons, this defines the text on the button.

```
<input id="Submit1" type="submit" value="submit" />
<input id="Reset1" type="reset" value="reset" />
```

Tag Mappings for HtmlControls

Tag	HtmlControl Class
	HtmlImage
<input type=file runat=server/>	HtmlInputFile
<input type=hidden runat=server/>	HtmlInputHidden
<input type=image runat=server/>	HtmlInputImage
<input type=radio runat=server/>	HtmlInputRadioButton
<input type=text runat=server/>	HtmlInputText
<input type=checkbox runat=server/>	HtmlInputCheckBox
<form runat=server>	HtmlForm
     <p runat=server> etc. (all other elements)	HtmlGenericControl
<select runat=server/>	HtmlSelect
<table runat=server/>	HtmlTable
<td> (within a server-side table) <th> (within a server-side table)	HtmlTableCell
<tr> (within a server-side table)	HtmlTableRow
<textarea runat=server/>	HtmlTextArea
<a runat=server/>	HtmlAnchor
<input type=button runat=server />	HtmlInputButton
<input type=submit runat=server />	HtmlInputButton
<input type=reset runat=server />	HtmlInputButton

2. **Web Controls**

Web controls are defined in the *System.Web.UI.WebControls* namespace and represent an alternative approach to HTML server controls.

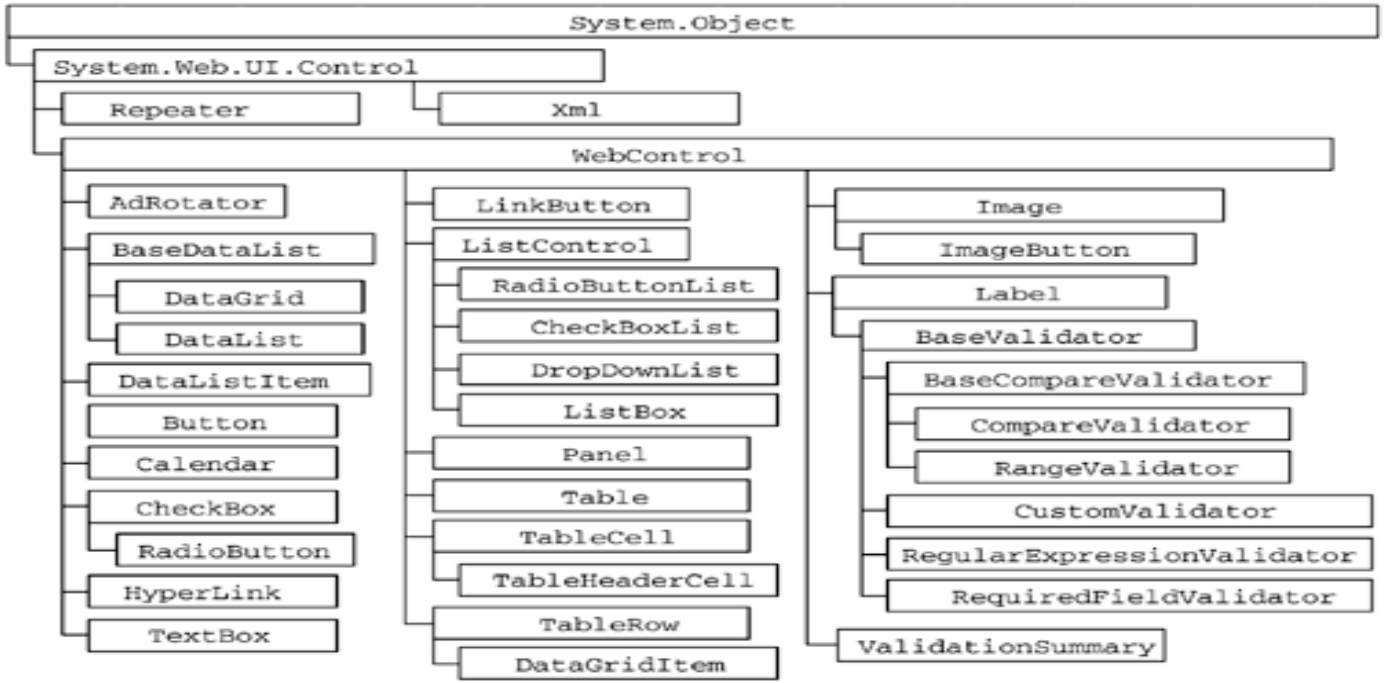
Web controls and HTML controls overlap and generate almost the same client code, although they do it through different programming interfaces.

For example, the Web controls namespace defines the *TextBox* control and makes it available through the *<asp:textbox>* tag;

Similarly, the HTML controls namespace provides the *HtmlInputText* control and declares it using the *<input>* tag. The output that both produce is nearly identical.

WebControl Hierarchy

Figure 2-8. WebControl Hierarchy



WebControl Class

- It is base class for all web controls
- Namespace-System.Web.UI.WebControls
- Serves as the base class that defines the methods, properties and events common to all controls in the System.Web.UI.WebControls
- namespace..

Properties of Web Controls

Property	Description
BackColor	Background color.
BorderColor	Border color.
CssClass	CSS class.
Enabled	Indicates whether the control is grayed out.
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 Web 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.
CheckedChanged	The CheckBox, RadioButton control sends this event when the control becomes checked or unchecked.
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.
DataBinding	Called when the control is bound to a data source.
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.

