

Web Technology

Unit I- Web Development Process, Front End Tools

Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet,Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component,Transforming XML into XSLT,

DTD: Schema, elements, attributes,

Introduction to JSON.

Web Technology

- The methods by which computers communicate with each other through the use of markup languages and multimedia packages is known as **web technology**.

Web Technologies

- HTML
- XHTML
- CSS
- XML
- JavaScript
- VBSCRIPT
- DOM
- DHTML
- AJAX
- E4X
- WMLScript
- SQL
- ASP
- ADO
- PHP
- .NET
- SMIL
- SVG
- FLASH
- Java applets
- Java servlets
- Java Server Page

Internet and WWW

- Inter-network and World Wide Web
- Interlinked hypertext documents accessed using HTTP Protocol
- Client - Server architecture

What is Internet?

- The Internet is essentially a global network of computing resources. You can think of the Internet as a physical collection of routers and circuits as a set of shared resources.

Internet-Based Services

- **Email** – A fast, easy, and inexpensive way to communicate with other Internet users around the world.
- **Telnet** – Allows a user to log into a remote computer as though it were a local system.
- **FTP** – Allows a user to transfer virtually every kind of file that can be stored on a computer from one Internet-connected computer to another.
- **UseNet news** – A distributed bulletin board that offers a combination news and discussion service on thousands of topics.
- **World Wide Web (WWW)** – A hypertext interface to Internet information resources.

What is WWW?

- WWW stands for **World Wide Web**.
- A technical definition of the World Wide Web is – All the resources and users on the Internet that are using the Hypertext Transfer Protocol (HTTP).
- In simple terms, The World Wide Web is a way of exchanging information between computers on the Internet

What is HTTP?

- HTTP stands for **Hypertext Transfer Protocol**. This is the protocol being used to transfer hypertext documents that makes the World Wide Web possible.
- A standard web address such as **Google.com** is called a URL and here the prefix **http** indicates its protocol

What is URL?

- URL stands for **Uniform Resource Locator**, and is used to specify addresses on the World Wide Web.
- A URL will have the following format –
 - **protocol://hostname/other_information**
- The protocol is followed by a colon, two slashes, and then the domain name. The domain name is the computer on which the resource is located.

What is Website?

- which is a collection of various pages written in HTML markup language.
- Each page available on the website is called a *web page* and first page of any website is called *home page* for that site.

What is Web Server?

- Every Website sits on a computer known as a Web server.
- This server is always connected to the internet.
- Every Web server that is connected to the Internet is given a unique address. For example, 68.178.157.132
- When you register a Web address, also known as a domain name, such as tutorialspoint.com you have to specify the IP address of the Web server that will host the site.
- **Examples of Web Servers**
 - Apache Tomcat
 - IIS
 - Glassfish

What is Web Browser?

- Web Browsers are software installed on your PC. To access the Web you need a web browsers.
- **Examples of Web Browsers**
 - Netscape Navigator,
 - Microsoft Internet Explorer
 - Mozilla Firefox.

What is ISP?

- ISP stands for **Internet Service Provider**. They are the companies who provide you service in terms of internet connection to connect to the internet.
- You will buy space on a Web Server from any Internet Service Provider. This space will be used to host your Website.
- **Examples of ISP Providers**
 - Reliance
 - Airtel
 - BSNL

Technologies- List of Technologies

- **Client Side Technologies**
 - HTML, CSS, JavaScript, VBScript
 - XHTML, DHTML, WML, AJAX
 - FLASH
- **Server Side Technologies**
 - ASP, PHP, Perl, JSP
 - ASP.NET, Java
 - MySQL, SQL Server, Access
- **Some More Advanced Technologies**
 - XML, XSLT, RSS, Atom
 - X-Path, XQuery, WSDL
 - XML-DOM, RDF
 - Ruby on Rails, GRAIL Framework
 - REST, SOAP

How to choose a Technology?

- **Depends on:**
 - What is the type of content?
 - Who is your audience?
 - Who will modify your content?
 - What are your Future Plans?
 - Availability of technology?
 - Your previous experience?
 - Portability and Data sharing

What is HTML?

- HTML stands for **H**yper **T**ext **M**arkup **L**anguage.
- This is the language in which we write web pages for any Website.
- This is a subset of Standard Generalized Mark-Up Language (SGML) for electronic publishing, the specific standard used for the World Wide Web.

What is Hyperlink?

- A hyperlink or simply a link is a selectable element in an electronic document that serves as an access point to other electronic resources.
- Typically, you click the hyperlink to access the linked resource.
- Familiar hyperlinks include buttons, icons, image maps, and clickable text links.

Web - Domain Names & Extension Types

- A domain name is the part of your Internet address that comes after "www". For example, in Tutorialspoint.com the domain name is tutorialspoint.com.
- Some Domain Extensions are as mentioned below
 - **.com** – Stands for company/commercial, but it can be used for any website.
 - **.net** – Stands for network and is usually used for a network of sites.
 - **.org** – Stands for organization and is supposed to be for non-profit bodies.
 - **.us, .in** – They are based on your country names so that you can go for country specific domain extensions
 - **.biz** – A newer extension on the Internet and can be used to indicate that this site is purely related to business.

Website Designing steps

- Information Gathering
- Planning
- Design
- Development
- Testing and Delivery
- Maintenance

Website Planning

- Define your target audiences.
 - Set goals and objectives.
 - Create a sitemap.
 - Request an estimate.
 - Set a budget.
 - Define roles and responsibilities.
-
- <https://www.farreachinc.com/blog/far-reach/2014/08/14/6-steps-how-to-plan-a-successful-website-redesign>

Website Design Issues

- Simplicity –less animations, texts, visuals
- Identity –web apps through design
- Consistency –e.g. uniform style, color, etc.
- Robustness–required function should not miss
- Navigability –navigation should be simple
- Visual appeal–look & feel of content
- Compatibility –compatible to all browsers, internet connection types, OS, etc.

Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet,Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component,Transforming XML into XSLT,

DTD: Schema, elements, attributes,

Introduction to JSON.



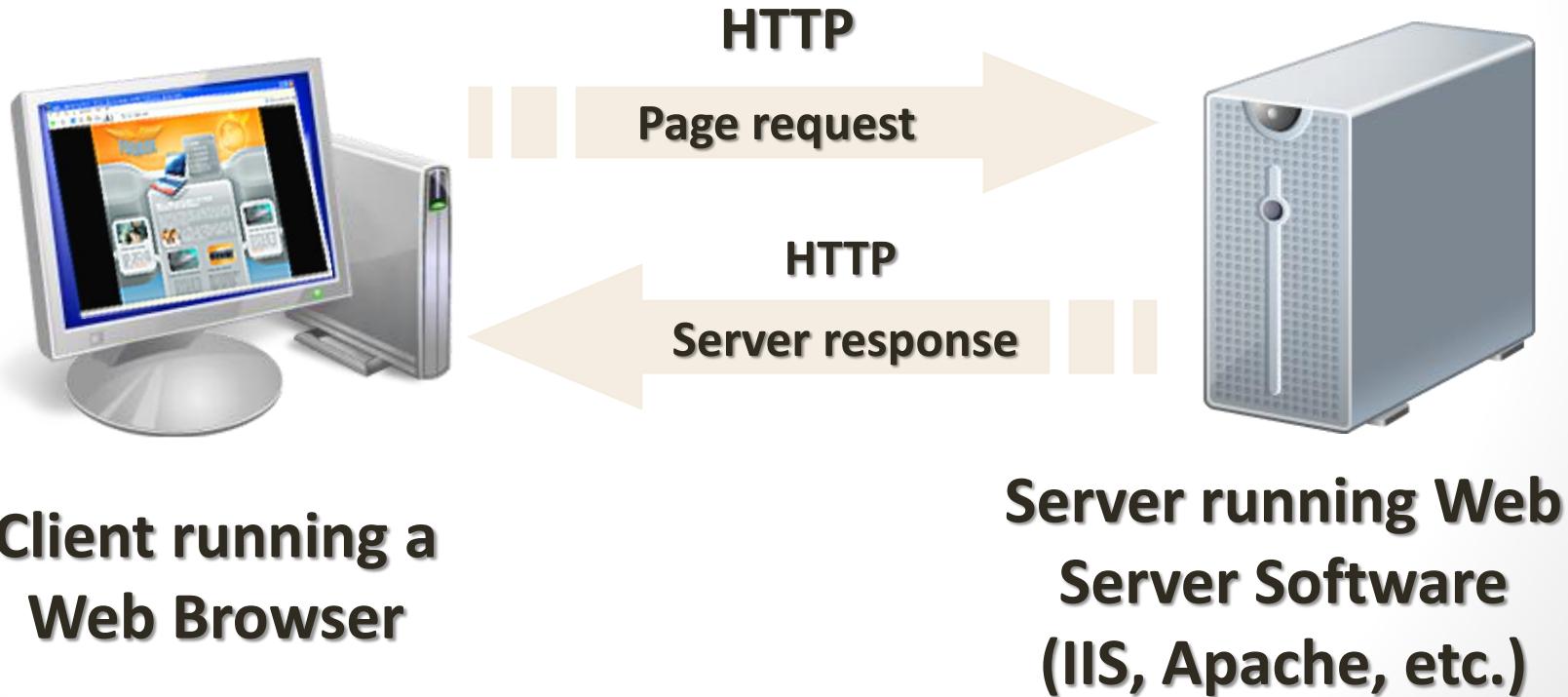
```
1<!DOCTYPE HTML PUBLIC "-//W3C//DTD
2 "http://www.w3.org/TR/html4/strict.
3 <html>
4 <head>
5   <title>Example</title>
6   <link rel="stylesheet" href="s.
7 </head>
8 <body>
9   <div id="header">
10    <h1><a href="#" title="Back .
11 </div>
12 <div id="toolbar">
13   <span class="left">Today <sp.
14   <span class="right">
15     <span id="time">&ampnbsp</sp.
16     <select id="timezone">
17       <option value="-12">(GMT-
18       <option value="-11">(GMT-
```

