



Copyright Notice

1

This presentation is intended to be used only by the participants who attended the training session conducted by Prakash Badhe. This presentation is shared for education purpose only.

Sharing/selling of this presentation in any form is NOT permitted.

Others found using this presentation or violation of above terms is considered as legal offence.

Internet applications with Java-EE Servlets

Prakash Badhe

prakash.badhe@vishwasoft.in



Oracle



About Me : Prakash Badhe

3

Web Technologies Consultant-Trainer for 19+ years.

Passion for technologies and frameworks.

Specialized in Java Technologies.

SCJP and Oracle Certified Trainer.

Trainer-Consultant for Oracle University.

Skills in Web2.0 Technology Frameworks.

J2ee web application developer.

Proficient with Html5,Ajax,XML standards.

Working with java and JavaScript frameworks.



About you..

4

- ✓ **Job Role**
- ✓ **Skill-Sets**
- ✓ **Objective**
- ✓ **Java development experience**
- ✓ **Experience with J2ee applications**



The Internet - WWW

5

- The internet is a large, large network of computer systems across the world.
- The internet contains different types of hardware, processors, operating systems, different types of networks and different applications written in different programming languages.
- The internet works on client-server architecture.
- The protocol used for internet communication is **http**.
- The internet is also called as World Wide Web.
- The group world wide web consortium [w3c.org] takes care of internet standards and implementations.

Internet Terminologies

6

- **Network:** group of computer systems connected together e.g. Internet. LAN etc.
- **Server:** is the program which provides the data and information to other requesting programs.e.g.web server like Apache Tomcat, Microsoft IIS.
- **Client :** is the program which requests the information/data from the server.e.g.web browser like MS Internet Explorer, Netscape Navigator, Firefox etc.
- **Protocol:** Set of standard rules, procedures.e.g. tcp/ip.
- **Address :**Unique number used to identify a particular system on the network. e.g. IP Address on internet 192.168.103.12
- **Domain Name System :**Instead of remembering ip address of every server on the internet, domain name System provides unique names which will be mapped to ip addresses by the DNS Servers on the internet.e.g. yahoo.com, google.com etc.
- **Port No:** A unique integer number to identify the program running on a particular computer system. Range of values is from 0 to 65535.The standard port no for internet servers is 80.



Web server : The information provider ⁷

- Web server is the application program waiting for request from web clients (web browsers from users) on a standard port i.e. 80.
- Web server reads the web client request and processes the request and decides the response to be sent to the web client program on user side.
- Web server provides network management, user management, file and resource management to applications running on its side.
- Different web servers such as Microsoft IIS web server, Netscape server, Tomcat web server, apache web server are available.



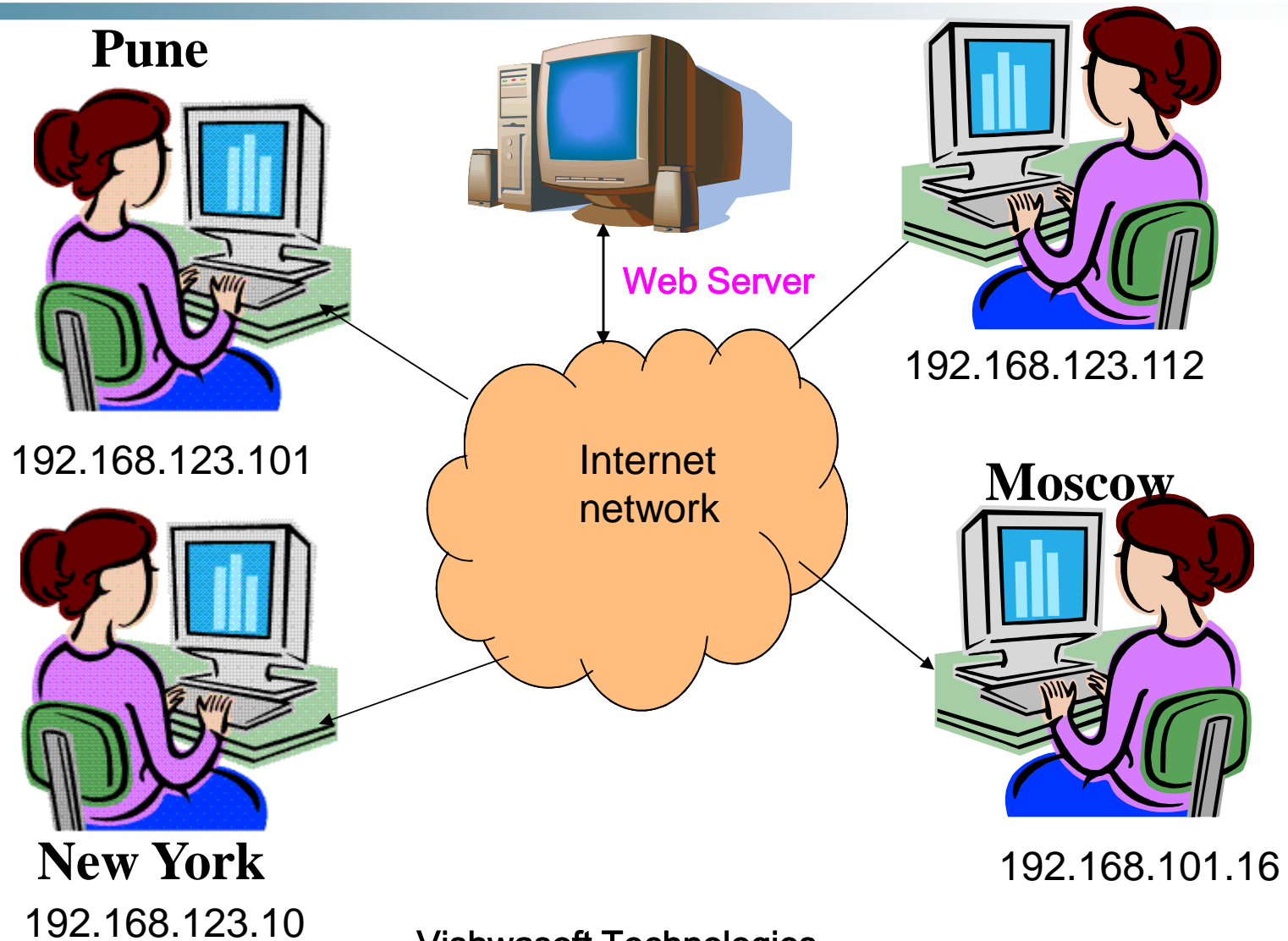
Browser : the Web Client

8

- The browser is defined to connect to the Internet web server on standard port (80) and request information from the web server in form of documents/files/resources.
- The web server sends the requested contents to browser as part of response.
- The browser understands the contents and displays them as graphics in its window.
- Prominent browsers available are Microsoft Internet Explorer, Netscape Navigator, Firefox, Opera etc.

The Internet View...

9



Vishwasoft Technologies

Uniform Resource Locator

10

- Identifying information documents on the Internet:
 - ▶ The (unique) Identifier of a document is the URL (Uniform Resource Locator).
 - ▶ This is the information we usually enter in the browser's Address Bar when we want to view a specific Web page.
- The DNS resolves the name to find IP address of web server.

<http://www.microsoft.com:80/home.html>

protocol	Host name	Port No.	Document
	(translates to		/file name
	IP address)		

http: // 192.168.46.34 :80 / home.html

Vishwasoft Technologies



Internet architecture

11

- The internet has client-server architecture.
- Web Server is running and listening (waiting) for client requests on a particular port number (default =80) and ip address computer.
- The client program, the web browser sends request to the web server by specifying the protocol, ip address, port number, and the requested resource name.
- The web server reads the client request, checks for the requested information and if available and satisfactory sends the contents to from the resource file to web client as response.
- So it is a request-response mechanism for communication.



Web Server and web client

12

- *The web server provides the information service on the web and web client requests the web service and internet is the medium used for connections and contents. Standard web servers such as MS IIS, Apache web server ,Netscape servers are available and web clients such as MS Internet Explorer and Netscape Navigator are available.*
- *The web server and clients use standard set of rules(i.e. **Protocol**) for connection management ,addressing and content dispatching so that any client can connect to any web server over any medium and get the contents retrieved from the web server. The language used for contents is the **HTML** and the protocol is **http**.*



Internet Rules : Http Protocol

13

- The protocol used for communication between client and server is *http* i.e. hyper text transfer protocol which is state less.
- The http client i.e. browser sends the http request to the web container/server
- The http client gets the http response from the web container in the form of html.
- The http client gets the requested resource (html page contents/image etc.) delivered by the web container/web application.
- CGI standard defines the client server interactions on the web.



