MC404 WEB TECHNOLOGY 3 0 0 100

UNIT I

Web Publishing: A Melding of Technologies – Setting up an Extensible Web Publishing Frame/Work. The Web Publishing Foundation: The function of HTML in contemporary Web Publishing – Basic Structural Elements and their usage – Traditional text and formatting – Style Sheets Formatting for the future – Using tables for Organization and layouts – Advanced layout and Positioning with style sheets – Creating forms with HTML – Frames and Frame sets – using Images with HTML – Merging Multimedia, Controls and plug – Ins with HTML.

UNIT II

Client – Side Scripting: Scripting basics – Client –Side Image Maps – Introduction Java Script – Creating simple Java Scripts – Using Java script for forms – using Java Script with Style Sheets.

UNIT III

Web Publishing With Java: WHY Java – The Java Language - Introduction to Applet Programming Java Beans – JARS and Safe Computing – Integrating Java and Java Script.

UNIT IV

CGI and Controlling the Web from the Server: Putting your server to work – Traditional CGI programming – The Anatomy of a CGI Application Server – Specific Technologies Netscape ONE Vs Microsoft Windows DNA – Serious Applications for serious Web Publishing – Server – Independent Technologies – The JAVA Servlet API. 9

UNIT V

Engineering A Web Set: Using the HTML Object Model and Creating Dynamic HTML Pages – Manipulating Objects and Responding to user Interaction – Saving User Preferences: Cookies and Ops. Emerging and Alternate Web Technologies: Active-X controls for the www-XML.

REFERENCES

1. Shelley Powers et.al. “Dynamic Web Publishing”, Tech Media, 1998.

2. Achyat.S.Godbole and Atul Kahate, “Web Technologies”, Tata McGraw Hill Pub. Co., Delhi, 2003.

UNIT I

CONTENTS

1. A Melding of Technologies

About the Internet

A network of network is called internet. The internet is used to perform the following types of tasks:

- Electronic Mail
- Research
- Discussion
- Interactive Games

A short history of HTML

- The secret of the internet is the separation of the act of transmission of data from the display of data. By separating the processes, transmission could be greatly speeded because the server did not have to worry about how the data looked at the other end, since the display was left to the local receiving computer.
- The information alongside the data (called tags) tells the receiving computer how to display different types of information. These coding methods, called markup languages, travel with data and are interpreted by the retrieving software.
- Codes to tell the computer how to interpret hypertext documents are called Hypertext Markup Language.
- HTML became the standard way to tag pages of information traveling over the internet.

Browsers and the Web

- The web is simply a way of looking at the internet. This vast network consists of computers that manage data and the communication links via software and hardware called server.
- Web servers receive requests for information, go out and find it in t heir databases, and return the proper pages of data to the requesting computer. Browsers assisted users in finding information on the web and properly displaying it on computer screens.
- Hypertext Transfer Protocol (HTTP) was developed as a way to find information and retrieve it over telephone lines.

Component Software and the Web

- Components are small programs that can be fit together to form a modular, customized application in real time. Components can be written in cross platform languages and rapidly complied to work on many platforms.
- Microsoft depends on the Distributed Component Object Model (DCOM) whereas Netscape, Apple, IBM and Sun have agreed upon the Common Object Request Broker Architecture (CORBA).

Java and JavaScript

- The hottest thing to hit the web is the programming language Java, which Sun Microsystems introduced to make programs portable.
- Netscape created a lighter version of Java, which lets users customize their browsers and create a more interactive environment on the web. This capability is called Web-delivered scripting, which enables you to execute programs within a browser.
- The scripting language based on Java is called JavaScript. Java and JavaScript provide web pages with the capability to interact with the user.

ActiveX

ActiveX is a set of technologies that lets you create more interactive web pages. ActiveX technologies provide support for programs on both the client and server sides. There are several ActiveX components used to perform different functions.

- ActiveX Controls – These are applets that function as objects you can embed in your HTML that provide interactive and user controllable functions.
- ActiveX Documents – Let you view non-HTML documents through a web browser.
- Active Scripting – Lets you integrate the functions of several ActiveX controls or Java applets from the browser or server.
- ActiveX Server Framework – Provides web server functions such as security, database access, and others to web pages.

Multimedia Tools

We are standing on the edge of the ability to broadcast on-demand video and video over the web, such as live event. In addition, working in three dimensions in real time motion (Called virtual reality) opens up new vistas.

Databases for Multimedia

- Multimedia databases such as Illustra Information Technologies automatically open information sent via email and store images, captions and text in separate fields.
- Because of growing size and number of HTML pages, and audio/video files that make up a site, you need databases to manage the storage and use of multimedia items.
- There four types of databases available for managing web sites: traditional relational databases such as Oracle; object oriented databases such as Objectstore; relational-object hybrid databases such as Illustra; and specialty databases such as Cinebase.
- Oracle now supports multimedia storage via its new Universal Server Enterprise Edition, which includes Oracle Spatial Data option for storage and retrieval of images, the ConText option for full text retrieval through SQL, and Oracle Video option, which serves video files to multiple clients.
- Object oriented databases are useful for storing images, video and audio because they use an object model rather than a tabular model in t he database design. An object can be anything.
- Hybrid mixtures of object-oriented and relational databases use the benefits of tabular data and object-based data.
- Specialty databases are built for single purpose, such as Media-On-Demand.

Audio Broadcasting

- One of the exciting new technologies coming out of multimedia developers workshops is the browser capability to play sound data as it is received. This technology is based on a data type called MIME, where the computer sends the data packet first so that the browser can interpret information as it comes screaming down the wire.

- The problem today is that telecommunications lines are not screaming, but sputtering - there is no such thing as a continuous, error-free flow of data. So you have to install a plug-in that buffers some of the information so that lost or delayed data can be accommodated.
- Netscape provide the following plug-in players.
    - Crescendo Plus by LiveUpdate.
    - InterVU MPEG Player by Intervu, Inc.
    - Koan by Sseyo.
    - Maczilla by knowledge engineering.
    - MidPlug by Yamaha.
    - RealAudio by Progressive networks.
    - StreamWorks by Xing Technology.

