Swami Sahajanand College of Computer Science

B.C.A. SEM - VI

Subject: Web Application Development Using ASP.Net

UNIT 1

Introduction

- Introduction to ASP.NET.
- IDE (Integrated Development Environment). Web Configuration File(Web.config).Config & Global
- Application Class File(Global.asax) State Management – Session & Cookies. C#.Net Programming Language – Variable, Datatype, Operator,
- Conditional Statement, Looping Statement, Array. Basic Control – Label, Textbox, Button, Radio button,
- Checkbox, Drop down list

Introduction of Visual Web Developer

Visual Web Developer:

- Visual Web Developer is a type of software which is used to develop website using graphical interface. It is allow to programmer to develop web application by clicking, drag and drop and need to write minimum code.
- Visual Web Developer is developed specially for building ASP.NET web sites, and as such, it includes lots of tools that will help you in quickly creating ASP.NET web applications.
- Visual Web Developer available with two editions - standalone and free version as Microsoft Visual Web Developer 2010 and as part of the larger development suite called Visual Studio 2010, which is also available in different versions.
- Express version of Visual Web Developer is free, it contains all the features and tools you need to create complex and feature-rich web applications.
- Visual Web Developer is a free webdevelopment tool, which allows developers to evaluate the web development and editing capabilities of the other Visual Studio versions at no charge. Its main function is to create ASP.NET websites.
- Visual Web Developer offers you the following features:
  - **Web page designs** A powerful Web page editor that includes WYSIWYG editing and an HTML editing mode with IntelliSense and validation.
  - **Page design** features Consistent site layout with master pages and consistent page appearance with themes and skins.
  - **Code editing** A code editor that enables you to write code for your dynamic Web pages in Visual Basic or C#. The code editor includes syntax coloration and IntelliSense.
  - **Debugging** A debugger that helps you find errors in your programs.
  - **Controls** An extensive suite of ASP.NET Web server controls that incorporate much of the functionality you need for creating Web sites.
  - **Data access** Support for displaying and editing data in your Web pages. The data can be in a variety of data stores, including databases or XML files. In many cases, you can add data display and editing to your Web pages without writing any code.
  - **Security**, personalization, and more Built-in application services that enable you to add membership for login security to your site, profile properties that enable you to maintain user-specific information, and other features, most without requiring any code.
  - **Development** for hosted sites Tools for publishing sites to your hosting site, including a local Web server for testing.

Q.-1 What is ASP.Net? Explain in detail.

- ASP.NET is a Web application framework developed and marketed by Microsoft.
- It allows programmers to build dynamic Web sites, Web applications and Web services.
- It was first released in January 2002 with version 1.0 of the .NET Framework.
- ASP.NET is built on the Common Language Runtime (CLR), allowing programmers to write ASP.NET code using any supported .NET language.
- ASP.NET takes an object-oriented programming approach to Web page execution.
- Every element in an ASP.NET page is treated as an object and run on the server thus it is known as server side scripting language.
- ASP.NET creates Web pages and web-sites HTML, CSS, JavaScript and server scripting.

How ASP.net Works?

- Basically an ASP.NET page is just the same as an HTML page.
- An HTML page has the extension .htm. If a browser requests an HTML page from the server, the server sends the page to the browser without any modifications.
- An ASP.NET page has the extension .aspx. If a browser requests an ASP.NET page, the server processes any executable code in the page, before the result is sent back to the browser.
- The ASP.NET page above does not contain any executable code, so nothing is executed. In the next examples we will add some executable code to the page to demonstrate the difference between static HTML pages and dynamic ASP pages.

ASP.Net Built-in Objects:

- ASP.Net has some built-in objects that run on a web server. These objects have methods, properties and collections that are used in application development.
- The following table lists the ASP.Net built-in objects with a brief description:

Object	Description
Application	Describes the methods, properties, and collections of the object that stores information related to the entire Web application, including variables and objects that exist for the lifetime of the application. You use this object to store and retrieve information to be shared among all users of an application. For example, you can use an Application object to create an e-commerce page.
Request	Describes the methods, properties, and collections of the object that stores information related to the HTTP request. This includes forms, cookies, server variables, and certificate data. You use this object to access the information sent in a request from a browser to the server. For example, you can use a Request object to access information entered by a user in an HTML form.
Response	Describes the methods, properties, and collections of the object that stores information related to the server's response. This includes displaying content, manipulating headers, setting locales, and redirecting requests. You use this object to send information to the browser. For example, you use a Response object to send output from your scripts to a browser.
Server	Describes the methods and properties of the object that provides methods for various server tasks. With these methods you can execute code, get error conditions, encode text strings, create objects for use by the Web page, and map physical paths. You use this object to access various utility functions on the server. For example, you may use the Server object to set a time out for a script.
Session	Describes the methods, properties, and collections of the object that stores information related to the user's session, including variables and objects that exist for the lifetime of the session. You use this object to store and retrieve information about particular user sessions. For example, you can use Session object to keep information about the user and his preference and keep track of pending operations.

Features of ASP.Net

Output Caching:

- Since ASP.NET, output caching has is being used to store the generated output of pages and controls.
- On subsequent requests ASP.NET can serve the HTML content more quickly from memory instead of generating the output from scratch .
- However this approach has the limitation that the generated output always has to be stored in memory and if the website experiences heavy traffic the memory requirements can be drastic.
- ASP.NET improves output caching by providing output-cache providers.
- Output-cache providers can use any storage medium to store the generated output. These storage medium can vary from local or remote disks to cache engines.

Redirecting page to a new link

- The website content is moved during the lifetime of the web application from one URL to other.
- This was handled by using the Redirect method to redirect to the new URL.
- One disadvantage with this method is that the Redirect method results in extra roundtrip to the server.

Enabling Viewstate on Control basis.

- In the previous versions of ASP.NET we can't turn off the viewstate for entire page and then unable it for some of the controls that need it. We can do so in ASP.NET.
- This is more convenient than disabling the viewstate on control by basis.You can set the ViewStateMode for the page to false and then enable it for individual controls.

Row Selection in GridView and ListView

- In previous versions of ASP.NET in Gridview and Listviewrowselection was on the basis of RowIndex. If we selected nth row in a page and moved to other page then nth row was selected on the moved page as well.

Session State Compression

- There are two options to store session state data out of process one is session state provider that stores data in session state server and second is session state provider that stores data in a Microsoft SQL Server database.
- The size of the serialized data stored in the session state server and SQL Server can be too large. In ASP.NET 4 there is compression option to store the serialized data. This option can be enabled using compressionEnabled attribute in Session State element.

ClientIDMode Property for generated ClientID

- Every asp.net control is rendered as an HTML element and a corresponding ClientId is generated for it.
- If we want to reference the generated HTML element in JavaScript we must know the Id attribute of the generated element.
- The ID attribute in HTML that is rendered for Web server controls is generated based on the ClientID property of the control.

ListView Control

- Until ASP.NET 3.5 ListView control required ItemTemplate and LayoutTemplate.