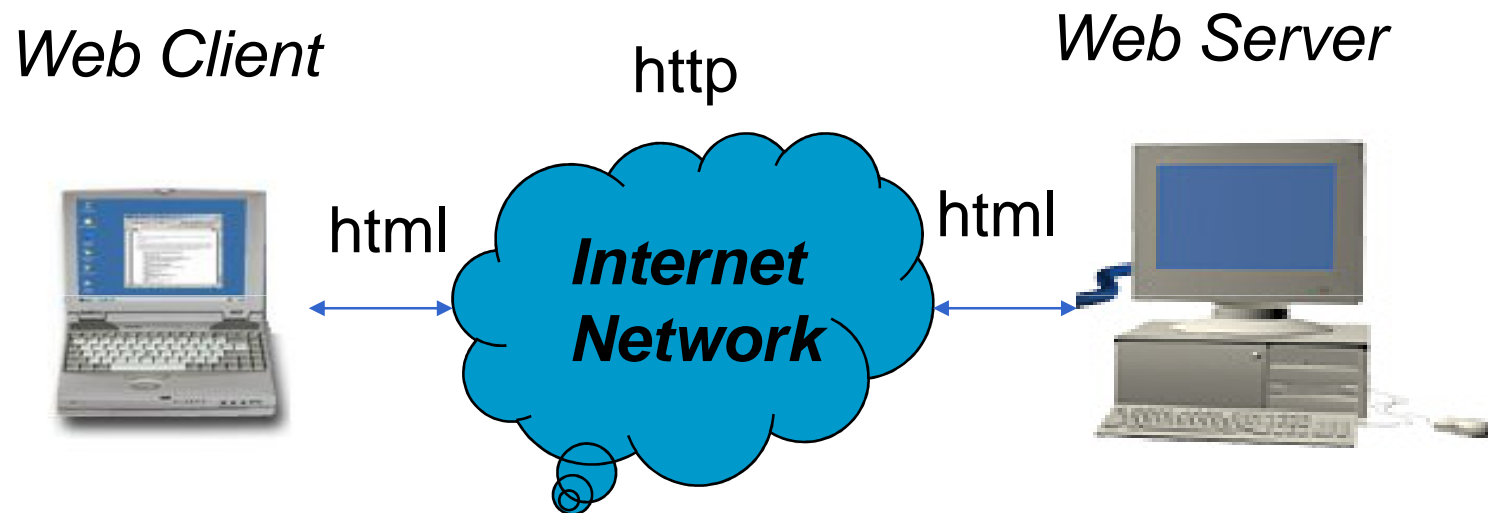
Http Communication

14

- The protocol used for communication between client and server is *http* i.e. hyper text transfer protocol which is state less.
- The http client i.e. browser sends the http request to the web container/server
- The http client gets the http response from the web container in the form of html.
- The http client gets the requested resource (html page contents/image etc.) delivered by the web container/web application.
- CGI and http standards defines the client server interactions on the web and is followed by every web server and web client on the internet.

Internet Communication

15



URL :

<protocol>:<Host name : Port Number>/<Resource Name>

<http://www.microsoft.com/msdn/index.html>



Web Server – Web Browser Information Transfer

16

- The 'http' protocol defines standard rules of communication on internet.
- The language of communication between web browser and web server is 'html'.
- The web browser sends a request to web server for a certain document page or resource and web server responds by sending the html contents from the resource/page to web browser.
- The web browser understands the html contents and displays the corresponding graphics in the window to the user.



What is HTML ?

17

- In an exponentially growing medium such as the Internet, it is literally impossible to count the number of Web pages that exist – however, there is no doubt about it - there are zillions!
Almost all of them are built from a simple, yet powerful language, that enables any user on the planet to view the identical content from his or her own personal computer at home.
- We will discuss the basic elements needed to construct a Web page, learn what these elements actually do, as well as discover how to customize each one to the requirements of our specific page.



Beginning html...

18

Some Basic Facts About HTML

- **HTML** stands for '*Hyper Text Markup Language*'.
- It is not a programming language. Therefore, it doesn't contain functions, variables etc.
- HTML contains **graphic data** that is to be displayed, as well as information informing us **how** to display it (Meta-Data).
- It can include text, images, audio, video, applications (applets, server-side code), and other additional gadgets (Flash, VRML, Shockwave, IPIX etc.).
- Last but not least – HTML can contain *Hyperlinks*, which can redirect the user to other resources available on the net.



Simple HTML page

19

```
<html>
<head>
  <title> This is my Beginning.</title>
</head>
<body>
  Welcome to my First Page!
  <br/>
  Here you will find lot of html tags...
</body>
</html>
```

Simple.html

Vishwasoft Technologies



Getting to Work!

20

- An HTML file is a plain **TEXT** file, which usually has a **.htm** or **.html** (better yet) extension. As such, it can be created using a simple word processor or text editor (such as, Notepad).
- The file can contain data that is to be displayed, as well as information used to change its appearance, behavior or role (**meta-data**).
- The **meta-data** is created by keywords which are called **tags**. **Tags** are enclosed within angular brackets **< >**, and are generally paired with corresponding closing tags, which are similar, but include the addition of a slash (**/**).
- When you use a closing **tag**, the text bracketed within the opening and closing tags will be affected by their intent:
 - ▶ Writing “** Hello! **” will result in the “**Hello!**” being affected by the function of the **** tag (i.e., it will appear in **bold** text).



HTML Document Structure

21

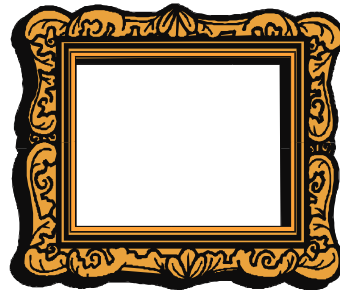
- Every HTML page begins with an **<HTML>** tag and ends with an **</HTML>** closing tag.
- The area within these tags is divided into two sections:
 - ▶ **<HEAD>**, **</HEAD>** - Bracket the HEAD section, which contains mostly hidden meta-data regarding the document (such as, author, keywords, descriptions and code resources).
 - ▶ The second section can be one of the following:
 - **<BODY>**, **</BODY>** - Bracket the BODY section, which contains the document content (text, images, etc.).
 - **<FRAMESET>**, **</FRAMESET>** - Divides the page into several display areas. This feature will be discussed later on.

HTML Schematic Structures

22

```
<HTML>
  <HEAD>
    Here is your head section.
  </HEAD>
  <BODY>
    And here is your page content.
  </BODY>
</HTML>
```

A single page.



Several pages within frames.

```
<HTML>
  <HEAD>
    Here is your head section.
  </HEAD>
  <FRAMESET>
    And here are all of your frame
    definitions
  </FRAMESET>
</HTML>
```



HTML Commands/tags

23

- Each HTML command/tag has its own **attributes** (properties), which are written within the opening tag and are initialized with a proper value.
 - ▶ For example: ** Hello! **, would change the color of “Hello!” to red, since we are dealing with the **color attribute** of the FONT tag (which effects the text bracketed within it with the font properties).
- HTML is **not case-sensitive**, therefore , and are all equivalent.

The HEAD Section

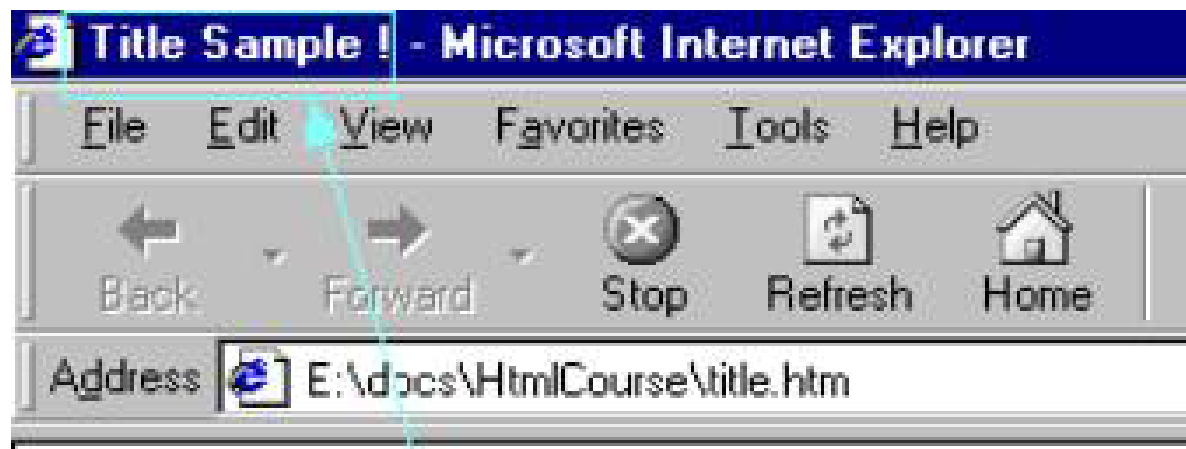
24

- **<TITLE>** - Defines the title of an HTML document and is the most frequently used tag within the HEAD section.

- It is only valid in the head section.

For example: **<TITLE> Title Sample! </TITLE>**

The browser will display the title, “Title Sample!” at the top of the window, also known as the title bar:



The BODY Tag Attributes

25

- **TEXT** – Specifies the default color of the document's text.
`<BODY TEXT="GREEN">`
- **BACKGROUND** – Specifies an image file (URL), that is used as a backdrop for the contents of the page.
`<BODY BACKGROUND="logo.gif">`
- **BGCOLOR** – Specifies a fill color for the entire document. When combining BGCOLOR and BACKGROUND, any transparent areas of the background image allow the background color to show through.
- We can merge BGCOLOR and BACKGROUND
`<BODY BGCOLOR="Blue">`



The Body's Attributes – II

26

- ***TOPMARGIN / BOTTOMMARGIN*** – Specify the quantity of blank space to be left between the start/end of the content and the top/bottom of a page, designated in pixels.
- ***RIGHTMARGIN / LEFTMARGIN*** – Specify the amount of blank space to be left between the right/left edge of the content area of a window and the right/left edge of the page, designated in pixels.