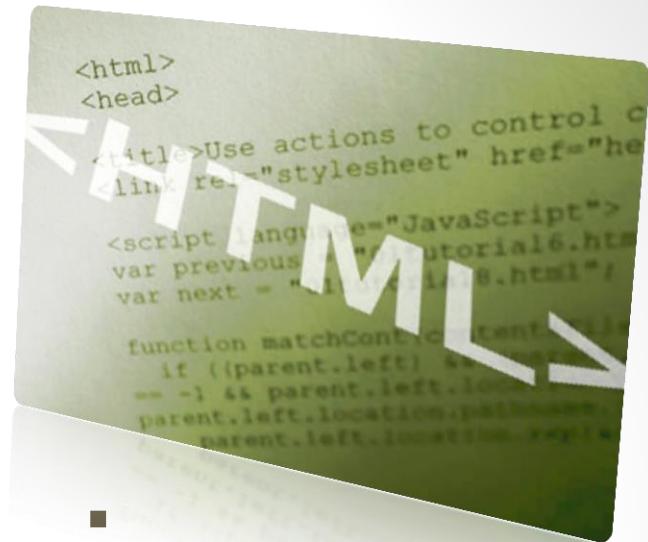
HTML Basics



How the Web Works?

- WWW use classical client / server architecture
 - HTTP is text-based request-response protocol

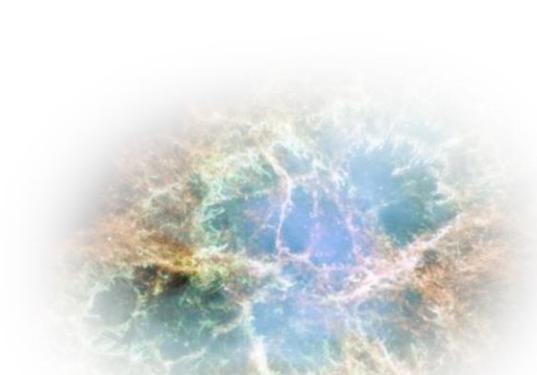




HTML Basics

Text, Images, Tables, Forms

ROW	ALPHABET SUBSTRINGS	NUMBER, ALPHABET NUMBER	DATE	NAME, DATE	ROW, DATE
0	abcde	abcde	1994/01/01	1994/01/01	0, 1994
1	bcde	bcde	1994/01/1994	1994/01/1994	1, 1994
2	cde	cde	1994/01/1993	1994/01/1993	2, 1993
3	de	de	1994/01/1992	1994/01/1992	3, 1992
4	e	e	1994/01/1991	1994/01/1991	4, 1991
5			1994/01/1990	1994/01/1990	5, 1990
6			1994/01/1989	1994/01/1989	6, 1989
7			1994/01/1988	1994/01/1988	7, 1988
8			1994/01/1987	1994/01/1987	8, 1987
9			1994/01/1986	1994/01/1986	9, 1986
10			1994/01/1985	1994/01/1985	10, 1985
11			1994/01/1984	1994/01/1984	11, 1984
12			1994/01/1983	1994/01/1983	12, 1983
13			1994/01/1982	1994/01/1982	13, 1982
14			1994/01/1981	1994/01/1981	14, 1981
15			1994/01/1980	1994/01/1980	15, 1980
16			1994/01/1979	1994/01/1979	16, 1979
17			1994/01/1978	1994/01/1978	17, 1978
18			1994/01/1977	1994/01/1977	18, 1977
19			1994/01/1976	1994/01/1976	19, 1976
20			1994/01/1975	1994/01/1975	20, 1975
21			1994/01/1974	1994/01/1974	21, 1974
22			1994/01/1973	1994/01/1973	22, 1973
23			1994/01/1972	1994/01/1972	23, 1972
24			1994/01/1971	1994/01/1971	24, 1971
25			1994/01/1970	1994/01/1970	25, 1970
26			1994/01/1969	1994/01/1969	26, 1969
27			1994/01/1968	1994/01/1968	27, 1968
28			1994/01/1967	1994/01/1967	28, 1967
29			1994/01/1966	1994/01/1966	29, 1966
30			1994/01/1965	1994/01/1965	30, 1965
31			1994/01/1964	1994/01/1964	31, 1964
32			1994/01/1963	1994/01/1963	32, 1963
33			1994/01/1962	1994/01/1962	33, 1962
34			1994/01/1961	1994/01/1961	34, 1961
35			1994/01/1960	1994/01/1960	35, 1960
36			1994/01/1959	1994/01/1959	36, 1959
37			1994/01/1958	1994/01/1958	37, 1958
38			1994/01/1957	1994/01/1957	38, 1957
39			1994/01/1956	1994/01/1956	39, 1956
40			1994/01/1955	1994/01/1955	40, 1955
41			1994/01/1954	1994/01/1954	41, 1954
42			1994/01/1953	1994/01/1953	42, 1953
43			1994/01/1952	1994/01/1952	43, 1952
44			1994/01/1951	1994/01/1951	44, 1951
45			1994/01/1950	1994/01/1950	45, 1950
46			1994/01/1949	1994/01/1949	46, 1949
47			1994/01/1948	1994/01/1948	47, 1948
48			1994/01/1947	1994/01/1947	48, 1947
49			1994/01/1946	1994/01/1946	49, 1946
50			1994/01/1945	1994/01/1945	50, 1945
51			1994/01/1944	1994/01/1944	51, 1944
52			1994/01/1943	1994/01/1943	52, 1943
53			1994/01/1942	1994/01/1942	53, 1942
54			1994/01/1941	1994/01/1941	54, 1941
55			1994/01/1940	1994/01/1940	55, 1940
56			1994/01/1939	1994/01/1939	56, 1939
57			1994/01/1938	1994/01/1938	57, 1938
58			1994/01/1937	1994/01/1937	58, 1937
59			1994/01/1936	1994/01/1936	59, 1936
60			1994/01/1935	1994/01/1935	60, 1935
61			1994/01/1934	1994/01/1934	61, 1934
62			1994/01/1933	1994/01/1933	62, 1933
63			1994/01/1932	1994/01/1932	63, 1932
64			1994/01/1931	1994/01/1931	64, 1931
65			1994/01/1930	1994/01/1930	65, 1930
66			1994/01/1929	1994/01/1929	66, 1929
67			1994/01/1928	1994/01/1928	67, 1928
68			1994/01/1927	1994/01/1927	68, 1927
69			1994/01/1926	1994/01/1926	69, 1926
70			1994/01/1925	1994/01/1925	70, 1925
71			1994/01/1924	1994/01/1924	71, 1924
72			1994/01/1923	1994/01/1923	72, 1923
73			1994/01/1922	1994/01/1922	73, 1922
74			1994/01/1921	1994/01/1921	74, 1921
75			1994/01/1920	1994/01/1920	75, 1920
76			1994/01/1919	1994/01/1919	76, 1919
77			1994/01/1918	1994/01/1918	77, 1918
78			1994/01/1917	1994/01/1917	78, 1917
79			1994/01/1916	1994/01/1916	79, 1916
80			1994/01/1915	1994/01/1915	80, 1915
81			1994/01/1914	1994/01/1914	81, 1914
82			1994/01/1913	1994/01/1913	82, 1913
83			1994/01/1912	1994/01/1912	83, 1912
84			1994/01/1911	1994/01/1911	84, 1911
85			1994/01/1910	1994/01/1910	85, 1910
86			1994/01/1909	1994/01/1909	86, 1909
87			1994/01/1908	1994/01/1908	87, 1908
88			1994/01/1907	1994/01/1907	88, 1907
89			1994/01/1906	1994/01/1906	89, 1906
90			1994/01/1905	1994/01/1905	90, 1905
91			1994/01/1904	1994/01/1904	91, 1904
92			1994/01/1903	1994/01/1903	92, 1903
93			1994/01/1902	1994/01/1902	93, 1902
94			1994/01/1901	1994/01/1901	94, 1901
95			1994/01/1900	1994/01/1900	95, 1900
96			1994/01/1900	1994/01/1900	96, 1900
97			1994/01/1900	1994/01/1900	97, 1900
98			1994/01/1900	1994/01/1900	98, 1900
99			1994/01/1900	1994/01/1900	99, 1900
100			1994/01/1900	1994/01/1900	100, 1900



What is HTML?