1.Button

- Displays a 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.

1.Button

- Special Attributes:
- 1.**ImageUrl** - For image button control only. The image to be displayed for the button.
- 2. **PostBackUrl** - The URL of the page that is requested when the user clicks the button.
- **Example**
- <asp:Button ID="Button1" runat="server" Text="Click Me" />

2.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.

2.Textbox

Special Attributes:

1.**TextMode** - Specifies the type of text box. SingleLine creates a standard text box, MultiLlne 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.

2.**MaxLength** - The maximum number of characters that can be entered into the text box.

3. **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>
```

3.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

⑩<asp:Label ID=" Label1" runat="server" Visiable="true" Text="Hi this is darshan"></asp:Label>

4.Checkbox

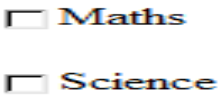
- This control is used to select multiple values from the list.
- Display a check box that allows the user to select a true or false condition means user can either check or uncheck.

Special Attributes:

1. **Checked** - Specifies whether it is selected or not, default is false.

Example:-

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



5. 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:

1. GroupName - Name of the group the control belongs to.

Example

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

○ CE

○ IT

6. Image

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

Special Attributes:

1. **AlternateText** - Alternate text to be displayed in absence of the image.
2. **ImageAlign** - Alignment options for the control.
3. **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" />
```

7.Hyperlink

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

Special Attributes:

1.**ImageUrl** - Path of the image to be displayed by the control.

2.**NavigateUrl** - The target URL of the link.

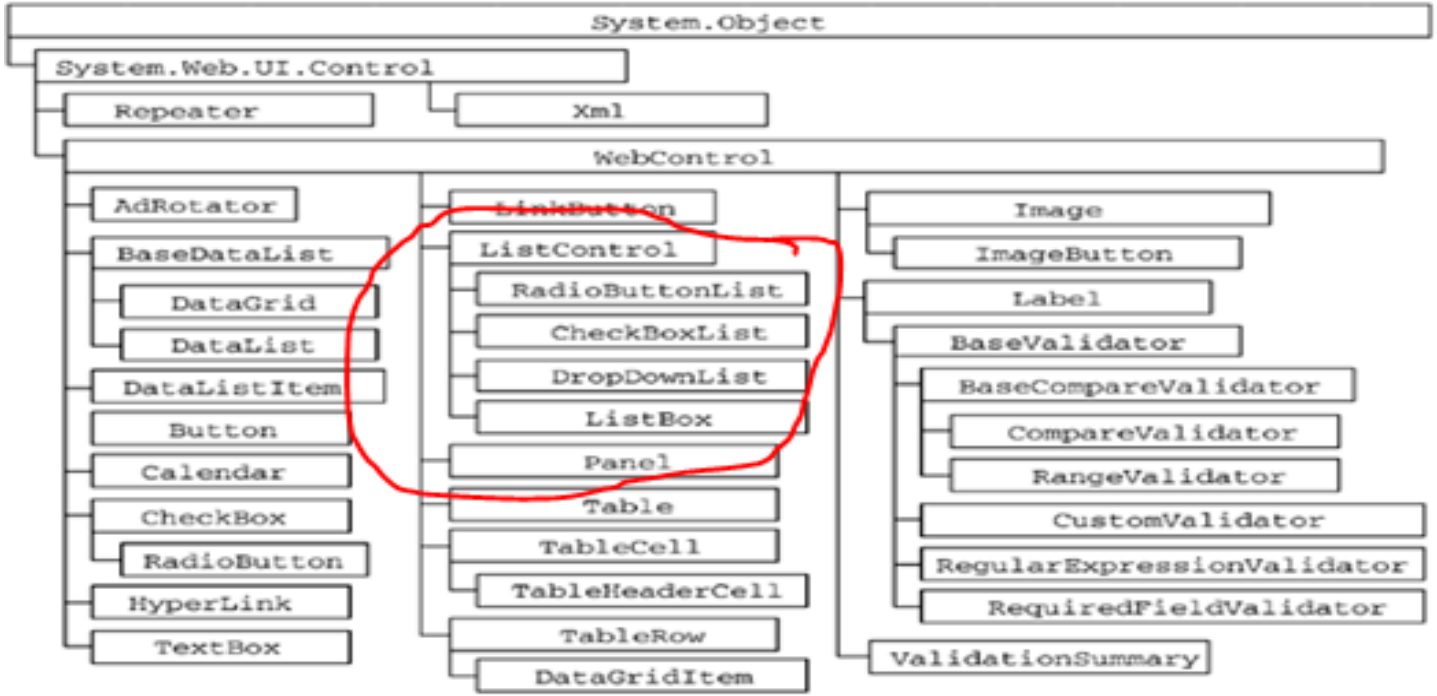
3. **Target** - The target frame of the URL.(option-blank ,Self)

Example

```
<asp:HyperLink ID="HyperLink1" runat="server" ImageUrl="~/App_Data/bird.jpg“
NavigateUrl="https://www.google.co.in" Target="_blank">HyperLink</asp:HyperLink>
```

WebControl Hierarchy

Figure 2-8. WebControl Hierarchy



2. List Controls

These **controls display list of options to select. You can select one or more options.**

They all derive from the **System.Web.UI.WebControls.ListControl** class

Drop-down list

List box

Radio button list

Check box list

2. List Controls---Properties

SelectedValue: Get the value of the selected item from the dropdown list.