<BODY BOTTOMMARGIN=5>

<BODY LEFTMARGIN=5>

Body's Attributes – III

- The body tag allows for the specification of colors to mark links within a page, using the following three attributes :
 - ▶ **LINK** – Specifies the color of links that have not been visited.
 - ▶ **VLINK** – Specifies the color of visited links.
 - ▶ **ALINK** – Specifies the color of links, which have been activated (have been clicked on) by the user.

For example: `<BODY LINK="BLUE" ALINK="CYAN" VLINK="YELLOW">`

In this case, links will be colored blue. When they are clicked they will change to cyan, from that point on (once they have been clicked), they will be colored yellow.

Text Formatting Headlines

- **<H>** – Represents a group of six different tags, each one stands for a different headline size. H1 is the largest size, while the fonts gradually decrease in size as one moves to the smallest size - H6.

For example:

```
<HTML>
```

```
<BODY>
```

```
    This is normal size text.
```

```
    <H1>This is the largest headline. </H1>
```

```
    <H3>This is the size of headline H3</H3>
```

```
    <H5>While this is H5</H5>
```

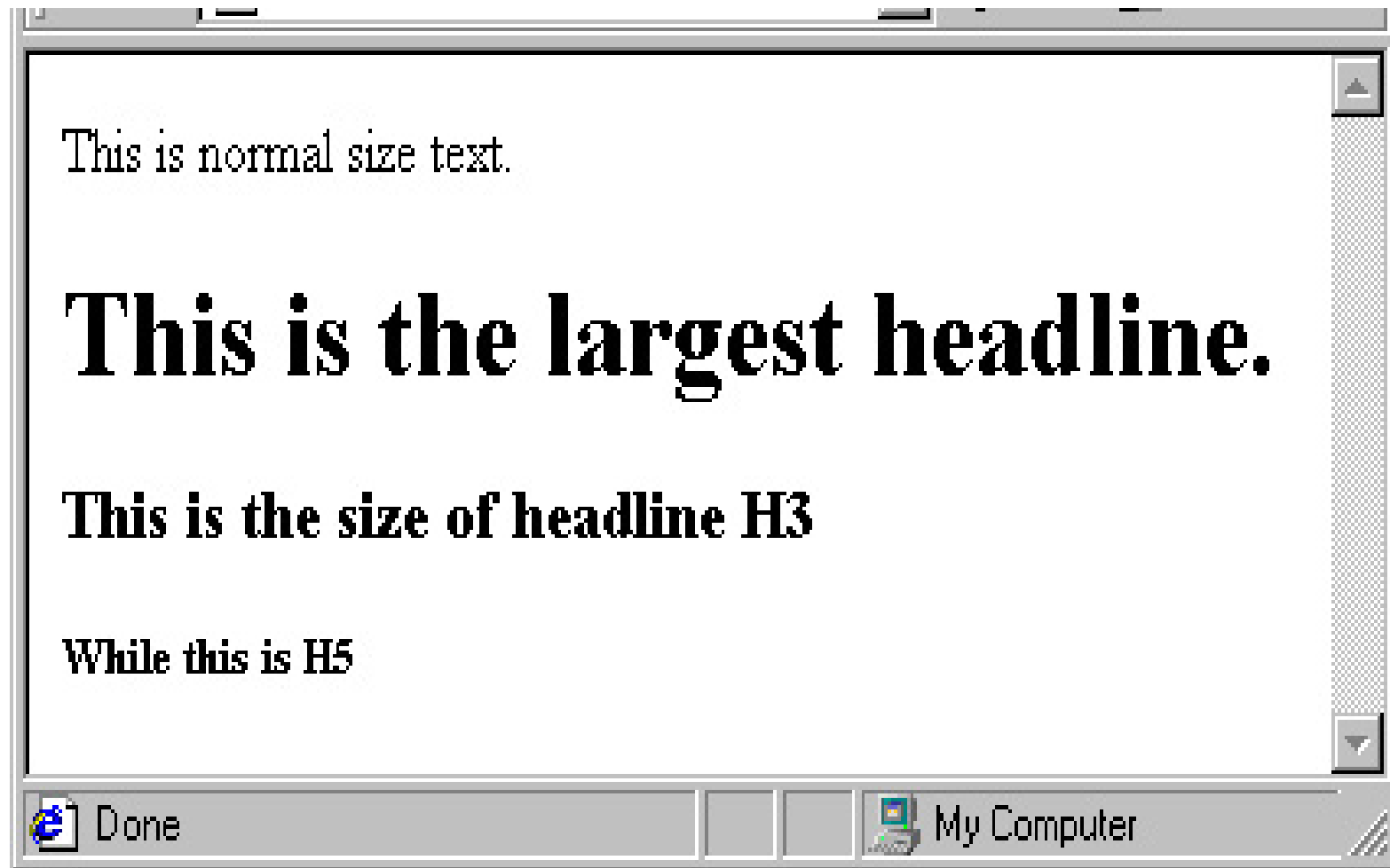
```
</BODY>
```

```
</HTML>
```



Headlines – A Sample

29



Text Flow Tags

- Line breaks in HTML code are interpreted by browsers as spaces. Tags such as `<P>` `<DIV>` and `
` enable us to force new lines.
- `<P>` and `<DIV>` also allow us to control paragraph alignment.
- `<P>` – Begins a new paragraph.
- Main attributes:
 - ▶ **ID** - Labels the element with a unique identifier.
 - ▶ **ALIGN** - Aligns the paragraph content to the left, right or center. The default is left alignment.
- `<DIV>` will be presented later on in this course.



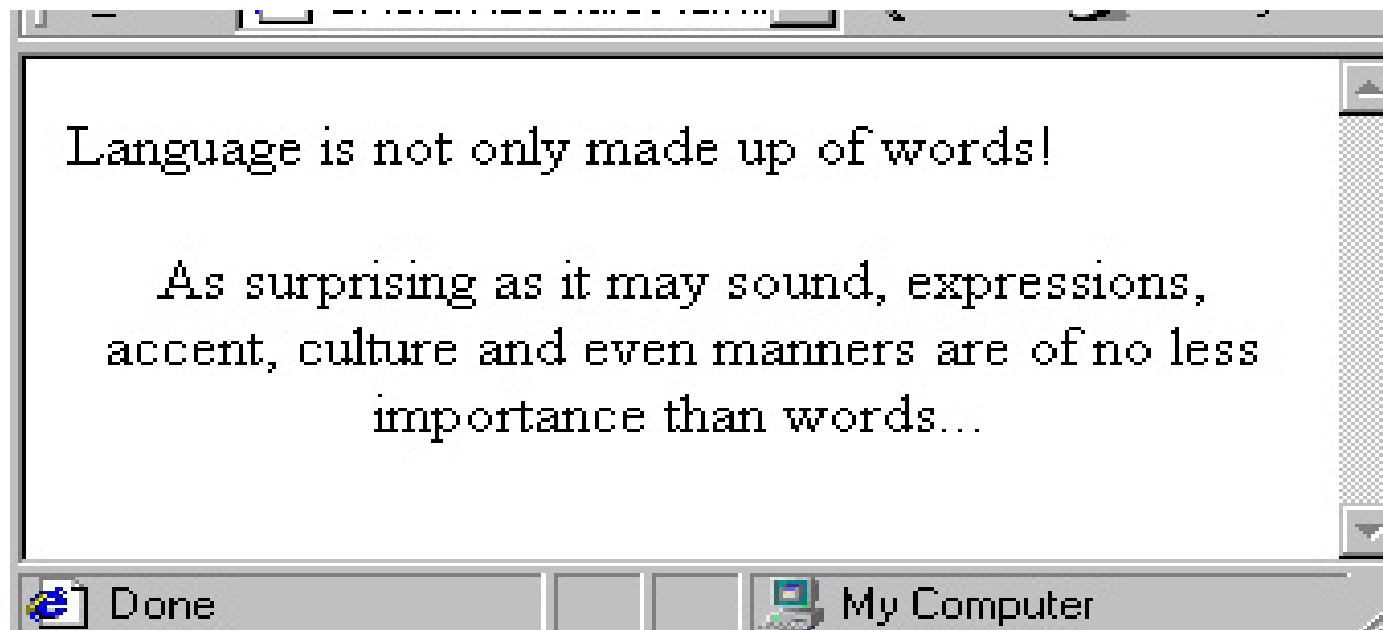
Text Flow – <P> - Example

31

Example:...

