DOTNET UNIT-4

TOPICS

S.NO	NAME OF THE TOPICS
1.	INTRO TO ASP.NET &DIFF BETWEEN ASP.NET VS ASP 3.0
2.	PAGE FRAMEWORK
3.	HTML SERVER CONTROLS
4.	WEB CONTROLS
5.	VALIDATION CONTROLS
6.	EVENTS
7.	CSS
8.	STATE MANAGEMENT

1.INTRO TO ASP.NET:

- ASP NET is built upon the .NET FRAMEWORK.
- ASP-NET is used to build web Forms & web services.
- It's an integrated web development platform that allows you to use Various tools to build rich web application.
- It offers great advance in State management, Scalability, Caching, deployment, Security, performance & Support For a Web Services.
- It also contains a collection of Controls to we & build on within the Context of webpage.
- ASP.NET was developed by Microsoft as [ASP+] Active Server Page Plus, later name changed as ASP.NET.
- It became a part of the NET family.

ASP.NET CONTROLS

- It Includes a number Of Controls to program.
- There Controls are used within a web Forms to produce Forms with the exact functionality.

- **The web Form are a number of new available Controls such as HTML CONTROLS, Web Controls, Validation Controls, User Controls.**
- **It allows you to program this Controls use in Vb.NET to Provide Specific Functionality.**
- **ASP. NET has a set of Validation Controls that allows you to specify Client-side Validation rules.**

Key Difference in ASP.NET & ASP. 3.0:

- **ASP. NET provides a number of new & exciting Features.**
- **ASP.NET provides a better way to build webpages.**
- **It is easier & Faster for development.**

Key features:

1. **Page extension.**
2. **Code neutrality**
3. **Notepad**
4. **Web Form.**
5. **Security.**
6. **The IDE.**
7. **App Configure.**
8. **Code Compile.**

DIFFERENCE BETWEEN ASP.NET & ASP 3.0

S.NO	ASP 3.0	S.NO	ASP.NET
1	Active server page 3.0	1	Active server page.net
2	Year=>2000 and below	2	Year=>2002 and above
3	Interpreted (line by line)	3	compiled
4	Languages :Vb-script and j-script	4	Languages : C#,vb.net
5	Procedural scripting	5	Oops concepts
6	Slower in performance	6	Faster in performance
7	Used in limited & small app only	7	Used in high & enterprise-level apps
8	Basic and dynamic pages	8	Webforms, webapp, server control

2.PAGE FRAME WORK:

- ASP.NET was a Considerable amount of HTML to layout pages.
- HTML Stands For Hyper Text Markup Languages, that marks up text to the browser.
- You Can Create a basic form using HTML Form elements.
- Using HTML, You will be able to layout your ASP.NET pages with <html> </html> tags

<html>
Content
</html>

NEXT tag is <head> tag:

- The head section is the First Section within the HTML document.
- The head section of the document can be Contain a number of items, It Contain Vb.NET Scripts.

Syntax:

<html>

<head>

Content

</head>

</html>

Eg:

<html>

<head>

<title> My First Doc </title>

<script language = "Vb" runat ="server" >

</script>

</head>

</html>

- **The Information heads tags. Can be title, Script tag, meta, Style Sheet.**

Next tag in html <body>tags

- **The Second Section is the body Section of the document.**
- **The body Section as Constructs using opening closing <body> tags.**

SYNTAX :

<html>

<head> </head>

<body>

<Content of body >

</body>

</html>

Example:

```
<html>
<head> </head>
<body>
<b> Hello </b> John!
</body>
</html>
```

Example program Far Page Framework:

```
<%@ Page Language="VB" >
<html>
<head>
<title> ASPNET Page Framework</title>
</head>
<body>
<form id = "Form 1" runat = "Server”>
<asp:Label ID="label1" runat="server"  Text="Welcome"> </asp:Label>
<br>
<asp:Button ID="Butlon1" runat=Server" Text="Click">
</form>
</body>
</html>
```

3. CSS:

- **Css [Cascading Style Sheet] Is used to define the look & Feel [presentation] of a web Page Such as layout, colors, Fonts, etc.**

- **In ASP.NET, CSS is used the Same way it is wed in any web application development.**
- **It is used to Style HTML elements and ASP.NET Server Control**

Using css in ASP.NET:

There are three main ways to use css in ASP.NET.