SelectedIndex: Gets the index of the selected item from the dropdown box.

SelectedItem: Gets the text of selected item from the list.

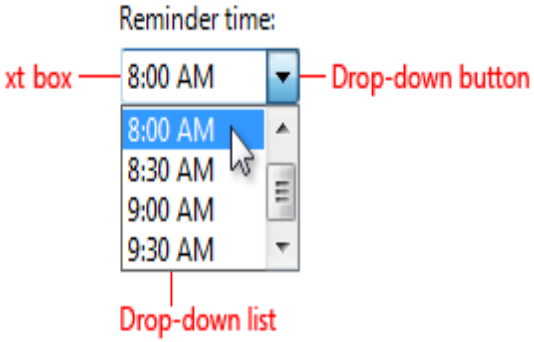
Items: Gets the collection of items from the dropdown list.

Example

```
Label4.Text = RadioButtonList1.SelectedValue;
Label4.Text = RadioButtonList1.SelectedItem.Text;
```

1.Dropdown List

- With a drop-down ,**users make a choice among a** list of mutually exclusive values. Users can **choose one and only one option.**
- With a standard drop-down list, users are **limited to choices one item in the list**
- Dropdown **list is a collection of ListItem Objects.**



DropDown List

How to add items At design Time

```
<asp:DropDownList ID="DropDownList1" runat="server"">
    <asp:ListItem>Red</asp:ListItem>
    <asp:ListItem>Blue</asp:ListItem>
    <asp:ListItem>Green</asp:ListItem>
    <asp:ListItem>Yellow</asp:ListItem>
  </asp:DropDownList>
```

How to add items At Load Time

```
DropDownList2.Items.Add("A");
DropDownList2.Items.Add("B");
  DropDownList2.Items.Add("C");
  DropDownList2.Items.Add("D");
  DropDownList2.Items.Add("E");
```

2.ListBox Control

- The ListBox control is similar to the **DropDownList but main difference is that you can select multiple items from ListBox at a time.**
- ListBox control has **SelectionMode property** that enables you to select multiple items from ListBox control.
- By default **SelectionMode property is set as single.** If you want to select **multiple items from the ListBox,** then set SelectionMode property value as **Multiple and press Ctrl or Shift key when clicking more than one list item.**
- It is also collection of ListItem Objects.
- Useful Property

  1. Row
  2. Selection Mode ---Single & Multiple

How to Add Items in listbox

In Page Load

```
ListBox2.Items.Add("A");
ListBox2.Items.Add("B");
ListBox2.Items.Add("C");
ListBox2.Items.Add("D");
ListBox2.Items.Add("E");
```

At Design Time

```
<asp:ListBox ID="ListBox2"
 runat="server" >
        <asp:ListItem>A</asp:ListItem>
         <asp:ListItem>B</asp:ListItem>
        <asp:ListItem>C</asp:ListItem>
        <asp:ListItem>D</asp:ListItem>
          <asp:ListItem>E</asp:ListItem>
</asp:ListBox>
```

How to choose/Display Multiple items from listbox

Set Selection Mode Property=Multiple

```
Label2.Text="";
foreach (int i in  ListBox2.GetSelectedIndices())
{
    Label2.Text += ListBox2.Items[i].Value + " ,";
    }
```

3.CheckboxList

- CheckBoxList control in ASP.NET is used **to select or deselect the item**.
- **CheckBoxList control contains CheckBoxes in a group.**
- The user is **not restrict for to choose only one answer from CkeckBoxList control.**

Add Items in checkboxlist at design time.

```
<asp:CheckBoxList ID="CheckBoxList1" runat="server" >
  <asp:ListItem Value="1">ABC</asp:ListItem>
      <asp:ListItem Value="2">DEF</asp:ListItem>
        <asp:ListItem Value="3">GHI</asp:ListItem>
        <asp:ListItem Value="4">PQR</asp:ListItem>
       <asp:ListItem Value="5">STU</asp:ListItem>
      <asp:ListItem Value="6">XYZ</asp:ListItem>
  </asp:CheckBoxList>
```

3.CheckboxList

- Add items at load time.

```
CheckBoxList1.Items.Add("A");
CheckBoxList1.Items.Add("B");
CheckBoxList1.Items.Add("C");
CheckBoxList1.Items.Add("D");
```

3.CheckboxList

How to get Selected items

```
Label3.Text = "";
        foreach(ListItem li in CheckBoxList1.Items)
        {
            if (li.Selected == true)
          {
                //  CheckBoxList1.Items.RemoveAt(i);
                  Label3.Text +=li.Text+"   " ;
            }
        }
```

4.RadioButtonList

- The **RadioButtonList control is used to create a group of radio buttons.**
- **Each selectable item in a RadioButtonList control** is defined by a **ListItem element**

Add item at design time

```
<asp:RadioButtonList ID="RadioButtonList1"
runat="server" >
        <asp:ListItem>ColdFusion</asp:ListItem>
            <asp:ListItem>Asp.Net</asp:ListItem>
              <asp:ListItem>PHP</asp:ListItem>
</asp:RadioButtonList>
```

4.RadioButtonList

Add item at load time

```
RadioButtonList1.Items.Add("AAA");
RadioButtonList1.Items.Add("BBB");
RadioButtonList1.Items.Add("CCC");
```

Get Selected Item value of RadioButtonList

```
Label4.Text =
RadioButtonList1.SelectedItem.Text;
```

4.Validation Controls

Why we use validation controls?