- ItemTemplate specifies the markup for each item bound to the ListView while LayoutTemplate defines just the enclosing markup for the items in the List View.

Asp.net Directories :

- In general, the ASP.NET directory structure can be determined by the developer's preferences.
- Apart from a few reserved directory names, the site can span any number of directories. The structure is typically reflected directly in the URLs.
- Although ASP.NET provides means for intercepting the request at any point during processing, the developer is not forced to funnel requests through a central application or front controller.
- The special directory names (from ASP.NET 2.0 on) are:

App_Code

- This is the "raw code" directory. The ASP.NET server automatically compiles files (and subdirectories) in this folder into an assembly which is accessible in the code of every page of the site. App_Code will typically be used for data access abstraction code, model code and business code. Also any site-specific http handlers and modules and Web service implementation go in this directory. As an alternative to using App_Code the developer may opt to provide a separate assembly with precompiled code.

App_Data

- The App_Data ASP.NET Directory is the default directory for any database used by the ASP.NET Website. These databases might include Access (mdb) files or SQL Server (mdf) files. The App_Data is the only directory with Write Access enabled for the ASP.NET web application.:

App_LocalResources

- E.g. a file called CheckOut.aspx.fr-FR.resx holds localized resources for the French version of the CheckOut.aspx page. When the UI culture is set to French, ASP.NET will automatically find and use this file for localization.

App_GlobalResources

- Holds resx files with localized resources available to every page of the site. This is where the ASP.NET developer will typically store localized messages etc. which are used on more than one page.

App_Themes

- Adds a folder that holds files related to themes which is a new ASP.NET feature that helps ensure a consistent appearance throughout a Web site and makes it easier to change the Web site’s appearance when necessary.

App_WebReferences

- holds discovery files and WSDL files for references to Web services to be consumed in the site.

Bin

- Contains compiled code (.dll files) for controls, components, or other code that you want to reference in your application. Any classes represented by code in the Bin folder are automatically referenced in your application.

Asp.Net Directives:

- Directives specify settings that are used by the page and user-control compilers when the compilers process ASP.NET Web Forms pages (.aspx files) and user control (.ascx) files.
- ASP.NET treats any directive block (<%@ %>) that does not contain an explicit directive name as an @ Pagedirective (for a page) or as an @ Control directive (for a user control).
- For syntax information and descriptions of the attributes that are available for each directive, use the links that are listed below.

@ Assembly
- Links an assembly to the current page or user control declaratively.

@ Control
- Defines control-specific attributes used by the ASP.NET page parser and compiler and can be included only in .ascx files (user controls).

@ Implements
- Indicates that a page or user control implements a specified .NET Framework interface declaratively.

@ Import
- Imports a namespace into a page or user control explicitly.

@ Master
- Identifies a page as a master page and defines attributes used by the ASP.NET page parser and compiler and can be included only in .master files.

@ MasterType
- Defines the class or virtual path used to type the Master property of a page.

@ OutputCache
- Controls the output caching policies of a page or user control declaratively.

@ Page
- Defines page-specific attributes used by the ASP.NET page parser and compiler and can be included only in .aspx files.

@ PreviousPageType
- Creates a strongly typed reference to the source page from the target of a cross-page posting.

@ Reference
- Links a page, user control, or COM control to the current page or user control declaratively.

@ Register
- Associates aliases with namespaces and classes, which allow user controls and custom server controls to be rendered when included in a requested page or user control.

Advantages of ASP.NET
- ASP.NET drastically reduces the amount of code required to build large applications.
- With built-in Windows authentication and per-application configuration, your applications are safe and secured.
- It provides better performance by taking advantage of early binding, just-in-time compilation, native optimization, and caching services right out of the box.
- The ASP.NET framework is complemented by a rich toolbox and designer in the Visual Studio integrated development environment. WYSIWYG editing, drag-and-drop server controls, and automatic deployment are just a few of the features this powerful tool provides.
- Provides simplicity as ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration.
- The source code and HTML are together therefore ASP.NET pages are easy to maintain and write. Also the source code is executed on the server. This provides a lot of power and flexibility to the web pages.
- All the processes are closely monitored and managed by the ASP.NET runtime, so that if process is dead, a new process can be created in its place, which helps keep your application constantly available to handle requests.

- It is purely server-side technology so, ASP.NET code executes on the server before it is sent to the browser.
- Being language-independent, it allows you to choose the language that best applies to your application or partition your application across many languages.
- ASP.NET makes for easy deployment. There is no need to register components because the configuration information is built-in.
- The Web server continuously monitors the pages, components and applications running on it. If it notices any memory leaks, infinite loops, other illegal activities, it immediately destroys those activities and restarts itself.
- Easily works with ADO.NET using data-binding and page formatting features. It is an application which runs faster and counters large volumes of users without having performance problems

Disadvantages of ASP.Net

- Does not allow for easy unit tests - the framework tends not to support automatic unit testing with tools like NUnit very well, which makes test-driven development difficult. MVC based frameworks like Struts or ASP.NET MVC do a better job of this.
- View state - often times viewstate can get really large or have negative effects on performance. This is especially true for some of the more complex server controls.
- Abstracts the webiness away from web programming - some people feel that ASP.NET does not fit the general architecture of internet and web based applications. They feel like Microsoft tried to follow the Windows Forms desktop paradigm too closely and it makes for developers that don't have a good grasp on how web apps are different than desktop apps.
- Clicking the browser’s “back” button may not return the user to an earlier state of the Ajax-enabled page.
- Dynamic web page updates also caused some troubles for a user to bookmark a particular state of the application.
- Ajax opens up another attack vector for malicious code that web developers might not expected for.
- Any user whose browser does not support Ajax or JavaScript, or simply has JavaScript disabled, will not be able to use its functionality.
- Platform-specific (only Windows)
- Complex to use.

Basic of Web Application Development

Q.-2. Explain web application development?

- Web application is also one type of program or we can say software that runs using a web browser over internet or intranet.
- Then we must say what web browser is?
- Web browser is built in software that is used to view web pages downloaded from either internet or intranet.
- To learn why we should use web application over desktop application we need to understand how web application works?