- HTML is the standard markup language for creating Web pages.
- HTML stands for **Hyper Text Markup Language**
- HTML **describes the structure of Web pages** using markup
- HTML elements are the building blocks of HTML pages
- HTML elements are represented by **tags**
- Browsers do not display the HTML tags, but use them to render the content of the page

A Simple HTML Document

Example

```
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Explanation

- The **<html>** element is the root element of an HTML page
- The **<head>** element contains meta information about the document
- The **<title>** element specifies a title for the document
- The **<body>** element contains the visible page content
- The **<h1>** element defines a large heading
- The **<p>** element defines a paragraph

HTML Tags

- HTML tags are element names **surrounded by angle brackets**:
- <tagname>content goes here...</tagname>
- HTML tags normally come **in pairs** like <p> and </p>
- The first tag in a pair is the **start tag**, the second tag is the **end tag**
- The **end tag** is written like the start tag, but with a **forward slash** inserted before the tag name

HTML Page Structure

```
<html>
```

```
  <head>
```

```
    <title>Page title</title>
```

```
  </head>
```

```
<body>
```

```
  <h1>This is a heading</h1>
```

```
  <p>This is a paragraph.</p>
```

```
  <p>This is another paragraph.</p>
```

```
</body>
```

```
</html>
```

HTML Versions

Version	Year
HTML	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2014

Creating HTML Page

Write HTML Using [Notepad orTextEdit](#)

Save the file on your computer using [.html or .htm extension](#) and set the encoding to UTF-8

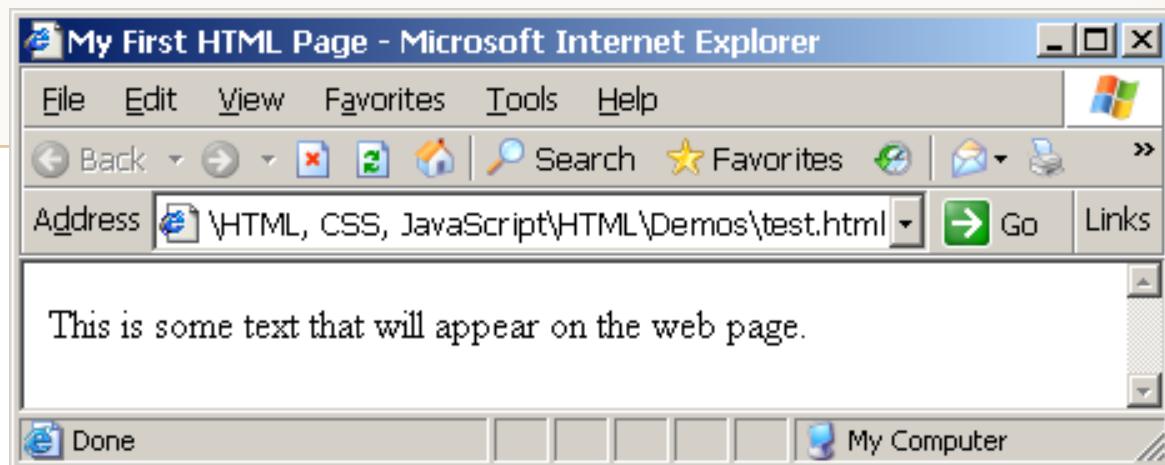
View the HTML Page in Your [Browser](#)

First HTML Page

test.html

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
    <p>This is some text...</p>
  </body>
</html>
```

33



First HTML Page: Tags

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
    <p>This is some text...</p>
  </body>
</html>
```

Opening tag

Closing tag

An HTML element consists of an opening tag, a closing tag and the content inside.

First HTML Page: Header

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
    <p>This is some text...</p>
  </body>
</html>
```

HTML header

First HTML Page: Body

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
    <p>This is some text...</p>
  </body>
</html>
```

HTML body

HTML Headings-

HTML headings are defined with the `<h1>` to `<h6>` tags. `<h1>` defines the most important heading. `<h6>` defines the least important heading:

HTML Code

```
<!DOCTYPE html>
<html>
<body>

<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>

</body>
</html>
```

Output

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

Headings and Paragraphs

- Heading Tags (h1 – h6)

```
<h1>Heading 1</h1>
<h2>Sub heading 2</h2>
<h3>Sub heading 3</h3>
```

- Paragraph Tags

```
<p>This is my first paragraph</p>
<p>This is my second paragraph</p>
```

- Sections: div and span

```
<div style="background: skyblue;">
    This is a div</div>
```

Text Formatting

- Text formatting tags modify the text between the opening tag and the closing tag
 - Ex. **Hello** makes “Hello” bold

	bold
<i></i>	<i>italicized</i>
<u></u>	<u>underlined</u>
	Sample ^{superscript}
	Sample _{subscript}
	strong
	<i>emphasized</i>
<pre></pre>	Preformatted text
<blockquote></blockquote>	Quoted text block
	Deleted text – strike through

Text Formatting Example

HTML Code

```
<!DOCTYPE html>
<html>
<body>
<b>This text is bold.</b>
<br/>
<i>This text is Italic.</i>
<br/>
<strong>This text is strong</strong>
<br/>
This is <sup>superscripted</sup> text.
</body>
</html>
```

Output

This text is bold.
This text is Italic.
This text is strong
This is ^{superscripted} text.

Hyperlinks: <a> Tag

- Link to a document called `form.html` on the same server in the same directory:

```
<a href="form.html">Fill Our Form</a>
```

- Link to a document called `parent.html` on the same server in the parent directory:

```
<a href="../parent.html">Parent</a>
```

- Link to a document called `cat.html` on the same server in the subdirectory `stuff`:

```
<a href="stuff/cat.html">Catalog</a>
```

- Link to an external Web site:

```
<a href="http://www.devbg.org" target="_blank">BASD</a>
```

Links to the Same Document – Example

links-to-same-document.html

```
<h1>Table of Contents</h1>

<p><a href="#section1">Introduction</a><br />
<a href="#section2">Some background</A><br />
<a href="#section2.1">Project History</a><br />
...the rest of the table of contents...

<!-- The document text follows here -->

<h2 id="section1">Introduction</h2>
... Section 1 follows here ...
<h2 id="section2">Some background</h2>
... Section 2 follows here ...
<h3 id="section2.1">Project History</h3>
... Section 2.1 follows here ...
```

Images: tag

- ◆ Inserting an image with tag:

```

```

- ◆ Image attributes:

src	Location of image file (relative or absolute)
alt	Substitute text for display (e.g. in text mode)
height	Number of pixels of the height
width	Number of pixels of the width
border	Size of border, 0 for no border

- ◆ Example:

```

```

Ordered Lists: Tag

- Create an Ordered List using :

```
<ol type="1" >
  <li>Apple</li>
  <li>Orange</li>
  <li>Grapefruit</li>
</ol>
```

- Attribute values for type are 1, A, a, I, or i
 - 1. Apple
 - 2. Orange
 - 3. Grapefruit
- a. Apple
 - b. Orange
 - c. Grapefruit
- I. Apple
 - II. Orange
 - III. Grapefruit
- i. Apple
 - ii. Orange
 - iii. Grapefruit

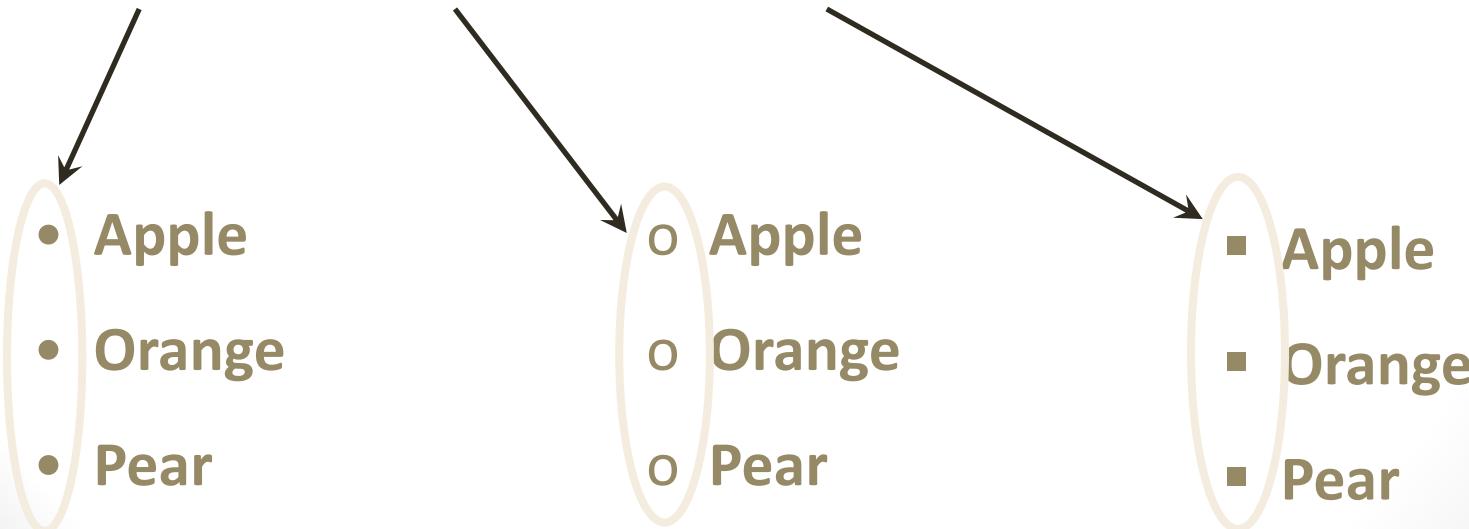
Unordered Lists: Tag

- Create an Unordered List using :

```
<ul type="disk" >  
  <li>Apple</li>  
  <li>Orange</li>  
  <li>Grapefruit</li>  
</ul>
```

- Attribute values for type are:

- **disc, circle or square**



Definition lists: <dl> tag

- Create definition lists using <dl>
 - Pairs of text and associated definition; text is in <dt> tag, definition in <dd> tag

