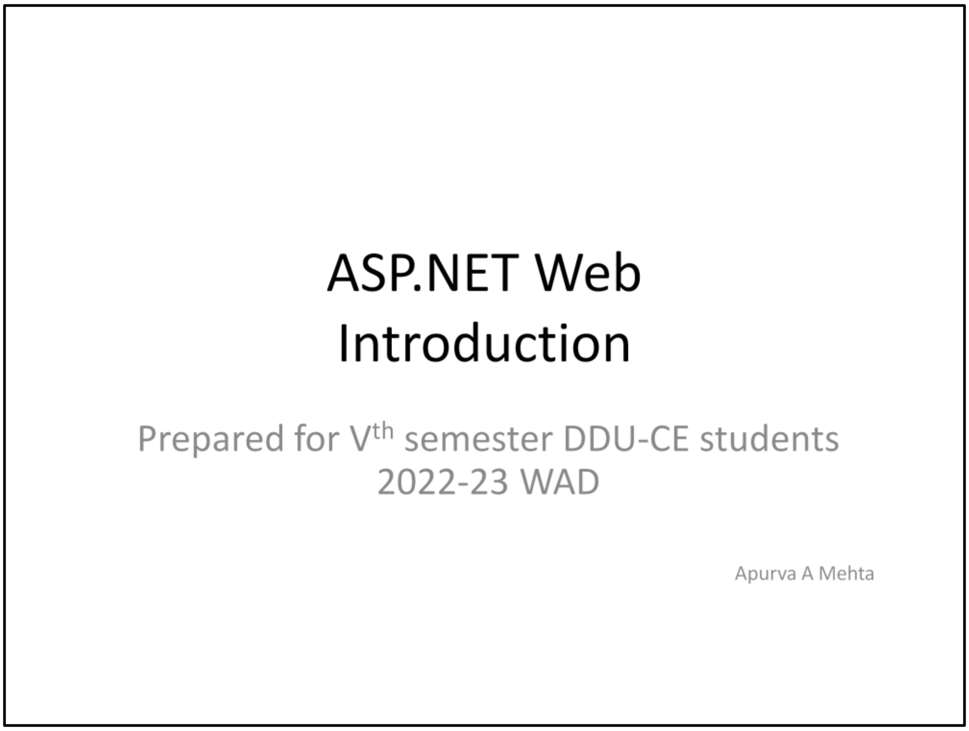
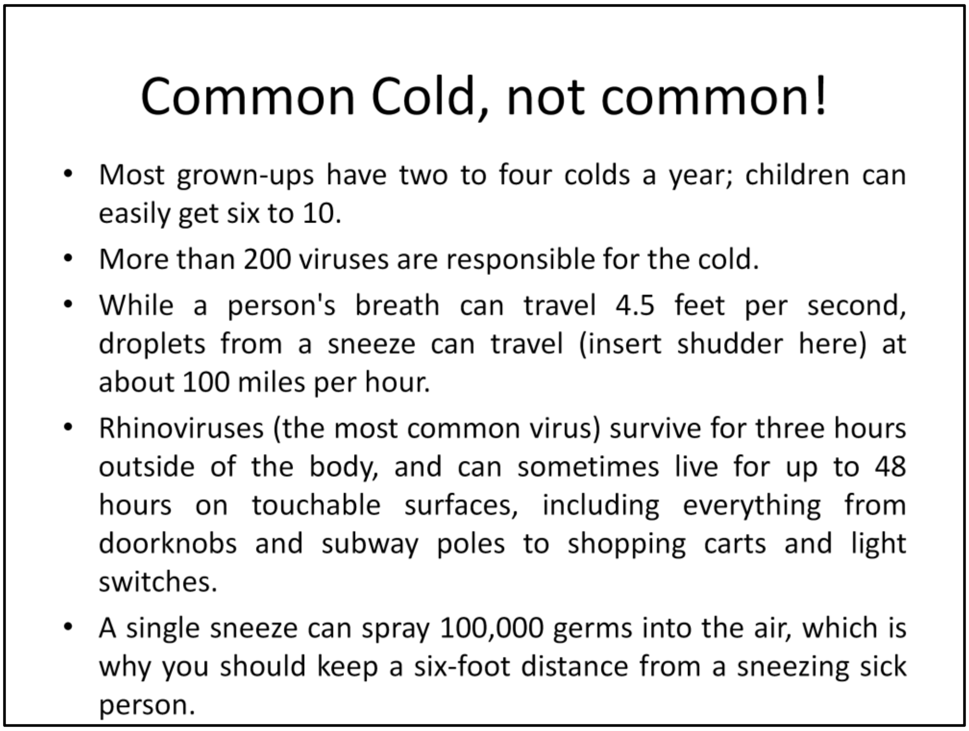