Validation is important part of any web application. User's input must always be validated before sending across different layers of the application.

Validation controls are used to,

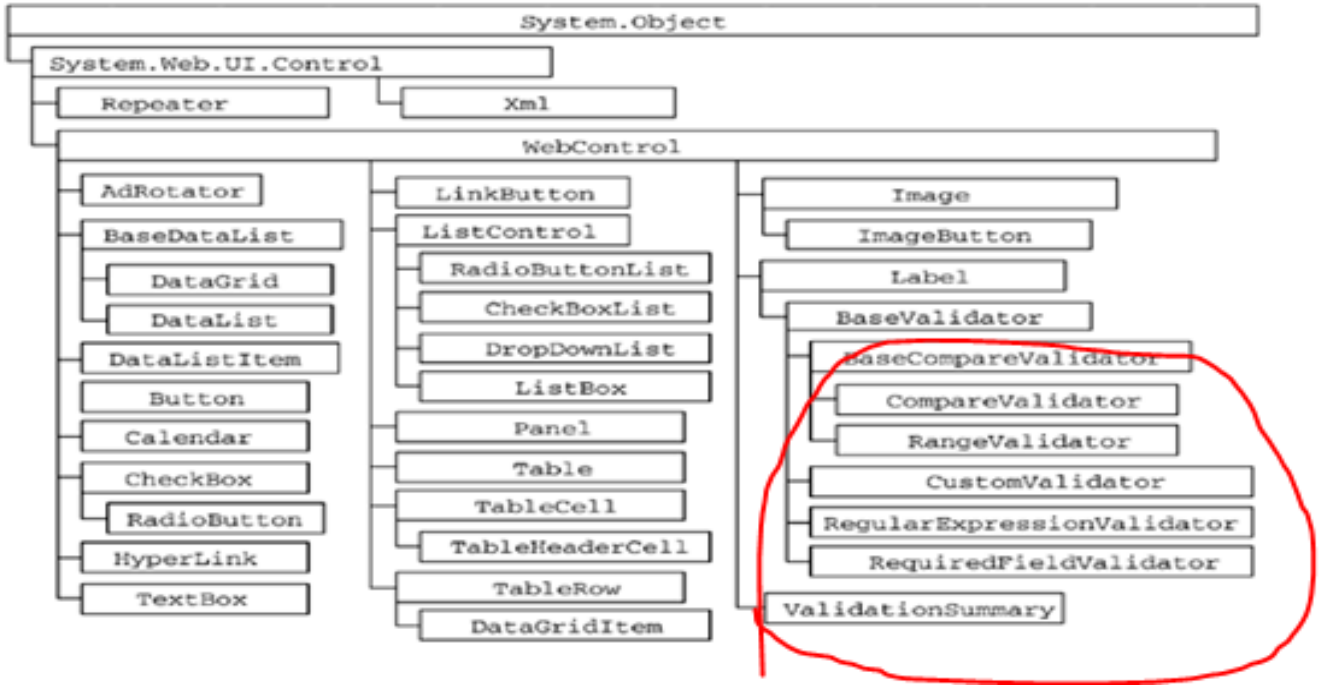
1. Implement presentation logic.
2. To validate user input data.
3. Data format, data type and data range is used for validation.

Validation is of two types

1. Client Side 2. Server Side

WebControl Hierarchy

Figure 2-8. WebControl Hierarchy



3.Validation Controls

Types Of Validation Controls

4.Validation Controls

1.RequiredFieldValidation Control

- The RequiredFieldValidator control is simple validation control, which checks to see if the data is entered for the input control.
- You can have a RequiredFieldValidator control for each form element on which you wish to enforce Mandatory Field rule.

Important Property

ControlToValidate, ErrorMessage

4.Validation Controls

2. CompareValidator Control

The CompareValidator control allows you to make comparison to compare data entered in an input control with a constant value or a value in a different control.(**Property** ☐ **ControlToValidate, ErrorMessage)**

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.

4.Validation Controls

3. RangeValidator Control

The RangeValidator Server Control is another validator control, which checks to see if a control value is within a valid range.**(Property**

- **ControlToValidate, ErrorMessage)**

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.

4.Validation Controls

4. RegularExpressionValidator Control

- A regular expression is a powerful pattern matching mechanism.
- we can check a user's input based on a pattern that you define using a regular expression.
- It is used to validate complex expressions.
- These expressions can be **phone number, email address, zip code and many more.**
- Using Regular Expression Validator is very simple.
- Simply set **the ValidationExpression property t**o any type of expression you want and it will validate it.
- **(Property** **ControlToValidate, ErrorMessage)**

8. ASP.Net Built in Objects

1. Request
2. Response
3. Session
4. Application

8. ASP.Net Built in Objects

1.Request

- When a browser asks for a page from a server, it is called a request.
- The Request object is used to get information from a user/visitor.

Important collection

Query String(Next in State Management topic)

8. ASP.Net Built in Objects

2. Response Object

- The Response object is used **to send output to the user from the server.**
- Its collections, properties, and methods are described below:

Collection	Description
Cookies	Sets a cookie value. If the cookie does not exist, it will be created.