Digital Video

- There are two video compression formats vying for power on the web: QuickTime and MPEG.
- Digital video does not have the picture quality of broadcast video because personal computers cannot support the bandwidth required to transmit all the data involved.
- Digital video is also called animation because the images that comprise a moving picture are sent one-by-one in a single pass without interlacing and at half the number of frames per second as broadcast video.
- Netscape supports the following plug-ins for viewing and creating digital video animations:
    - Action by Open2U.
    - CineWeb by Digigami.
    - CoolFusion by Iterated Data Systems.
    - InterVU by InterVu, Inc.
    - MovieStar by Intelligence at large.
    - QuickTime for Netscape by Apple Computer.
    - ViewMovie by Ivan Cavero Belaunde,

Helper Applications

- A helper application is basically a program that cooperates with Netscape and other browsers to perform functions such as displaying the contents of files, which the browsers cannot do.
- Some of the helper applications are:

  Movie Viewing – AVI Video player for windows

  MPEG Animation viewing – Ghostscript

Electronic Publishing

- It is one thing to HTML to send pictures and text over telephone wires to remote computers where a browser interprets arcane codes to display a replica of what was broadcast.
- It is quite another to be able to reproduce published documents with their complex layouts, colors and art on a remote computer. The remote computer may not have the fonts installed that were used to produce the document. What is needed is a way to make documents electronically portable.

2. Setting up an Extensible Web Publishing Framework

- The extensible means that the specification has to hold and support all of the new technologies driving the web.

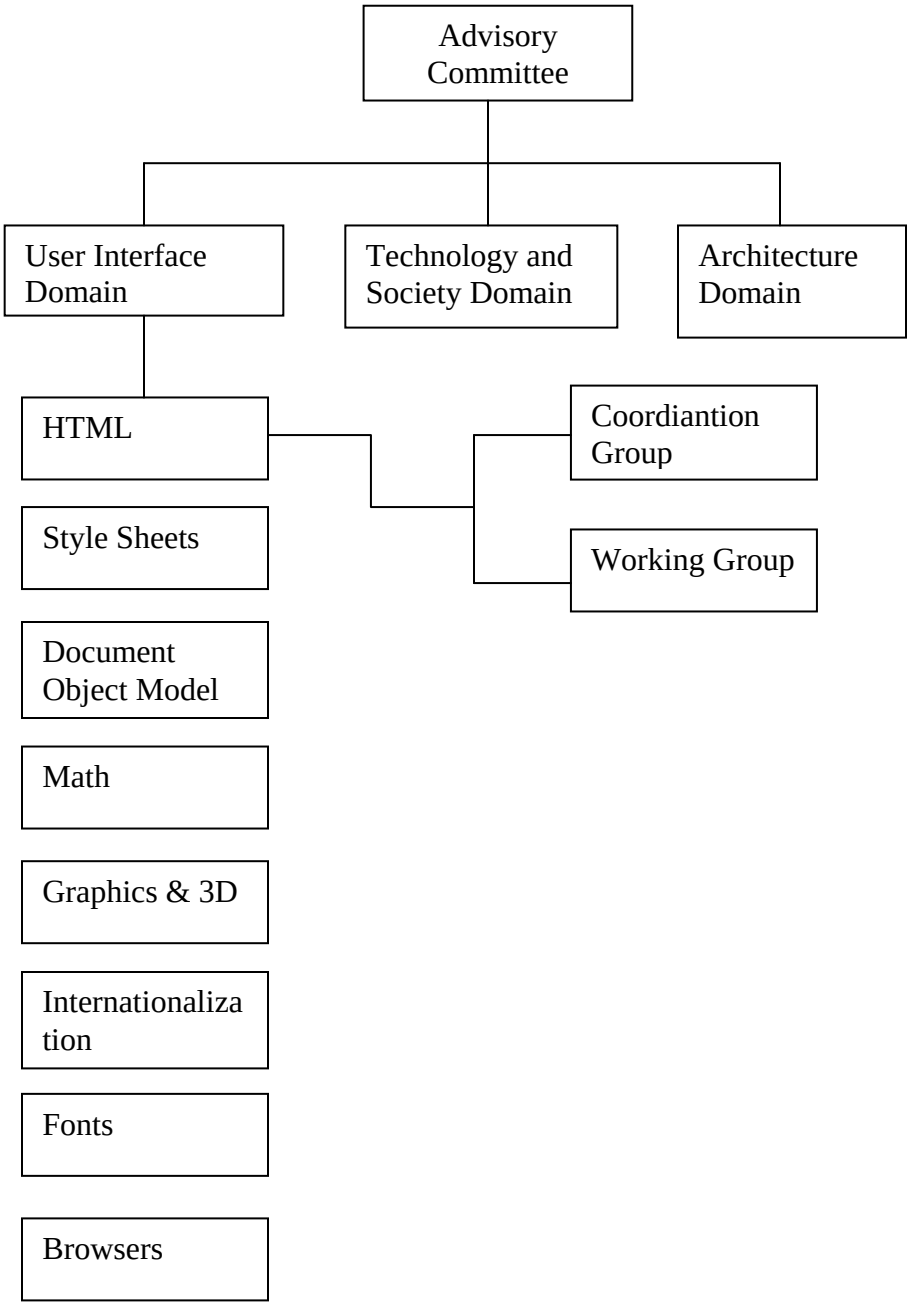
What is the World Wide Web Consortium (W3C)?

- W3C was founded to manage the evolution of the World Wide Web and its design components.
- W3C was founded by combination of hardware, software, and content providers to ensure that the internet remains open, meaning that the web continues to be interoperable and has not become the province of one major provider.
- There are currently 180 commercial and academic members worldwide. W3C members include representatives from hardware and software vendors, telecommunication providers, content developers and government and academic users of the internet.

How is the W3C Organized?

- The W3C is organized around three domains: User Interface, Technology and Society, and Architecture.
- The user interface group works on issues dealing with web content and design.
- The technology and society group works on issues dealing with security, privacy, copyrights, and intellectual property rights.
- The architecture group works on issues dealing with infrastructure.

Organization chart for W3C



What is the Html Working Group?

- The Html working group is the current organization within W3Cresponsible for developing Html specifications and standards.

A quick review of Markup Concepts

- The goal of markup languages is to create a universal way to identify the structure and content of a document.
- SGML is the earliest specification for markup. SGML is based upon the use of a special file called a Document Type Description (DTD)document. The DTD describes the structure of a document.
- The document's basic framework is defined, including types of elements used and how these elements are related to each other.
- The contents of a document are labeled with tags that identify the document's structure. Each tag has a beginning and ending.
- HTML was developed as a subset of SGML for handling hypertext link.

What is HTML 4.0?

