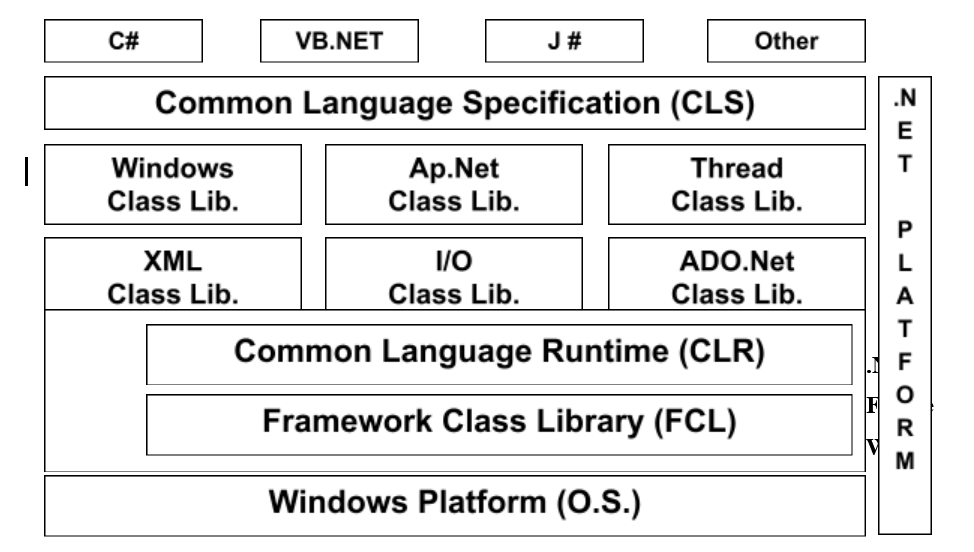
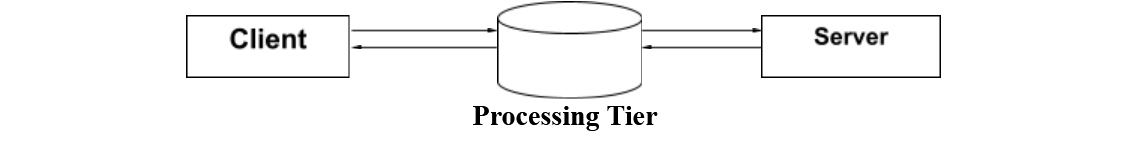
**CHAPTER-1: Building Asp.Net Pages**

Explain ASP.NET Framework

* .NET Framework is really a cluster of several technologies:
* **The .NET languages:** These include C# and VB .NET (Visual Basic .NET), the object oriented and modernized successor to Visual Basic 6.0; these languages also include JScript .NET (a server-side version of JavaScript), J# (a Java clone), and C++ with Managed Extensions.
* **The CLR (Common Language Runtime):** The CLR is the engine that executes all .NET programs and provides automatic services for these applications, such as security checking, memory management, and optimization.
* **The .NET Framework class library:** The class library collects thousands of pieces of prebuilt functionality that you can “snap in” to your applications. These features are sometimes organized into technology sets, such as ADO.NET (the technology for creating database applications) and Windows Forms (the technology for creating desktop user interfaces).
* **ASP.NET:** This is the engine that hosts web applications and web services, with almost any feature from the .NET class library. ASP.NET also includes a set of web-specific services.
* **Visual Studio:** This optional development tool contains a rich set of productivity and debugging features. The Visual Studio setup CDs (or DVD) include the complete .NET Framework, so you won’t need to download it separately.
* There are many features provided by .NET Framework which has made .NET popular and reliable in software development and web development industry. Following are features of .NET Platform / .NET Framework.
* **Multilanguage Development**
* **Multi-Device Development**
* **Platform and Processor independence**
* **Automatic memory management**
* **Easy Deployment**
* **Distributed Architecture**
* **Interoperability with Unmanaged code**
* **Security**
* **Performance and Scalability**
* **XML Support**
* **.NET Framework :**
* .NET Framework is a service or platform for building, deploying and running applications.
* The .NET Framework consists of 2 main parts: CLR and FC**L.**
* **Explain Client-Server Architecture:**