`<P ID="LangArticle"> Language is not only made up of words!`

`<P align=center> As surprising as it may sound, expressions, accent, culture and even manners are of no less importance than words...`



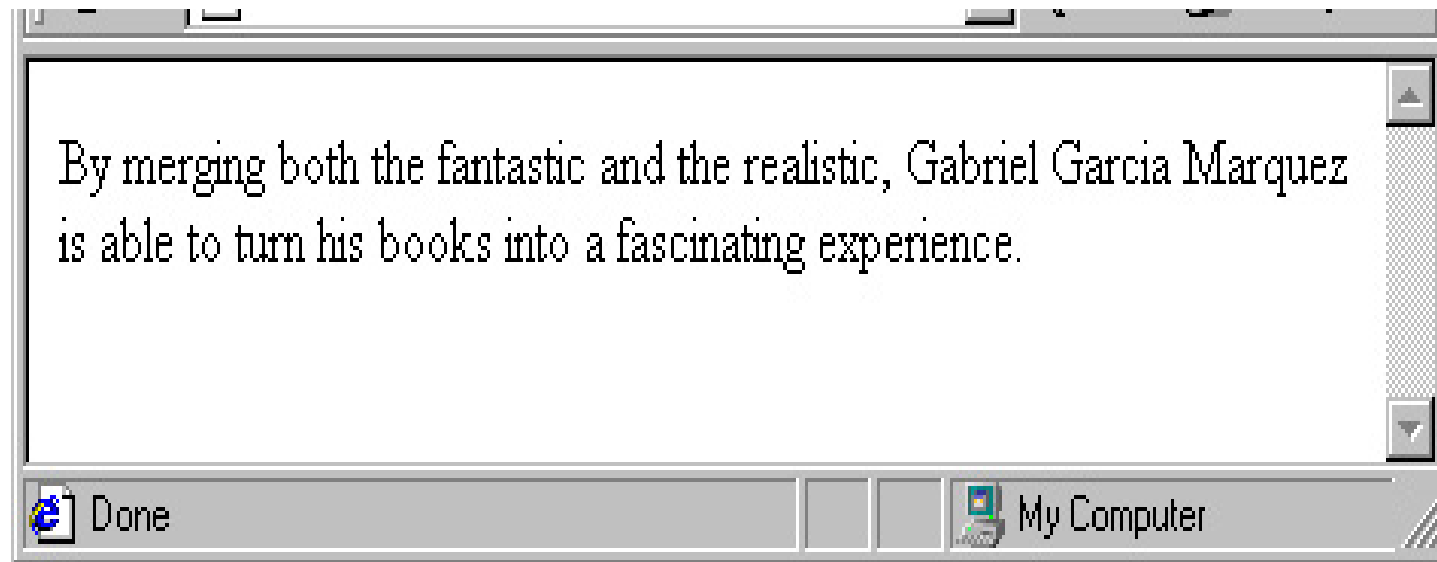
Text Flow –

32

-
 – Breaks the current line of text.

For example:

By merging both the fantastic and the realistic,
Gabriel Garcia Marquez
 is able to turn his
books into a fascinating experience.





Text Flow – `<PRE>`

33

- `<PRE>` – Forces text to be displayed exactly as set by the page author, which means that line breaks occur whenever they appear broken in the code, etc.

The next slide presents an example:

<PRE> - A Sample of Code

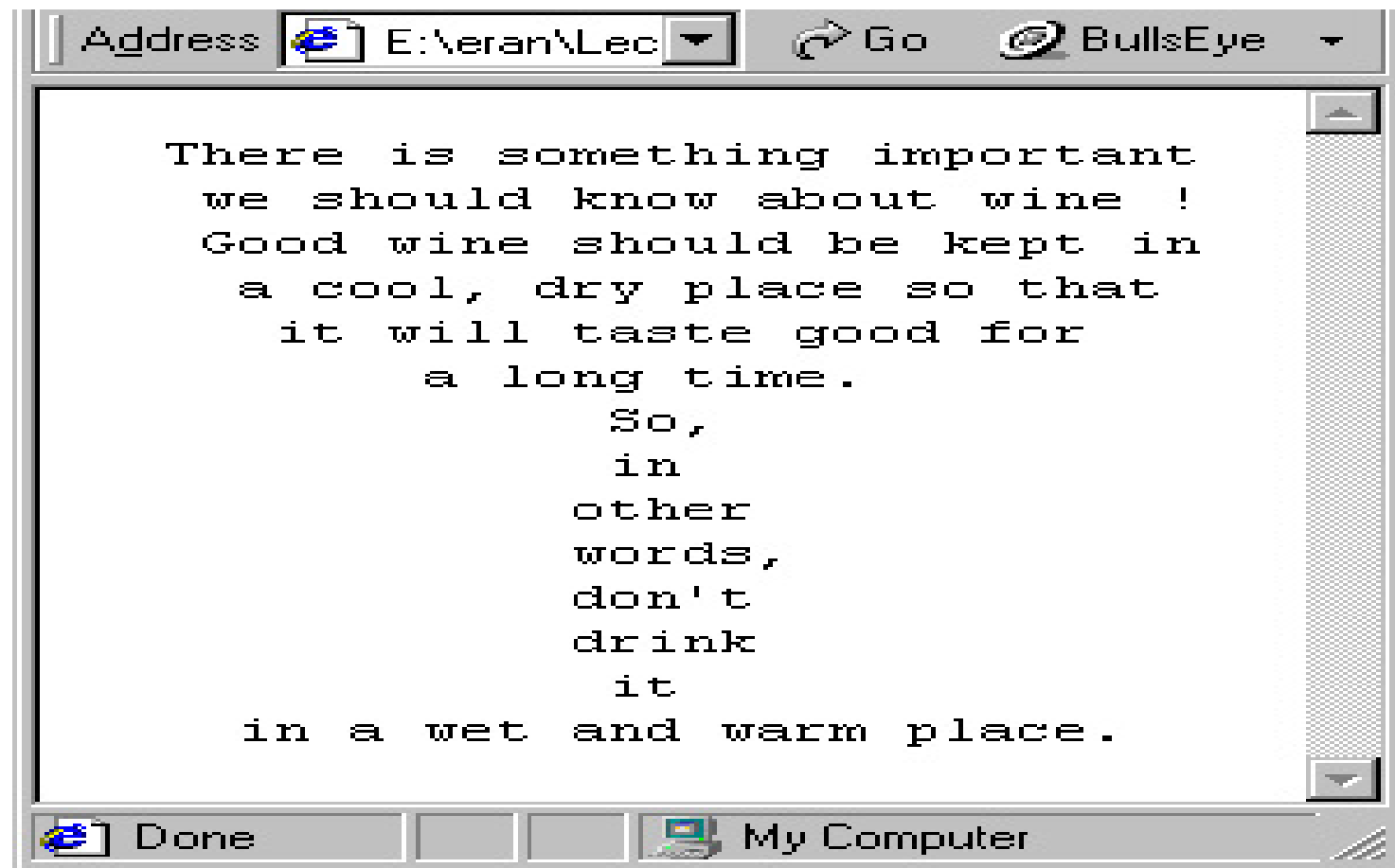
34

```
1. ...
2. <PRE>
3.   There is something important
4.   we should know about wine!
5.   Good wine should be kept in
6.   a cool, dry place so that
7.   it will taste good for
8.       a long time.
9.           So,
10.          in
11.          other
12.          words,
13.          don't
14.          drink
15.          it
16.   in a wet and warm place.
17. </PRE>
```

What is hidden here?

<PRE> - Demo

35



The screenshot shows a web browser window with a single tab titled "Address". The address bar contains the path "E:\veran\Lec" and a dropdown arrow. To the right of the address bar are "Go" and "BullsEye" buttons. The main content area displays a poem in a monospaced font, with lines indented to form a stanza. The status bar at the bottom shows "Done" and "My Computer" icons.

```
There is something important
we should know about wine !
Good wine should be kept in
  a cool, dry place so that
    it will taste good for
      a long time.
        So,
        in
        other
        words,
        don't
        drink
          it
    in a wet and warm place.
```

Text Font Styles:, <I>, <U>

36

- - Designates text that should be displayed in boldface.

For example: what does that have to do with this?.

What does **that** have to do with **this**?

The tag does not force spaces:

For example: you-niversity.com

you-niversity.com

- <I> - Designates italicized text .
- <U> - Underlines the text enclosed within the start and end tags.

For example: This is <I>italicized</I>, while this is <U>underlined</U>.

This is *italicized*, while this is underlined.



Font Styles : tag

37

**** - Changes the font of the text. Main attributes:

- ▶ **SIZE** – Value may be fixed(1,2, ... 7), or relative to size number 3:

(-1,+2...).

`text size is 2`

- ▶ **COLOR** – Value may be one of a set of predefined color names (red, blue...), or a string with the hash sign (#) and a 6 digit hexadecimal RGB (#FF0000, #0000FF).

`blue colored text`

- ▶ **FACE** - Indicates font type such as: Times New Roman, Monospace, serif, Courier, Ariel, etc.

`This text is
monospace`

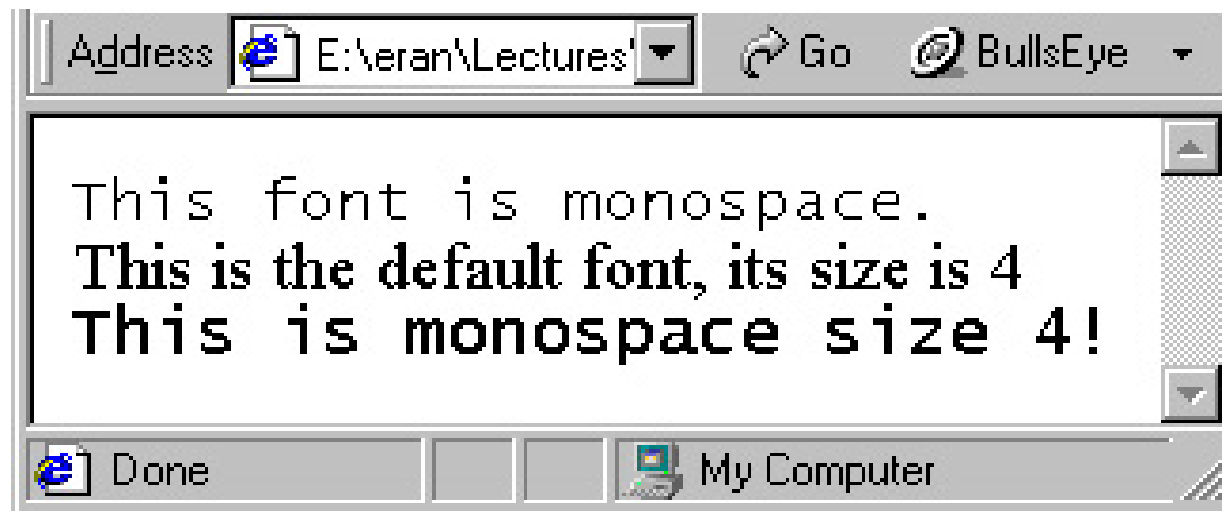
Font Style Changes – Demo

38

```
<FONT FACE="monospace">This font is  
monospace.</FONT><BR>
```

```
<FONT SIZE="+1">This is the default font, its size is  
4</FONT><BR>
```

```
<FONT SIZE="4" FACE="monospace">This is monospace  
size 4!</FONT><BR>
```





Editing Tag– <HR>

39

- **<HR>** - Results in a horizontal line (rule, being drawn across the screen).