Components of web application

- **Client**
  - Client is simple node, machine, terminal, computer that request for the information or some resources using communication channel.
  - Client computer must be connected with network/internet to send and receive information to and from server.
- **Web server**
  - Server computer usually provide information, resource, required by client computer.
  - Server generally gives this information in HTML pages which is universal format for WWW.
  - Server can be web server, mail server, database server or any other server.
- **Database server**
  - A database server is software that provides database services to other computer programs or computers.
  - Database server is accessed via Front-end server to dynamically generate content of web page.
  - Some examples of Database servers are Oracle, DB2, Informix, Ingres, SQL Server, and MySQL.
- **Communication channel**
  - A communication channel, or channel, refers either to a physical transmission medium such as a wire, or to a logical connection over a multiplexed medium such as a radio channel.
  - A channel is used to convey an information signal, for example a digital bit stream, from one or several senders (or transmitters) to one or several receivers.
  - A channel has a certain capacity for transmitting information, often measured by its bandwidth in Hz or its data rate in bits per second

Web browser

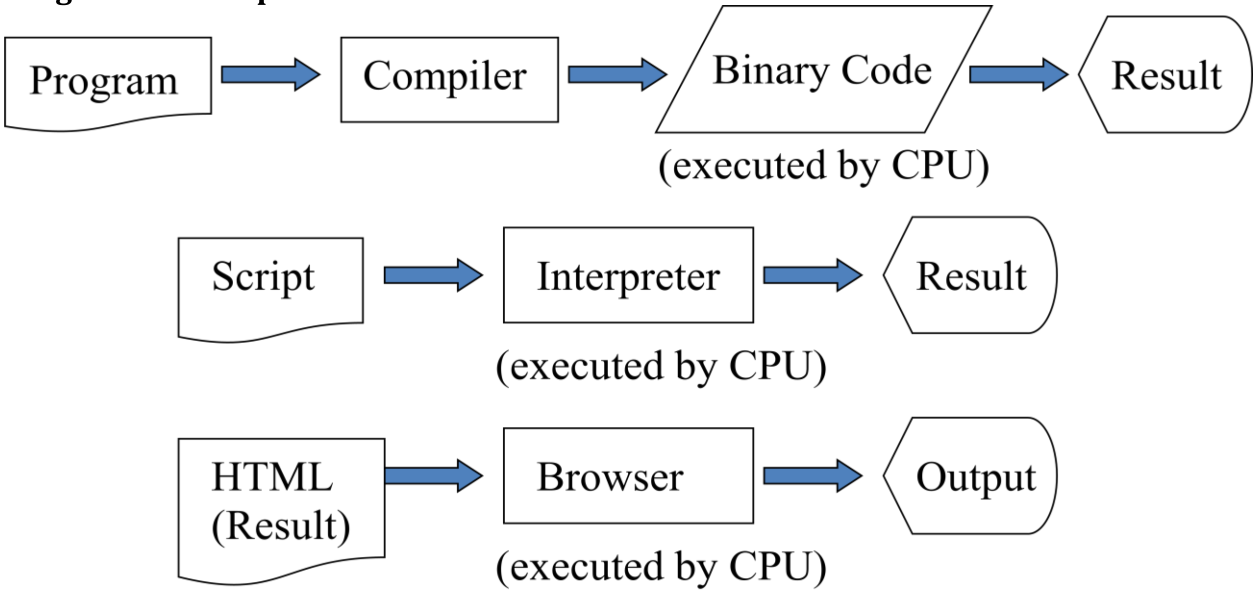
- A web browser is a software application for retrieving, presenting, and traversing information resources on the World Wide Web.
- An information resource is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video, or other piece of content.

What is difference between script and program?

- Program is set of instruction that is first converted into machine level code using program known as compiler in one pass or two pass.
- Script is also set of instruction that is first converted into machine level code using another program known as interpreter in line by line fashion.

- Compare to compiler, interpreter require less resource (hard disk, main memory, processing power) to perform conversion.

Programs vs. Scripts



Advantages of web application

- No installation is required on each individual machine.
- Web application is very easy to update and maintain.
- Higher security of confidential and important data.
- Lower infrastructure IT Cost.
- Remote access of application and data is possible.
- Update are available to each person in time.

How to create a web application is ASP.Net?

- Create a simple SQL Server Database using SQL Express
- Add Tables to the Database and set foreign key relationships
- Add data to the tables created
- Add a MasterPage and ContentPage to the ASP.NET
- Modify the Master Page to provide consistent look and feel to the site
- Add datacontrols like GridView and bind them to SQL Express database
- Add additional web page and use DataView to display more data
- Use QueryBuilder to fetch data from multiple tables
- Use AJAX to refresh only parts of the page rather than complete page
- Hook up various pages and complete the experience….

Application and State

Q.-3. What is application? Explain LifeCycle of an application.

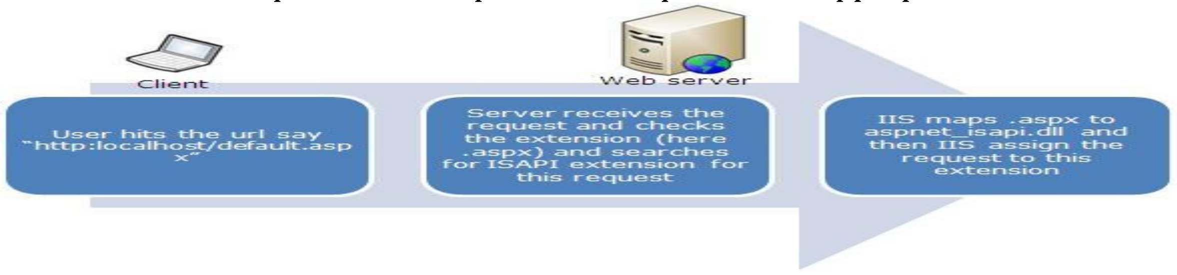
- An application on the Web may consist of several ASP files that work together to perform some purpose.
- The Application object is used to tie these files together.
- The Application object is used to store and access variables from any page, just like the Session object.
- The difference is that ALL users share ONE Application object (with Sessions there is ONE Session object for EACH user).
- The Application object holds information that will be used by many pages in the application (like database connection information). The information can be accessed from any page.
- The information can also be changed in one place, and the changes will automatically be reflected on all pages.

Application LifeCycle

- One needs to really understand the application Lifecycle, so that one can code efficiently and use the resources available. Also it is very important to discuss, as we are going to Application level events, objects, etc.
- ASP.NET uses **lazy initialization** technique for creating the application domains, i.e., Application domain for an application is created only when the first request is received by the web server. We can categorise Application life cycle in several stages. These can be:
  - **Stage 1:** User first requests any resource from the webserver.
  - **Stage 2:** Application receives very first request from the application.
  - **Stage 3:** Application basic Objects are created.
  - **Stage 4:** An HTTPapplication object is assigned to the request.
  - **Stage 5:** And the request is processed by the HTTPApplication pipeline.

Stage 1:

- The Application life cycle starts when a user hits the URL by typing it in the browser. The browser sends this request to the webserver.
- When webserver receives the request from the browser, it examines the file extension of the requested file and checks which ISAPI extension is required to handle this request and then passes the request to the appropriate ISAPI extension.