Explain Client Server Architecture

* **Every website has architecture depending upon its requirements. There are two types of architecture:**
* **2-tier architecture**
* **3-tier architecture**
* **2-tier architecture:**
* In this type of architecture, only one client and one server is there that is client is directly connected to database.
* In this type of architecture, client requests the web page and this request is sent to the server. The server then searches the particular web page in the database and if it is found then client request is fulfilled. 2-tier architecture is known as client-server architecture
* The **disadvantage** of 2-tier architecture is that server is loaded when the traffic increases.
* 
* **3-tier Architecture:**
* In this type of architecture, all interactions between client and sever is handled by processing tier.
* This architecture contains client, sever and one processing tier.
* The **advantage** of 3-tier architecture is that server load is reduced as all the interactions between client and server is handled by processing tier**.**
* 

Explain Application Web Server

* When you develop Web projects in Visual Studio, you need a Web server to test or run them. Visual Studio lets you test with different Web servers, including IIS Express, **Internet Information Services (IIS), or the built-in Visual Studio Development Server.** You can use any of these servers with a file-based Web application project. For a file-based Web site project, you can use IIS Express or the built-in Visual Studio Development Server.
* The following table provides summary guidance for choosing a Web server in Visual Web Developer.

|  |  |
| --- | --- |
| * **Web server** | * + - * **When to use** |
| * **IIS Express** | * **Use when the target Web server is IIS 7 but you do not want to (or cannot) use the full version of IIS 7. This requires Visual Studio 2010 Service Pack 1, and IIS Express must be installed separately. IIS Express hosts sites in a manner that is very similar to IIS 7.** |
| * **Visual Studio Development Server** | * **Use when you are working with an existing project or your site targets an older version of IIS, such as IIS 6, and it is not very important that your testing environment match the production environment closely. This server option is the default in Visual Studio. However, the Visual Studio Development Server runs in a different security context than full IIS, and may fail to reveal errors that can occur when you deploy to a production version of IIS.** |
| * **IIS** | * **Use when you want to test your Web application using the server environment that is closest to what the live site will run under, and it is practical for you to install and work with IIS on your development computer. However, it can be more complex to configure debugging and other tasks than if you use IIS Express or the Visual Studio Development Environment. Requires you to run Visual Studio as an administrator.** |

**Installation of IIS:**

Explain Installation Of IIS.

* **IIS Express offers the following features:**
* It supports and enables the same extensibility model and **Web.config file settings as IIS 7.**
* It does not require changes in your Web application code.
* It can be installed side-by-side with the full IIS web server as well as with the Visual Studio Development Server. You can choose a different Web server for each project.
* **In corporate environments, IIS Express offers the following features:**
* It does not require an administrator account in order to run or debug applications.
* It does not serve requests to a browser on another computer, making its approval easier in corporate environments.
* It supports multiple developers on the same computer. Configuration files, settings, and Web content are maintained on a per-user basis under the **%systemdrive%\users\<username> folder.**
* It can be installed on versions of Windows that do not support IIS 7.
* **IIS Express Requirements**
  + To install IIS Express in your computer, you must have the following:
* Windows XP, Vista, or Windows 7, or Windows Server 2008 or 2008 R2.
* The .NET Framework 4 or later.
* Visual Studio 2010 SP1 or Visual Web Developer 2010 Express SP1.
* **Installing IIS Express**
  + IIS Express is not installed automatically as part of Visual Studio 2010 SP1. To install IIS Express, you can use the Microsoft Web Platform Installer.
* **To install IIS Express in Visual Studio 2010 SP1**
* In a browser, go to the installation page of the Microsoft.com/Web site.
* Download the installer and then follow the steps to finish the installation.

Explain Types Of Files In Asp.Net.