- Html 4.0 corrects design deficiencies produced by Html 3.2, especially in the support of forms and tables.

Changes between Html 3.2 and Html 4.0

1. New elements and attributes - Q, INS, BUTTON, LABEL, FIELDSET
2. Modified elements and attributes - Table frame and rules attributes
3. Deprecated Elements - applet, center, font, basefont, u
4. Obsolete Elements - XMP, PlainText

Design Concepts Underlying Html 4.0

- HTML 4.0 is an extensible web framework because it allows the specification to grow with the times.
- The working group used the following concepts to ensure that their specifications met that goal:

- Interoperability
- Internationalization
- Accessibility
- Enhanced tables

- Style sheets
- Scripting – enabled
- Maintain html's ease of use

3. The Function of HTML in Contemporary Web Publishing

- HTML is a subset of SGML, Standard Generalized Markup Language.
- SGML is itself a rigorous format meant for designing other markup languages.
- The basic concept behind HTML or any other markup languages is to embed commands that describe how different elements within a document should be interpreted in that document.

The Web's Defining Framework

- HTML provides Web designers both a matrix and a mechanism.
- As a matrix, its purpose is to the basic framework in which content is embedded.
- As a mechanism, it provides ways to specify the manner in which that content is presented and, to a limited extent, allows various options for its functioning.
- Paired with newer developments of scripting languages and style sheets, the true potential of HTML seems about to be fully realized, and the World Wide Web to really come into flower.

Tags and Elements in HTML

- Tags are the main resources for HTML authors. They define the existence of all the elements a web page contains. Indeed, they define the existence of the page itself. Each element is said to be contained by tags.
- When a web browser or other client agent parses the HTML code in a web page, it looks for these defining indicators to tell it how to process and display the elements.
- The code structure of elements is a start tag, followed by the contents, if any, and the end tag.
- Most elements have required start and end tags, the function of which is to delimit they beginning and ending of the element.
- Other elements have required start tags, but optional end tags. Still other elements have only a start tag and no end tag. Each element has its own attributes.

- The start of an HTML page is defined by the <HTML> start tag and the end is defined by </HTML> tag. The HEAD and BODY elements are within that HTML element.
- The material between <HEAD> </HEAD> tags is largely informational, although this is also where JavaScript and style-connection information are placed. The BODY element contains the majority of the other elements.
- The elements contained within a Web Page's body tags are many and various.
- They include elements for controlling the insertion of images and sounds, tying in Java applets and other objects, setting the appearance and size of text, adding tables and forms.

4. Basic Structural Elements and Their Usage

- There are four basic structural elements in HTML.
- The <!DOCTYPE> tag represents the Document Type Declaration (DTD), which defines the version of HTML used on the page.
- The next is the <HTML> tag itself, which simply states that the page is an HTML document.
- The third is the HEAD element, which contains the document's title and other information.
- Then there is the BODY element, where the actual web page itself is contained.

A basic HTML page:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 // EN">
<HTML>
<HEAD>
<TITLE> PAGE TITLE GOES HERE </TITLE>
OPTIONAL META DATA GOES HERE..
</HEAD>
<BODY>
WEB PAGE GOES HERE…….
>/BODY>
</HTML>

The <DOCTYPE> element

- Its purpose is to declare to browsers exactly what version of HTML was used to create the document. The general DOCTYPE declaration for HTML 4.0 is
  <! DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 // EN">
- The reason for the EN is that the HTML specification has not yet been developed in any language but English.
- If has not yet been developed in any language but English.
- If you are replacing the BODY element with FRAMESET, you should use the following DTD.
  <! DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 FRAMESET // EN">
- There is another technique for supplying the DOCTYPE declaration. This is called the System Identifier DTD. A System Identifier DTD is declared as follows:
  <! DOCTYPE
  HTML
  SYSTEM
  "http:// WWW.W3.Org/DTD/HTML 4 - strict.dtd">
- The difference in this approach is that the DTD is referenced not by name, but by specifying the URL where it is found.

The HTML Element

- It is the first and last thing in a web page, and its absence means that no web browser recognizes your work as an HTML page.
- HTML element must have both start and end tags in order to function properly. Thus, the document begins with <HTML> and ends with </HTML>.
- The HTML start tag can have three attributes. The first is version, and it takes a URL value. The URL points to a location that has the Document Type Definition (DTD) for the version of HTML in use on that page.
- The HTML start tag can also contain the lang and dir attributes, which respectively, establish the human language in which the web page is written and the direction of the printing.

- <HTML> tag might look like:

  <HTML version = http://www.w3.org/DTD/HTML4- strict.dtd lang=en dir=rtl>

The HEAD Element

- The HEAD element is a container for an HTML documents' header information. It contains one required element – TITLE and several other optional attributes.
- The HEAD element has both start and end tags, beginning with <HEAD> and ending with </HEAD>. A typical HEAD element look like this:

  <HEAD>

  <TITLE> My Home Page </TITLE>

  </HEAD>
- The text between the TITLE tags is displayed in the title bar of a visitor's web browser.

MetaData

- In addition to the title, a head element can contain other elements known as metadata.
- It is data that is not shown to the person viewing the page, but useful to user agents and search engines.
- The currently used elements in this category are:

  Base

  Link

  Meta

  Script

  Style

The Base Element

- The base element establishes a base url from which relative URLs referenced in the html document can be calculated. Example,

  <base href=http://www.test.org/>
- The result of this declaration is that any relative url in the html document is appended to this base url to achieve he full url.
- Thus, the relative url index.html would be interpreted as http://www.test.org/index.html.

The Link Element

- Its purpose is to establish relationships among different html documents. Used only in the head element.
- Link takes three main attributes in addition to href: rel, rev and title.
- Rel defines a forward relationship and rev defines a reverse one.
- Example,

  <link rev=”previous” href=”chapter1.html”

  <link rel=”next” href=”chapter3.html”>
- The title attribute siply gives a title to the document referenced by the url.
- Link types are:

  Alternate - Points to an optional replacement for the current document

  Appendix - points to an appendix

  Bookmark - Points to a named anchor

  Chapter – Points to a chapter

  Copyright - Points to a copyright statement

  Glossary - Points to s glossary of terms

  Help - Points to a help document

  Index - Points to an index

  Next - Points to the document that comes after the current one

  Previous - Points to the document that comes before the current one

  Start - Points to the beginning document

The Meta Element

- When used with the name and content attributes, its main function is to establish metadata variable information for use by web search agents.
- For example,

  <meta name=”wt” content=”web technology”>

The script and style elements

- Used for including script and style sheet into an html document.