```
<dl>
  <dt>HTML</dt>
  <dd>A markup language ...</dd>
  <dt>CSS</dt>
  <dd>Language used to ...</dd>
</dl>
```

- Renders without bullets
- Definition is indented

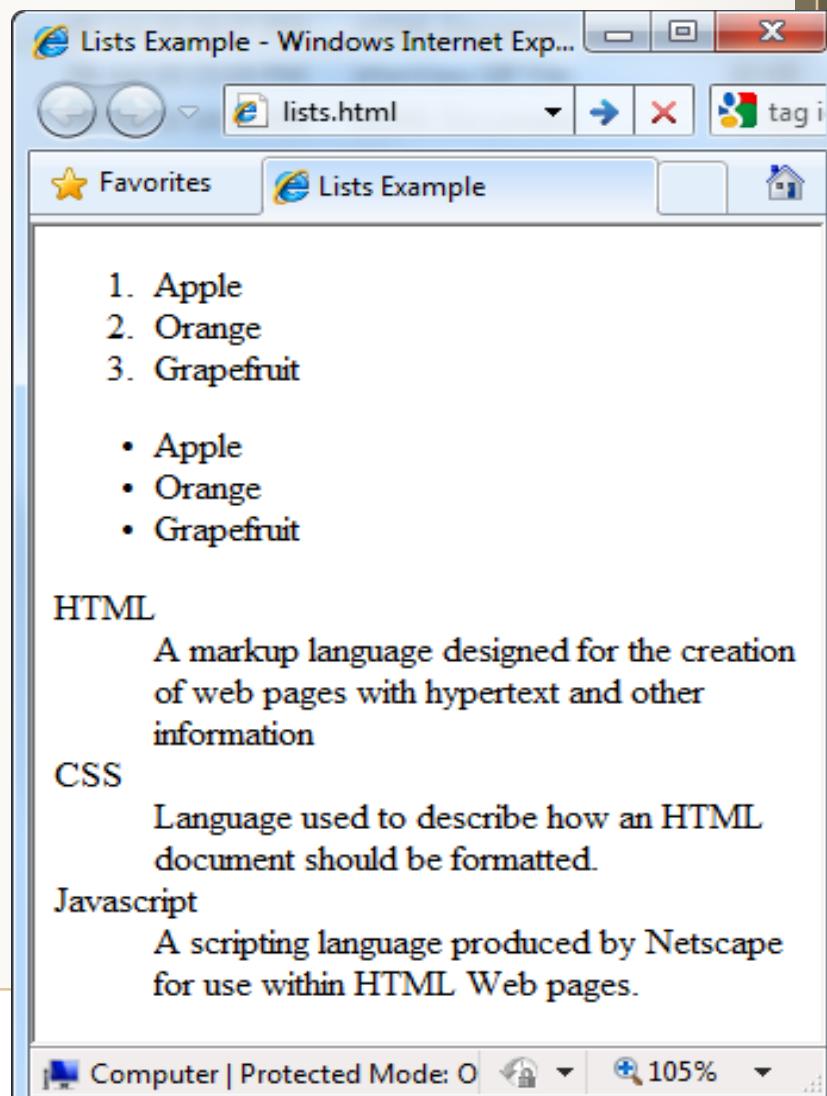
Lists – Example

```
<ol type="1">
  <li>Apple</li>
  <li>Orange</li>
  <li>Grapefruit</li>
</ol>

<ul type="disc">
  <li>Apple</li>
  <li>Orange</li>
  <li>Grapefruit</li>
</ul>

<dl>
  <dt>HTML</dt>
  <dd>A markup lang...</dd>
</dl>
```

lists.html



HTML Tables

htmltable1 - Notepad

```

<html>
<head>
<title>How To Create HTML Tables</title>
</head>
<body>
<table border=1 cellspacing=0 cellpadding=0>
<tr>
<td width=110 valign=top>
<br><upper left corner>
<td>
<td width=110 valign=top>
<br><upper right corner>
<td>
</tr>
<tr>
<td width=110 valign=top>
<br><left center cell>
<td width=110 valign=top>
<br><right center cell>
<td>
</tr>
<tr>
<td width=110 valign=top>
<br><lower left corner>
<td>
<td width=110 valign=top>
<br><lower right corner>
<td>
</tr>
</table>
</body>
</html>

```

US time	European date (D/M/Y) & time	Y-M-D date & time	Dollar	Chinese money	IP addresses	Names	Numbers
29/10/1965		83-03-24		YMB 4	98.176.35.80		26.32 E +03
Fri Mar 22 21:48:49 UTC+0200 1957		1967-08-22 06:07:16 PM		YMB -81.38	162.117.253.34	dyse chidi	-191.45E-05
Fri, 14 Feb 2002 04:24:20 UTC	06/07/99 06:46:01 AM	81-02-04 09:09:54 AM		YMB -108.83	122.205.50.6	bochai dychai	-131.20E+01
Monday, May 30, 1994 4:47:31 PM	06/09/05 05:11:16 AM			YMB 33.16		dydy balie	-131.20E+01
09/28/2000	24/11/1957		\$-38.77	YMB 112.42	15.192.151.209		
		97-08-13 00:01:33 AM	\$14.5	YMB -1.75	99.93.147.150	dychai tonchai	-187.28E-05
Mon, 29 Oct 1979 00:44:03 UTC		87-10-16	\$14.66	YMB 61.14		chite malie	- 125.19 E -03
Sat, 9 Jan 1982 05:45:06 UTC	04/06/68	74-10-20	\$20.47		121.169.225.22	dyma bama	138.11E+02
04/05/75		2000-03-20	\$68.84	YMB 88.19	239.133.227.68	made liete	195.44 E +03
Monday, July 15, 2002 1:05:02 AM	01/02/1961 09:40:16 AM		\$97.9	YMB 44.28	223.66.228.116	mava sete	-107
This is footer	row	number	ONE!	adsf	adsf	adsf	adsf

Title	Title	Title	Title	Title	Title
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data

HTML Table Tags

Tag	Description
<u><table></u>	Defines a table
<u><th></u>	Defines a header cell in a table
<u><tr></u>	Defines a row in a table
<u><td></u>	Defines a cell in a table
<u><caption></u>	Defines a table caption
<u><colgroup></u>	Specifies a group of one or more columns in a table for formatting
<u><col></u>	Specifies column properties for each column within a <colgroup> element
<u><thead></u>	Groups the header content in a table
<u><tbody></u>	Groups the body content in a table
<u><tfoot></u>	Groups the footer content in a table

HTML Tables (2)

- Start and end of a table

```
<table> ... </table>
```

- Start and end of a row

```
<tr> ... </tr>
```

- Start and end of a cell in a row

```
<td> ... </td>
```

Simple HTML Tables – Example

```
<html>
<body>

<table width=100% border = "1" bgcolor = "yellow">
<tr>
  <th>Firstname</th>
  <th>Lastname</th>
  <th>Age</th>
</tr>
<tr>
  <td>Jill</td>
  <td>Smith</td>
  <td>50</td>
</tr>
</table>

</body>
</html>
```

Firstname	Lastname	Age
Jill	Smith	50

Simple HTML Tables – Example

```
<html>
<body>

<table width=50% border = "1" >
  <tr>
    <th colspan=2>Firstname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
</table>

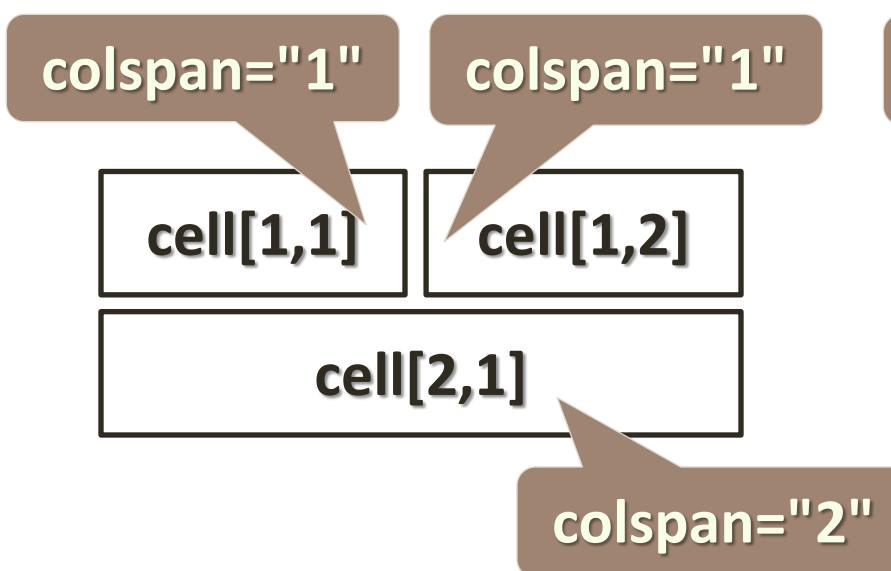
</body>
</html>
```

Firstname		Age
Jill	Smith	50

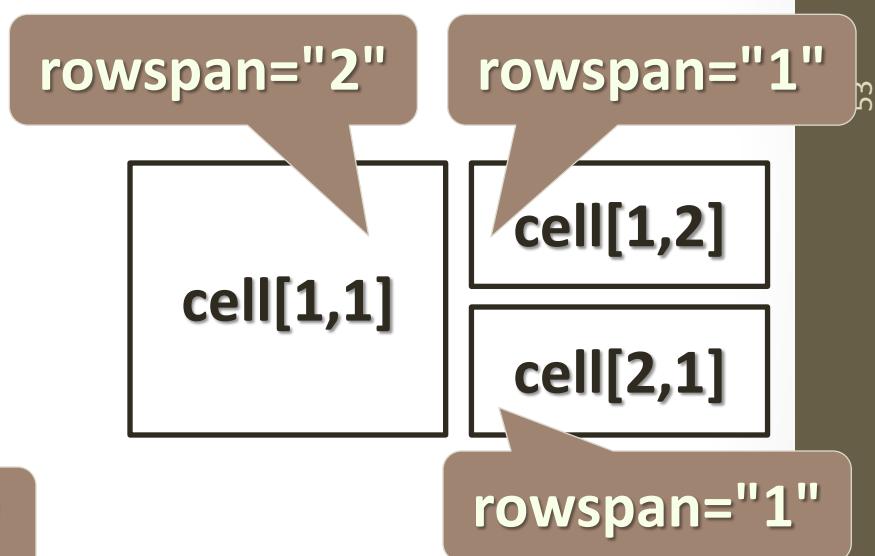
Column and Row Span

- Table cells have two important attributes:

◆ **colspan**



◆



◆ **Defines how many columns the cell occupies**

◆ **Defines how many rows the cell occupies**

HTML Tables – colspan Example