* ASP.NET have many types of files. They are:
* **.aspx :** These are **ASP.NET web pages** (the .NET equivalent of the .asp file in an ASP application). They contain the user interface and, optionally, the underlying application code. Users request or navigate directly to one of these pages to start your web application.

* **.ascx :** These are ASP.NET **user controls**. User controls are similar to web pages, except that they can’t be accessed directly. Instead, they must be hosted inside an ASP.NET web page. User controls allow you to develop a small piece of user interface and reuse it in as many web forms as you want without repetitive code.
* **.asmx :** These are ASP.NET **web services**. Web services work differently than web pages, but they still share the same application resources, configuration settings, and memory.
* **web.config :** This is the **XML-based configuration file** for your ASP.NET application. It includes settings for customizing security, state management, memory management, and much more.
* **global.asax :** This is the **global application file.** You can use this file to define global variables (variables that can be accessed from any web page in the web application) and react to global events (such as when a web application first starts).
* **.cs / .vb :** These are **code-behind files** that contain **C# code or VB code**. They allow you to separate the application from the user interface of a web page. **The Page Class :** Every web page is a custom class that inherits from System.Web.UI.Page. By inheriting from this class, your web page class acquires a number of properties that your code can use. These include properties for enabling caching, validation, and tracing.

Explain types of controls in ASP.NET.

* There are two types of controls available in ASP.NET.
* They are known as:
* **Server Controls**
* **Client Controls**
* **Server Controls:**
* These are the controls which execute on server side. Most of the controls in ASP.NET controls works or can work as Server Controls. The controls which execute on server side have two very important properties.
* All server controls has one very important attribute called “runat”, which is written as **runat=”server”.**
* **Client Controls:**
* Client controls are controls which don’t execute on server side. Client controls don’t expose the functionality of event handling because event handling is generally done on the control which executes on server side.
* HTML controls are generally called as Client control but they can be converted to Server Control after adding ID and runat attribute to the code.
* The ASP.NET Framework (version 3.5) contains over 70 controls. These controls can be divided into following groups:
  + **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,etc.
  + **Login Controls :** The login controls enable you to display login, change password, and registration forms.
  + **Web Part Controls :** The Web Part controls enable you to build personalizable portal applications.
  + **HTML Controls :** The HTML controls enable you to convert any HTML tag into a server-side control.
  + **ASP.NET Mobile Controls :**   These are the controls which are somewhat similar to Web Controls but these are customized to support mobile clients. Mobile clients can be PDAs, Smart Phones, etc.

Explain Page Architecture or structure of ASP.NET

* Following are the elements of Asp.Net Page.
* **Directives:**
* A *directive* controls how an ASP.NET page is compiled. The beginning of a directive is marked with the characters <%@ and the end of a directive is marked with the characters %>. A directive can appear anywhere within a page. By convention, however, a directive typically appears at the top of an ASP.NET page.
* There are several types of directives that you can add to an ASP.NET page. Two of the most useful types are page and import directives.
* **Page Directives:**
* Page Directive can be used to enable tracing and debugging for a page.
* To change the default programming language of an Asp.Net you can use following page directive:
  + - **<%@Page%>**
* **Import Directives:**
* By default, only certain namespaces are automatically imported into an ASP.NET page. If you want to refer to a class that isn't a member of one of the default namespaces, then you must explicitly import the namespace of the class or you must use the fully qualified name of the class.
* For example, suppose that you want to send an email from an ASP.NET page by using the Send method of the SmtpMail class. The SmtpMail class is contained in the System.Web.Mail namespace. This is not one of the default namespaces imported into an ASP.NET page.