ASP.NET Web Introduction

Prepared for Vth semester DDU-CE students 2022-23 WAD

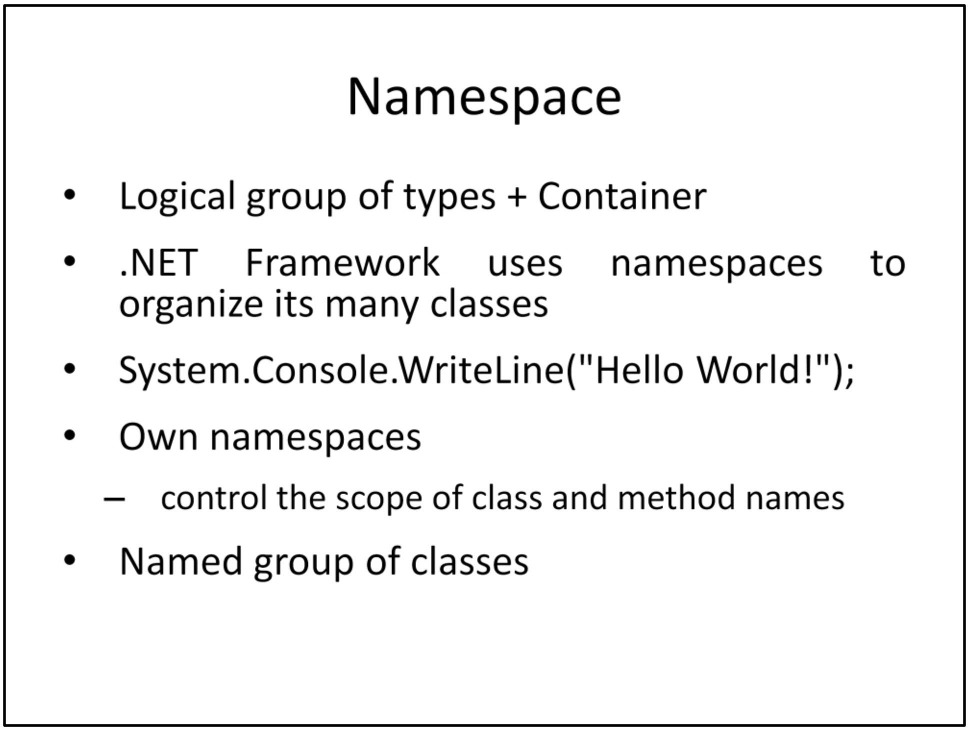
Apurva A Mehta



Common Cold, not common!

- Most grown-ups have two to four colds a year; children can easily get six to 10.
- More than 200 viruses are responsible for the cold.
- While a person's breath can travel 4.5 feet per second, droplets from a sneeze can travel (insert shudder here) at about 100 miles per hour.
- Rhinoviruses (the most common virus) survive for three hours outside of the body, and can sometimes live for up to 48 hours on touchable surfaces, including everything from doorknobs and subway poles to shopping carts and light switches.
- A single sneeze can spray 100,000 germs into the air, which is why you should keep a six-foot distance from a sneezing sick person.

https://www.treehugger.com/crazy-facts-about-common-cold-4858428



Namespace

- Logical group of types + Container
- .NET Framework uses namespaces to organize its many classes
- System.Console.WriteLine("Hello World!");
- Own namespaces
  - control the scope of class and method names
- Named group of classes