Cookies(How to deal with cookies with example **in State Management topic)**

8.ASP.Net Built in Objects

Methods

Method	Description
Clear	Clears any buffered HTML output
Redirect	Redirects the user to a different URL
Write	Writes a specified string to the output

8. ASP.Net Built in Objects

3.Session

- The Session object stores information about about **a particular user.**
- Variables stored in a Session object hold **information about one single user, and are available to all pages in one application.**
- **Common information stored in session variables are name, id, and preferences.**
- The **server creates a new Session object for each new user, and destroys the Session object when the session expires/Sign out of user.**

8. ASP.Net Built in Objects

3.Session

For Example

One application.

1. User login to a shopping website.
2. User gives some order.
3. Then User just closed the browser .But not sign out from website.
4. When session expires user will automatically sign out from website.(Default session expire time =20min)

8.ASP.Net Built in Objects

Property

Property	Description
SessionID	**Returns a unique id for** each user. The unique id is generated by the **server**
Timeout	Sets the timeout period (Default 20 minutes)

8.ASP.Net Built in Objects

4. Application Object

- **A group of All files that work together to perform some purpose is called an application**. The **Application object is used to tie these files** together.
- The **Application object is used to store and access variables from any page, just like the Session object. The difference is that ALL users share ONE Application object** (**with Sessions there is ONE Session object for EACH user**).

Difference between application & session variable

- **Application variables** are the variables which remain common for the whole application
- Their **value can be used across the whole application…**
- And they are destroyed only **when the application stops** or probably when they are **destroyed by calling function.**
- **Session variables are variables which remain common for the whole** application but for **one particular user.**
- They also can **be used across the whole application…**
- But **they** are destroyed **when a particular user session ends** or **probably when they are destroyed by calling function.**

8.ASP.Net Built in Objects

Methods

Lock	Prevents other users from modifying the variables in the Application object
Unlock	Enables other users to modify the variables in the Application object (after it has been locked using the Lock method)

8.ASP.Net Built in Objects

Events

Event	Description
Application_OnEnd	Occurs when the application ends./**When web server** of that **application stops working.**
Application_OnStart	Occurs when the application start first time.(Using this count no of site visitors)

State Management in ASP.Net

- HTTP is stateless Protocol.
- Now the question arises here, what does stateless actually mean?
- Stateless means, whenever we visit a website, our browser communicates with the respective server depending on our requested functionality or the request. The browser communicates with the respective server using the HTTP or HTTPs protocol.
- But after that response, what's next or what will happen when we visit that website again after closing our web browser?
- So our browsers are stateless.

Create your Google Account

accounts.google.com/signup/v2/webcreateaccount?continue=https%3A%2F%2Fwww.google.co.in%2F&hl=en&dsh=S-317844835%3A1586234318939651&gmb=exp...

Incognito

Google

Create your Google Account

First name: abc

Last name: xyz

Username: abc123 @gmail.com

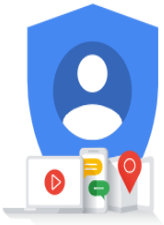
You can use letters, numbers & periods

Use my current email address instead

Password: ••••

Confirm

Use 8 or more characters with a mix of letters, numbers & symbols



One account. All of Google working for you.

Sign in instead

Next

English (United States)

Help Privacy Terms

Type here to search

ENG 10:09 07-04-2020

New Tab

:: Online Job Applic... p2p CE/IT FACULTIES PPS_CO_PO_1.docx... Ganpat University -... Meet - PPS_Lecture... Sci-Hub | Towards a...

Incognito

You've gone incognito

Now you can browse privately, and other people who use this device won't see your activity. However, downloads and bookmarks will be saved. Learn more

Chrome won't save the following information:

- Your browsing history
- Cookies and site data
- Information entered in forms

Your activity might still be visible to:

- Websites you visit
- Your employer or school
- Your internet service provider

Type here to search

ENG 10:10 07-04-2020

Sign in - Google Accounts

accounts.google.com/signin/v2/identifier?hl=en&passive=true&continue=https%3A%2F%2Fwww.google.co.in%2F&flowName=GlifWebSignIn&flowEntry=ServiceLogin

Incognito

Google

Sign in

Use your Google Account

Email or phone

Forgot email?

Not your computer? Use Guest mode to sign in privately.
Learn more

Create account

Next

English (United States)

Help Privacy Terms

Activate Windows
Go to Settings to activate Windows.

Type here to search

ENG 10:11 07-04-2020

Create your Google Account

accounts.google.com/signup/v2/webcreateaccount?continue=https%3A%2F%2Fwww.google.co.in%2F&hl=en&dsh=S1544062911%3A1586234467540067&gmb=exp...

Incognito

Google

Create your Google Account

First name

Last name

Username @gmail.com

You can use letters, numbers & periods

Use my current email address instead

Password

Confirm

Use 8 or more characters with a mix of letters, numbers & symbols

One account. All of Google working for you.

Sign in instead

Next

English (United States)

Help Privacy Terms

Waiting for ssl.gstatic.com...

9. State Management in ASP.Net

- All **web applications are stateless**. means in asp.net each **page posted to the server**, the state of controls is lost.
- In other word, all user can send request to web server but **web server does not know about request** from they **coming same user or new user.**
- We must make web pages which will remember about the **user/we can save control values.**
- Using state management technique we identify the **each user uniquely on same web pages.**

9. State Management in ASP.Net

State management in ASP.NET can be classified into

1. **Client-side state management**
2. **Server-side state management**

1. Client-side state management

- It is a way by which the information is **stored on client's machine or in page itself(requested by client)**
- In this server resources are not utilized.