```
<html>
<body>

<table width=50% border = "1" >
  <tr>
    <th colspan=2>Firstname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
</table>

</body>
</html>
```

Firstname		Age
Jill	Smith	50

HTML Tables – rowspan Example

```
<h2 align=center> Cell that spans two rows: </h2>
<table style="width:50%" border="1" align=center>
  <tr>
    <th>Name:</th>
    <td>Bill Gates</td>
  </tr>
  <tr>
    <th rowspan="2">Telephone:</th>
    <td>55577854</td>
  </tr>
  <tr>
    <td>55577855</td>
  </tr>
</table>
</body>
</html>
```

Cell that spans two rows:

Name:	Bill Gates
Telephone:	55577854
	55577855

HTML Tables – rowspan

Example with <style>

```
<html>
  <head>
    <style>
      th, td {
        padding: 5px;
        text-align: left;
      }
    </style>
  </head>
  <body>
```

```
  <h2>Cell that spans two rows:</h2>
  <table style="width:50%" border=1>
    <tr>
      <th>Name:</th>
      <td>Bill Gates</td>
    </tr>
    <tr>
      <th rowspan="2">Telephone:</th>
      <td>55577854</td>
    </tr>
    <tr>
      <td>55577855</td>
    </tr>
  </table>

  </body>
</html>
```

Cell that spans two rows:

Name:	Bill Gates
Telephone:	55577854
	55577855

Complete HTML Tables

- Table rows split into three semantic sections: header, body and footer
 - <thead> denotes table header and contains <th> elements, instead of <td> elements
 - <tbody> denotes collection of table rows that contain the very data
 - <tfoot> denotes table footer but comes BEFORE the <tbody> tag
 - <colgroup> and <col> define columns (most often used to set column widths)



HTML Forms

Registration Form - Mozilla Fi...

User name:

Password:

Gender: Male Female

Click to accept our terms:

Done

Done

HTML

Entering User Data from a Web Page

Art School Form - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Go Norton Antivirus

Address: C:\E-class\forms.html

First Name:

Last Name:

Age: 1-17 yrs
 18 yrs and over

I would like to learn to work with:

watercolors
 acrylics
 pastels

I am interested in art lessons because:

Send me an application now!

HTML Form

- **The <form> Element**
- The HTML <form> element defines a form that is used to collect user input:
- **Syntax**

```
<form>  
  form elements  
</form>
```

- Form elements are different types of input elements, like
 - text fields,
 - checkboxes,
 - radio buttons,
 - submit buttons, and more.

The <input> Element

- The <input> element is the most important form element.
- Here are some examples:
 - **Input Type Text**
 - **Input Type Password**
 - **Input Type Submit**
 - **Input Type Radio**
 - **Input Type Reset**
 - **Input Type Checkbox**
 - **Input Type Button ...etc**

- **HTML5 Input Types**
- HTML5 added several new input types
 - color
 - date
 - datetime-local
 - email
 - month
 - number
 - range
 - search
 - tel
 - time
 - url
 - week

Input Type Text

- `<input type="text">` defines a **one-line text input field**:
- Example

```
<form>  
  First name:<br>  
  <input type="text" name="firstname"><br>  
  Last name:<br>  
  <input type="text" name="lastname">  
</form>
```

First name:

Last name:

Input Type Password

- **<input type="password">**
 - defines a **password field**:
 - Example
- This is how the HTML code above will be displayed in a browser:

```
<form>  
User name:<br>  
<input type="text" name="username"><br>  
User password:<br>  
<input type="password" name="psw">  
</form>
```

User name:

User password:

The characters in a password field are masked (shown as asterisks or circles).

Input Type Submit

- defines a button for **submitting** form data to a **form-handler**.
- The form-handler is specified in the form's **action** attribute:
- **Example**

```
<form action="/action_page.php">
```

First name:


```
<input type="text" name="firstname" value="Mickey"> <br>
```

Last name:


```
<input type="text" name="lastname" value="Mouse"><br><br>
```

```
<input type="submit" value="Submit">
```

```
</form>
```

The image shows a screenshot of a web browser displaying a form. The form consists of three main elements: a text input field labeled "First name:" containing the value "Mickey", a text input field labeled "Last name:" containing the value "Mouse", and a large blue rectangular button labeled "Submit". The "Submit" button is significantly larger than the standard input fields, emphasizing its role as the primary action button for the form.

Input Type Reset

- **<input type="reset">** defines a **reset button** that will reset all form values to their default values:
- **click the "Reset" button, the form-data will be reset.**
- **Example**

```
<form action="/action_page.php">
First name:<br>
<input type="text" name="firstname" >
<br>
Last name:<br>
<input type="text" name="lastname" >
<br><br>
<input type="submit" value="Submit">
<input type="reset">
</form>
```

The diagram illustrates the mapping of form fields from the provided HTML code to a browser's user interface. A large green arrow points from the left side of the slide towards the right, where a screenshot of a web browser is displayed. The browser shows a form with two text input fields: 'First name:' and 'Last name:', each with an empty white box. Below the inputs are two buttons: 'Submit' and 'Reset'. The 'Submit' button is blue with white text, and the 'Reset' button is grey with white text.

First name:

Last name:

Submit Reset

Input Type Button

- `<input type="button">` defines a **button**:
- **Example**
- `<input type="button" onclick="alert('Hello World!')"
value="Click Me!">`



- After clicking above button it shows output as below:



Input Type Radio

- `<input type="radio">` defines a **radio button**.
- Radio buttons let a user select ONLY ONE of a limited number of choices:
- `<form>`
`<input type="radio" name="gender" value="male" checked> Male
`
`<input type="radio" name="gender" value="female"> Female
`
`<input type="radio" name="gender" value="other"> Other`
`</form>`
- This is how the HTML code above will be displayed in a browser:

Male
 Female
 Other

Input Type Checkbox

- `<input type="checkbox">` defines a **checkbox**.
- Checkboxes let a user select ZERO or MORE options of a limited number of choices.
- **Example**

```
<form>
  <input type="checkbox" name="vehicle1" value="Bike"> I have a bike
  <br>
  <input type="checkbox" name="vehicle2" value="Car"> I have a car
</form>
```

- This is how the HTML code above will be displayed in a browser:

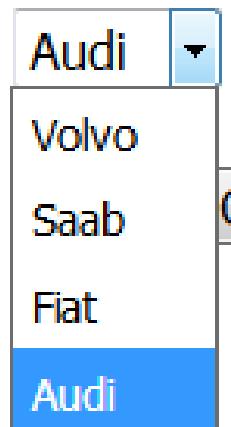
I have a bike
 I have a car

HTML Input Attributes

- The value Attribute
- The readonly Attribute
- The disabled Attribute
- The size Attribute
- The maxlength Attribute

The <select> Element (Dropdown menus)

- The <select> element defines a **drop-down list**:
- ```
<select name="cars">
 <option value="volvo">Volvo</option>
 <option value="saab">Saab</option>
 <option value="fiat">Fiat</option>
 <option value="audi">Audi</option>
</select>
```



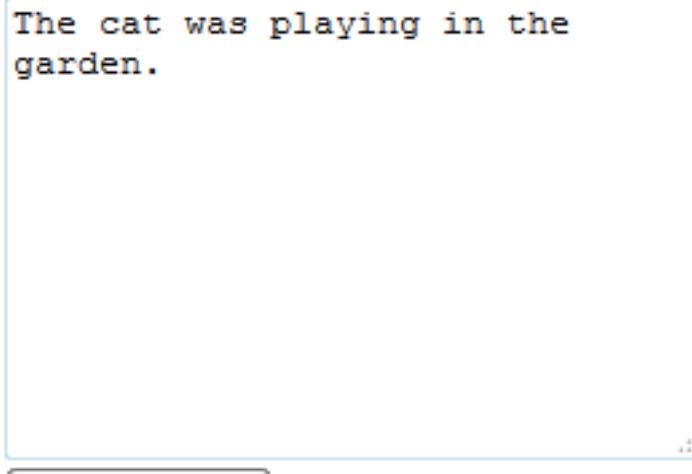
# The <select> Element with multiple selection

- <select name="cars" size="4" multiple>  
    <option value="volvo">Volvo</option>  
    <option value="saab">Saab</option>  
    <option value="fiat">Fiat</option>  
    <option value="audi">Audi</option>  
  </select>



# The <textarea> Element

- <textarea name="message" rows="10" cols="30">  
The cat was playing in the garden.  
</textarea>



The cat was playing in the  
garden.

- The **rows** attribute specifies the visible number of lines in a text area.
- The **cols** attribute specifies the visible width of a text area.

# HTML Forms – Example

## form.html

```
<form method="post" action="apply-now.php">
 <input name="subject" type="hidden" value="Class" />
 <fieldset><legend>Academic information</legend>
 <label for="degree">Degree</label>
 <select name="degree" id="degree">
 <option value="BA">Bachelor of Art</option>
 <option value="BS">Bachelor of Science</option>
 <option value="MBA" selected="selected">Master of
 Business Administration</option>
 </select>

 <label for="studentid">Student ID</label>
 <input type="password" name="studentid" />
 </fieldset>
 <fieldset><legend>Personal Details</legend>
 <label for="fname">First Name</label>
 <input type="text" name="fname" id="fname" />

 <label for="lname">Last Name</label>
 <input type="text" name="lname" id="lname" />
```

# HTML Forms – Example (2)

## form.html (continued)

```


Gender:
<input name="gender" type="radio" id="gm" value="m" />
<label for="gm">Male</label>
<input name="gender" type="radio" id="gf" value="f" />
<label for="gf">Female</label>