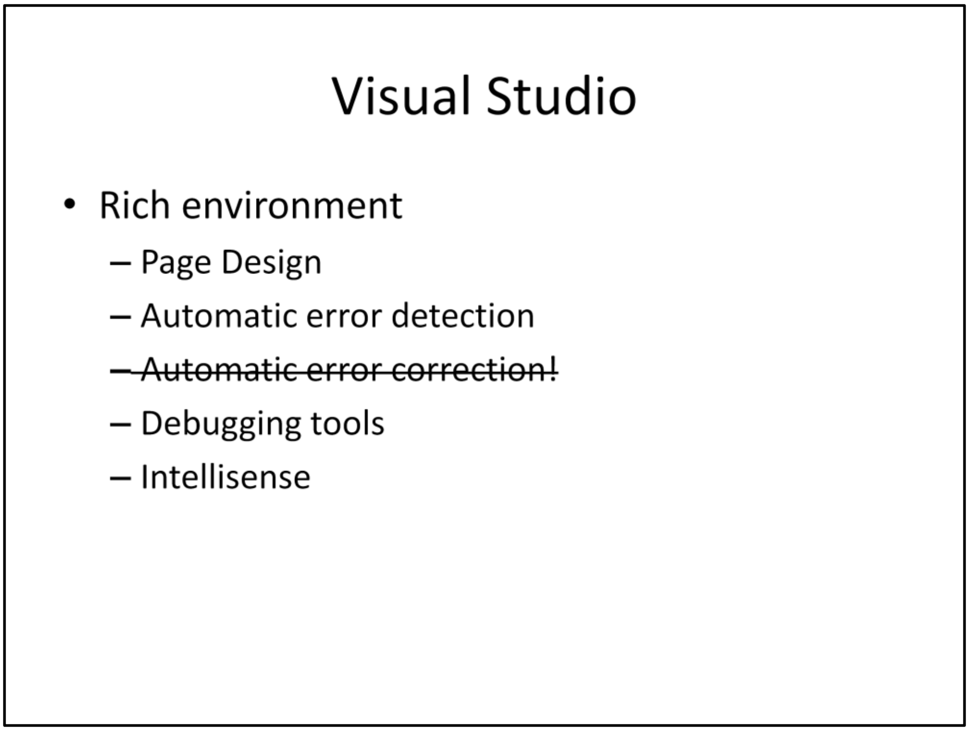
NameSpace is the Logical group of types or we can say namespace is a container

Namespaces are the basic building block for the .NET framework.

Namespaces are a way to define the classes and other types of information into one hierarchical structure

System is the basic namespace used by every .NET code.
System is a namespace and Console is a class in that namespace.
declaring your own namespaces can help you control the scope of class and method names in larger programming projects.

The biggest advantage of using namespace is that the class names which are declared in one namespace will not clash with the same class names declared in another namespace. It is also referred as **named group of classes** having common features. The members of a namespace can be **namespaces**, **interfaces**, **structures**, **and delegates.**



Visual Studio

- Rich environment
  - Page Design
  - Automatic error detection
  - ~~Automatic error correction!~~
  - Debugging tools
  - Intellisense

Visual Studio is a complete set of development tools for building ASP.NET Web applications, XML Web Services, desktop applications, and mobile applications.

It has rich code editor, debugger, designer and other development utilities.

Some of the features of Visual Studio include the following:

Page design: You can create an attractive page with drag-and-drop ease by using Visual Studio's integrated web form designer. You don't need to understand HTML.

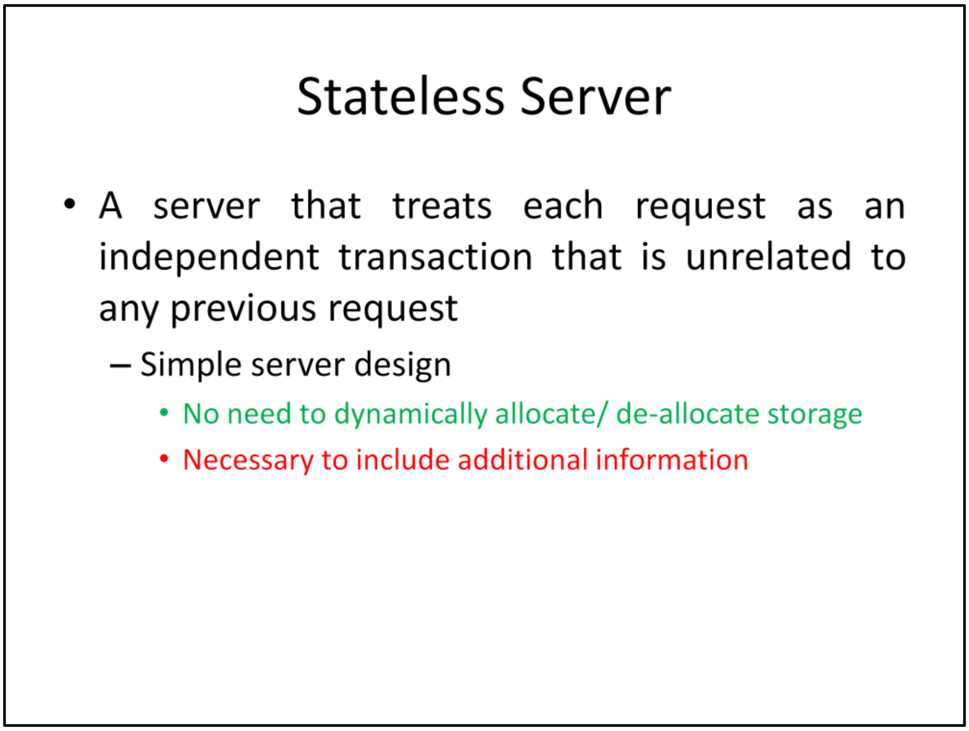
Automatic error detection: You could save hours of work when Visual Studio detects and reports an error before you run your application. Potential problems are underlined, just like the "spell-as-you-go" feature found in many word processors.