Client Side State management

Client Side State management

- Cookies
- Hidden Fields
- View State
- Query String

Client Side State management

1. View State

- ViewState is a important client side state management technique. **ViewState is used to store user data on page before we are sending that data to server.**
- **ViewState does not hold the controls,** it holds the **values of controls**
- ViewState can hold the **value on single web page, if we go to other page using response.redirect then ViewState will be null.**

Example

View State Example

Name

Label

Submit Restore

Example

```
protected void Button1_Click(object sender, EventArgs e)
  {
      ViewState["name"] = TextBox1.Text;
      TextBox1.Text = "";
  }
  protected void Button2_Click(object sender, EventArgs e)
  {
      Label3.Text= ViewState["name"].ToString();
  }
```

2. Hidden Field

- A hidden field is used **for storing small amounts of data on the client side.**
- It is **invisible in** the browser.
- Hidden fields store only **one value** in their value property.
- The value **is saved as a string** and therefore in order to use it for other types you need to **perform casting.**

Example


HiddenField - HiddenField1
Hidden State Example
Label
Button
Get Text

Client Side State management

Example

```
protected void Button3_Click(object sender, EventArgs e)
  {
      HiddenField1.Value = TextBox2.Text;
      TextBox2.Text = "";

   }
   protected void Button4_Click(object sender, EventArgs e)
   {
      Label5.Text = HiddenField1.Value;
   }
```

Client Side State management

3.Query Strings

- Query string is a simple way to **pass some information from one page to another.**
- With query string method the information **passed in url of page request.**
- Using this method we can pass maximum 255 characters from one page to another.
- Because 255 is max length of URL.
- The value passed in URL is visible to all.

Client Side State management

- For send information to other page Response.Redirect() method used and for retrieve information from url use Request.QueryString collection of Request object.
- Syntax of Query String
- Send information to other page

Response.Redirect("nextpage.aspx?name="+value);

Example

WebForm1.aspx

No

Name

Button

Req_forward.aspx

Label

Label

On webForm1.aspx

```
protected void Button1_Click(object sender, EventArgs e)
  {
      int no = Convert.ToInt32(TextBox2.Text);
      string name = TextBox1.Text;
   Response.Redirect("req_forward.aspx?no1="+no+"&name1="+name);
  }
```

On req_forward.aspx

```
protected void Page_Load(object sender, EventArgs e)
  {
      Label3.Text = "N0 = " + Request.QueryString["no1"];
      Label2.Text = "Name = " + Request.QueryString["name1"];
  }
```

Client Side State management

4.Cookies

- Cookies is a small pieces of text information which is stored inside the browser for identify users.
- It may contain username, ID, password or any information(address of PC). Cookie does not use server memory.

Cookies are created inside the browser.

Client Side State management

Example to create cookie

```
HttpCookie cookie = new HttpCookie("name");
 //Parameterized constructor is called.
  //HttpCookie cookie = new HttpCookie("Name of Cookie");

   cookie.Value = TextBox2.Text;
    cookie.Expires = DateTime.Now.AddDays(1);
    Response.Cookies.Add(cookie);
```

Server-Side State Management

- Server-Side State Management is different from Client-Side State Management
- In Server-Side State Management all the information is stored in **server memory.**
- More Secure

Server Side State management

1. **Session State Management in ASP.Net**

- When a user connects to an ASP.NET website, a **new session object** is created.
- In simple word we can say**, At the same time more than one users** login to system, all user identification name store separately until they logout.
- session can store username or any unique identification of user for a login period time.
- Session value **can be accessible from all pages from** website.
- By default, Session state is enabled for all ASP.NET applications.
- Session state is based on the **System.Web.HttpSessionState** class.

Server Side State management

Example FirstPage.aspx

```
protected void Button1_Click(object sender, EventArgs e)
    {
        Session["name"] = TextBox1.Text;
        Session["pwd"] = TextBox2.Text;
        Response.Redirect("session_Response.aspx");
    }
```

On Second.aspx

```
protected void Page_Load(object sender, EventArgs e)
  {
      Label2.Text = Session["name"].ToString();
      Label1.Text = Session["pwd"].ToString();
   }
```

Server Side State management

Application State Management in ASP.Net

- It is also **server side management technique.**
- It is also called **application level state management**.
- Application state is a global storage mechanism that used **to stored** data on the **server and shared for all users, means data stored in Application state is common for all user.**
- **Data from Application state can be accessible anywhere in the application.**
- Application state is based on the System.Web.HttpApplicationState class.