- Main attributes:
 - ▶ **SIZE** - Indicates the line's thickness.
May be measured in pixels or percent.
 - ▶ **WIDTH** - Indicates line's length. May be measured in pixels or percent.
 - ▶ **ALIGN** - Indicates the way a line will be aligned within a document. May be: left, right or center.
 - ▶ **NOSHADE** - Turns line into 2D rule instead of 3D rule.

For example:

```
<HR SIZE="3" WIDTH="250" ALIGN="LEFT" NOSHADE>
```

Editing – <HR> - demo

40

```
<H2>The WindMills Of Your Mind </H2>
```

```
<I>Noel Harrison</I>
```

```
<HR ALIGN=LEFT SIZE=4 WIDTH=310>
```





<div> element tag

41

- A block-level element, simply defines a block of content in the page
- Beyond defining a block, it doesn't do anything by itself.
- Because <div> is a block level element, visual browsers (Netscape and MSIE) render <div> elements with a line break before and after them
- A style can be applied to the <div> element

- **Example**

`<div>`

This is after Div

`<p>This is inside para</p>`

`</div>`

 element tag

42

- The tag is used to modify the characteristics of a particular section of text
- The difference between <DIV> and is that is used within a paragraph, sentence or even word, while <DIV> is intended for block-level formatting.
- allows you to dynamically add effects to the included text such as
 - ▶ Highlighting text
 - ▶ Coloring text
 - ▶ Add background image to the text
 - ▶ Add mouseover effects

■ Example

This is the text here...

Symbols

43

- Some characters are problematic for inclusion in an HTML file because:
 - ▶ They are hard to create with a text editor (e.g., ©, ®, etc.).
 - ▶ They function as a component of HTML instructions (e.g., <, >, &).
- HTML supports two types of instructions for representing these characters
 - ▶ ***&charname*** -
 - e.g.: ***<*** for <, ***>*** for >, etc.
 - ▶ ***&#charcode*** -
 - e.g.: ***&*** for &, ***%*** for %, ***<*** for < etc.
- The next slide displays some of these characters.

Character Symbols

44

A partial table of special characters:

Char Code	Char Name	Symbol	Description
"	"	"	Quotation
®	®	®	Registered Trademark
¢	¢	¢	Cent Sign
°	°	°	Degree Sign
¼	¼	¼	Fraction – Quarter
½	½	½	Fraction – Half
¾	¾	¾	Fraction – Three Quarters
 	 		Nonbreaking Space
&	&	&	Ampersand



HTML Links

45

- ✓ Up to this point, the pages we created only contained (formatted) text.
- ✓ As you may remember, HTML stands for **Hypertext** Markup Language.
- ✓ We have seen where the “Markup” component comes from.
- ✓ Let us now move on to “Hypertext” ...



<A> - Link tag

46

- <A> – Is used for two different purposes:
 - ▶ First, it is used to **create a hyperlink** – it defines an area within the document, that when clicked, causes the browser to display (jump to):
 - A new document, or
 - A new location within the current document.
 - ▶ Second, it is used to **create an anchor** – that is, a location within the document to which links can refer.



<A> – Anchors

47

- The use of the **NAME** attribute creates an anchor to provide linkage in another page:

```
<A NAME="HugoAnchor">Hemingway, Hugo...</A>
```



<A> - Link

48

- The use of the HREF attribute creates a link to one of the following:
 - ▶ A specific location on the page, when used with a hash sign '#' followed by an anchor* name.

```
<A HREF="#HugoAnchor">Go To Hugo's</A>
```

When the "Go To Hugo's" text is clicked, the browser will scroll to the location where the anchor containing the "HugoAnchor" name has been recorded (as demonstrated in the previous slide).

<A> - Link – cont'd

49

- ▶ **A new document**, when used together with a URL:

```
Let's go <A HREF="http://www.yahoo.com">to YAHOO</A>.
```

Will result in: **Let's go** [to YAHOO](http://www.yahoo.com).

When the "to YAHOO" is clicked, the current page will be replaced with the home page of Yahoo!'s Web site.

We can also combine the two:

A link may also lead to a specific location within a different document if a hashed anchor name is included in the URL as follows:

```
<A HREF="http://www.books.com/Authors.html#HugoAnchor">  
  Go To Hugo's</A>
```

When such a link is clicked, the page will be loaded and the section that includes that location will be displayed.



Links to NAMES and IDs

50

Links can refer to any tag that contains either the *NAME* or *ID* attribute (not only anchors created with the `<A>` tag).

Example:

In the following code, you will find a link named “**More**”, which takes you to a paragraph within the document through the use of the paragraph’s **ID**:

```
<A HREF="#MovieSite">More</A>
```

And also to the paragraph at the bottom of the page:

```
<P ID="MovieSite">Want to get to a movie site...</P>
```



Link- TARGET Attribute

51

- Another <A> tag attribute is *TARGET*:
- *TARGET* – Determines where the destination document will be displayed. The options are:
 - ▶ **Within the current window or frame** - this is the default;
 - ▶ **In another window** - This is accomplished by doing the following:

Providing the name of the desired window (in the event that no such window exists, the browser will create a new window and name it accordingly).

 - Assigning “_blank” to the attribute - this will result in the creation of a new nameless window.
 - ▶ **In a specified frame** – this will be discussed in more detail within the section on frames.



Target Attribute – Sample

52

Example:

```
<A HREF="http://www.dictionary.com" TARGET="_BLANK">  
Open Dictionary</A>
```

Each click on this link will open a **new** browser window that displays dictionary.com:

Images tag -

53

- – Inserts a graphic image into an HTML document.
- Main Attributes:
 - ▶ **SRC** - The image file's URL. Usually '.jpg' or '.gif' files.
 - ▶ **ALIGN** - Causes the image to be aligned with respect to the text. Options:
 - Top - text aligned with the top of the picture;
 - Middle;
 - Bottom;
 - Left - picture is to the left of the text; or
 - Right.
 - ▶ **WIDTH / HEIGHT** – Indicates width and height. Given in:
 - Number of pixels – for a fixed result or,
 - Percentage – for a dynamic fit.

Image & Text Sample Code

54

1. `<P>`
2. `This is a beautiful guard dog.</P>`

3. `<P>`
4. `This is another great guard dog.</P>`

5. `<P>`
6. ``
7. `The Doberman and the Alsecian`
8. `are two great guard dogs.</P>`

Image & Text Alignment

55

Note the text alignment, based on the ALIGN values.