The body element

- The body element is where the displayable web page is found.
- A typical body in an html document looks like:

```
<body background=”bg.gif”text=”000000” link=”0000FF”vlink=”FF8C00” alink=”000000”>
        Text, images, etc.  are found here…
</body>
```

5. Traditional Text and Formatting

Deprecated and Obsolete Elements

- As HTML evolves, the function of some elements is replaced or suppressed by newer.
- Some are still fully functional and useful, but can be done more economically or efficiently.
- For example, both the <IMG> and <APPLET> tags still work, but both perform similar tasks – embedding particular object in the HTML page.
- A single <OBJECT> element would be designed that would encompass all possible embeddable objects.
- Thus, both <IMG> and <APPLET> are now deprecated in favor of <OBJECT> tag.
- When an element is deprecated, that means that the W3C recommends that you no longer use it, but use, the newer solution.
- However, deprecated elements are still a part of HTML specification, and still be supported by browsers.
- Obsolete elements, are no longer defined in the HTML specification, and W3C does not require that client agents support them.

Text Layout

- There are 2 basic types of text affecting elements in html.
- The first kind performs text layout tasks and the second affects text’s appearance.

The P Element

- The <p> tag is used to denote the beginning of a new paragraph.
- Although the end tag </P> exists, its use is optional.
- If the end tag is not used, the beginning of the next block level element is interpreted as the end of the paragraph.

- The align attribute is used to set the alignment of the paragraph with respect to the page size. Values are LEFT, RIGHT, CENTER and JUSTIFY. Example: <p align="center">

<P> tag Example

```
<P align=left>
        This paragraph is Left Aligned.
</P>
<P align=center>
        This paragraph is Centered.
</P>
<P align=right>
        This paragraph is Right Aligned.
</P>
<P align=justify>
        This paragraph is justified.
</P>
```

The BR Element

- The
 tag inserts a single line break.
- The
 tag is an empty tag which means that it has no end tag.
- When placed after images, the clear attribute controls how text is handled when wrapping around those images.
- The clear attribute has four possible values: none, left, right and all.

The CENTER Element

- The CENTER element causes all text between its start and end tags to be centered between the margins.
- <CENTER> Text Goes Here. </CENTER>

The HN Element

- The highest level of heading is represented by <H1> tag, the lowest by the <H6> tag.

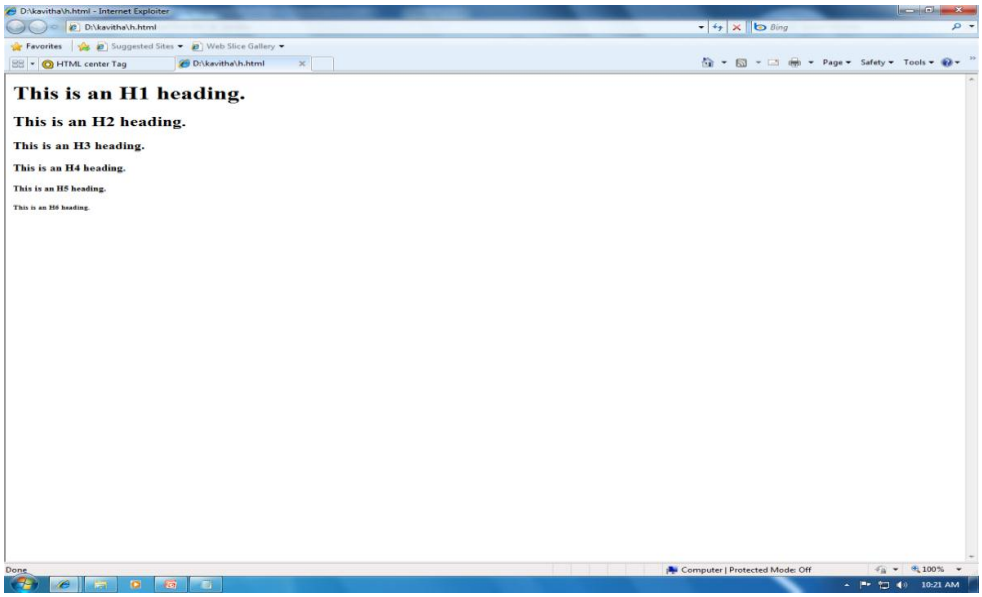
```
<H1> This is an H1 heading. </H1>
<H2> This is an H2 heading. </H2>
```

<H3> This is an H3 heading. </H3>

<H4> This is an H4 heading. </H4>

<H5> This is an H5 heading. </H5>

<H6> This is an H6 heading. </H6>



The HR Element

- Horizontal rules are used to visually divide different segments of web pages from one another.
- A simple horizontal rule is coded as follows:

<HR>

Lists

- One of the most popular methods for organizing information is by using lists.
- HTML presents three basic kinds of lists: unordered lists, ordered lists, and definition lists.
- In unordered lists, the list items are marked with bullets.
- In ordered lists, they are marked with numbers, Roman numerals, or letters.
- Definition lists are a little different; they have a pair of values, one for the term, the other for its definition.

Unordered Lists

- Unordered lists are specified with the <UL> tag.
- Unordered lists are used when the order of the list items is unimportant.
- The type attribute defines the type of bullets used to denote the individual list items.
- The three options are: disc, circle and square.

Example

```
<UL type="disc">
<LH> UG Courses </LH>
<LI> BE (CSE) </LI>
<LI> BE (ECE) </LI>
<LI> BE (EEE) </LI>
</UL>
<UL type="square">
<LH> PG Courses </LH>
<LI> MCA </LI>
<LI> ME (CSE) </LI>
<LI> MBA  </LI>
</UL>
```

Ordered Lists

- Ordered lists are specified with the <OL> tag.
- They are used when the order of the list item is significant.
- OL elements have the type and start attributes.
- The type attribute selects the kinds of numbering system utilized to order the list.