1. **Inline Css**
2. **Internal Css**
3. **EXTERNAL CSS**

l. Inline css

- **You can directly add css styles to a specific element to Using the style attributes**

eg:-

<asp : Label ID="Label 1" runat = "server" Text = "Hello"

Style = "Color: blue; font-size:20px > </asp:label>

2. Internal CSS

- **Css can be defined inside the <style> tag within <head>Section of the aspx.page.**

```
<html>
<head>
<style>
body {
background-Color: #fofofo;
}
.title{
Color: green;
font-size: 24px;
}
```

</style>

</head>

<body>

<asp:Label ID= "label2", runat=”Serves” CssClass="title">

<asp:Label Text="Hello">

3. EXTERNAL CSS

- **Create a .css File & link it in <head> Section.**

<link href="styles/style.css" rel="stylesheet"/>

How css works with ASP.NET Controls.

- **Asp.Net Controls [like Label, Button, GridView, etc] are renderred as HTML elements in the browser**
- **use the css Class property to apply css internal css. &external css**

4. HTML SERVER CONTROLS, WEB CONTROLS, AND VALIDATION CONTROLS IN ASP.NET

- ASP.NET provides different types of server-side controls that make web development easier by allowing programmers to manipulate web page elements using code on the server (VB.NET).
- These are classified mainly into:

1. HTML Server Controls.
2. Web Server Controls.
3. Validation Controls.

1. HTML Server Controls

📄 Definition

- HTML Server Controls are standard HTML elements that can be processed on the server side by ASP.NET.
- They are created by adding the runat="server" attribute to normal HTML tags.
- Once this attribute is added, the control becomes a server object that can be accessed and manipulated in server-side code.

❓ Example

```
<form id="form1" runat="server">
  <input type="text" id="txtName" runat="server" />
  <input type="submit" id="btnSubmit" value="Submit" runat="server" />
</form>
```

⚙ How It Works

- When the page runs, ASP.NET converts each runat="server" tag into an object. The control's state and properties are managed on the server.
- You can read or modify its value in the code-behind file (e.g., txtName.Value in VB)

Features

- Directly correspond to standard HTML tags.
- Easy for developers who already know HTML.
- Fewer built-in features compared to Web Controls.
- Render pure HTML output on the browser (lightweight).
- Event handling supported (like ServerClick).

📋 Common HTML Server Controls

HTML Tag	ASP.NET Class	Description
<form>	HtmlForm	Defines a form on a web page.
<input type="text"> .	HtmlInputText	Single-line text box
<input type="password">	HtmlInputPassword	Password input field.
<input type="submit">	HtmlInputSubmit	Submit button.
<input type="checkbox">	HtmlInputCheckBox	Checkbox input.
<select> .	HtmlSelect	Drop-down list
<textarea>	HtmlTextArea	Multi-line text box.
<a>	HtmlAnchor	Hyperlink element.

🧠 Example (Code-Behind)

VB.NET:

```
Protected Sub btnSubmit_ServerClick(sender As Object, e As EventArgs)
  lblOutput.InnerText = "Welcome, " & txtName.Value
End Sub
```

2. Web Server Controls

📄 Definition

- Web Server Controls are ASP.NET-specific controls derived from the System.Web.UI.WebControls namespace.

- They are not limited to HTML — they are more powerful and flexible

components created by Microsoft for dynamic web development.

Example

```
<asp:TextBox ID="txtName" runat="server"></asp:TextBox>
<asp:Button ID="btnSubmit" runat="server" Text="Submit" />
<asp:Label ID="lblOutput" runat="server"></asp:Label>
```

Features

- Rich set of properties, methods, and events.
- Provide consistent look and feel across browsers.
- Support AutoPostBack — automatically submit form data when value changes.
- Can be data-bound to databases, XML, or collections.
- Easier to handle complex logic through event-driven programming.
- Generate complex HTML automatically.