<label for="email">Email</label>
<input type="text" name="email" id="email" />
</fieldset>
<p>
<textarea name="terms" cols="30" rows="4"
readonly="readonly">TERMS AND CONDITIONS...</textarea>
</p>
<p>
<input type="submit" name="submit" value="Send Form" />
<input type="reset" value="Clear Form" />
</p>
</form>
```

# HTML Forms – Example (3)

## form.html (continued)

HTML Forms Example - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Academic information

Degree Master of Business Administration

Student ID

Classes attended

Geography  
Mathematics  
English

Personal Details

First Name

Last Name

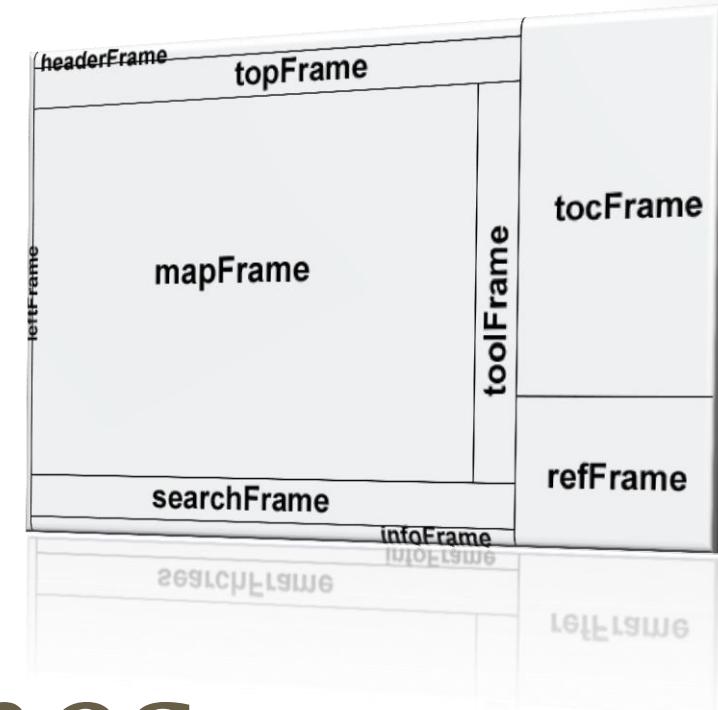
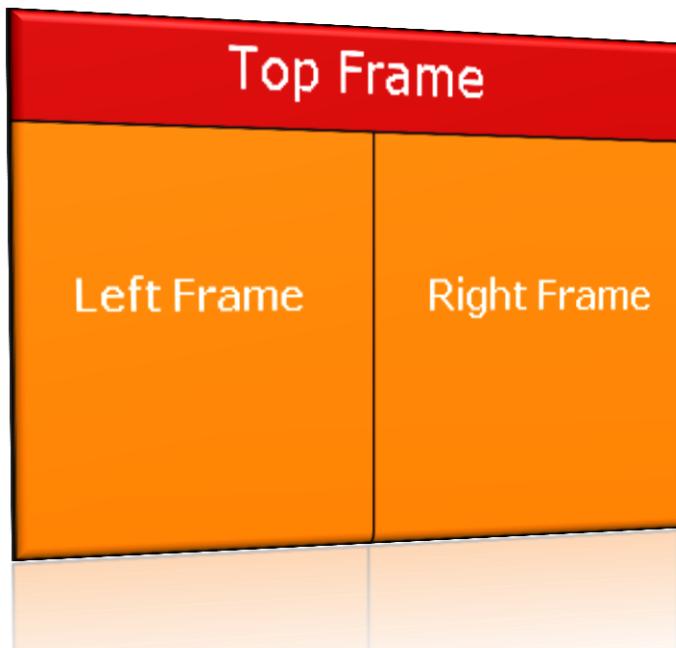
Gender:  Male  Female

Email

TERMS AND CONDITIONS...

Send Form Clear Form

Done Fiddler: Disabled 0 errors / 0 warnings



# HTML Frames

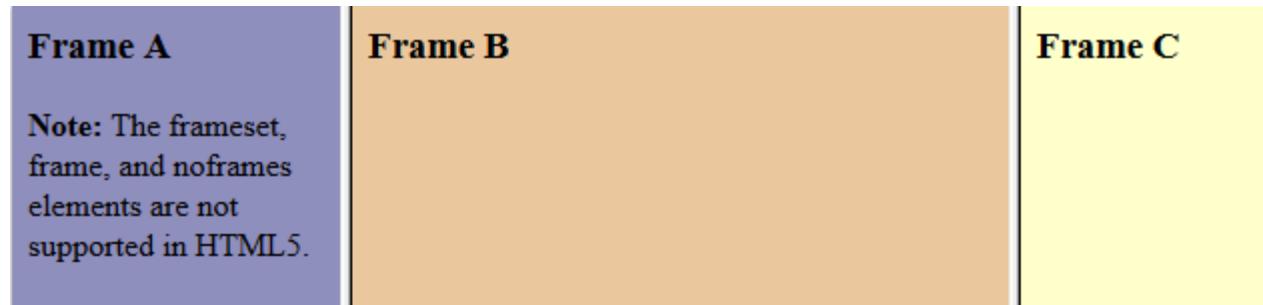
<frameset>, <frame> and <iframe>

# HTML Frames

- Frames provide a way to show multiple HTML documents in a single Web page
- The page can be split into separate views (frames) horizontally and vertically
- Frames were popular in the early ages of HTML development, but now their usage is rejected
- Frames are not supported by all user agents (browsers, search engines, etc.)
  - A <noframes> element is used to provide content for non-compatible agents.

# HTML <frame> Tag.

- **Example**
- A simple three-framed page:
- ```
<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
  <frame src="frame_c.htm">
</frameset>
```



- Each <frame> in a <frameset> can have different attributes, such as border, scrolling, the ability to resize, etc.

Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet,Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component,Transforming XML into XSLT,

DTD: Schema, elements, attributes,

Introduction to JSON.

Html	Html5
<p>Doctype declaration in Html is too longer</p> <pre data-bbox="19 155 952 308"><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/ strict.dtd"></pre>	<p>DOCTYPE declaration in Html5 is very simple "<!DOCTYPE html></p>
<p>character encoding in Html is also longer</p> <pre data-bbox="19 406 952 501"><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"></pre>	<p>character encoding (charset) declaration is also very simple <meta charset="UTF-8"></p>
<p>Audio and Video are not part of HTML4</p>	<p>Audio and Videos are integral part of HTML5 e.g. <audio> and <video> tags.</p>
<p>Vector Graphics is possible with the help of technologies such as VML, Silverlight, Flash etc</p>	<p>Vector graphics is integral part of HTML5 e.g. SVG and canvas</p>
<p>It is almost impossible to get true GeoLocation of user browsing any website especially if it comes to mobile devices.</p>	<p>JS GeoLocation API in HTML5 helps identify location of user browsing any website (provided user allows it)</p>
<p>Html5 use cookies.</p>	<p>It provides local storage in place of cookies.</p>
<p>Not possible to draw shapes like circle, rectangle, triangle.</p>	<p>Using Html5 you can draw shapes like circle, rectangle, triangle.</p>
<p>Does not allow JavaScript to run in browser. JS runs in same thread as browser interface.</p>	<p>Allows JavaScript to run in background. This is possible due to JS Web worker API in HTML5</p>
<p>Works with all old browsers</p>	<p>Supported by all new browser</p>

Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet, Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component,Transforming XML into XSLT,

DTD: Schema, elements, attributes,

Introduction to JSON.



Cascading Style Sheets (CSS)

```
171 #content .article img.left.border {  
172     padding: 0 9px 9px 0;  
173     border-right: 1px dotted #999;  
174     border-bottom: 1px dotted #999; }  
175 #content .article blockquote {  
176     margin-left: 10px;  
177     padding-left: 10px;  
178     border-left: 3px solid #252525; }  
179 #content .article ul {  
180     padding-left: 1em;  
181     list-style-type: circle; }
```

Introduction of CSS

- **Cascading Style Sheets**, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.
- CSS handles the look and feel part of a web page.
- Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

Advantages of CSS

- CSS saves time
- Pages load faster
- Easy maintenance
- Superior styles to HTML
- Multiple Device Compatibility
- Global web standards
- Offline Browsing
- Platform Independence

CSS3 Modules

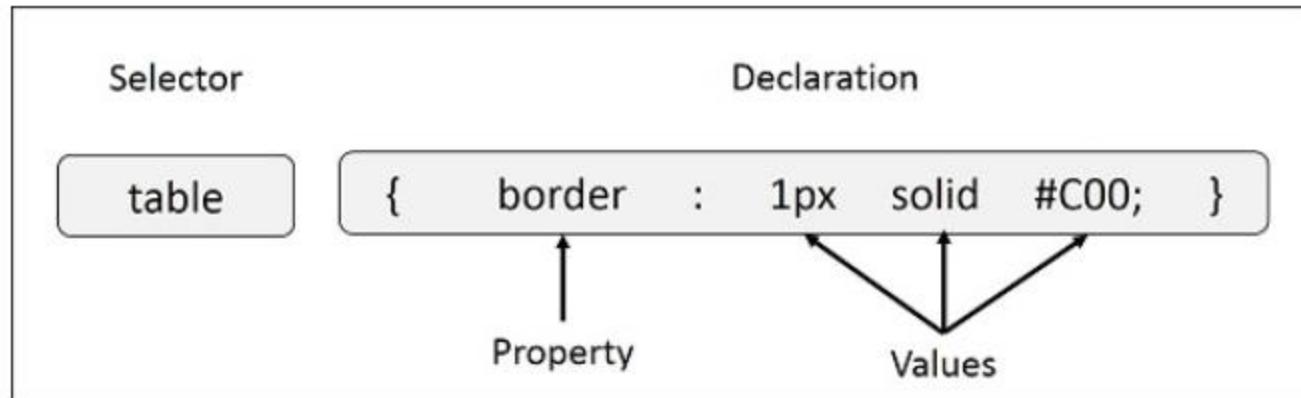
- Selectors
- Box Model
- Backgrounds and Borders
- Image Values and Replaced Content
- Text Effects
- 2D/3D Transformations
- Animations
- Multiple Column Layout
- User Interface

CSS - Syntax

- A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document.
- Style rule is made of three parts –
- **Selector** – A selector is an HTML tag at which a style will be applied. This could be any tag like `<h1>` or `<table>` etc.
- **Property** - A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be *color*, *border* etc.
- **Value** - Values are assigned to properties. For example, *color* property can have value either *red* or `#F1F1F1` etc.

CSS - Syntax

- Syntax:
 - selector { property: value }
 - Example:
 - `table{ border :1px solid #C00; }`



CSS selectors (1)

- CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more.
- **The element Selector**
 - The element selector selects elements based on the element name.
 - You can select all `<p>` elements on a page like this (in this case, all `<p>` elements will be center-aligned, with a red text color):
 - **Example**
 - `p {
 text-align: center;
 color: red;
}`

Example with output

HTML Code with CSS