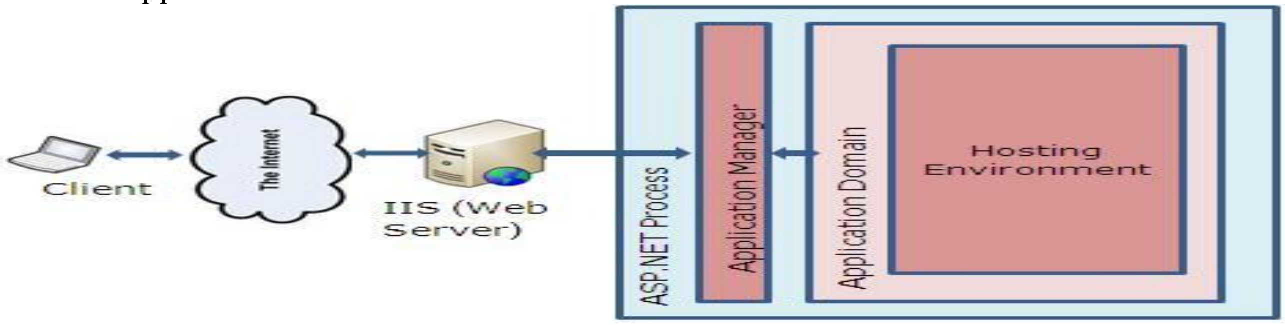

Client
Web server
User hits the url say "http:localhost/default.aspx"
Server receives the request and checks the extension (here .aspx) and searches for ISAPI extension for this request
IIS maps .aspx to aspnet_isapi.dll and then IIS assign the request to this extension

- **Note 1**: If any extension is not mapped to any ISAPI extension, then ASP.NET will not receive the request and the request is handled by the server itself and ASP.NET authentication, etc. will not be applied.
- **Note 2**: We can also make our own **custom handler**, to process any specific file extension.

Stage 2:

- When ASP.NET receives the first request, the Application manager creates an application **domain** for it.

- Application domains are very important because they provide the **isolation** amongst various applications on the webserver and every application domain is loaded and unloaded separately, and in application domain an instance of class HostingEnvironment is created which provides access to information about all application resources.

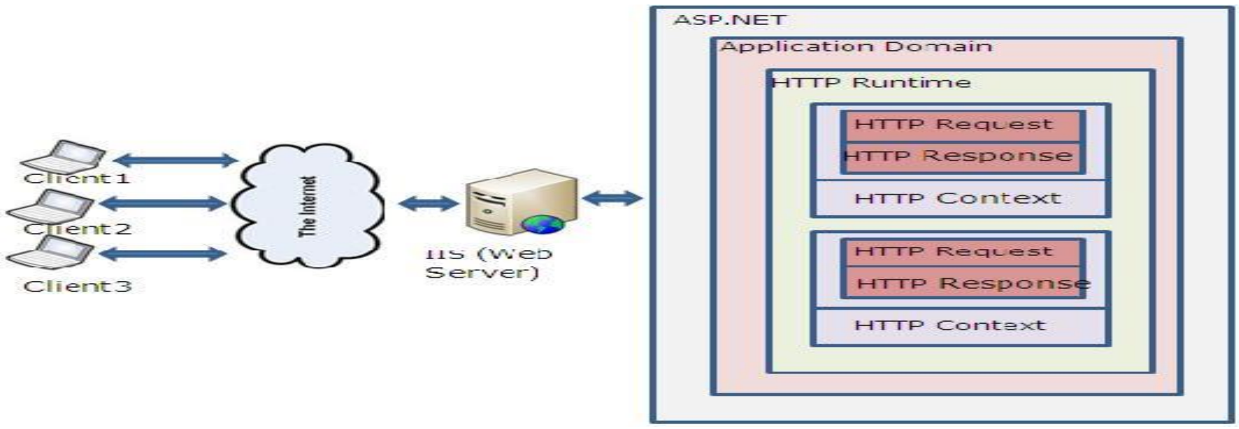


Stage 3:

- After creating the application domain and hosting environment, ASP.NET initializes the basic objects as HTTPContext, HTTPRequest and HTTPResponse. HTTPContext holds objects to the specific application request as HTTPRequest and HTTPResponse.
- HTTPRequest contains all the information regarding the current request like cookies, browser information, etc. and HTTPResponse contains the response that is sent to client.

Stage 4:

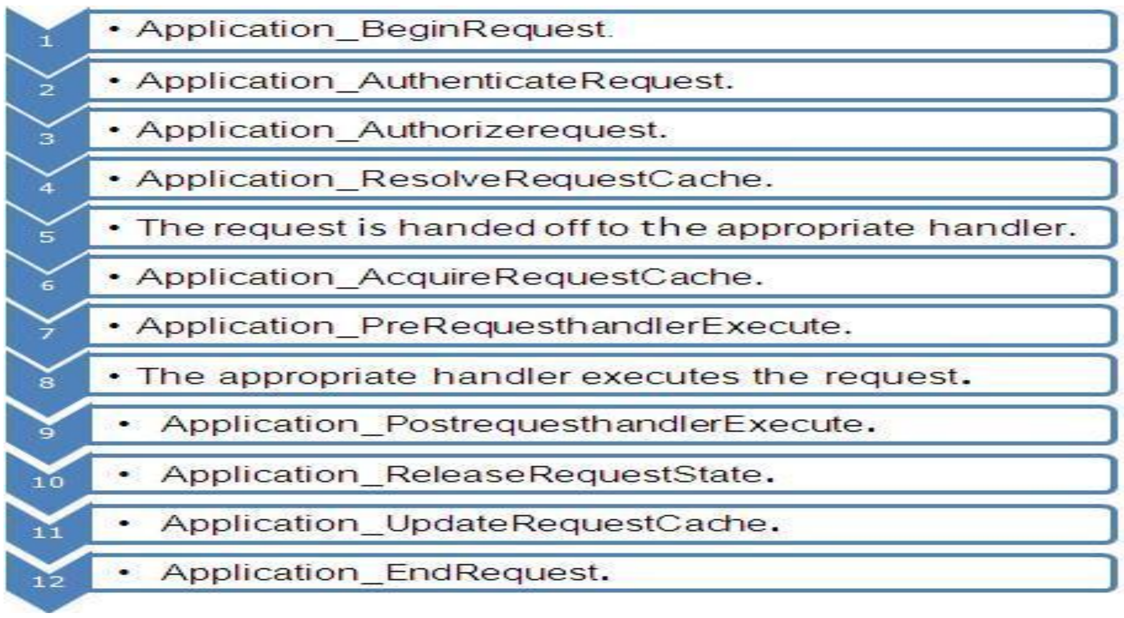
- Here all the basic objects are being initialized and the application is being started with the creation of HTTPApplication class, if there is *Global.asax*(It is derived from HTTPApplication class) in the application, then that is instantiated.



Note: When the application is accessed for the first time, the HTTPApplication instance is created for further requests. It may be used for other requests as well.

Stage 5:

- There are a lot of events executed by the HTTPApplication class. Here, I have listed down a few important ones. These events can be used for any specific requirement.


1 • Application_BeginRequest.
2 • Application_AuthenticateRequest.
3 • Application_Authorizerequest.
4 • Application_ResolveRequestCache.
5 • The request is handed off to the appropriate handler.
6 • Application_AcquireRequestCache.
7 • Application_PreRequesthandlerExecute.
8 • The appropriate handler executes the request.
9 • Application_PostrequesthandlerExecute.
10 • Application_ReleaseRequestState.
11 • Application_UpdateRequestCache.
12 • Application_EndRequest.