Debugging tools: Visual Studio retains its legendary debugging tools, which allow you to watch your code in action and track the contents of variables. And you can test web applications just as easily as any other application type, because Visual Studio has a built-in web server that works just for debugging.

IntelliSense: Visual Studio provides statement completion for recognized objects and automatically lists information such as function parameters in helpful tooltips.

Lets start with Web

https://www.google.com/imgres?imgurl=https%3A%2F%2Fupload.wikimedia.org%2Fwikipedia%2Fcommons%2F2%2F24%2FA_classic_circular_form_spider%2527s_web.jpg&imgrefurl=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FSpider_web&docid=qi4AUIxDSu9t0M&tbnid=dEe42IcZEeJbZM%3A&vet=10ahUKEwiCuJC6xqnjAhWC73MBHTc8B0wQMwjkAShZMFk..i&w=3088&h=2056&bih=657&biw=1366&q=web&ved=0ahUKEwiCuJC6xqnjAhWC73MBHTc8B0wQMwjkAShZMFk&iact=mrc&uact=8



Stateless Server

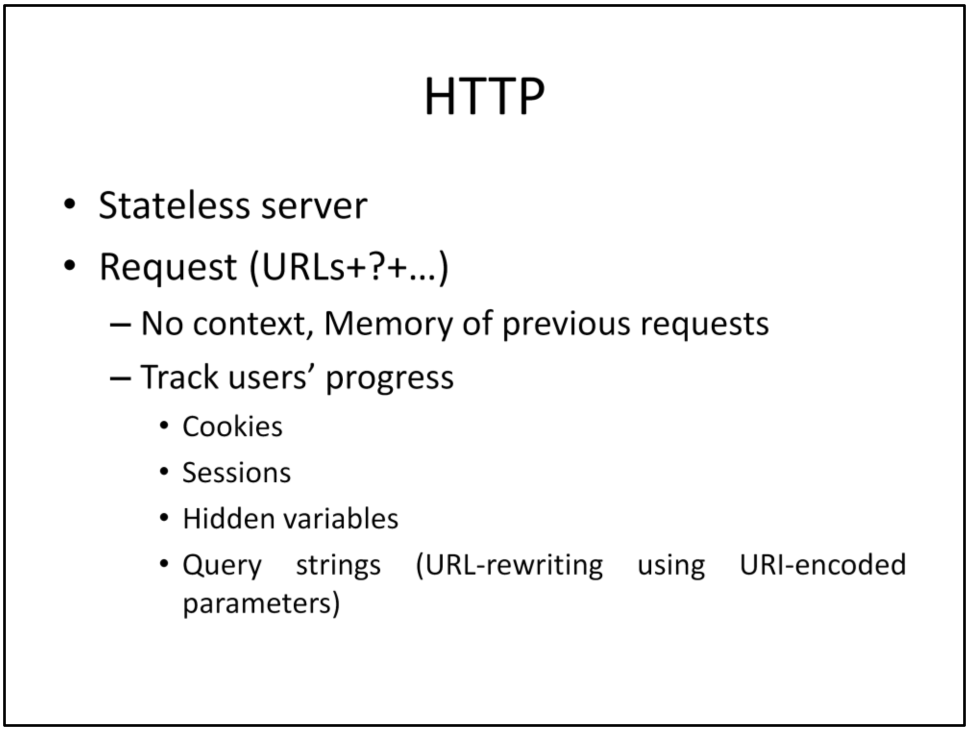
- A server that treats each request as an independent transaction that is unrelated to any previous request
  - Simple server design
    - No need to dynamically allocate/ de-allocate storage
    - Necessary to include additional information

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

Advantages and Disadvantages

The stateless design simplifies the server design because **there is no need to dynamically allocate storage** to deal with conversations in progress. If a client dies in mid-transaction, no part of the system needs to be responsible for cleaning the present state of the server.

A disadvantage is that it may be necessary to include additional information in every request and this extra information will need to be interpreted by the server.