```
<html>
<head>
<style>
p {
    text-align: center;
    color: red;
}
</style>
</head>
<body>

<p>Every paragraph will be affected by the style.</p>
<p id="para1">Me too!</p>
<p>And me!</p>

</body>
</html>
```

Output

Every paragraph will be affected by the style.

Me too!

And me!

CSS selectors (2)

- **The id Selector**
- The id selector uses the id attribute of an HTML element to select a specific element.
- The id of an element should be unique within a page, so the id selector is used to select one unique element!
- To select an element with a specific id, write a hash (#) character, followed by the id of the element.
- The style rule below will be applied to the HTML element with id="para1":
- **Example**
- ```
#para1 {
 text-align: center;
 color: red;
}
```

# Example with output

```
<!DOCTYPE html>
<html>
<head>
<style>
#para1 {
 text-align: center;
 color: red;
}
</style>
</head>
<body>

<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>

</body>
</html>
```

HTML Code with CSS



Hello World!

This paragraph is not affected by the style.

Output

# CSS selectors (3)

- **The class Selector**
- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the name of the class.
- In the example below, all HTML elements with class="center" will be red and center-aligned:
- **Example**
- ```
.center {  
    text-align: center;  
    color: red;  
}
```

Example with output

HTML Code with CSS

```
<!DOCTYPE html>
<html>
<head>
<style>
.center {
    text-align: center;
    color: red;
}
</style>
</head>
<body>

<h1 class="center">Red and center-aligned heading</h1>
<p class="center">Red and center-aligned paragraph.</p>

</body>
</html>
```

Output

Red and center-aligned heading

Red and center-aligned paragraph.

CSS selectors (4)

- **The class Selector continued....**
- You can also specify that only specific HTML elements should be affected by a class.
- In the example below, only `<p>` elements with `class="center"` will be center-aligned:
- **Example**
- ```
p.center {
 text-align: center;
 color: red;
}
```

# CSS selectors (5)

- **Grouping Selectors**
- If you have elements with the same style definitions, like this:
  - ```
h1 {  
    text-align: center;  
    color: red;}  
  
h2 {  
    text-align: center;  
    color: red; }  
  
p {  
    text-align: center;  
    color: red;}
```
- It will be better to group the selectors, to minimize the code.
 - ```
h1, h2, p {
 text-align: center;
 color: red;
}
```

# Example with output

HTML Code with CSS

```
<!DOCTYPE html>
<html>
<head>
<style>
h1, h2, p {
 text-align: center;
 color: red;
}
</style>
</head>
<body>

<h1>Hello World!</h1>
<h2>Smaller heading!</h2>
<p>This is a paragraph.</p>

</body>
</html>
```

Output

Hello World!  
Smaller heading!  
This is a paragraph.

# Insert CSS in HTML

- **Three Ways to Insert CSS**
  1. External style sheet
  2. Internal style sheet
  3. Inline style

# External Style Sheet

- With an external style sheet, you can change the look of an entire website by changing just one file!
- Each page must include a reference to the external style sheet file inside the `<link>` element. The `<link>` element goes inside the `<head>` section:
  - **Example**
  - `<head>`  
`<link rel="stylesheet" type="text/css" href="mystyle.css">`  
`</head>`

# External Style Sheet - Example

## HTML Code

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

## CSS File named- mystyle.css

```
body {
 background-color: lightblue;
}

h1 {
 color: navy;
 margin-left: 20px;
}
```

# Internal Style Sheet

- An internal style sheet may be used if one single page has a unique style.
- Internal styles are defined within the `<style>` element, inside the `<head>` section of an HTML page:
- **Example**

```
<head>
<style>
body {
 background-color: linen;
}

h1 {
 color: maroon;
 margin-left: 40px;
}
</style>
</head>
```

# Inline Styles

- An inline style may be used to apply a unique style for a single element.
- To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.
- The example below shows how to change the color and the left margin of a `<h1>` element:
- **Example**

```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;margin-left:30px;">This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

# Text-related CSS Properties

- **color** – specifies the color of the text
- **font-size** – size of font: **xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger** or numeric value
- **font-family** – comma separated font names
  - Example: **verdana, sans-serif**, etc.
  - The browser loads the first one that is available
  - There should always be at least one generic font
- **font-weight** can be **normal, bold, bolder, lighter** or a number in range [100 ... 900]

# CSS Rules for Fonts (2)

- **font-style** – styles the font
  - Values: **normal, italic, oblique**
- **text-decoration** – decorates the text
  - Values: **none, underline, line-through, overline, blink**
- **text-align** – defines the alignment of text or other content
  - Values: **left, right, center, justify**

# Shorthand Font Property

- **font**
  - Shorthand rule for setting multiple font properties at the same time

```
font:italic normal bold 12px/16px verdana
```

is equal to writing this:

```
font-style: italic;
font-variant: normal;
font-weight: bold;
font-size: 12px;
line-height: 16px;
font-family: verdana;
```

# Backgrounds

- background-image
  - URL of image to be used as background, e.g.:

```
background-image:url("back.gif");
```

- background-color
  - Using color and image at the same time
- background-repeat
  - repeat-x, repeat-y, repeat, no-repeat
- background-attachment
  - fixed / scroll

# Backgrounds (2)

- **background-position**: specifies vertical and horizontal position of the background image
  - Vertical position: top, center, bottom
  - Horizontal position: left, center, right
  - Both can be specified in percentage or other numerical values
  - Examples:

```
background-position: top left;
```

```
background-position: -5px 50%;
```

# Borders

- border-width: thin, medium, thick or numerical value (e.g. 10px)
- border-color: color alias or RGB value
- border-style: none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
- Each property can be defined separately for left, top, bottom and right
  - border-top-style, border-left-color, ...

# Border Shorthand Property

- border: shorthand rule for setting border properties at once:

```
border: 1px solid red
```

is equal to writing:

```
border-width:1px;
border-color:red;
border-style:solid;
```

- Specify different borders for the sides via shorthand rules:  
border-top, border-left, border-right, border-bottom
- When to avoid border:0

# Width and Height

- **width** – defines numerical value for the width of element,  
e.g. 200px
- **height** – defines numerical value for the height of element,  
e.g. 100px
  - By default the height of an element is defined by its content
  - Inline elements do not apply height, unless you change their **display** style.

# Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet,Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component, Transforming XML into XSLT,

DTD: Schema, elements, attributes,

Introduction to JSON.

# XML

- **Introduction to XML**
  - XML is a software- and hardware-independent tool for storing and transporting data.
- **What is XML?**
  - XML stands for eXtensible Markup Language
  - XML is a markup language much like HTML
  - XML was designed to store and transport data
  - XML was designed to be self-descriptive
  - XML is a W3C Recommendation

# XML Does Not DO Anything

- This note is a note to Tove from Jani, stored as XML::
  - <note>

```
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```
  - The XML above is quite self-descriptive:
    - It has sender information.
    - It has receiver information
    - It has a heading
    - It has a message body.
  - But still, the XML above does not DO anything. XML is just information wrapped in tags.

# The Difference Between XML and HTML

- XML and HTML were designed with different goals:
- XML was designed to carry data - with focus on what data is
- HTML was designed to display data - with focus on how data looks
- XML Does Not Use Predefined Tags like HTML tags

# XML Used For (1)

- It simplifies data sharing
- It simplifies data transport
- It simplifies platform changes
- It simplifies data availability

# XML Used For (2)

- Many computer systems contain data in incompatible formats. Exchanging data between incompatible systems (or upgraded systems) is a time-consuming task for web developers. Large amounts of data must be converted, and incompatible data is often lost.
- XML stores data in plain text format. This provides a software- and hardware-independent way of storing, transporting, and sharing data.
- XML also makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

# XML Example 1

- <?xml version="1.0" encoding="UTF-8"?>  
<note>  
    <to>Amit</to>  
    <from>Neha</from>  
    <heading>Reminder</heading>  
    <body>Don't forget me this weekend!</body>  
</note>
- Save the file with .xml extension and when run o/p is like below