Web Page Life Cycle:

- When a page is requested, it is loaded into the server memory, processed and sent to the browser.
- Then it is unloaded from the memory. At each of this steps, methods and events are available, which could be overridden according to the need of the application.
- All the components on the page, except the directives are part of this control tree.
- You can see the control tree by adding trace= "true" to the Page directive.
- We will cover page directives and tracing under 'directives' and 'error handling'.
- The page life cycle stages are:
  - Initialization
  - Instantiation of the controls on the page
  - Restoration and maintenance of the state
  - Execution of the event handler codes
  - Page rendering
- Understanding the page cycle helps in writing codes for making some specific thing happen at any stage of the page life cycle.
- It also helps in writing custom controls and initializing them at right time, populate their properties with view-state data and run control behavior code.
- Following are the different stages of an ASP.Net page:
  - **Page request**: when ASP.Net gets a page request, it decides whether to parse and compile the page or there would be a cached version of the page; accordingly the response is sent
  - **Starting of page life cycle:**At this stage, the Request and Response objects are set. If the request is an old request or post back, the IsPostBack property of the page is set to true. The UICulture property of the page is also set.
  - **Page initialization**:At this stage, the controls on the page are assigned unique ID by setting the UniqueID property and themes are applied. For a new request post back data is loaded and the control properties are restored to the view-state values.
  - **Page load**: At this stage, control properties are set using the view state and control state values.
  - **Validation**: Validate method of the validation control is called and if it runs successfully, the IsValid property of the page is set to true.

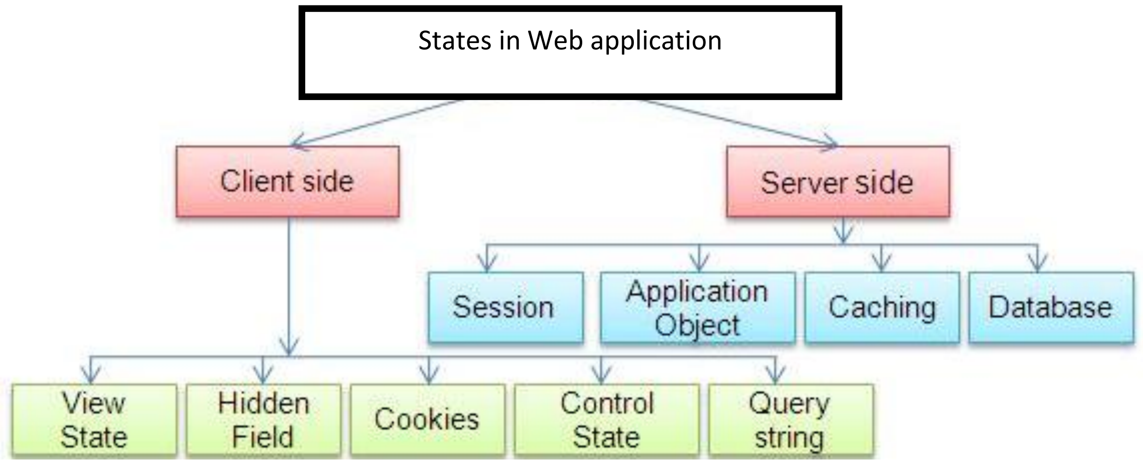
- **Postback event handling:**If the request is a post back (old request), the related event handler is called.
- **Page rendering**:At this stage, view state for the page and all controls are saved. The page calls the Render method for each control and the output of rendering is written to the Output Stream class of the Page's Response property.
- **Unload:** The rendered page is sent to the client and page properties, such as Response and Request are unloaded and all cleanup done.

Web Page Events:

- At each stage of the page life cycle, the page raises some events, which could be coded. An event handler is basically a function or subroutine, bound to the event, using declarative attributes like Onclick or handle.
- Following are the web page events:
  - **PreInit:**PreInit is the first event in page life cycle. It checks the IsPostBack property and determines whether the page is a post back. It sets the themes and master pages, creates dynamic controls and gets and sets profile property values. This event can be handled by overloading the OnPreInit method or creating a Page_PreInit handler.
  - **Init:** Init event initializes the control property and the control tree is built. This event can be handled by overloading the OnInit method or creating a Page_Init handler.
  - **InitComplete:**InitComplete event allows tracking of view state. All the controls turn on view-state tracking.
  - **LoadViewState**:LoadViewState event allows loading view state information into the controls.
  - **LoadPostData:** During this phase, the contents of all the input fields defined with the <form> tag are processed.
  - **PreLoad**:PreLoad occurs before the post back data is loaded in the controls. This event can be handled by overloading the OnPreLoad method or creating a Page_PreLoad handler.
  - **Load**: The Load event is raised for the page first and then recursively for all child controls. The controls in the control tree are created. This event can be handled by overloading the OnLoad method or creating a Page Load handler.
  - **LoadComplete**:The loading process is completed, control event handlers are run and page validation takes place. This event can be handled by overloading the OnLoadComplete method or creating a Page_LoadComplete handler.
  - **PreRender**:ThePreRender event occurs just before the output is rendered. By handling this event, pages and controls can perform any updates before the output is rendered.
  - **PreRenderComplete:**As the PreRender event is recursively fired for all child controls, this event ensures the completion of the pre-rendering phase.
  - **SaveStateComplete:**State of control on the page is saved. Personalization, control state and view state information is saved. The HTML markup is generated. This stage can be handled by overriding the Render method or creating a Page_Render handler.
  - **UnLoad:**TheUnLoad phase is the last phase of the page life cycle. It raises the UnLoad event for all controls recursively and lastly for the page itself. Final cleanup is done and all resources and references, such as database connections, are freed. This event can be handled by modifying the OnUnLoad method or creating a Page_UnLoad handler.

Q.-4. What is state? Explain in brief. OR Write a note about State Management.

- As we all know, web is stateless. A Web page is recreated every time it is posted back to the server.
- In traditional web programming, all the information within the page and control gets wiped off on every postback.
- To overcome this problem, ASP.NET Framework provides various ways to preserve the states at various stages like controlstate, viewstate, cookies, session, etc.
- These can be defined in client side and server side state management. Please see the image below:



- State management is an important aspect of any Web application.
- Because state information is lost between subsequent requests, ASP.NET provides a variety of way to preserve state both server-side and client-side, when your application or controls need to round-trip information across requests.

Client Side State:

View State

- This is used to store request-specific values. ASP.NET provides the server-side notion of a view state for each control.
- A control can save its internal state between requests using the ViewState property on an instance of the class StateBag.
- The StateBag class provides a dictionary-like interface to store objects associated with a string key.

Hidden field

- Refers to the control that is not visible when a Web application is viewed in the browser.
- The content of the control is sent in the HTTP Form collection control along with the values of other controls to the server on page reloads.
- This control acts as a storage area for any page-specific storing information.

Cookies state

- Client-side cookies are used to store volatile user preferences. Storing cookies on the client is one of the methods that ASP.NET's session state uses to associate requests with sessions.

- Cookies can also be used directly to persistance data between requests, but the data is then stored on the client and sent to the server with every request.
- Browsers place limits on the size of a cookie; therefore, only a maximum of 4096 bytes is guaranteed to be acceptable.When the data is stored on the client, the Page_Load method in the file, checks whether the client has sent a cookie. If not, a new cookie is created and initialized and stored on the client.