* **Code Declaration Block:**
* A *code declaration block* contains all the application logic for your ASP.NET page and all the global variable declarations, subroutines, and functions. It must appear within a <Script Runat="Server"> tag.
* **Asp.Net Controls:**
* *ASP.NET controls* can be freely interspersed with the text and HTML content of a page. The only requirement is that the controls should appear within a <form Runat="Server"> tag. And, for certain tags such as <span Runat="Server"> and <ASP:Label Runat="Server"/>, this requirement can be ignored without any dire consequences.
* One significant limitation of ASP.NET pages is that they can contain only one <form Runat="Server"> tag. This means that you cannot group ASP.NET into multiple forms on a page. If you try, you get an error.
* **Code Render Blocks:**
* If you need to execute code within the HTML or text content of your ASP.NET page, you can do so within *code render blocks*. The two types of code render blocks are *inline code* and *inline expressions*. Inline code executes a statement or series of statements. This type of code begins with the characters <% and ends with the characters %>.
* Inline expressions, on the other hand, display the value of a variable or method (this type of code is shorthand for Response.Write). Inline expressions begin with the characters <%= and end with the characters %>.
* **Server Side Comments:**
* You can add comments to your ASP.NET pages by using *server-side comment* blocks. The beginning of a server-side comment is marked with the characters <%-- and the end of the comment is marked with the characters --%>.
* Server-side comments can be added to a page for the purposes of documentation. Note that you cannot see the contents of server-side comment tags, unlike normal HTML comment tags, by using the View Source command on your Web browser.
* Server-side comments can also be useful when you're debugging an ASP.NET page. You can temporarily remove both ASP.NET controls and code render blocks from a page by surrounding these elements with server-side comments.
* **Server Side include Directives:**
* You can include a file in an ASP.NET page by using one of the two forms of the *server-side include directive*. If you want to include a file that is located in the same directory or in a subdirectory of the page including the file, you would use the following directive:
  + <!-- #INCLUDE file="includefile.aspx" -->
* Alternatively, you can include a file by supplying the full virtual path. For example, if you have a subdirectory named myDirectory under the wwwroot directory, you can include a file from that directory like this:
  + <!-- #INCLUDE virtual="/myDirectory/includefile.aspx" -->
* The include directive is executed before any of the code in a page. One implication is that you cannot use variables to specify the path to the file that you want to include. For example, the following directive would generate an error:
  + <!-- #INCLUDE file="<%=myVar%>" -->
* **Literal Text and HTML Tags:**
* The static portion of your page is built with plain old HTML tags and text. HTML content in a page is represented with the LiteralControl class. You can use the Text property of the LiteralControl class to manipulate the pure HTML portion of an ASP.NET page

Explain Web Form.

* Web Pages which you create in ASP.NET are known as Web Forms. Web Form is one of the strong part ASP.NET application because it provides the actual output to the client.
* Actually, web forms are displayed to clients in their browser and then after user’s interaction in terms of event, processed HTML output is shown back to the client.
* Web Forms allow to control based pages, which can have different types of controls on it.
* To Run Web Form ASP.NET engine reads .aspx file, executes series of events which has code written as per requirement.
* All Web Forms are stored .aspx extension. In ASP.NET all Web Forms are actually combination of two different files.
* **Presentation Logic File (.aspx)**
* **Code Behind File (.cs /.vb /.js)**

* Presentation Logic File stores HTML code which will be displayed on client’s browser. Whereas Code Behind File is a file where all events are wired up. Cole Behind File stores your application logic.
* <[%@PAGE...%](about:blank)> is one of the important page directive, which specifies some important properties about particular page like:
* **Language:** Which is the language will be used to write application logic . It can be C#, VB or J#.
* **AutoEventWireup :** This is set as true. It means all the events should be automatically wired up as per event delegation model. All efvents which can be generated under this page, should add event delegates automatically while execution.
* **CodeFile :** One of the very important part which says which file will be used as Code Behind File. The extension of Code File depends on the language which you have selected. It may be “.cs” or “.vb” or “.js”.
* **Inherits :** This specifies the name of the class of your web form which will be created in Code Behind File.
* Apart from some default properties <[%@PAGE...%](about:blank)> directive can have some more properties too which may be **Buffer, ContentType, ErrorPage, Language, MasterPageFile, Theme, EnableViewState, etc.**
* **~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
* **<%@ Page Language=”C#” =”true” CodeFile=”**[**Home.aspx.cs**](http://home.aspx.cs)**” Inherits=”Home” %>**
* **<html>**
* **<head runat=”server”>**
* **<title>Sample Web Page</title>**
* **</head>**
* **<body>**
* **<form id=”form1” runat=”server”>**
* **<div>**
  + **Hello Web Developer.<br/>**
* **</div>**
* **</form>**
* **</body>**
* **</html>**

Introduction to standard controls.