Commonly Used Web Controls

Control	Description
Label	Displays static text.
TextBox	Accepts user input (single/multi-line).
Button	Triggers server-side events.
DropDownList	Displays list of selectable items.
ListBox	Displays multiple items for selection.
CheckBox	Represents a true/false option.
RadioButton	Used for mutually exclusive options.
HyperLink	Creates a clickable link.
Image	Displays an image.
Panel	Groups other controls.
GridView	Displays and manages tabular data.
DataList/Repeater	Display records with templates.

Example with Event

```
<asp:TextBox ID="txtName" runat="server"></asp:TextBox>
<asp:Button ID="btnShow" runat="server" Text="Show Name" OnClick="btnShow_Click" />
<asp:Label ID="lblMessage" runat="server"></asp:Label>
```

Code-Behind (VB.NET):

```
Protected Sub btnShow_Click(sender As Object, e As EventArgs)
    lblMessage.Text = "Welcome " & txtName.Text
End Sub
```

3. Validation Controls

📄 Definition

- Validation Controls are special ASP.NET controls used to validate user input.
- They ensure that form fields contain correct, required, and properly formatted data before submission to the server.

Validation can occur:

Client side (browser):	Faster, uses JavaScript.
Server side:	Ensures data security even if JavaScript is disabled.

❓ Features

- Automatically check input values.
- Easy to use — no manual validation code needed.
- Display error messages when validation fails.
- Can be combined with a Validation Summary to show all errors together.

📋 Types of Validation Controls

Control	Description	Example Usage
RequiredFieldValidator	Ensures field is not empty.	Name, Email fields.
CompareValidator	Compares two controls or a control and a constant value.	Confirm password.
RangeValidator	Ensures a value is within a specific range.	Age between 18–60.
RegularExpressionValidator	Checks format using regular expressions.	Email, Phone format.
CustomValidator	Allows custom validation logic.	Complex business rules.
ValidationSummary	Displays summary of all errors.	At top of form.

 Example

```
<asp:TextBox ID="txtEmail" runat="server"></asp:TextBox>
<asp:RequiredFieldValidator ID="reqEmail" runat="server">

   ControlToValidate="txtEmail"
   ErrorMessage="Email is required!"
   ForeColor="Red" />

<asp:RegularExpressionValidator ID="regEmail" runat="server"
   ControlToValidate="txtEmail"
   ErrorMessage="Invalid email format!"
   ValidationExpression="\w+@\w+\.\w+"
   ForeColor="Red" />

<asp:ValidationSummary ID="valSummary" runat="server"
   HeaderText="Please fix the following errors:"
   ForeColor="Red" />
```

⚖ Difference Between HTML Server Control and Web Server Control

Feature	HTML Server Control	Web Server Control
Definition	HTML tags made server-side using runat="server".	ASP.NET-specific controls.
Namespace	System.Web.UI.HtmlControls	System.Web.UI.WebControls
Rendering	Direct HTML output.	ASP.NET renders complex HTML.
Look & Feel	Browser dependent.	Consistent across browsers.
Events	Limited (e.g., ServerClick).	Rich event model (Click, TextChanged).
Data Binding	Not supported.	Supported.
Ease of Use	Simple for HTML developers.	Powerful for ASP.NET developers.
Performance	Lightweight.	Slightly heavier.

6. Events

- **An event occurs when an object on the page sends a message to the server that some Sort of action to taken place.**
- **Controls are objects that warts for specific events & notify the server when those specified events take place.**
- **The act that initiates the notification is called trigger.**
- **The object that sends the trigger is called an event sender**
- **The Code that runs when an event is triggered Ps called an event handler.**

EXAMPLE:

- **An common example of an object triggering an event a button on a page.**
- **The button 98 the object, & the user clicking on it initiates the trigger (the OnClick event), then they sent to the appropriate event handler.**
- ***In ASP.NET, events are used to handle User Inter-actions with web Controls on a web page [like Button, TextBox, DropDown etc)**
- **Each Control Can raise Certain Server - Side events which are processed on the Server after a post back.**

eg:- event in <asp:button>

1. OnClick

2. On Command.

3. OnInit.

4. OnLoad.

& On Disposed.