Control State

- The control state is a way to save a control's state information when the EnableViewState property is turned off.
- Unlike ViewState a developer can't turn off control state. The control state can be implemented only in controls that you implement.
- Control state implementation is easy.
- First, override the OnInit method of the control and add the call for the Page.RegisterRequiresControlState method with the instance of the control to register.
- Then, override the LoadControlState and SaveControlState in order to save the required state information.

Query String

- The HTTP specification allows for client browsers to pass data as part of the request's query string.
- This feature is typically used in HTTP GET requests to pass arguments to an application, but you can also use it to pass state information back and forth between the client and server.
- For example, in the URL http://www.asp.net/somepage.aspx?userid=12&location=Boston, the query string contains two attribute-value pairs: one named userid and the other named location.
- Use query strings when:
  - You have to store small amounts of state information.
  - The state does not include secret or sensitive information.
  - The state does not provide access to or drive secured parts of your application.
  - Your application runs on a server farm and you have no means of providing a centralized state store.
  - Cookies are disabled.
  - The state must survive across page requests or even across application invocations.
  - Your application will still function if the state is not available.

Server Side State:

Session State

- The session state used to store volatile user preferences. To provide individual data for a user during a session, data can be stored with session scope.

Application Object State

- An application state to read a dataset in Application_Start. The HttpApplicationState collection used for backward-compatibility with classic ASP and will be familiar to ASP developers. However, the use of static fields in ASP.NET is generally preferred over the use of HttpApplicationState.

- Because an application and all the objects it stores can be concurrently accessed by different threads, it is better to store only infrequently modified data with application scope. Ideally an object is initialized in the Application_Start event and further access is read-only.

Caching

- Caching is a technique used to improve the performance and scalability of applications.
- Caching is most useful in server-based applications, in particular Web applications, but it also applies to many smart client situations.
- The purpose of caching is to keep state that is expensive to create or retrieve in a more easily accessible form or location.
- For example, when a user has an active session on your Web application, if you keep a copy of their profile in memory instead of always reading from a remote SQL Server, your application performance improves.

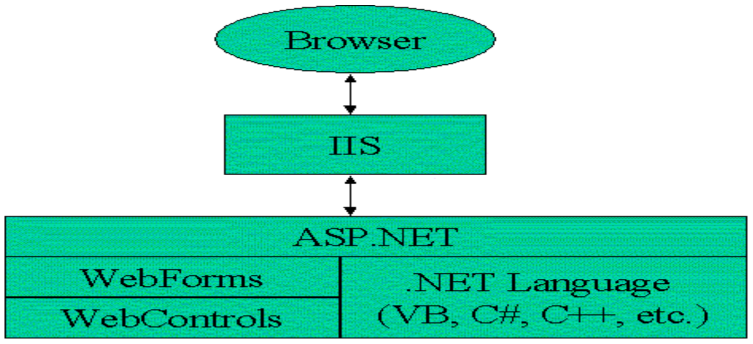
Database

- ASP.NET sessions can also be stored in a SQL Server database.
- Storing sessions in SQL Server offers resilience that can serve sessions to a large web farm that persists across IIS restarts.
- SQL based Session stateis configured with aspnet_regsql.exe. This utility is located in .NET Framework's installed directory C:\<windows>\microsoft.net\framework\<version>.
- Running this utility will create a database which will manage the session state.

Introduction of Web-Forms

Q.-5. What is form in asp.net? Also explain Features of ASP.NET Web Forms.

- ASP.NET Web Forms is a part of the ASP.NET web application framework.
- It is one of the three different programming models you can use to create ASP.NET web applications,
- The others are ASP.NET MVC and ASP.NET Web Pages.
- Web Forms are pages that your users request through their browser and that makes the user interface (UI) that give your web applications their look and feel.
- These pages are written using a combination of HTML, server controls, and server code.
- When users request a page, it is compiled and executed on the server, and then it generates the HTML markup that the browser can render.
- Using Visual Studio, you can create ASP.NET Web Forms using a powerful IDE.
- For example, this lets you drag and drop server controls to lay out your Web Forms page.
- You can then easily set properties, methods, and events for controls or for the page in order to define the page's behavior, look and feel, and so on.
- To write server code to handle the logic for the page, you can use a .NET language like Visual Basic or C#.
- ASP.NET Web Forms offer:
  - Separation of HTML and other UI code from application logic.
  - A rich suite of server controls for common tasks, including data access.
  - Powerful data binding, with great tool support.
  - Support for using Ajax, even if developer don't know JavaScript.
- Server controls must appear within a <form> tag, and the <form> tag must contain the runat="server" attribute.
- The runat="server" attribute indicates that the form should be processed on the server.
- It also indicates that the enclosed controls can be accessed by server scripts.
- An .aspx page can only contain ONE <form runat="server"> control.
- Web Forms are made up of two components: the visual portion (the ASPX file), and the code behind the form, which resides in a separate class file.



How to submit a Form?

- A form is most often submitted by clicking on a button. The Button server control in ASP.NET has the following format:
- <asp:Button id="id" text="label" OnClick="sub" runat="server" />
- The id attribute defines a unique name for the button and the text attribute assigns a label to the button.
- The onClick event handler specifies a named subroutine to execute.

Features of ASP.NET Web Forms

Server Controls

- ASP.NET Web server controls are objects on ASP.NET Web pages that run when the page is requested and that render markup to the browser. Many Web server controls are similar to familiar HTML elements, such as buttons and text boxes. Other controls encompass complex behavior, such as a calendar controls, and controls that you can use to connect to data sources and display data.

Master Pages

- ASP.NET master pages allow you to create a consistent layout for the pages in your application. A single master page defines the look and feel and standard behavior that you want for all of the pages (or a group of pages) in your application. You can then create individual content pages that contain the content you want to display. When users request the content pages, they merge with the master page to produce output that combines the layout of the master page with the content from the content page.

Working with Data

- ASP.NET provides many options for storing, retrieving, and displaying data. In an ASP.NET Web Forms application, you use data-bound controls to automate the presentation or input of data in web page UI elements such as tables and text boxes and drop-down lists.

Membership

- ASP.NET Identity stores your users' credentials in a database created by the application. When your users log in, the application validates their credentials by reading the database. Your project's Account folder contains the files that implement the various parts of membership: registering, logging in, changing a password, and authorizing access. Additionally, ASP.NET Web Forms supports OAuth and OpenID. These authentication enhancements allow users to log into your site using existing credentials, from such accounts as Facebook, Twitter, Windows Live, and Google. By default, the template creates a membership database using a default database name on an instance of SQL Server Express LocalDB, the development database server that comes with Visual Studio Express 2013 for Web.

Client Script and Client Frameworks

- You can enhance the server-based features of ASP.NET by including client-script functionality in ASP.NET Web Form pages. You can use client script to provide a richer, more responsive user interface to users. You can also use client script to make asynchronous calls to the Web server while a page is running in the browser.