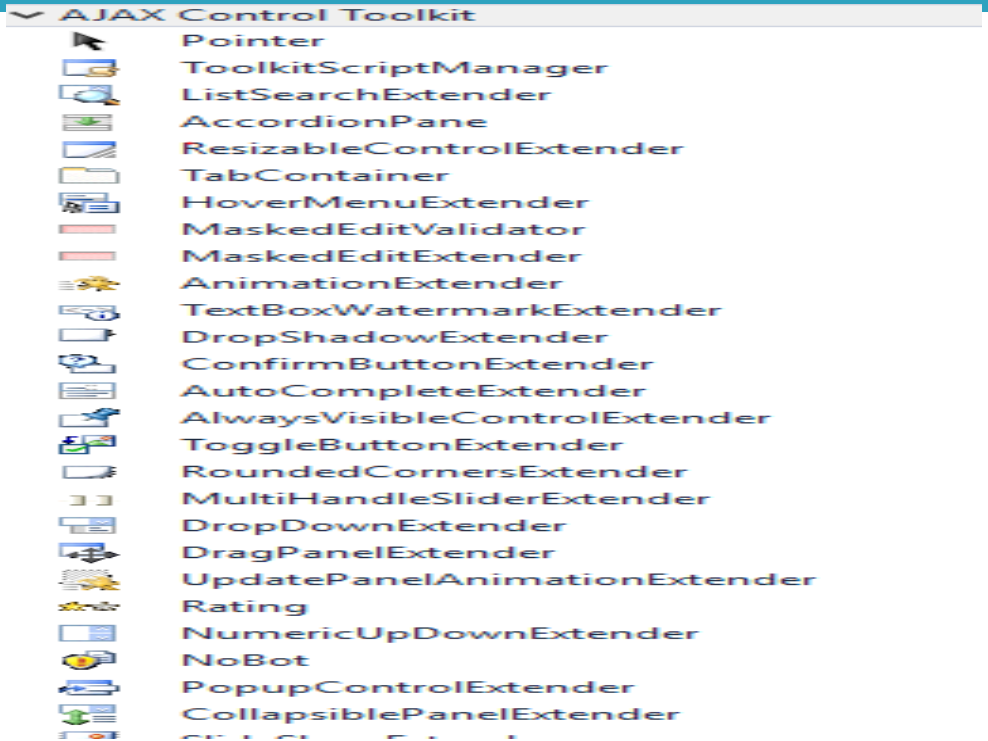
Example

```
protected void Page_Load(object sender, EventArgs e)
  {
      Label2.Text = Application[“count"].ToString();
   }
```

10. Introduction To ASP.Net AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- This is a **cross platform technology/Platform independent** which speeds up response time.
- **No need to refresh the whole page**/ at each interaction with the server.
- **Reduce the traffic travels between the client** and the server.
- **Response time is faster so increases performance and speed.**

10. Introduction To ASP.Net AJAX



1. The ScriptManager Control

- The ScriptManager control is the most important control and must be present on the page **for other controls to work.**
- It is **required on every page on which AJAX controls** are used to manage **the communication between client and server**
- We can add only one scriptManager control on one page.
- It has the basic syntax:

```
<asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>
```

1. The ScriptManager Control

- The UpdatePanel control is **a container control** and derives from the Control class.

1.UpdatePanel Control

- The <asp:UpdatePanel> tag has two childtags - the **ContentTemplate and the Triggers tags.**
- ContentTemplate is used to hold the **content of the page** means suppose we designed page **with some controls we will place controls inside of the ContentTemplate**
- Triggers we **used in a situation like need to refresh updatepanel** only **whenever I click some button control in that situation I will define those controls with this Triggers child tag.**
- **When a control inside it triggers a post back, the UpdatePanel initiates postback and update just that portion of the page.**

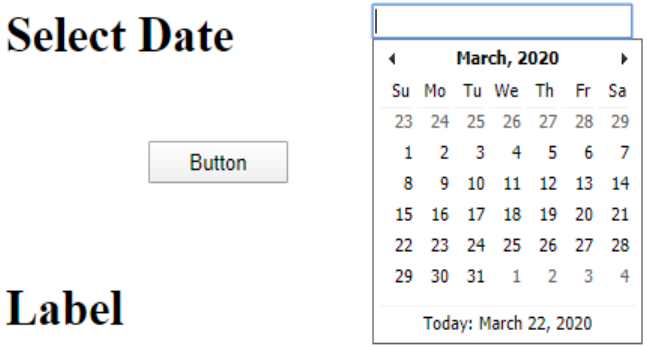
1. The ScriptManager Control

- For example, **if a button control is inside the update panel and it is clicked, only the controls within the update panel will be affected,**
- **the controls on the other parts of the page will not be affected.** This is called the **partial post back** or the **asynchronous post back.**

2.Calender Extender

- The CalendarExtender is an ASP.NET AJAX control that is associated with a **TextBox control.**
- When the user clicks on the TextBox, a client-side Calendar control pops up.
- The user can then set a date by clicking on a day, navigate months by clicking on the left and right arrow and perform other such actions without a postback.