* **Label Control**
* Whenever you need to **modify the text displayed in a page dynamically,** you can use the Label control. Any string that you assign to the Label control's Text property is displayed by the Label when the control is rendered. You can assign simple text to the Text property or you can  assign HTML content.
* As an alternative to assigning text to the Text property, you can place the text between the Label control's opening and closing tags. Any text that you place before the opening and closing tags gets assigned to the Text property.
* The Label control supports several properties you can use to format the text displayed by the Label (this is not a complete list):
  + - **BackColor :** Enables you to change the background color of the label.
    - **BorderColor :** Enables you to set the color of a border rendered around the label.
    - **BorderStyle :** Enables you to display a border around the label. Possible values are NotSet, None, Dotted, Dashed, Solid, Double, Groove, Ridge, Inset, and Outset.
    - **BorderWidth :** Enables you to set the size of a border rendered around the label.
    - **CssClass :** Enables you to associate a Cascading Style Sheet class with label.
    - **Font** : Enables you to set the label's font properties.
    - **ForeColor :** Enables you to set the color of the content rendered by the label.
    - **Style :** Enables you to assign style attributes to the label.
    - **ToolTip :** Enables you to set a label's title attribute. (In Microsoft Internet Explorer, the title attribute is displayed as a floating tooltip.)
* **TextBox Control:**
* The TextBox control can be used to **display three different types of input fields depending on the value of its TextMode property**. The TextMode property accepts the following three values:
  + **SingleLine :** Displays a single-line input field.
  + **MultiLine :** Displays a multi-line input field.
  + **Password :** Displays a single-line input field in which the text is hidden.
* You can use the following properties to control the rendering characteristics of the TextBox control (this is not a complete list):
  + - **AccessKey :** Enables you to specify a key that navigates to the TextBox control.
    - **AutoCompleteType :** Enables you to associate an AutoComplete class with the TextBox control.
    - **AutoPostBack :** Enables you to post the form containing the TextBox back to the server automatically when the contents of the TextBox is changed.
    - **Columns :** Enables you to specify the number of columns to display.
    - **Enabled :** Enables you to disable the text box.
    - **MaxLength** Enables you to specify the maximum length of data that a user can enter in a text box (does not work when TextMode is set to Multiline).
    - **ReadOnly :** Enables you to prevent users from changing the text in a text box.
    - **Rows :** Enables you to specify the number of rows to display.
    - **TabIndex :** Enables you to specify the tab order of the text box.
    - **Wrap :** Enables you to specify whether text word-wraps when the TextMode is set to Multiline.
* The TextBox control also supports the following method:
  + - **Focus :** Enables you to set the initial form focus to the text box.
* The TextBox control supports the following event:
  + - **TextChanged :** Raised on the server when the contents of the text box are changed.
* Notice that the TextBox control also includes a property that enables you to associate the TextBox with a particular AutoComplete class. When **AutoComplete** is enabled, the user does not need to re-enter common informationsuch as a first name, last name, or phone numberin a form field. If the user has not disabled AutoComplete on his browser, then his browser prompts him to enter the same value that he entered previously for the form field (even if the user entered the value for a form field at a different website).