Example

```
<html>
<body>
<h4>An Ordered List:</h4>
<ol type=I>
  <li>Coffee</li>
  <li>Tea</li>
```

```
  <li>Milk</li>
</ol>
</body>
</html>
```

Definition Lists

- Definition lists are specified with the <DL> tag.
- Definition lists consist of pairs of values, the first being the term to be defined, and the second being the definition of the tem.
- Example

```
<DL>
<DT>  Satellite Dish
<DD>  Antenna like device which functions to receive and concentrate television
      signals.
```

Nested Lists

- Any kind of list – ordered, unordered or definition – can be nested within another list.

```
<UL type=disc>
<LI> UG COurses </LI>
<OL type=i>
<LI> BE (CSE) </LI>
<LI> BE (ECE) </LI>
<LI> BE (EEE) </LI>
</OL>
<LI> PG COurses </LI>
<OL type=i>
<LI> ME (CSE) </LI>
<LI> MCA </LI>
<LI> MBA </LI>
</OL>
</UL>
```

Text Styles

The B and STRONG Elements

- Using the <B> and <STRONG> tags has the effect of rendering text in bold print.

The I and EM elements

- Using the <I> and <EM> tags has the effect of rendering text in italicized print.

STRIKE and U Elements

- The STRIKE element causes text to be struck through.
- The U element underlines the affected text.

The BIG, SMALL, SUP and SUB elements

- These four elements actually change the size or position of the affected text.

The FONT and BASEFONT Elements

- The FONT and BASEFONT Elements perform the same task and use the same methods for doing so.
- The difference between them is the scope of their effect.
- Both set the size, color and font face for the text, but the basefont element is global for all body text in the document, whereas the FONT element is strictly local and affects only the text between its start and end tags.
- Example

  <BASEFONT size=4 color="#000000" face="arial">

  <P>

  This is body text using the base font size.

  <P>

  <FONT size=+3> This is locally increased font. </FONT>

6. Style Sheets : Formatting for the future

CSS1 standard

- Style sheets are embedded directly into a page, or linked in from an external file, and provide web page developers the ability to redefine the appearance of all elements of a certain type, or one specific element.
- The following code is a simple style sheet, embedded into the head section of the web page.

```
<STYLE type="text/css">
        H1  { color: red;font-family: Arial; margin: 1.0in }
</STYLE>
```

- With this style sheet setting, all H1 elements within the document are set to use the new style.

Including Style Sheets

- There are 4 different techniques to Include style sheet definitions
  1. Including a style sheet in the document's head section.
  2. Linking in a style sheet stored externally.
  3. Importing in a style sheet stored externally.
  4. Including an inline style sheet definition directly in an html element.
- The most common approach is to embed style sheets into the head of the web page and provide settings for all of the elements within the web page.
- Example

```
<STYLE type="text/css">
        BODY { background-color: aqua; color: firebrick;
                margin: 0.5 in}
</STYLE>
```

- A second technique is to link the style sheet into the web page using the link syntax:

```
<HEAD> <TITLE> CSS Test Content </TITLE>
<LINK rel=stylesheet type="text/css" href=style.css>
</HEAD>
```

- Another approach to incorporate style sheets into a page is to import the style sheet into the file.
- The code to import style sheet:

```
<STYLE type="text/css">
@import url(style.css);
</STYLE>
```

- A last technique to include style sheet settings is to embed them directly into an element.

- The syntax for this is:

  <P style="color: yellow; font-style: italic">

Applying styles to Specific Groups of Elements

Using the class name style sheet selector

- A class name is a way to apply style sheet settings to a group of named elements, using the following syntax:

  <style type="text/css">

  P.someclass {color: red; margin-left: 1.5 in }

  .otherclass {color: green; font-size: 18 pt }

  </style>

Example:

```
<html>
<head> <title> css </title>
<style type="text/css">
        P.someclass {color: red; margin-left: 1.5 in }
        .otherclass {color: green; background-color: yellow }
</style> </head>
<body>
<p class=someclass> text1.
<p class=otherclass> text2
<h1 class = otherclass> This is heading1.
</body>
</html>
```

Creating an overall look for the web page

- The body tag is the tag specifier to use when applying style sheets to an entire document page.
- Background-color and image and setting the page margins are the common attributes with the body tag.
- Example

  <style type="text/css">

  body{ margin: 0.3 in; color: white; background-color: black;

background-image: url(back1.jpg)

background-repeat: repeat-x; background-attachment: fixed}

The background, margin and color css1 attributes

1. Background
2. Background-color
3. Background-image
4. Background-repeat - repeat-x // repeat horizontally
5. Background-attachment - scroll // scroll with page
6. Background-position - top center
7. Color - white
8. Margin - 20 px 10 px // top/bottom set to 20 pixels, left/right margins set to 10 pixels.
9. Margin-left
10. Margin-right
11. Margin-top
12. Margin-bottom

Modifying Font and Text Appearance

- Font and text css1 properties

CSS1 attribute	Sample Value
Font	Italic bold 18pt arial
Font-family	“times new roman” Arial
Font-size	Larger
Font-weight	800
Letter-spacing	0.1 em
Word-spacing	1.5em
Text-decoration	Underline
Text-transform	Capitalize

Text-align	Center
Text-indent	10%
Font-style	normal | italic | oblique
Font-variant	normal | small-caps

Creating Borders Around Elements

- There are certain CSS1 attributes that impact the box that surrounds a block level Html element.
- Padding and border CSS1 attributes

CSS1 attribute	Value
Border	Thick groove yellow
Border-color	Yellow red blue
Border-width	Thin thick
Border-style	Inset
Border-top	3px solid red (width,style,color)
Border-right	Yellow
Border-bottom	5px solid
Border-left	Solid
Border-top-style	Ridge
Border-bottom-style	Double
Border-left-style	None
Border-right-style	groove
Border-top-color	#FFFF00

CSS1 attribute	Value
Border-bottom-color	Black
Border-right-color	#0000CC
Border-left-color	Blue
Border-top-width	Thin
Border-left-width	Thick
Padding	12% 18px
Padding-left	18 px
Padding-right	0.25 in
Padding-top	4%
Padding-bottom	5px

Controlling the Appearance of Lists and other HTML elements

- The classifying and display CSS1 attributes

CSS1 Attribute	Value
Display	None
White-space	Pre
List-style	Square outside
List-style-type	Disc
List-style-image	url(someimage.jpg)
List-style-position	Inside

Width	150 px
Height	25%

- The display attribute can define that an element display as a list item or not display at all.
- The following example turns off the display of any element using the nodisplay class:

```
<style type="text/css">
    .nodisplay { display: none }
</style>
```

The Pseudo – Elements and classes

- A pseudo element is an html elelment in which some external factor influences the presentation of the element.
- Style settings can be applied based on this external factor.
- The following code demonstrates the use of this pseudo-class:

```
A:visited { color:red }
A:unvisited { color: yellow}
A:active { color: lime }
```

7. Using Tables for Organization and Layout

What are the HTML Table Elements

- Html tables begin and end with the table tags, <table> and </table>
- They contain rows, defined with the row tags <tr> and </tr>.
- Cells defined with cell tags, <td> and </td>
- Captions for tables are created with the begin and end caption tags, <caption> and </caption>.
- Some cells can be designated as table row headers with the use of the <th> and </th> tags.