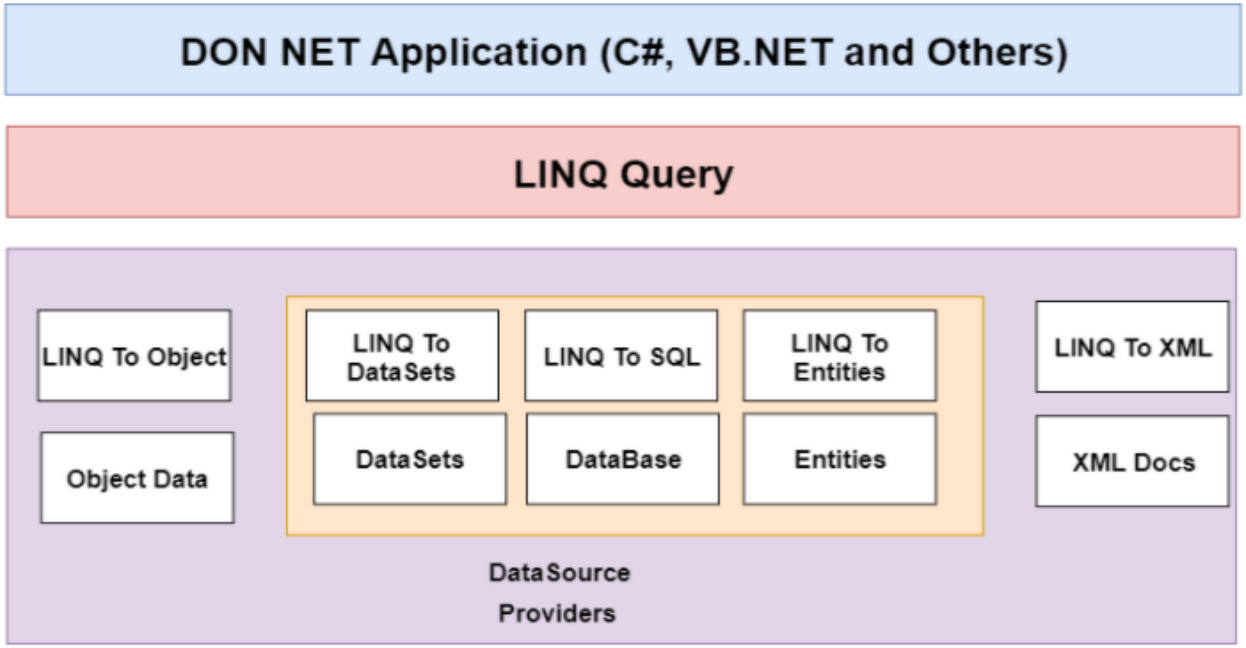
2.Calender Extender



11. Introduction to LINQ

- The full form of LINQ is '**Language Integrated Query**,' and introduced in **.NET Framework 3.5**
- It is used to query the data from different sources of data such as collections, generics, XML documents, ADO.NET Datasets, SQL, Web Services, etc. in C# and VB.NET.
- LINQ provides the rich, standardized query syntax in a .NET programming language such as C# and VB.NET, which allows the developers to interact with any data sources.
- In C# or VB.NET, LINQ functionality can be achieved by importing the **System.Linq** namespace in our application

LINQ Architecture



LINQ Architecture

LINK PROVIDERS

- The responsibility of the LINQ provider is to convert the **LINQ Query** into a format so that the data source can understand it.
- **Example:** Here, we will take a scenario, let us say the application wants to fetch the data from SQL Database. In this case, LINQ Query will fit into the **LINQ to SQL Provider**. In this case, it will convert the **LINQ Query into T-SQL** so that the underlying database can understand in the same way if there is a need to fetch the data from the **XML** document. We will use the same LINQ query in this case as well, which is LINQ to XML Provider. XML provider would convert the **LINQ query into XLST** so that the **XMLDataSource** can understand.

LINQ Syntax

The requirement for writing the LINQ Query

- **To write the LINQ Query, we need the following three things:**
- Data Source (in-memory objects, SQL, XML)
- Query
- Execution of the Query

Each Query is a combination of three things; they are:

- Initialization(to work with a particular data source)
- Condition(where, filter, sorting condition)
- Selection (single selection, group selection or joining)

LINQ Query Syntax

- To write the LINQ Query, we need the following three things:

1. Data Source (in-memory objects, SQL, XML)
2. Query
3. Execution of the Query

Syntax of LINQ is as:

Example-1

We have an integer list, and we need to write a LINQ query that will return all the integers, which are higher than 5. Here we will create a console application.

```
using System;
usingSystem.Collections.Generic;
usingSystem.Linq;
usingSystem.Text;
usingSystem.Threading.Tasks;

namespace ConsoleApp1
{
classProgram
   {
staticvoid Main(string[] args)
      {
//Data Source
   List<int>integerList = new List<int>()
         {
            1, 2, 3, 4, 5, 6, 7, 8, 9, 10
         };
//LINQ Query using Query Syntax
varQuerySyntax = fromobjinintegerList
whereobj> 5
selectobj;
//Execution
foreach (var item inQuerySyntax)
         {
Console.Write(item + " ");
         }
Console.ReadKey();
      }
   }
}
```

```
//LINQ Query using Query Syntax
var QuerySyntax = from obj in integerList
                  where obj > 5
                  select obj;

//Execution
```

Initialization

Condition

Selection

12. ASP.NET Security

- Authentication
  - Who do you say you are? User id
  - Do you have proof? Password
- Authorization
  - Do you have the priviledges to do a requested action?

Asp.Net Authentication

- Asp.Net directly supports three models:
  - Authentication mode = None
    - Application supplied security
  - Authentication mode = Windows
    - Based on Windows Accounts
    - Suitable only for local network
  - Authentication mode = Forms
    - Manged by application with support for redirection and accessing identities provided by Asp.Net
  - Authentication mode = PassPort
    - Authentication credentials stored on Microsoft server
    - Sites license the service

No Authentication

- By Default Authentication is None.
- That means all users can access all pages of a website.
- Web.Config file specifies:

```
<authentication mode="None"/>

<authorization>
    <allow users="*"/>
  </authorization>
```

Windows Authentication

- Requires all users to have Windows accounts on server.
- Suitable only for site serving a local network.
- The major advantage of Windows Integrated Authentication is that you can use all of the Windows role-based security mechanisms.
- It's easy to restrict access to a page to one or more roles and roles can be configured with specific permissions.


THANK YOU!
ANY QUESTIONS?