Routing

- URL routing allows you to configure an application to accept request URLs that do not map to physical files. A request URL is simply the URL a user enters into their browser to find a page on your web site. You use routing to define URLs that are semantically meaningful to users and that can help with search-engine optimization (SEO).

State Management

- ASP.NET Web Forms includes several options that help you preserve data on both a per-page basis and an application-wide basis.

Security

- An important part of developing a more secure application is to understand the threats to it. Microsoft has developed a way to categorize threats: Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE). In ASP.NET Web Forms, you can add extensibility points and configuration options that enable you to customize various security behaviors in ASP.NET Web Forms.

Performance

- Performance can be a key factor in a successful Web site or project. ASP.NET Web Forms allows you to modify performance related to page and server control processing, state management, data access, application configuration and loading, and efficient coding practices.

Internationalization

- ASP.NET Web Forms enables you to create web pages that can obtain content and other data based on the preferred language setting for the browser or based on the user's explicit choice of language. Content and other data is referred to as resources and such data can be stored in resource files or other sources. In an ASP.NET Web Forms page, you configure controls to get their property values from resources. At run time, the resource expressions are replaced by resources from the appropriate localized resource file.

Debugging and Error Handling

- ASP.NET includes features to help you diagnose problems that might arise in your Web Forms application. Debugging and error handling are well supported within ASP.NET Web Forms so that your applications compile and run effectively.

Deployment and Hosting

- Visual Studio, ASP.NET, Azure, and IIS provide tools that help you with the process of deploying and hosting your Web Forms application.

<u>Basics Controls</u>

Q.-6. What are controls? Explain Types of controls in asp.net.

- ASP.NET Web controls are objects on ASP.NET Web pages that run when the page is requested and that render markup to a browser.
- Many Web controls resemble familiar HTML elements, such as buttons and text boxes.
- Other controls encompass complex behavior, such as a calendar controls, and controls that manage data connections.

Types of controls in asp.net

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.

Web Server Controls:

- ASP.NET Web server controls are objects on ASP.NET Web pages that run when the page is requested and that render markup to the browser. Many Web server controls are similar to familiar HTML elements, such as buttons and text boxes. Other controls encompass complex behavior, such as a calendar controls, and controls that you can use to connect to data sources and display data.

Web Part Controls:

- ASP.NET Web Parts controls are an integrated set of controls for creating Web sites that enable end users to modify the content, appearance, and behavior of Web pages directly in a browser. The topics in this section provide information on what Web Parts are, how they work, and how to use them to create user-customizable ASP.NET Web pages.

Types of asp.net framework controls:

- The ASP.NET Framework contains over 70 controls. These controls can be spread in different types, these types are below:

Standard Controls

- The standard controls enable you to render standard form elements such as buttons, input fields, and labels. We examine these controls in detail in the following chapter, "Using the Standard Controls.

Validation Controls

- The validation controls enable you to validate form data before you submit the data to the server. For example, you can use a RequiredFieldValidator control to check whether a user entered a value for a required input field.

Rich Controls

- The rich controls enable you to render things such as calendars, file upload buttons, rotating banner advertisements, and multi-step wizards.

Data Controls

- The data controls enable you to work with data such as database data. For example, you can use these controls to submit new records to a database table or display a list of database records.

Navigation Controls

- The navigation controls enable you to display standard navigation elements such as menus, tree views, and bread crumb trails.

Login Controls

- The login controls enable you to display login, change password, and registration forms.

HTML Controls

- The HTML controls enable you to convert any HTML tag into a server- side control.

Basic Web Server Controls

The Label Server Control

- The Label server control is used to display text in the browser.
- Because this is a server control, it is possible to dynamically change the text from your server-side code. It has Text Property which is used to assign caption to text.
- Label control can also have hot keys which is also known as access keys which are used to set focus in particular control like text box or button.
- Hot keys are assigned with the AccessKey attribute. In this case, Label1 uses a, and Label2 uses b. The second new attribute for the Label control is the AssociatedControlID attribute.
- The String value placed here associates the Label control with another server control on the form. The value must be one of the other server controls on the form. If not, the page gives you an error when invoked.

- See the following example of label control

```
<asp:Label ID="Label1" runat="server" Text="Name" AccessKey="a"
AssociatedControlID="TextBox1"></asp:Label><br />
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox><br />
<asp:Label ID="Label2" runat="server" Text="address" AccessKey="b"
AssociatedControlID="TextBox2"></asp:Label><br />
<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
```

The Literal Server Control

- The Literal server control works very much like the Label server control.
- It has Mode property that enables you to dictate how the text assigned to the control is interpreted by the ASP.NET engine.
- If you place some HTML code in the string that is output (for instance, <b>Here is some text</b>), the Literal control outputs just that and the consuming browser shows the text as bold:
- Here is some text
- If Mode="Encode" then it display tags <b></b> with the text instead of applying bold effect.
- See the following example

```
asp:Literal ID="Literal1" Runat="server" Mode="Encode" Text="<b>Here is some
text</b>"></asp:Literal>
```

The TextBox Server Control

- The TextBox server control is one of the most used controls in web forms.
- Text box on the form is used by the end user to input text. You can map the TextBox control to three different HTML elements used in your forms.
  - Simple TextBox
  - Password TextBox
  - Multiline TextBox
- First, the TextBox control can be used as a standard HTML text box, as shown in the following code

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

- Second, the TextBox control can allow end users to input their passwords into a form.as shown in following code

```
<asp:TextBox ID="TextBox1" Runat="server" TextMode="Password"></asp:TextBox>
```

- Third, the TextBox server control can be used as a multiline text box as shown in the following code.

```
asp:TextBox ID="TextBox1" Runat="server" TextMode="MultiLine"
Width="300px" Height="150px"></asp:TextBox>
```

The Button Server Control

- 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 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.

The CausesValidation Property

- If you have more than one button on your Web page and you are working with the validation server con-
- trols, you don't want to fire the validation for each button on the form. Setting the CausesValidation
- property to False is a way to use a button that will not fire the validation process.

The CommandName Property

- To have multiple buttons on your form all working from a single event, Command Name property is used.
- You must construct your Button controls in the manner explained below.

```
<asp:Button ID="Button1" Runat="server" Text="Button 1"
OnCommand="Button_Command" CommandName="DoSomething1" />
<asp:Button ID="Button2" Runat="server" Text="Button 2"
OnCommand="Button_Command" CommandName="DoSomething2" />
```

- Looking at these two instances of the Button control, you should pay attention to following things.
- The first thing to notice is the OnCommand event, which points to an event called Button_Command.
- You can see that both Button controls are working from the same event so to make a difference between two event,value placed in the CommandName property.