```
<note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>
```

# XML Example 2- Books.xml

- <?xml version="1.0" encoding="UTF-8"?>  
<bookstore>  
  
    <book category="children">  
        <title lang="en">Harry Potter</title>  
        <author>J K. Rowling</author>  
        <year>2005</year>  
        <price>29.99</price>  
    </book>  
  
    <book category="web" cover="paperback">  
        <title lang="en">Learning XML</title>  
        <author>Erik T. Ray</author>  
        <year>2003</year>  
        <price>39.95</price>  
    </book>  
  
  </bookstore>

# XML Example 2- Books.xml explanation

- XML uses a much self-describing syntax.
- A prolog defines the XML version and the character encoding:
- <?xml version="1.0" encoding="UTF-8"?>
- The next line is the **root element** of the document:
- <bookstore>
- The next line starts a <book> element:
- <book category="cooking">
- The <book> elements have **4 child elements**: <title>, <author>, <year>, <price>.
- The next line ends the book element:
- </book>

# XSLT

- XSLT (eXtensible Stylesheet Language Transformations) is the recommended style sheet language for XML.
- XSLT is far more sophisticated than CSS.
- With XSLT you can add/remove elements and attributes to or from the output file.
- You can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more.
- XSLT uses XPath to find information in an XML document.

# Displaying XML with XSLT

Create XML Page

Create XSLT Page according to your design criteria

Link XML page with XSLT Page

Get output on browser

# Step 1 : Create XML Document: students.xml

```
<?xml version = "1.0"?>
<class>
 <student rollno = "393">
 <firstname>Dinkar</firstname>
 <lastname>Kad</lastname>
 </student>
 <student rollno = "493">
 <firstname>Vaneet</firstname>
 <lastname>Gupta</lastname>
 </student>
</class>
```

# Step 2: XSLT Conversion criteria

- We need to define an XSLT style sheet document for the above XML document to meet the following criteria –
- Page should have a title **Students**.
- Page should have a table of student details.
- Columns should have following headers:
  - Roll No, First Name, Last Name
- Table must contain details of the students accordingly.

# Step 2: Create XSLT document according to design criteria: students.xsl

```
<?xml version = "1.0" encoding = "UTF-8"?>
<xsl:stylesheet version = "1.0" xmlns:xsl = "http://www.w3.org/1999/XSL/Transform">
 <xsl:template match = "/">
 <html> <body>
 <h2>Students</h2>

 <table border = "1">
 <tr bgcolor = "#9acd32">
 <th>Roll No</th>
 <th>First Name</th>
 <th>Last Name</th>
 </tr>

 <xsl:for-each select="class/student">
 <tr>
 <td><xsl:value-of select = "@rollno"/></td>
 <td><xsl:value-of select = "firstname"/></td>
 <td><xsl:value-of select = "lastname"/></td>
 </tr>
 </xsl:for-each> </table>
 </body> </html>
 </xsl:template>
 </xsl:stylesheet>
```

# Step 3: Link the XSLT Document to the XML Document

- <?xml version = "1.0"?>
- **<?xmlstylesheet type = "text/xsl" href = "students.xsl"?>**
- <class>
- ...
- </class>

# Step 4: View the XML Document in Internet Explorer

```
<?xml version = "1.0"?>
<?xml-stylesheet type = "text/xsl" href = "students.xsl"?>
<class>
 <student rollno = "393">
 <firstname>Dinkar</firstname>
 <lastname>Kad</lastname>
 </student>
 <student rollno = "493">
 <firstname>Vaneet</firstname>
 <lastname>Gupta</lastname>
 </student>
</class>
```

# Output

## Students

Roll No	First Name	Last Name
393	Dinkar	Kad
493	Vaneet	Gupta

# Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet,Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component,Transforming XML into XSLT,

**DTD: Schema, elements, attributes,**

Introduction to JSON.

# Introduction to DTD

- A DTD is a Document Type Definition.
- A DTD defines the structure and the legal elements and attributes of an XML document.
- With a DTD, independent groups of people can agree on a standard DTD for interchanging data.
- An application can use a DTD to verify that XML data is valid.

# An Internal DTD Example

- <?xml version="1.0"?>  
  <!DOCTYPE note [  
    <!ELEMENT note (to,from,heading,body)>  
    <!ELEMENT to (#PCDATA)>  
    <!ELEMENT from (#PCDATA)>  
    <!ELEMENT heading (#PCDATA)>  
    <!ELEMENT body (#PCDATA)>  
  ]>  
  <note>  
    <to>Tove</to>  
    <from>Jani</from>  
    <heading>Reminder</heading>  
    <body>Don't forget me this weekend</body>  
  </note>

# An Internal DTD Explanation

- The DTD in previous slide is interpreted like this:
- **!DOCTYPE note** defines that the root element of this document is note
- **!ELEMENT note** defines that the note element must contain four elements: "to,from,heading,body"
- **!ELEMENT to** defines the to element to be of type "#PCDATA"
- **!ELEMENT from** defines the from element to be of type "#PCDATA"
- **!ELEMENT heading** defines the heading element to be of type "#PCDATA"
- **!ELEMENT body** defines the body element to be of type "#PCDATA"

# An External DTD Example

- XML File
  - <?xml version="1.0"?>  
<!DOCTYPE note SYSTEM "note.dtd">  
<note>  
    <to>Tove</to>  
    <from>Jani</from>  
    <heading>Reminder</heading>  
    <body>Don't forget me this  
weekend!</body>  
</note>
- note.dtd
  - <!ELEMENT note  
(to,from,heading,body)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT heading  
(#PCDATA)>  
<!ELEMENT body (#PCDATA)>

# Building Blocks of XML Documents as per DTD

- **Elements :** <body>some text</body>
- **Attributes :** 
- **Entities :** &lt; &amp;
- **PCDATA :** PCDATA means parsed character data.
  - **PCDATA is text that WILL be parsed by a parser.**  
Character data is the text found between the start tag and the end tag of an XML element.
- **CDATA:** CDATA means character data.
  - **CDATA is text that will NOT be parsed by a parser.**

# Elements

- XML elements can be defined as building blocks of an XML document.
  - Elements can behave as a container to hold text, elements, attributes, media objects or mix of all.
  - Each XML document contains one or more elements, the boundaries of which are either delimited by start-tags and end-tags, or empty elements.
- 
- **Example**
  - **<name>Tutorials Point</name>**

# Attributes

- Attributes are part of the XML elements.
  - An element can have any number of unique attributes.
  - Attributes give more information about the XML element or more precisely it defines a property of the element.
  - An XML attribute is always a *name-value* pair.
- 
- **Example**
  - ****
  - Here *img* is the element name whereas *src* is an attribute name and *flower.jpg* is a value given for the attribute *src*.

# Entities

- Entities are placeholders in XML. These can be declared in the document prolog or in a DTD. Entities can be primarily categorized as:
  - Built-in entities
  - Character entities
  - General entities
  - Parameter entities
- There are five built-in entities that play in well-formed XML, they are:
  - ampersand: &amp;
  - Single quote: &apos;
  - Greater than: &gt;
  - Less than: &lt;
  - Double quote: &quot;

# Advantages & Disadvantages

- **Advantages of using DTD**
  - **Documentation** - You can define your own format for the XML files.
  - **Validation** - It gives a way to check the validity of XML files by checking whether the elements appear in the right order, mandatory elements and attributes are in place
- **Disadvantages of using DTD**
  - It does not support the namespaces.
  - It supports only the *text string data type*.
  - It is not object oriented.

# Outline

Introduction to web technology, internet and www, Web site planning and design issues,

HTML: structure of html document,HTML elements: headings, paragraphs, line break, colors & fonts, links, frames, lists, tables, images and forms,

Difference between HTML and HTML5.

CSS: Introduction to Style Sheet,Inserting CSS in an HTML page, CSS selectors,

XML: Introduction to XML, XML key component,Transforming XML into XSLT,

DTD: Schema, elements, attributes,

Introduction to JSON.

# JSON-Introduction

- JSON: **JavaScript Object Notation.**
- JSON is a syntax for storing and exchanging data.
- JSON is text, written with JavaScript object notation.
- JSON is language independent \*
- **Why use JSON?**
- Since the JSON format is text only, it can easily be sent to and from a server, and used as a data format by any programming language.
- JavaScript has a built in function to convert a string, written in JSON format, into native JavaScript objects:
- `JSON.parse()`
- So, if you receive data from a server, in JSON format, you can use it like any other JavaScript object.

# JSON Syntax Rules

- JSON syntax is derived from JavaScript object notation syntax:
- Data is in name/value pairs
- Data is separated by commas
- Curly braces hold objects
- Square brackets hold arrays
- **JSON Data - A Name and a Value**
  - JSON data is written as name/value pairs.
  - A name/value pair consists of a field name (in double quotes), followed by a colon, followed by a value:
- **Example**
  - "name":"John"

# JSON Values

- In JSON, *values* must be one of the following data types:
  - a string
  - a number
  - an object (JSON object)
  - an array
  - a boolean
  - null
- In JSON, *string values* must be written with double quotes:
  - { "name":"John" }

# Object Syntax

- **Example**
- { "name":"John", "age":30, "car":null }
- JSON objects are surrounded by curly braces {}.
- JSON objects are written in key/value pairs.
- Keys must be strings, and values must be a valid JSON data type (string, number, object, array, boolean or null).
- Keys and values are separated by a colon.
- Each key/value pair is separated by a comma.

# References

- <http://study.com/academy/lesson/what-is-web-technology-definition-trends.html>
- [https://www.tutorialspoint.com/web developers guide/web basic concepts.htm](https://www.tutorialspoint.com/web_developers_guide/web_basic_concepts.htm)
- <https://www.slideshare.net/vikramsingh.v85/introduction-to-web-technology>
- [www.telerik.com](http://www.telerik.com)
- <https://www.w3schools.com/html/>
- [https://www.w3schools.com/css/css intro.asp](https://www.w3schools.com/css/css_intro.asp)
- <https://www.w3schools.com/xml/>
- [https://www.w3schools.com/js/json intro.asp](https://www.w3schools.com/js/js_json_intro.asp)
- [https://www.w3schools.com/xml/xml dtd intro.asp](https://www.w3schools.com/xml/xml_dtd_intro.asp)