*** If you wanted the button to monitor click events, you can use OnClick event & then make the event handler [Buttont-Click] perform the Junction the event wanted.**

eg [.aspx File].

<asp:Button ID = "btnSubmit" runat="server" Text="Click me"

OnClick = "Button1_Click"/>

In Vb.NET.

Sub Button1_Click (sender As Object,E As EventArgs)

Button 1. Text = "The button text Can Change".

End Sub

- **when button (Click me) is Selected It process to the Vb.NET & print the text.**

Event argument:

- **The event handlers that you build. For web Form Application Contain two arguments [sender As object & e As EventArgs.]**

1. **sender As Object : An object that represents the object that raised the event**
2. **e As Event Args: Contains all the event specific data that the event sender passes along.**

Other Args:

1. File system EventArgs.

2. Key EventArgs.

3. Command EventArgs.

Event work with ASP.NET

1. **User Interact with control (Clicks a button).**
2. **Page sends a post back request to the Server.**
3. **ASP.NET processes. the page & raise the event.**
4. **The Corresponding event handler executes.**
5. **Page is re-rendered &sent back to the Client.**

8.STATE MANAGEMENT IN .NET

- **State Management in ASP.NET is the technique used to maintain the state of a web page between multiple requests.**
- **Since HTTP is a stateless protocol, it does not store user information or page data by default.**
- **To overcome this, ASP.NET provides several mechanisms to maintain state either on the client-side or on the server-side.**

TYPES OF STATE MANAGEMENT

There are two major categories of state management techniques in .NET:

1. CLIENT-SIDE STATE MANAGEMENT

In client-side state management, the data is stored on the user's browser or device. It is
useful for lightweight data storage and reduces server load.

View State

- Stores data of controls between postbacks using a hidden field named __VIEWSTATE.
- Data is encoded and maintained automatically by ASP.NET.
- Advantages: Easy to use, no server resource needed.
- Disadvantages: Increases page size, not secure for sensitive data.

Hidden Fields

- Hidden fields store data that is not visible to the user but available when the form is submitted.
- Advantages: Simple to implement, requires no server resource.
- Disadvantages: Data can be viewed or modified using browser developer tools.

Cookies

- Small text files stored on the client's machine.
- Used to remember user preferences or login data.
- Advantages: Persistent across sessions, simple to implement.
- Disadvantages: Size limited (4KB), can be deleted or disabled by the user.

Query String

- Passes data in the URL (e.g., page.aspx?id=101).
- Advantages: Easy to implement, allows data transfer between pages.
- Disadvantages: Limited length, visible to the user, not secure.

Control State

- Stores critical control information similar to ViewState but cannot be disabled.
- Advantages: Ensures important data (like paging info) is retained.
- Disadvantages: Slightly increases page size.

SERVER-SIDE STATE MANAGEMENT

In server-side state management, the data is stored on the web server. It is more secure and suitable for larger or sensitive data.

Session State

- Maintains user-specific data for the duration of a session.
- A unique SessionID is assigned to each user.
- Advantages: Secure, user-specific, can store large data.
- Disadvantages: Increases server memory usage, expires after timeout or server restart.

Application State

- Shared data across all sessions and users.
- Stored in the Application object.
- Advantages: Useful for global information like visitor count.
- Disadvantages: Memory-intensive, not user-specific.

Cache

- Temporary storage of frequently accessed data.
- Improves performance by reducing database calls.
- Advantages: Increases application performance.
- Disadvantages: Data can become outdated, consumes server memory.

State management is an essential concept in ASP.NET that ensures a smooth and interactive user experience by maintaining data across web requests. Choosing the right state management technique depends on the application's security, performance, and scalability requirement.