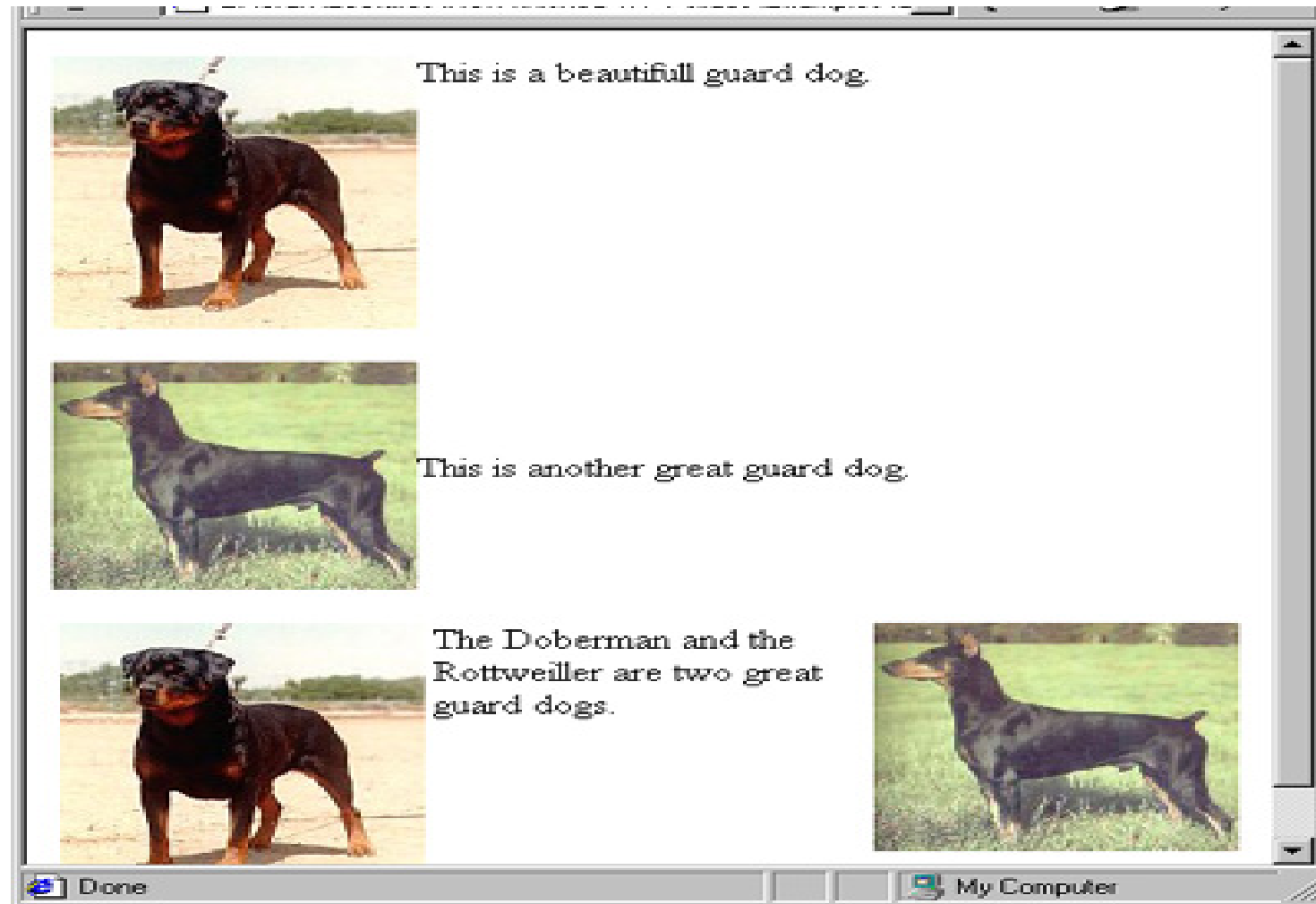


Image Links

56

- Images can also be used as links or bullets in a list.

For example:

```
▶ <A HREF="http://games.yahoo.com/">  
    <IMG SRC="sbr.gif">  
</A>
```

This will create an image that functions as a link to the Yahoo! card/board games site (bridge, blackjack, etc.).



Links & Images - Exercises

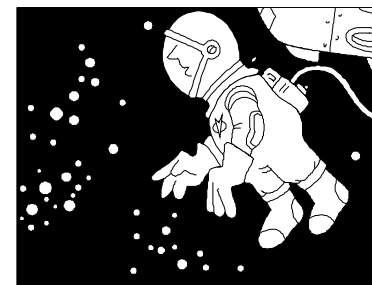
57

1. Image linking

- a. Create a web page that includes three anchors using three different pictures. The maximum width/height of each picture should be 300 pixels.
- b. Add links at the top of your document:
 - To each of your anchors.
 - To a site that enables you to find additional art pictures, which will:
 - Open a new window, or
 - Use the current window.
 - To any paragraph within your document.
- c. Add links within each anchor:
 - To a new window displaying the picture in a larger size (it could even be its original size). This time, the picture itself is to function as a link !

HTML Data Flow

- By default, elements within HTML are tiled (i.e., laid side by side), from left to right, and from top to bottom (just like a typewriter).
 - ▶ Therefore, adding an image after a paragraph of text would display the image to the right of the text (unless we reached the end of the line, in which it would move to the beginning of the next line).
- Spaces are ignored if there is more than one, thus:
 - ▶ **“There is a big gap here!”**
When written as HTML code, this would appear as, **“There is a big gap here!”**
- Nevertheless, one can position an HTML element in an exact spot on a page, by using a variety of methods (such as, CSS).





Exercise

59

1. Create an HTML page called, “catalog.html”, which has the title “Course Exercises”.
2. Add a BODY block and a simple caption saying, “Welcome to Course Exercises for HTML Training”.

This page will be used throughout the entire course, and at the end of each chapter we will add links to this chapter’s exercise.



HTML Forms

60

- Forms were designed to allow the viewer of HTML pages to send information to server applications, such as CGI's and Java Servlets.
- A form defines a collection of input items that can be user-defined values. The form includes a means of submission (usually a submit button), which submits the entire set of values when activated by the user.
- Forms are currently used for three major purposes:
 - ▶ Sending information to server applications.
 - ▶ Sending information by mail to a predefined recipient.
 - ▶ Allowing the containing page to communicate with the user, through the use of event handling and JavaScript. This option will be discussed in the chapter on Dynamic HTML.



<FORM>

61

- The <FORM> tag must have a corresponding </FORM> ending tag.
- The form is constructed from the input elements defined within its tags. All values from these input elements are sent at the time the form is submitted.
- FORM's main attributes are as follows:

<FORM> Attributes: ACTION

62

- ACTION – Its value should be a URL, that determines:
 - ▶ The location to which the collected information will be sent:
`<FORM ACTION="http://you-niversity.com/sela/parrot">`
or
 - ▶ A *mailto:* address to which the information will be sent via mail.
`<FORM ACTION="mailto:user@somewhere.com"
 EncType="text/plain" METHOD=POST>`
- Note: This option is problematic when used with the two major browsers – as a result of security related problems it does not truly work as it should.



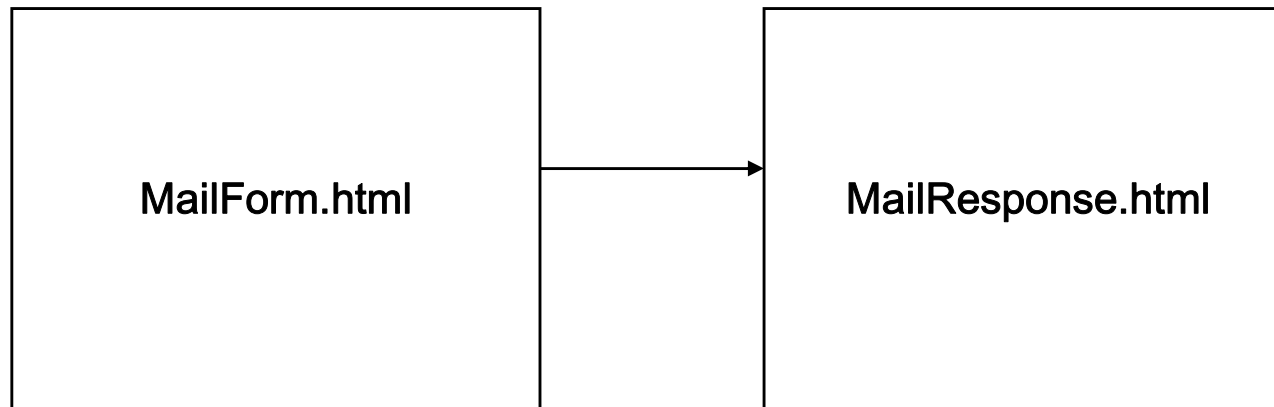
<FORM> Attributes: METHOD

63

- **METHOD** – Determines how the data will be sent to the server application. Possible values include:
 - ▶ **post** – the name value pairs, which correspond to the input items are sent as a text message.
This is the preferable option.
 - ▶ **get** – the values will be appended to the URL specified in the **ACTION** attribute.
- This attribute is not mandatory when using a **mailto:** URL as part of the **ACTION** attribute.

Example form

64



For all the different users sending different id and password they will get the same information from the page MailResponse.html



User needs Dynamic responses

65

- Some kind of dynamic programming is required on server side which will generate different html text responses for different users in different situations.
- This task can be achieved by using a service program with a programming language code to read the request parameters from http request and accordingly generate html responses.
- The programming languages such as Pearl, C, C++, Java, PHP , DotNet etc. can be used for this purpose.
- Each programming language will need its required runtime support from the server to execute the code.
- The applications written using such languages will require some facility for file management, user management network management etc.on the server side.



Server side programming

66

- *The web server applications provide the infrastructure for these service applications such as network connections, file management, client request /response dispatching etc. and **Service** applications running on server use these facilities provided by the server.*
- *The Service applications use different programming languages to generate **dynamic contents** and **responses** to be sent to the client applications some of these are perl, cold fusion, asp etc. Java and PHP.*
- *PHP is the simple and flexible scripting language for server side dynamic applications.*



Java Web Servers

67

- *Java Web Server 2.0 from Sun Microsystems*
- *Apache Tomcat web server*
- *Apache Http server with java engine.*
- *iPlanet web server*
- *Alliare JRUN web server*
- *JBoss Web server*



Java Web components

- **Java Servlets, java server pages and java beans** are the web components deployed in the web container/Web server
- Servlets provide *dynamic* interactive response to the web client
- Java Server Pages i.e. jsp are compiled to servlets at run time.
- Java beans are classes providing user interface components and server side processing embedded inside the reusable components.