* **Checkbox Control**
* Checkbox control is used to accept **the choice from user**. It is used to display multiple choices from which user can select none of them or many or all of them. For example, if you want to accept Hobbies of user, you can use CheckBox control.
* The CheckBox control supports the following properties (this is not a complete list):
  + - **AccessKey :** Enables you to specify a key that navigates to the TextBox control.
    - **AutoPostBack :** Enables you to post the form containing the CheckBox back to the server automatically when the CheckBox is checked or unchecked.
    - **Checked :** Enables you to get or set whether the CheckBox is checked.
    - **Enabled :** Enables you to disable the TextBox.
    - **TabIndex :** Enables you to specify the tab order of the check box.
    - **Text :** Enables you to provide a label for the check box.
    - **TextAlign :** Enables you to align the label for the check box. Possible values are Left and Right.
* The CheckBox control also supports the following method:
  + - **Focus :** Enables you to set the initial form focus to the check box.
* The CheckBox control supports the following event:
  + - **CheckedChanged :** Raised on the server when the check box is checked or unchecked.
* **Radio Button Control**
* You always use the RadioButton control in a group. Only one radio button in a group of RadioButton controls can be checked at a time.
* The RadioButton control supports the following properties (this is not a complete list):
  + - * **AccessKey :** Enables you to specify a key that navigates to the RadioButton control.
      * **AutoPostBack :** Enables you to post the form containing the RadioButton back to the server automatically when the radio button is checked or unchecked.
      * **Checked :** Enables you to get or set whether the RadioButton control is checked.
      * **Enabled :** Enables you to disable the RadioButton.
      * **GroupName :** Enables you to group RadioButton controls.
      * **TabIndex :** Enables you to specify the tab order of the RadioButton control.
      * **Text :** Enables you to label the RadioButton control.
      * **TextAlign :** Enables you to align the RadioButton label. Possible values are Left and Right.
* The RadioButton control supports the following method:
  + **Focus :** Enables you to set the initial form focus to the RadionButton control.
* Finally, the RadioButton control supports the following event:
  + **CheckedChanged :** Raised on the server when the RadioButton is checked or unchecked.
* **Button Control**
* Button control is used to submit the data to the server.  Button control works like a Push Button when you click the data is submitted to the server.
* The Button control supports the following properties (this is not a complete list):
  + - **AccessKey :** Enables you to specify a key that navigates to the Button control.
    - **CommandArgument :** Enables you to specify a command argument that is passed to the Command event.
    - **CommandName :** Enables you to specify a command name that is passed to the Command event.
    - **Enabled :** Enables you to disable the Button control.
    - **OnClientClick :** Enables you to specify a client-side script that executes when the button is clicked.
    - **PostBackUrl :** Enables you to post a form to a particular page.
    - **TabIndex :** Enables you to specify the tab order of the Button control.
    - **Text :** Enables you to label the Button control.
    - **UseSubmitBehavior :** Enables you to use JavaScript to post a form.
* The Button control also supports the following method:
  + - **Focus :** Enables you to set the initial form focus to the Button control.