HTTP

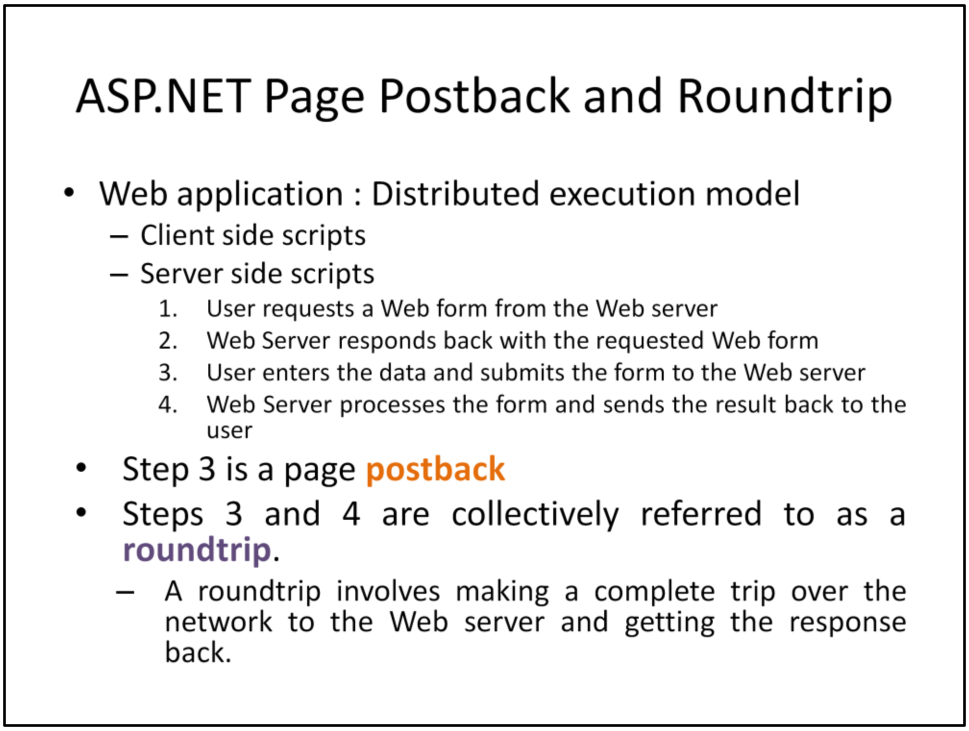
- Stateless server
- Request (URLs+?+…)
  - No context, Memory of previous requests
  - Track users' progress
    - Cookies
    - Sessions
    - Hidden variables
    - Query strings (URL-rewriting using URI-encoded parameters)

An example of a **stateless server** is an **HTTP** server. These take in requests in the form of URLs (which may be accompanied by client-side state-data such as **cookies**) which **completely specify the required document** and **do not require any context or memory of previous requests.**

HTTP Protocol stands for Hyper Text Transfer Protocol is also appliction layer protocol for the WWW. It is the protocol used to convey information of World Wide Web (WWW).HTTP protocol is a stateless and connectionless protocol. HTTP is called as a stateless protocol because each command is request is executed independently, without any knowledge of the requests that were executed before it. It is the protocol used for the web. It is based on a request/response paradigm. In this protocol the communication generally takes place over a TCP/IP protocol.

A stateless protocol does not require the server to retain information or status about each user for the duration of multiple requests.
But some web applications may have to track the user's progress from page to page, for example when a web server is required to customize the content of a web page for a user. Solutions for these cases include:
- the use of HTTP cookies.
- server side sessions,
- hidden variables (when the current page contains a form), and
- URL-rewriting using URI-encoded parameters, e.g., /index.php?session_id=some_unique_session_code.

What makes the protocol stateless is that the server is not required to track state over multiple requests, not that it cannot do so if it wants to. This simplifies the contract between client and server, and in many cases minimizes the amount of data that needs to be transferred.and the Internet is a stateless development environment" is often used. This simply means that the HTTP that is the backbone of the Web is unable to retain a memory of the identity of each client that connects to a Web site and therefore treats each request for a Web page as a unique and independent connection, with no relationship whatsoever to the connections that preceded it.