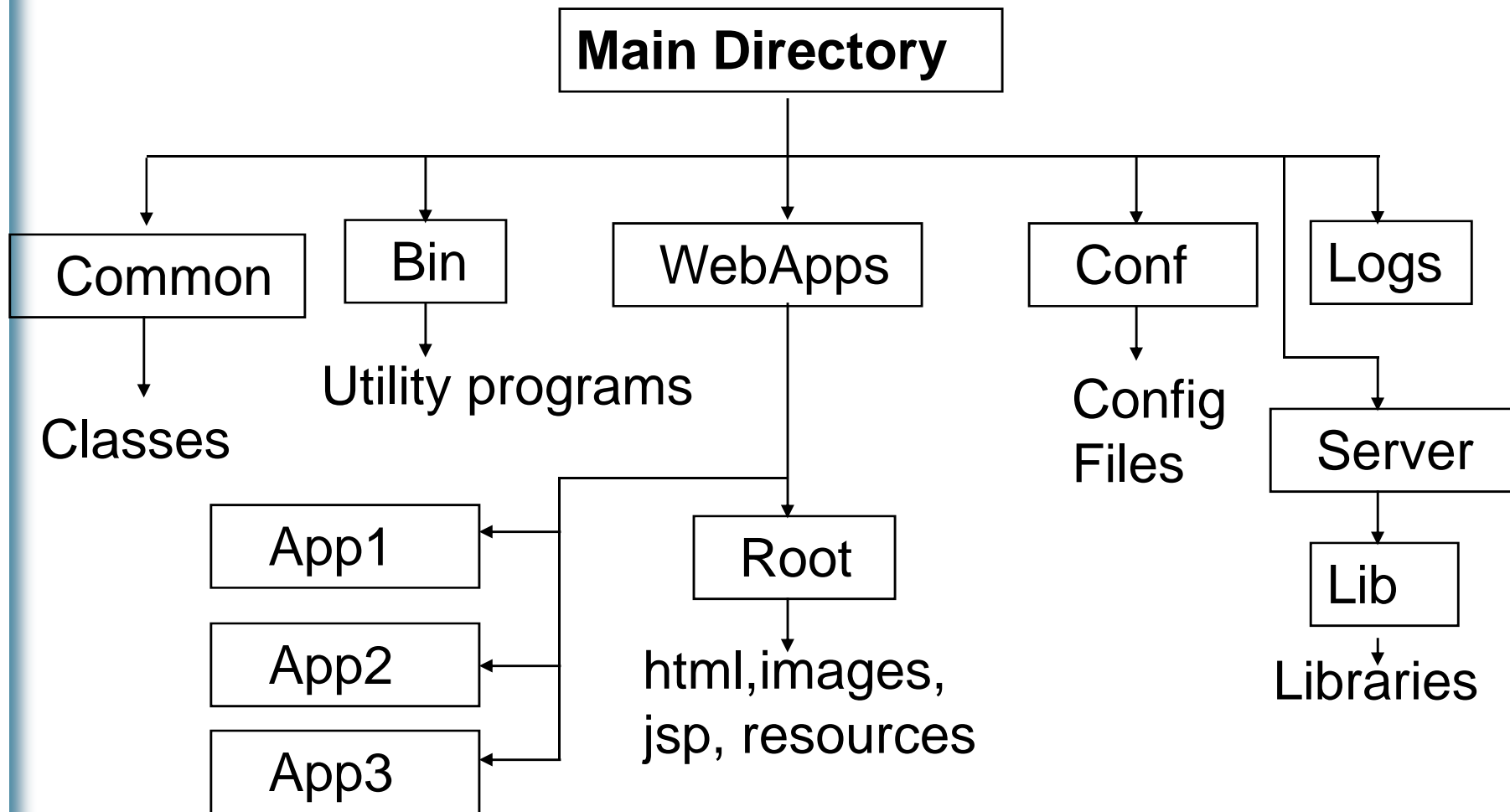
Java Servlets

69

- Java Servlets are distributed components similar to Applets, which are **not** the standalone applications with the public static void main(String args[]) method.
- Applets run into Web browser while Servlets run into the Web server
- Servlet components are the java classes with pre- defined life cycle methods.
- Servlet classes and interfaces are stored into two main packages [javax.servlet](#) and [javax.servlet.http](#).
- Servlets require java runtime support to execute and web servers having servlet engines or embedded java runtime supports them.

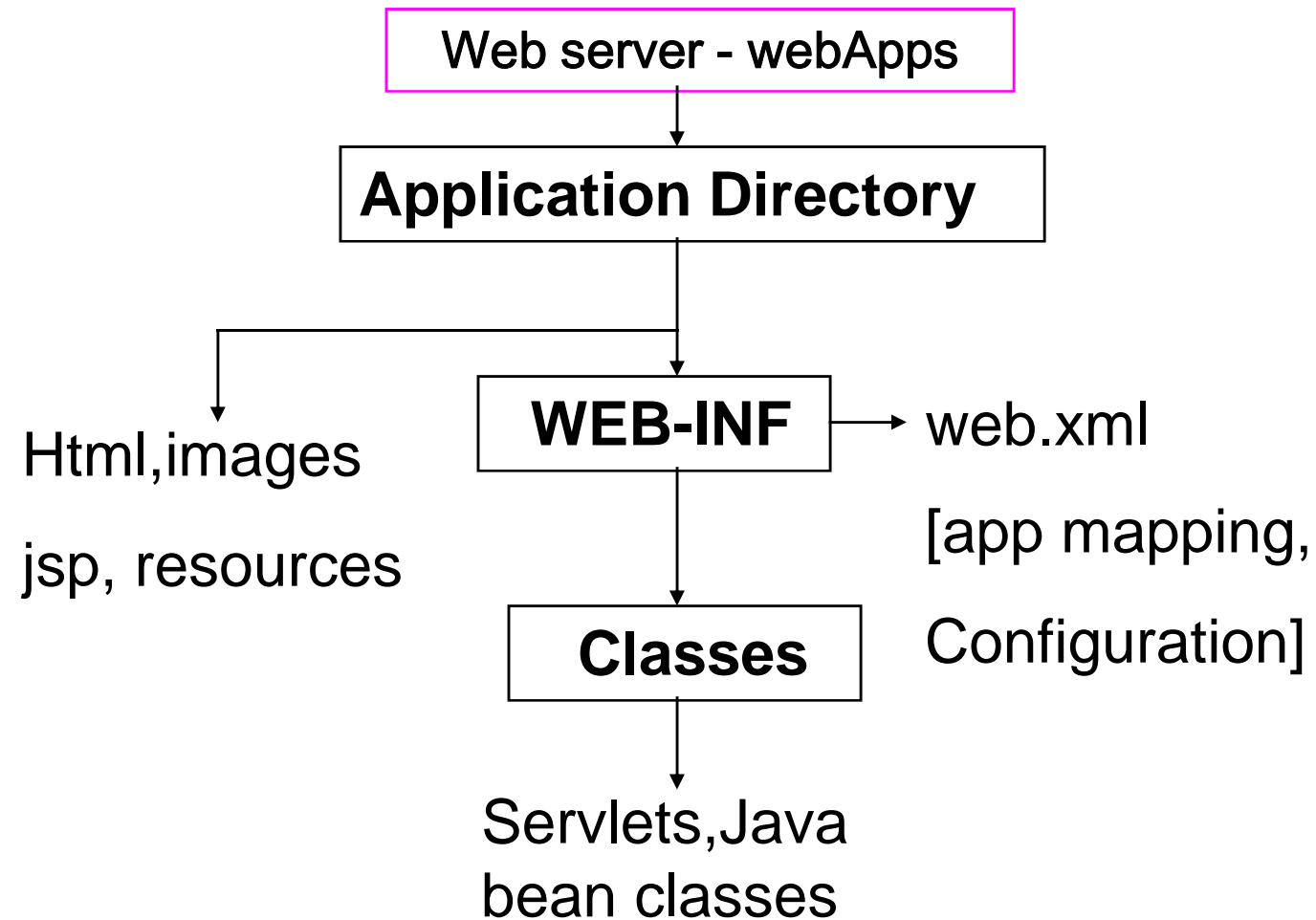
Tomcat Web Server Directory

70



Java Web application structure

71





Java Servlets

72

- Java Servlets are server independent java classes.
- How the Web server identifies the servlets ?
- Servlets are defined as java classes implementing a specific interface i.e. *Servlet* from *javax.servlet* package.
- The Servlet interface declares the abstract life cycle methods for servlet components which the servlet classes have to implement and are invoked by the web container/web server.



Servlet Life Cycle methods

73

These are invoked by the web container at particular situations.

- *`public void init(ServletConfig)` - invoked during initialization of servlet class object and called only once in the life time of the object. The initialization parameters are passed to the servlet object here.*
- *`public void service(ServletRequest req, ServletResponse res)` - invoked to handle each client request. The servlet receives the request information through ServletRequest object and sends data back to the client via ServletResponse object.*
- *`public void destroy()` - invoked when the servlet object is unloaded or the server is shut down*



Servlet implementation

74

To write a servlet class you must implement the **Servlet** interface with all its methods.

J2ee provides two classes which implement the Servlet and other interfaces so that your classes can extend these classes and override only the required methods.

These classes are **GenericServlet** from javax.servlet package and **HttpServlet** from javax.servlet.http package.

GenericServlet provides abstract Service method which should be overridden by subclasses.

HttpServlet Service method calls **doGet** or **doPost** methods depending on the method used by the browser to request data I.e. Get or Post methods.

Your First Servlet...