The LinkButton Server Control

- The LinkButton server control is a variation of the Button control.
- It is basically the same except that the LinkButton control looks like of a hyperlink. But, it isn't a typical hyperlink. When the end user clicks the link, it behaves like a button.
- This is an ideal control to use if you have a large number of buttons on your Web form.
- A LinkButton server control is constructed as follows:

```
<asp:LinkButton ID="LinkButton1" Runat="server" OnClick="LinkButton1_Click">
Submit your name to our database </asp:LinkButton>
```

The ImageButton Server Control

- The ImageButton control is also one type of Button control.
- It is almost exactly the same as the Button control except that it can show any custom image as the form's button instead of the typical buttons used on most forms.
- This means that one can create your own buttons as images and the end users can click the images to submit form data.
- A typical ImageButton is constructed as follows:

```
<asp:ImageButton ID="ImageButton1" Runat="server"
OnClick="ImageButton1_Click" ImageUrl="MyButton.jpg" />
```

- The ImageButton control specifies the location of the image used by using the ImageUrl property.

The HyperLink Server Control

- The HyperLink server control is used to provide hyperlinks on your Web pages.
- Hyperlinks are used to transfer from one page to another.
- You can set the text of a hyperlink using the control's Text attribute:

```
<asp:HyperLink ID=”HyperLink1” Runat=”server” Text=”Go to this page here”
NavigateUrl=”~/Default2.aspx”></asp:HyperLink>
```

- This server control creates a hyperlink on your page with the text Go to this page here.
- When the link is clicked, the user is redirected to the value that is placed in the NavigateUrl property.
- It can also display Image instead of Text using ImageUrl property.
- It is specially used when one need to show dynamic Links based on user input or value of database.

The DropDownList Server Control

- The DropDownList server control is similar to HTML select box on Web page.
- It should be used when you have a large collection of items from which you want the end user to select a single item.
- the select box generated by the DropDownList control displays a single item and allows the end user to make a selection from a larger list of items.
- Here's the code for DropDownList control:

```
<asp:DropDownList ID=”DropDownList1” Runat=”server”>
<asp:ListItem>Car</asp:ListItem>
<asp:ListItem>Airplane</asp:ListItem>
<asp:ListItem>Train</asp:ListItem>
</asp:DropDownList>
```

The ListBox Server Control

- The ListBox server control much like a the DropDownList control.
- It displays a collection of items.
- The ListBox control behaves differently from the DropDownList control in that it displays list of item to the end user at a time as well as end user can make multiple selections from the collection.
- A typical List Box control appears in code as follows:

```
<asp:ListBox ID=”ListBox1” Runat=”server”>
<asp:ListItem>Hematite</asp:ListItem>
<asp:ListItem>Halite</asp:ListItem>
<asp:ListItem>Limonite</asp:ListItem>
<asp:ListItem>Magnetite</asp:ListItem>
</asp:ListBox>
```

The CheckBox Server Control

- Check boxes on a Web form enables user to make multiple selections from a collection of items or specify a value of an item to be yes/no, on/off, or true/false.
- The CheckBox control allows you to place single or multiple check boxes on a form.
- All the Check Boxes are independent from each others.
- See the following example of checkbox control.

```
<asp:CheckBox ID=”CheckBox1” Runat=”server” Text=”CheckBox Demo example”
TextAlign=”Left” />
```

The CheckBoxList Server Control

- The CheckBoxList server control is quite similar to the CheckBox control, except that it allows working with a collection of items rather than a single item.
- CheckBoxList controls can be also bind with datasource like MsAccess, Oracle, MSSQL etc.
- See the following example of CheckBoxList control.

```
asp:CheckBoxList id="Check1"
RepeatLayout="flow"
runat="server">
<asp:ListItem>Item 1</asp:ListItem>
<asp:ListItem>Item 2</asp:ListItem>
<asp:ListItem>Item 3</asp:ListItem>
<asp:ListItem>Item 4</asp:ListItem>
<asp:ListItem>Item 5</asp:ListItem>
<asp:ListItem>Item 6</asp:ListItem>
</asp:CheckBoxList>
```

The RadioButton Server Control

- RadioButton Server control enables user to select single item from multiple items.
- The RadioButton control allows you to place single or multiple Radio Button on a form.
- All the Radio Buttons are mutually works together if GroupName property is set to same value for all radio buttons.
- See the following example of radio button control.

```
<asp:RadioButton ID="RadioButton1" runat="server" Text="Yes" GroupName="Set1" />
<asp:RadioButton ID="RadioButton2" runat="server" Text="No" GroupName="Set1"/>
The RadioButtonList Server Control
```

RadioButton List Control

- The RadioButtonList server is used to display a collection of radio buttons on a Web page.
- The RadioButtonList control is quite similar to the CheckBoxList and other list controls in that it allows you to iterate through to see what the user selected, to make counts, or to perform other actions.
- See the following example of radio button list control.

```
<asp:RadioButtonList ID="RadioButtonList1" runat="server">
<asp:ListItem Selected="True">English</asp:ListItem>
<asp:ListItem>Russian</asp:ListItem>
<asp:ListItem>Finnish</asp:ListItem>
<asp:ListItem>Swedish</asp:ListItem>
</asp:RadioButtonList>
```

Image Server Control

- The Image server control enables you to work with the images that appear on your Web page from the server-side code.
- Image server control has one important property, ImageUrl which is points to file & its location.
- It also has two other important property, height & width which are used to control height and width of the image.
- See the following example of image control.

```
<asp:Image ID="Image1" runat="server" ImageUrl="~/MyImage1.gif" />
```

Table Control

- The Table allows you to build an HTML table and specify its characteristics.
- A table can be built at design time with static content, but the Table control is often built programmatically with dynamic contents.
- Each Table control is made up of rows stored in the Rows collection of the control.
- Each row is made up of cells stored in the Cells collection of the each TableRow.
- You can display an image in the background of the Table control by setting the BackImageUrl property. By default, the horizontal alignment of the items in the table is not set.
- To add a single row to a Table control, you have to create new instances of the TableRow and TableCell objects.
- You create the TableCell objects first and then place them within a TableRow object that is added to a Table object.
- See the following example of table control.

```
<asp:Table ID="Table1" runat="server">
<asp:TableRowRunat="server" Font-Bold="True"
ForeColor="Black" BackColor="Silver">
<asp:TableHeaderCell>First Name</asp:TableHeaderCell>
<asp:TableHeaderCell>Last Name</asp:TableHeaderCell>
</asp:TableRow>
<asp:TableRow>
<asp:TableCell>one</asp:TableCell>
<asp:TableCell>two</asp:TableCell>
</asp:TableRow>
<asp:TableRow>
<asp:TableCell>three</asp:TableCell>
<asp:TableCell>four</asp:TableCell>
</asp:TableRow>
</asp:Table>
```

Panel Server Control

- The Panel server control used to hold a set of controls to process it or to lay out controls in asp.net web pages.
- It is basically a wrapper for other controls, which enable you to take a group of server controls along with other elements (such as HTML and images) and turn them into a single unit.
- The advantage of using the Panel control to hold a set of other control that can be processed as single unit.
- For example, setting the Font-Bold attribute to True causes each control or text within the Panel control to adopt this attribute, setting visible=false causes each control/text become invisiable in panel.
- Panel control has the capability to scroll with scrollbars that appear automatically depending on the amount of information that Panel control holds.