* The Button control also supports the following two events:
  + - **Click :** Raised when the Button control is clicked.
    - **Command :** Raised when the Button control is clicked. The CommandName and CommandArgument are passed to this event.
* **Link Button Control**

* The LinkButton control, like the Button control, enables you to post a form to the server. Unlike a Button control, however, the LinkButton control renders a link instead of a push button.
* Behind the scenes, the LinkButton control uses JavaScript to post the form back to the server. The hyperlink rendered by the LinkButton control looks like this:

<a id="lnkSubmit" href="javascript:\_\_doPostBack('lnkSubmit','')">Submit</a>

* Clicking the LinkButton invokes the JavaScript \_\_doPostBack() method, which posts the form to the server. When the form is posted, the values of all the other form fields in the page are also posted to the server.
* The LinkButton control supports the following properties (this is not a complete list):
  + - **AccessKey :** Enables you to specify a key that navigates to the Button control.
    - **CommandArgument :** Enables you to specify a command argument that is passed to the Command event.
    - **CommandName :** Enables you to specify a command name that is passed to the Command event.
    - **Enabled :** Enables you to disable the LinkButton control.
    - **OnClientClick :** Enables you to specify a client-side script that executes when the LinkButton is clicked.
    - **PostBackUrl :** Enables you to post a form to a particular page.
    - **TabIndex :** Enables you to specify the tab order of the LinkButton control.
    - **Text :** Enables you to label the LinkButton control.
* The LinkButton control also supports the following method:
  + - **Focus :** Enables you to set the initial form focus to the LinkButton control.
* The LinkButton control also supports the following two events:
  + - **Click :** Raised when the LinkButton control is clicked.
    - **Command :** Raised when the LinkButton control is clicked. The CommandName and CommandArgument are passed to this event.
* **Image Button Control**

* The ImageButton control, like the Button and LinkButton controls, enables you to post a form to the server. However, the ImageButton control always displays an image.
* The ImageButton includes both an ImageUrl and AlternateText property. The ImageUrl contains the path to the image that the ImageButton displays. The AlternateText property is used to provide alternate text for the image used by screen readers and text-only browsers.
* Notice that the event handler for an Image control's Click event is different than that for the other button controls. The second parameter passed to the event handler is an instance of the ImageClickEventArgs class. This class has the following properties:
  + - **X :** The x coordinate relative to the image the user clicked.
    - **Y :** The y coordinate relative to the image the user clicked.
* You can use the ImageButton control to create a simple image map. The ImageButton can be used to create a server-side image map. Server-side image maps are not accessible to persons with disabilities. A better method for creating an ImageMap is to use the ImageMap control, which enables you to create a client-side image map.
* The ImageButton control supports the following properties (this is not a complete list):
  + - **AccessKey :** Enables you to specify a key that navigates to the ImageButton control.
    - **AlternateText :** Enables you to provide alternate text for the image (required for accessibility).
    - **DescriptionUrl :** Enables you to provide a link to a page that contains a detailed description of the image (required to make a complex image accessible).
    - **CommandArgument :** Enables you to specify a command argument that is passed to the Command event.
    - **CommandName :** Enables you to specify a command name that is passed to the Command event.
    - **Enabled :** Enables you to disable the ImageButton control.
    - **GenerateEmptyAlternateText :** Enables you to set the AlternateText property to an empty string.
    - **ImageAlign :** Enables you to align the image relative to other HTML elements in the page. Possible values are AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop, and Top.
    - **ImageUrl :** Enables you to specify the URL to the image.
    - **OnClientClick :** Enables you to specify a client-side script that executes when the ImageButton is clicked.
    - **PostBackUrl :** Enables you to post a form to a particular page.
    - **TabIndex :** Enables you to specify the tab order of the ImageButton control.
* The ImageButton control also supports the following method:
  + - **Focus :** Enables you to set the initial form focus to the ImageButton control.

The ImageButton control also supports the following two events:

* + - **Click :** Raised when the ImageButton control is clicked.
    - **Command :** Raised when the ImageButton control is clicked. The CommandName and CommandArgument are passed to this event.
* **Image Control**

* Image control is used to display an image. It has got some following properties.
  + - **AlternateText** Enables you to provide alternate text for the image (required for accessibility).
    - **DescriptionUrl** Enables you to provide a link to a page that contains a detailed description of the image (required to make a complex image accessible).
    - **GenerateEmptyAlternateText** Enables you to set the AlternateText property to an empty string.
    - **ImageAlign** Enables you to align the image relative to other HTML elements in the page. Possible values are AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop, and Top.
    - **ImageUrl** Enables you to specify the URL to the image.

* The Image control supports three methods for supplying alternate text. If an image represents page content, then you should supply a value for the AlternateText property. For example, if you have an image for your company's logo, then you should assign the text "My Company Logo" to the AlternateText property.

* If an Image control represents something really complexsuch as a bar chart, pie graph, or company organizational chartthen you should supply a value for the DescriptionUrl property. The DescriptionUrl property links to a page that contains a long textual description of the image.