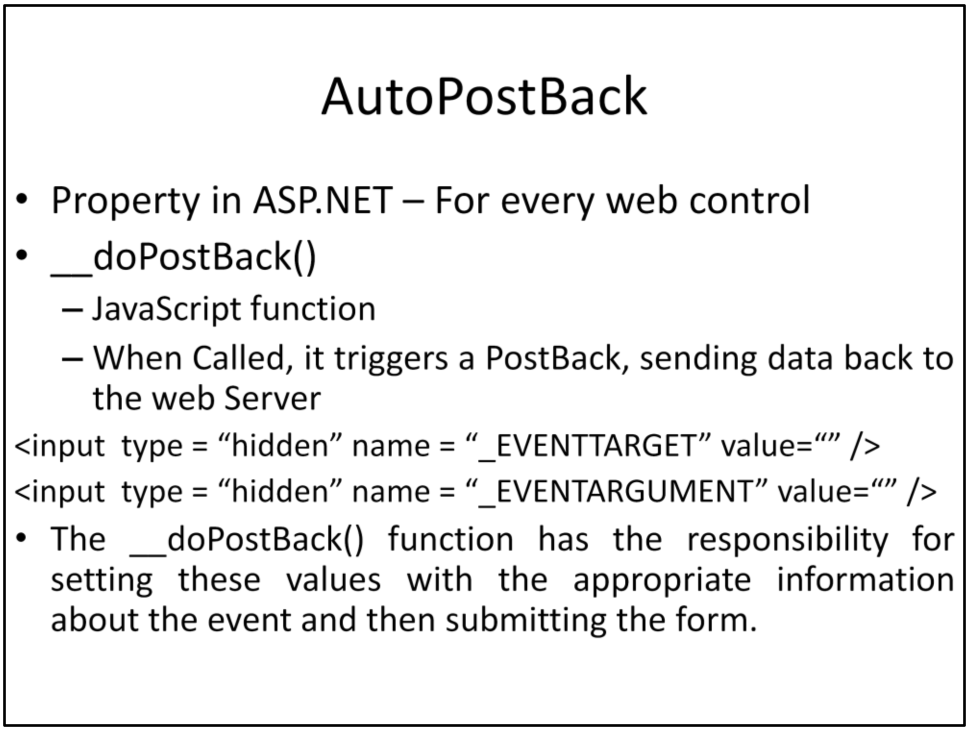


ASP.NET Page Postback and Roundtrip

- Web application : Distributed execution model
  - Client side scripts
  - Server side scripts
    1. User requests a Web form from the Web server
    2. Web Server responds back with the requested Web form
    3. User enters the data and submits the form to the Web server
    4. Web Server processes the form and sends the result back to the user
- Step 3 is a page **postback**
- Steps 3 and 4 are collectively referred to as a **roundtrip**.
  - A roundtrip involves making a complete trip over the network to the Web server and getting the response back.

Roundtrip and Postback
Web applications have a distributed execution model. When a user interacts with a Web form, the browser might respond to some of the user actions by executing client-side scripts while some other actions that require server resources must be sent to the Web server for processing. When server-side processing is involved, a typical interactive user session with a Web form consists of the following steps:
User requests a Web form from the Web server.
Web Server responds back with the requested Web form.
User enters the data and submits the form to the Web server.
Web Server processes the form and sends the result back to the user.
Step 3 is also referred to as a page postback, whereas steps 3 and 4 are collectively referred to as a roundtrip. A roundtrip involves making a complete trip over the network to the Web server and getting the response back.
However, ASP.NET makes this process so transparent that your code can treat your web page like a continuously running program that fires events.

PostBack is the name given to the process of submitting an ASP.NET page to the server for processing. PostBack is done if certain credentials of the page are to be checked against some sources (such as verification of username and password using database). This is something that a client machine is not able to accomplish and thus these details have to be 'posted back' to the server.



AutoPostBack

- Property in ASP.NET – For every web control
- __doPostBack()
  - JavaScript function
  - When Called, it triggers a PostBack, sending data back to the web Server

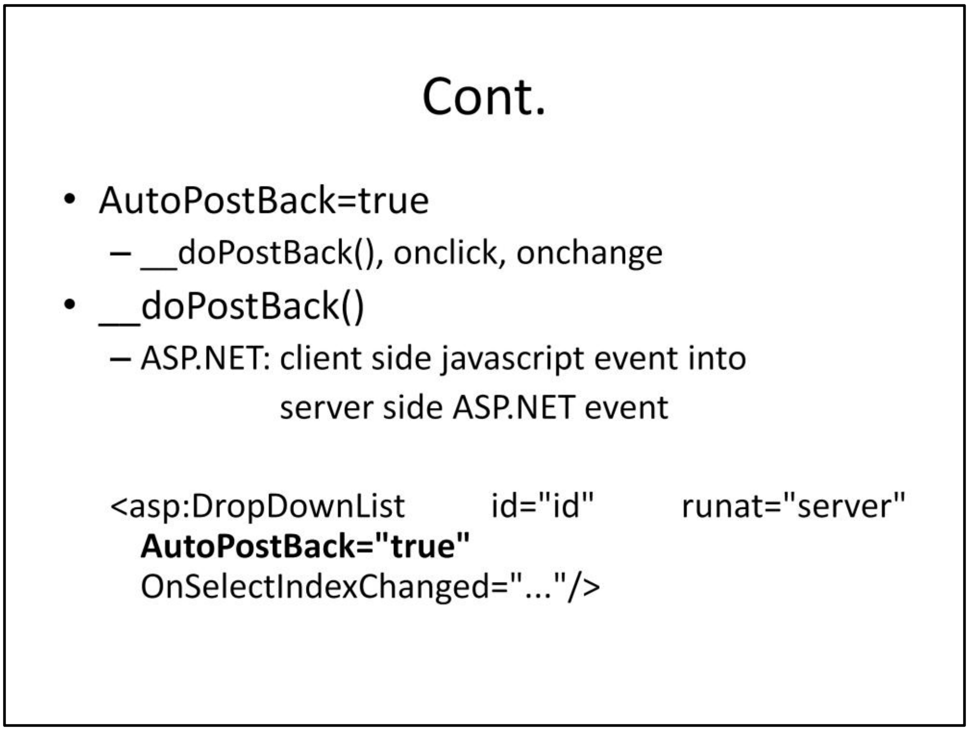
<input type = “hidden” name = “_EVENTTARGET” value=“” />