- **The basic Table Elements**

Attribute	Description
Align	How table aligns with other elements.
Bgcolor	Background color for table
Width	Width of table
Cols	Number of columns within table.
Border	Width in pixels of frame around table.
Frame	Which sides of the frame surrounding table are visible.
Rules	Which rules appear between table columns and rows
Cellspacing	The space between cell border and table frame.
Cellpadding	The space between cell border and cell contents.

- The frame attribute has several values.
    i. Void – no frame
    ii. Above – top side only
    iii. Below – bottom side only
    iv. Hsides – horizontal sides
    v. Vsides – vertical sides only
    vi. Lhs – left side only
    vii. Rhs – right side only
    viii. Box – all four sides
    ix. Border – all four sides
- The acceptable values for the rules attribute

    None – no rules

    Groups – rules appear only between groups

    Rows – rules appear between rows

    Cols – rules appear between columns

    All – rules appear between all elements.

- **TH and TD element attributes**

Attribute	Description
Nowrap	Disbale automatic text wrapping
Bgcolor	Background color of cell
Rowspan	Numbers of rows spanned by cell
Colspan	Number of columns spanned by cell.

Column Grouping with COLGROUP and COL

- The columns of the table can be grouped by using the colgroup element.
- Once grouped, you can apply a width to all of the columns included within the group.
- A second element col, can also supply specific width and alignment information for one or more columns within the group.
- Example

```
<table border="1">
 <colgroup span="3">
  <col width="50"></col>
  <col width="100"></col>
  <col width="20"></col>
 </colgroup>
 <tr>
  <td>col 1</td>
  <td>col 2</td>
  <td>col 3</td>
 </tr>
</table>
```

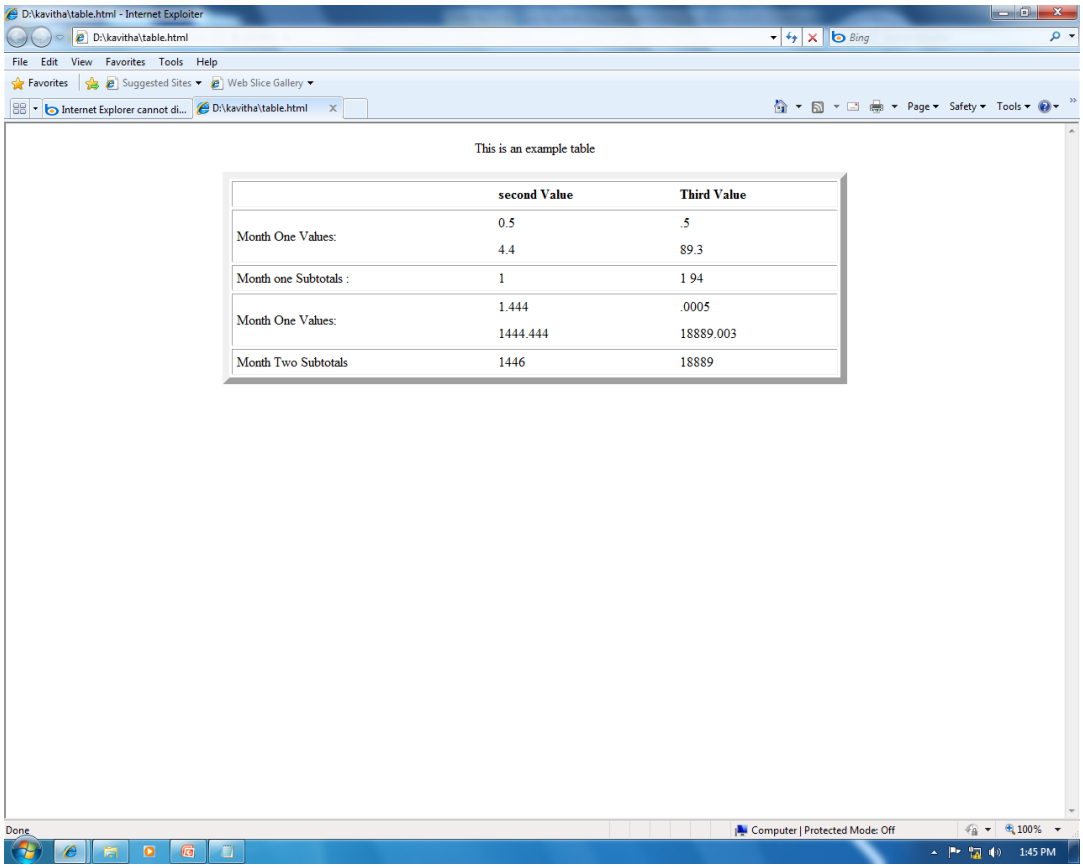
Row Grouping with thead, tfoot and tbody elements

- Each table has at least one row, and this becomes the default table body.

- However, the rows that make up the body can also be delimited with one or more tbody elements.
- The rows that make up the table head for a specific row grouping can be delimited with the thead element.
- The foot of the table row grouping can be delimited with the tfoot element.
- Example