* Finally, if the image is used purely for decoration (it expresses no content), then you should set the GenerateEmptyAlternateText property to the value TRue. When this property has the value TRue, then an alt="" attribute is included in the rendered <img> tag. Screen readers know to ignore images with empty alt attributes.
* **Image Map Control**
* The ImageMap control enables you to create a client-side image map. An image map displays an image. When you click different areas of the image, things happen.

* For example, you can use an image map as a fancy navigation bar. In that case, clicking different areas of the image map navigates to different pages in your website.
* You also can use an image map as an input mechanism. For example, you can click different product images to add a particular product to a shopping cart.
* An ImageMap control is composed out of instances of the HotSpot class. A HotSpot defines the clickable regions in an image map. The ASP.NET framework ships with three HotSpot classes:
  + - **CircleHotSpot :** Enables you to define a circular region in an image map.
    - **PolygonHotSpot :** Enables you to define an irregularly shaped region in an image map.
    - **RectangleHotSpot :** Enables you to define a rectangular region in an image map.
* Each RectangleHotSpot includes Left, Top, Right, and Bottom properties that describe the area of the rectangle. Each RectangleHotSpot also includes a NavigateUrl property that contains the URL to which the region of the image map links.
* The ImageMap control supports the following properties (this is not a complete list):
  + - **AccessKey** Enables you to specify a key that navigates to the ImageMap control.
    - **AlternateText** Enables you to provide alternate text for the image (required for accessibility).
    - **DescriptionUrl** Enables you to provide a link to a page which contains a detailed description of the image (required to make a complex image accessible).
    - **GenerateEmptyAlternateText** Enables you to set the AlternateText property to an empty string.
    - **HotSpotMode** Enables you to specify the behavior of the image map when you click a region. Possible values are Inactive, Navigate, NotSet, and PostBack.
    - **HotSpots** Enables you to retrieve the collection of HotSpots contained in the ImageMap control.
    - **ImageAlign** Enables you to align the image map with other HTML elements in the page. Possible values are AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop, and Top.
    - **ImageUrl** Enables you to specify the URL to the image.
    - **TabIndex** Enables you to specify the tab order of the ImageMap control.
    - **Target** Enables you to open a page in a new window.
* The ImageMap control also supports the following method:
  + - **Focus** Enables you to set the initial form focus to the ImageMap control.

* Finally, the ImageMap control supports the following event:
  + - **Click** Raised when you click a region of the ImageMap and the HotSpotMode property is set to the value PostBack.
* **Panel Control**

* The Panel control enables you to work with a group of ASP.NET controls. You can use a Panel control to hide or show a group of ASP.NET controls.
* The Panel control supports the following properties (this is not a complete list):
  + - **DefaultButton :** Enables you to specify the default button in a Panel. The default button is invoked when you press the Enter button.
    - **Direction :** Enables you to get or set the direction in which controls that display text are rendered. Possible values are NotSet, LeftToRight, and RightToLeft.
    - **GroupingText :** Enables you to render the Panel control as a fieldset with a particular legend.
    - **HorizontalAlign :** Enables you to specify the horizontal alignment of the contents of the Panel. Possible values are Center, Justify, Left, NotSet, and Right.
    - **ScrollBars :** Enables you to display scrollbars around the panel's contents. Possible values are Auto, Both, Horizontal, None, and Vertical.
* **Hyperlink Control**

* The HyperLink control enables you to create a link to a page. Unlike the LinkButton control, the HyperLink control does not submit a form to a server.
* The HyperLink control supports the following properties (this is not a complete list):
  + - **Enabled :** Enables you to disable the hyperlink.
    - **ImageUrl :** Enables you to specify an image for the hyperlink.
    - **NavigateUrl :** Enables you to specify the URL represented by the hyperlink.
    - **Target :** Enables you to open a new window.
    - **Text :** Enables you to label the hyperlink.
* Notice that you can specify an image for the HyperLink control by setting the ImageUrl property. If you set both the Text and ImageUrl properties, then the ImageUrl property takes precedence.