75

```
import javax.servlet.*;
import java.io.*;
public class myServlet extends GenericServlet
{ //Override the service method..
    public void service(ServletRequest req,ServletResponse res)
    {
        res.setContentType(text/html);
        PrintWriter out = res.getWriter( );
        out.println("<html>");
        out.println("<head><Title>Welcome to Java Server!</Title></head>");
        out.println("<h1>");
        out.println("Hello Java Client");
        out.println("</H1>");
        out.println("</html>");
    } }
```

Http Servlet Sub class Servlet

76

```
import javax.servlet.*;
import javax.servlet.http*;
import java.io.*;
public class myHttpServlet extends HttpServlet
{ //Override the doGet/doPost methods.
  public void doGet(HttpServletRequest req, HttpServletResponse res)
  {
    res.setContentType(text/html);
    PrintWriter out = res.getWriter( );
    out.println("<html>");
    out.println("<head><Title>Welcome to Web Server!</Title></head>");
    out.println("<h1>");
    out.println("Hello Web Client");
    out.println("</H1>");
    out.println("</html>");
  } }
```

Servlet compile and deploy

77

- Add the `servlet-api.jar` file from your java web server's lib directory to your system classpath
- Compile the class from command line.
- Keep the compiled class file into your web application's WEB-INF/classes directory
- Define the web.xml to contain the servlet class and url mapping.
- Open the browser and give the URL to request the servlet class from the web server.

i.e. <http://LocalHost:8080/myApp/hello>



Retrieving client request parameters

78

The **ServletRequest** and **HttpServletRequest** specifies methods to get the request attributes, headers and Request parameters.

Inside **Service** or **doGet/doPost** methods

```
String id = req.getParameter("id");
```

```
String pw = req.getParameter("pwd");
```

```
String val = req.getAttribute("myAttribute");
```

```
String lang = req. getHeader ("accept-lang");
```

Generating dynamic response

79

The **ServletResponse** and **HttpServletResponse** specifies methods to dynamically set the content type and generate the dynamic response.

Inside *Service* or *doGet/doPost* methods

```
response.setContentType("text/html");
```

```
PrintWriter out; = response.getWriter();
```

```
out.println("<html>");
```

```
out.println("<head><title>myResponse</title></head>");
```

```
out.println("<body>");
```



Servlet configuration

80

The **ServletConfig** interface specifies a set of methods that the container/server uses to pass information to a servlet during initialization.

```
public String getInitParameter (String param);
```

```
public Enumeration getInitParameterNames();
```

```
public ServletContext getServletContext();
```

```
public String getServletName();
```

This interface is implemented by **GenericServlet**.



Getting the Application Context

81

The **ServletContext** interface specifies a set of methods that a servlet uses to communicate with its servlet container/server, for example, to get the MIME type of a file, dispatch requests, or write to a log file. And is implemented by Servlet classes.

```
public Object getAttribute(String name);
```

```
public Enumeration getAttributeNames();
```

```
public void setAttribute(String name, Object object);
```

```
public void removeAttribute(String name);
```

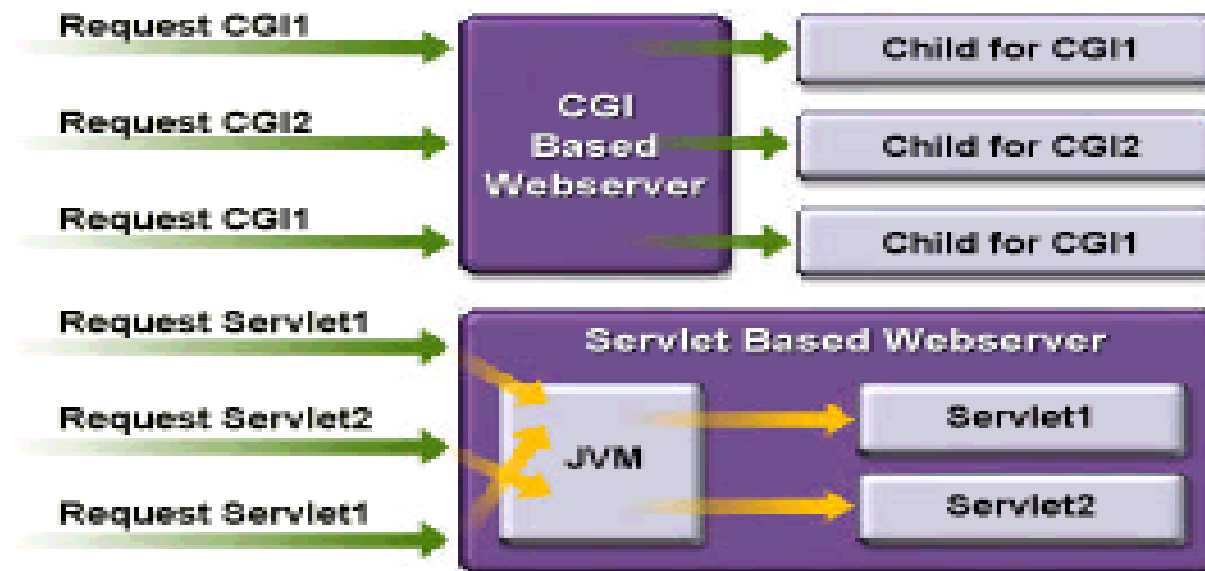
```
public void log(String msg);
```

```
public RequestDispatcher getRequestDispatcher(String path);
```

Servlet Execution

82

Servlets vs. CGI



- *Servlets are Java's answer to CGI/Perl.*



Servlet aliases and customization

83

We have seen that we can access the servlet by giving its url name in the address request.

<http://LocalHost:8080/myApps/hello> where 'hello' is the mapping defined for a servlet class name.

We can specify a dummy name to this Servlet so that nobody knows the real class name of the servlet ? Well.. This is provided by the web container as aliases.

The alias can be specified as mapping to a specific class name in the properties file or XML file.

This xml file is called as *Deployment Descriptor* (web.xml) where you can also specify initialization and other attributes to the web container and servlets. Through which the servlet is customized



Servlet web application

84

To deploy servlets you simply copy the compiled classes in the specific directory of the web server.

Instead of keeping servlets and all the required resources in their original form they all can be combined in a compressed file and kept in the server directory.

This compressed file is created by jar program and extension is **.war**. This is called a web application.

Each web application will have its own context provided by the web container.

So there can be multiple web applications running in isolation on the same server.



Servlet collaboration

The servlet api provides collaboration among the servlets and other resources through **RequestDispatcher** interface which specifies the methods as

public void forward(ServletRequest request, ServletResponse response) throws ServletException, IOException - forwards the current request to the specified resource

public void include(ServletRequest request, ServletResponse response) throws ServletException, IOException – includes the response from the specified resource.



RequestDispatcher

The **RequestDispatcher** is returned by two ways

`ServletContext.getRequestDispatcher("/Resource ");` Or

`HttpServletRequest.getRequestDispatcher("/Resource");`

The RequestDispatcher object reconstructs the client request and response objects and passes them to the specified resource which can be servlet,jsp or html page to be forwarded or included in the current response.



Internationalization support

The servlets provide internationalization support i.e. multi lingual response through **Writer** objects got from ServletResponse objects.

After identifying the client preferred language, the response can be customized in the client preferred language and character set.

The Unicode character set supported by java and the ,ResourceBundle objects specific to language properties will help in customizing the response to clients.



Servlet database connectivity

88

Since the servlet is java class you can add any java code to access the java functionality.

To provide responses from the database manipulation in the servlet you can initialize the database connections and other properties in the init method and in service or doGet/doPost methods access the database fields, manipulate them and generate dynamic responses to the clients.

Similarly file handling code can be added in servlets



Http session tracking

89

Http protocol used in web communication is stateless i.e. the moment the web client connects to the web server and gets the response the connection between them is disconnected.

For each new request the client has to create a new fresh connection and hence the protocol doesn't provide any means of storing or remembering these client transactions over number of requests to the web server.

In applications like online shopping cart the server side programming has to provide the technique to store the client transactions temporarily and is called as **Session tracking**.



Http session tracking

90

There are different ways to provide session tracking in web applications

Adding hidden fields in the generated html contents so that they will be passed to the next page and remembered.

Modifying the URL sent back to the server such that it will contain extra information appended to it in addition to the actual resource URL-this technique is called as URL Rewriting.e.g. a URL dynamically generated by the server to be sent to next page may look like this..

[Http://localhost:8080/bookstore1/cashier;jsessionid=c0o7fszeb1](http://localhost:8080/bookstore1/cashier;jsessionid=c0o7fszeb1) while the original url is <http://localhost:8080/bookstore1/cashier>.



Http session tracking with Cookies

Session tracking can also be done by using **Cookies** which are small text files generated and sent by server to the client and stored at client end. This cookie is sent by client to the server along with the request. Some browsers may disable the cookies, in that case another alternative has to be done.



Cookies for session tracking

92

The servlet api provides `javax.servlet.http.Cookie` class to manage cookie functions and through `HttpServletResponse` you can send cookie to the web client.

The `HttpServletResponse` specifies *`public String encodeURL(String url)` and `public String encodeRedirectURL(String url)` methods to manage URL rewriting in case cookies are disabled by the browser.*

The servlet api provides an interface `HttpSession` for session management between request and responses, which is implemented by the web container.



HttpSession methods

93

`public Object getAttribute(String name);` returns named attribute value

`public void setAttribute(String name, Object value);` -set the value with attribute name and new value .

`public Enumeration getAttributeNames();` - returns an enumeration of all attributes.

`public String getId()` - returns unique session id

`public ServletContext getServletContext();`

`public void invalidate ()` -invalidates/expires the current session with the client.

`public boolean isNew()` –finds whether the session is new or old.



Servlet loading and unloading

94

The servlets are loaded by the web container as normal classes and specific methods are invoked on these servlet objects by the container.

The container can also unload the servlets and it provides the admin screen to load, start and unload the specified web applications or specific servlets.