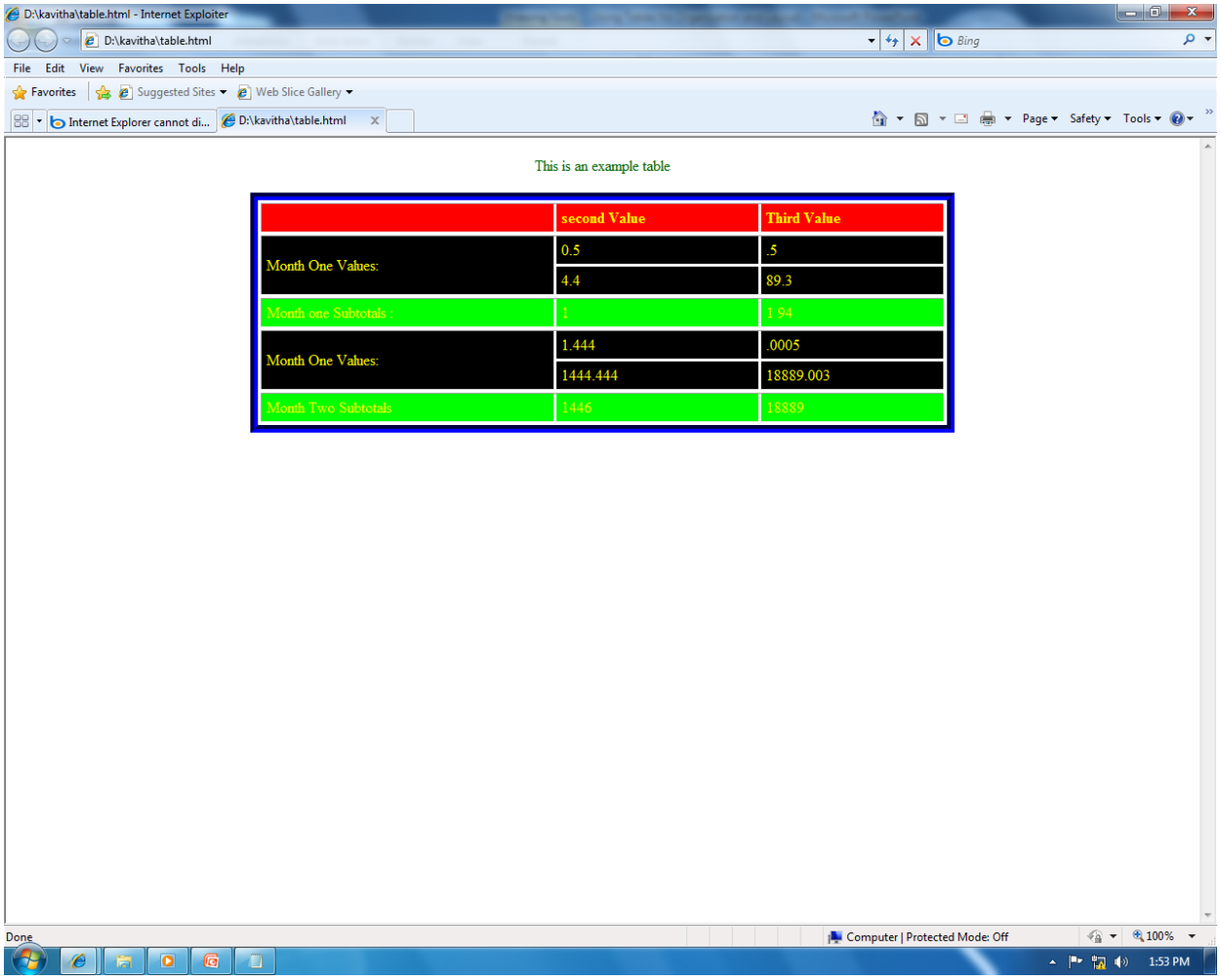
```
<table width=60% frame=border border=8 rules=groups cols=3 cellspacing=3 cellpadding=5 align=center>
<caption> This is an example table </caption>
<thead align=left>
<tr>
<th> </th>
<th> second Value </th>
<th> Third Value </th> </tr>
<tfoot>
<tr> <td> Month Two Subtotals </td>
<td> 1446 </td>
<td> 18889 </td>
</tr>
</tfoot>
<tbody>
<tr>
<td rowspan=2> Month One Values: </td>
<td> 0.5 </td>
<td> .5 </td>
</tr>
<tr><td> 4.4 </td>
<td> 89.3 </td>
</tr>
<tfoot>
<tr>
```

```
<td> Month one Subtotals : </td>
<td> 1 </td>
<td> 1  94 </td>
</tr>
<tbody>
<tr>
<td rowspan=2> Month One Values: </td>
<td> 1.444 </td>
<td> .0005 </td>
</tr>
<tr><td> 1444.444 </td>
<td> 18889.003 </td>
</tr>
</tbody>
</table>
```



Combining the Use of HTML Tables and CSS1 Style Sheets

```
<head>
<style type="text/css">
Thead { background-color: red; color: yellow}
Tbody  { background-color:black ; color: yellow}
Tfoot { background-color:lime; color: yellow}
Table { border-style: groove; border-color:blue;
        color: darkgreen;
Th { font-family : fantasy }
</style>
```



Using an html table to create a Layout for Links

- A popular use of html tables is to maintain the layout of a links page or a section of a page that contains resource links
- One column table used to control layout of a links list.
- Example

```
<html>
<body>
<table width=80% cellpadding=10 cellspacing=15 border=5 frame=border rules=none align=center>
<caption align=bottom> Internet Explorer Dynamic Html Links </caption>
<tr> <td> <h1> My IE Dynamic Html Resources </h1> </td> </tr>
<tr><td> <a href="http://www.w3.org/pub/www/tr/wd-positioning"> W3C Positioning Draft </a> </td></tr>
<tr><td> <a href="http://www.microsoft.com/ie/ie40"> Internet Explorer 4.0 Platform Preview </a> </td></tr>
</table>
</body>
</html>
```

Creating Page Columns with a Table

- Another popular use of html tables is to create columns within a page.

```
<table width=100% cols=3 cellspacing =10>
<colgroup span=3>
<col width=“30%”>
<col width=“35%”>
<col width=“35%”>
```

Controlling a Form Layout with a Table

- With a table, the form elements can be easily aligned.
- Example…

8. Advanced Layout and Positioning with Style Sheets

- CSS positioning attributes

Attribute	Description
Position	Determines whether element is positioned explicitly or relative to the natural flow of web page document.
Left	Position of the left side of the rectangular area enclosing the element.
Top	Position of the left side of the rectangular area enclosing the element.
Width	Width of the rectangular area enclosing the element.
Height	Width of the rectangular area enclosing the element.
Clip	The clipping shape and dimensions used to control what portion of the element displays.
Overflow	The portion of the element contents that exceeds the bounds of the rectangular area enclosing the element.
Z-index	The stacking order of the element if two or more elements are stacked on top of each other.
Visibility	Whether element is visible.

Positioning images and other elements

- Images have some positioning capability, with the hspace and vspace IMG attributes, and can be aligned with the align attribute.
- One technique web developers did not have before IE 4.0 is the ability to layer text and other html elements on images, or to layer images themselves.
- Also, developers could not exactly position images or any other element.
- CSS positioning changes all of this.