<input type = “hidden” name = “_EVENTARGUMENT” value=“” />

- The __doPostBack() function has the responsibility for setting these values with the appropriate information about the event and then submitting the form.

If we create a web Page, which consists of one or more Web Controls that are configured to use AutoPostBack (Every Web controls will have their own AutoPostBack property), the ASP.Net adds a special JavaScipt function to the rendered HTML Page. This function is named _doPostBack() . When Called, it triggers a PostBack, sending data back to the web Server.

ASP.NET also adds two additional hidden input fields that are used to pass information back to the server. This information consists of ID of the Control that raised the event and any additional information if needed. These fields will empty initially as shown below,



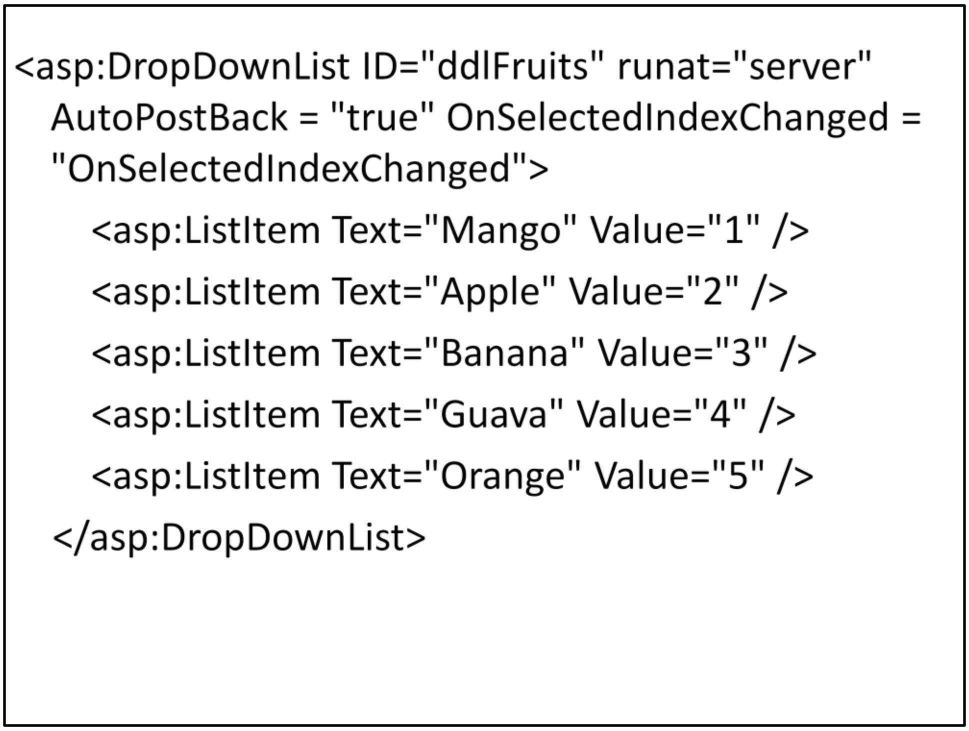
Cont.

- AutoPostBack=true
  - __doPostBack(), onclick, onchange
- __doPostBack()
  - ASP.NET: client side javascript event into
    server side ASP.NET event

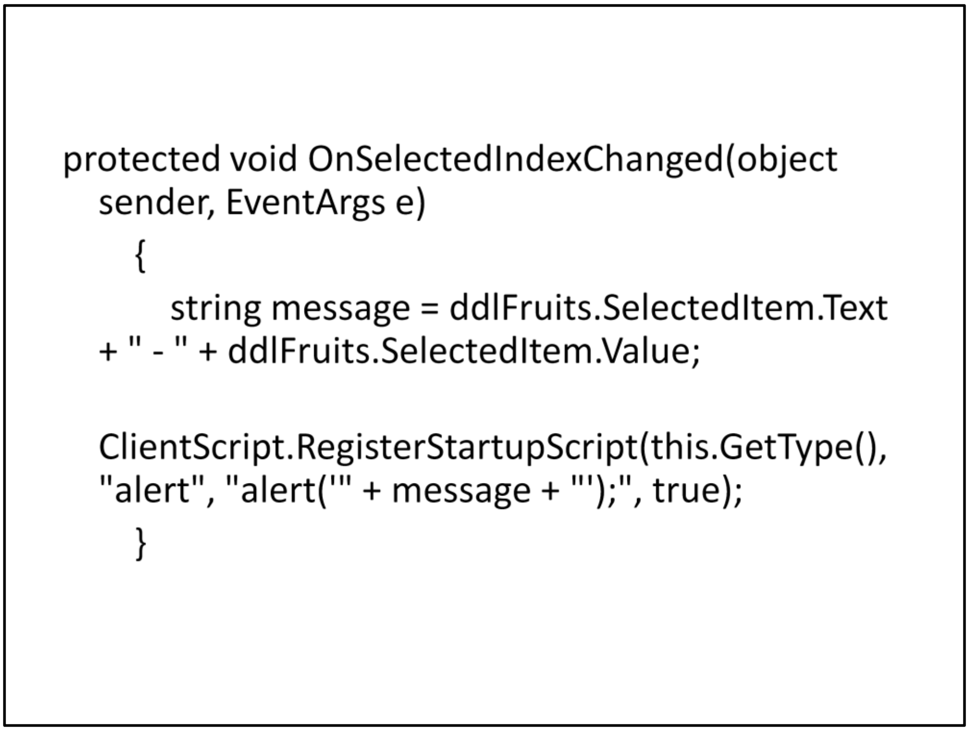
```
<asp:DropDownList       id="id"       runat="server"
  AutoPostBack="true"
  OnSelectIndexChanged="..."/>
```

Any Control that has its AutoPostBack Property set to true is connected to the _doPostBack() function using the onclick or onchange attributes. These attributes indicate what action should Browser take in response to the Client-Side javascript events onclick and onchange.

This ddl do not need asp:button for example in order to post, when you change ddl is autoposted.



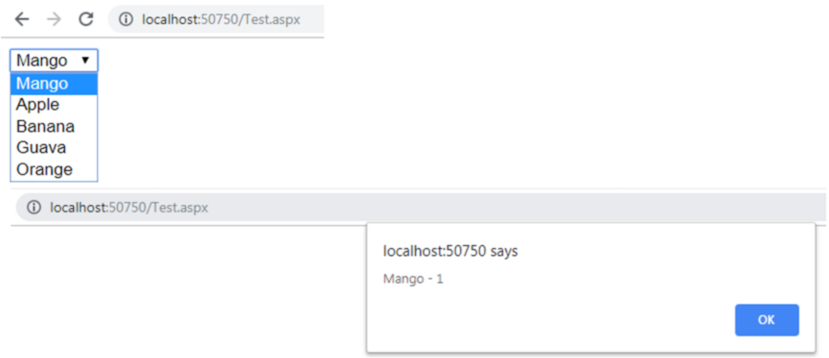
```
<asp:DropDownList ID="ddlFruits" runat="server"
   AutoPostBack = "true" OnSelectedIndexChanged =
   "OnSelectedIndexChanged">
      <asp:ListItem Text="Mango" Value="1" />
      <asp:ListItem Text="Apple" Value="2" />
      <asp:ListItem Text="Banana" Value="3" />
      <asp:ListItem Text="Guava" Value="4" />
      <asp:ListItem Text="Orange" Value="5" />
   </asp:DropDownList>
```



```
protected void OnSelectedIndexChanged(object
   sender, EventArgs e)
      {
          string message = ddlFruits.SelectedItem.Text
   + " - " + ddlFruits.SelectedItem.Value;

   ClientScript.RegisterStartupScript(this.GetType(),
   "alert", "alert('" + message + "');", true);
      }
```

https://www.youtube.com/watch?v=JzZOAZxhC1w
D:\Personal\WDDN\C# Codes 2019-20\AutoPostBack
RegisterStartupScript method places the javascript at the bottom of the page just before closing </form> element



localhost:50750 says

Mango - 1