- With CSS, the position attribute is set to a value of absolute, which means that the element is positioned exactly, regardless of how any other element is positioned.
- Using CSS positioning to position three images

```
<html><head><style type=text/css>
Img { position: absolute; width:90;height:90; top 100}
#one { left:100}
#two { left:200 }
#three {left:300}
</style>
</head>
<body>
<img src=red.jpg id=one>
 <img src=yellow.jpg id=two>
<img src=green.jpg id=three>
</body> </html>
```

- Using DIV blocks to position images

```
<html><head>
<style type=text/css>
DIV { position: absolute; top: 100}
#one { left:100}
#two { left:200 }
#three {left:300}
</style>
</head>
<body>
<div id=one> <img src=red.jpg>  </div>
<div id=two>  <img src=yellow.jpg > </div>
<div id=three> <img src=green.jpg>  </div>
</body> </html>
```

- You can layer html elements, including placing text above images.
- One key to for using layers is to set the z-index CSS positioning attribute to a higher integer for the element you want to display at the top of the stack.
- Layering text and images with z-index ordering

```
<html><head>
<style type="text/css">
    Body { font-family: arial; color:white; font-weight: bold}
    Div { position: absolute }
    #one { top:25; left:20; z-index:1 }
    #two { top:125; left:20; z-index:1 }
    #three { top:225; left:20; z-index:1 }
</style>
 </head>
<body>
    <div style="top:50; left:40; z-index:2">
    Product <br> One  </div>
    <div style="top:150; left:40; z-index:2">
    Product <br> two  </div>
    <div style="top:250; left:40; z-index:2">
    Product <br> Three  </div>
    <div id=one>   <img src=red.jpg width=90 height=90>   </div>
    <div id=two>   <img src=yellow.jpg width=90 height=90>   </div>
    <div id=three>   <img src=green.jpg width=90 height=90>   </div>
</body>
</html>
```

Creating page columns and using html Tables and CSS positioning together

- Web developers use html tables to create columnar contents.
- CSS positioning can be used to create multi column web page content.
- In addition html tables and CSS positioning can be used for one web page.

Converting an Existing Table – Based Web Page

- Using a table to control page layout isn't a bad approach, but there are limitations.

- First with the menu bar, you should layer the menu bar text on the image itself, rather than having to set each text block below the image.
- A second limitation is that because the text and images alternate, the images tend to have a large space surrounding them
- To convert the page using CSS positioning, the first step is to recreate the menu bar at the top of the page.
- A new document is started.
- Div blocks were used, which supports positioning directly with images.
- Menu bar section code:

```
<style type="text/css">
DIV { position: absolute; font-size: 10pt;font-family:arial;}
#logo ( position:absolute; top: 50 px; left: 50px; z-index: 5; width:105}
#image1 { position:absolute; top:10px; left:185 }
#image2 { position:absolute; top:10px; left:295 }
#image3 { position:absolute; top:10px; left:395 }
#image4 { position:absolute; top:10px; left:500 }
#title1 {position:absolute; top:20px; left:200; z-index:5; height: 20px;
width: 80px }
#title2 {position:absolute; top:20px; left:300; z-index:5; height: 20px;
width: 80px }
#title3 {position:absolute; top:20px; left:410; z-index:5; height: 20px;
width: 80px }
#title4 {position:absolute; top:20px; left:510; z-index:5; height: 20px;
        width: 80px }
</style> </head>
<body>
<!-- set menu images →
<div id=logo> <img src="logi.gif" width=104 height=153> </div>
<div id=image1> <a href=http://www.yasd.com>
<img src="menu.jpg" border=0 width=101 height=153> </a> </div>
<div id=image2> <a href=http://www.yasd.com/samples>
```

<img src="sample.jpg" border=0 width=101 height=153>

<img src="image.jpg" border=0 width=101 height=153>

<img src="photo.jpg" border=0 width=101 height=153>

<!-- set menu Titles -- >

  Main

.

.

.

9. Creating Forms with HTML

What are HTML forms?

- An HTML form is not a visual element.
- It is a container and can contain one or more buttons, textboxes or other form elements.
- The form elements can be used to access information from the reader and then process that information within the webpage.
- The information can also be sent to a CGI or web server application for further processing.

The FORM Objects and its Attributes

- A form is created using the begin and end form tags <FORM> and </FORM>.
- Though not required , there are form attributes that can control what happens to the information, the method used to deliver this information and where feedback derived from the form contents should be sent.
- Syntax for Creating Form

<Form name="mailForm"
action = "url" Method=post>
……
</Form>

- Form attributes are:
  i. name – Form name.

ii. target – Location of window where from responses are sent.

iii. action – URL of webserver application that process form information.

iv. enctype – By default this attribute has a value of application x/www-form-urlencoded, but can be set to multipart/form-data if the file upload element is used.

v. method – A value of get or post, which determines how form information is sent.

The forms array

- Each form has a separate entry in a built-in array called forms.
- This array can be accessed in script through the document object, which contains this array as a property.

Web page with 2 forms and 4 elements

```
<html>
<head>
<title>forms</title>
<script language="javascript">
<!--
function list_values()
 {
   var string="";
   for(i=0;i<document.forms.length;i++)
        {
                string+="form name: "+document.forms[i].name+"<br>";
                for(j=0;j<document.forms[i].elements.length;j++)
                string+=document.forms[i].elements[j].name+" ";
                document.forms[i].elements[j].value+ "<br>";
                string+="<p>";
        }
   document.writeln(string);
 }
-->
```

```
</script>
function
</head>
<body>
 <!--form 1 -->
<FORM name="form1">
<input type=text name="textname">
<input type=button name="button1" value="push me">
</FORM>
<!--form 2-->
<form name="form2">
<select name="randomvalues">
<option value="1">one
<option selected value="2">Two
</select>
<input type=button name="button2" value="no, push me" onclick="list_values()">
</form>
</body>
</htmL>
```

The FORM elements

- The Form elements are : button, check box, fileupload, hidden, password, radio, reset, select, submit, text and textarea.
- Each element has a different look and performs a different function.
- The INPUT tag creates most of these elements. As an example of creating an element, the following code creates a text field:
  <INPUT type="text" name=semefield>

The BUTTON Element

- Probably the most common of the form elements is the button element.
- You can create a button using the following code
  <input type="button" name="somebutton" value="PUSH ME">

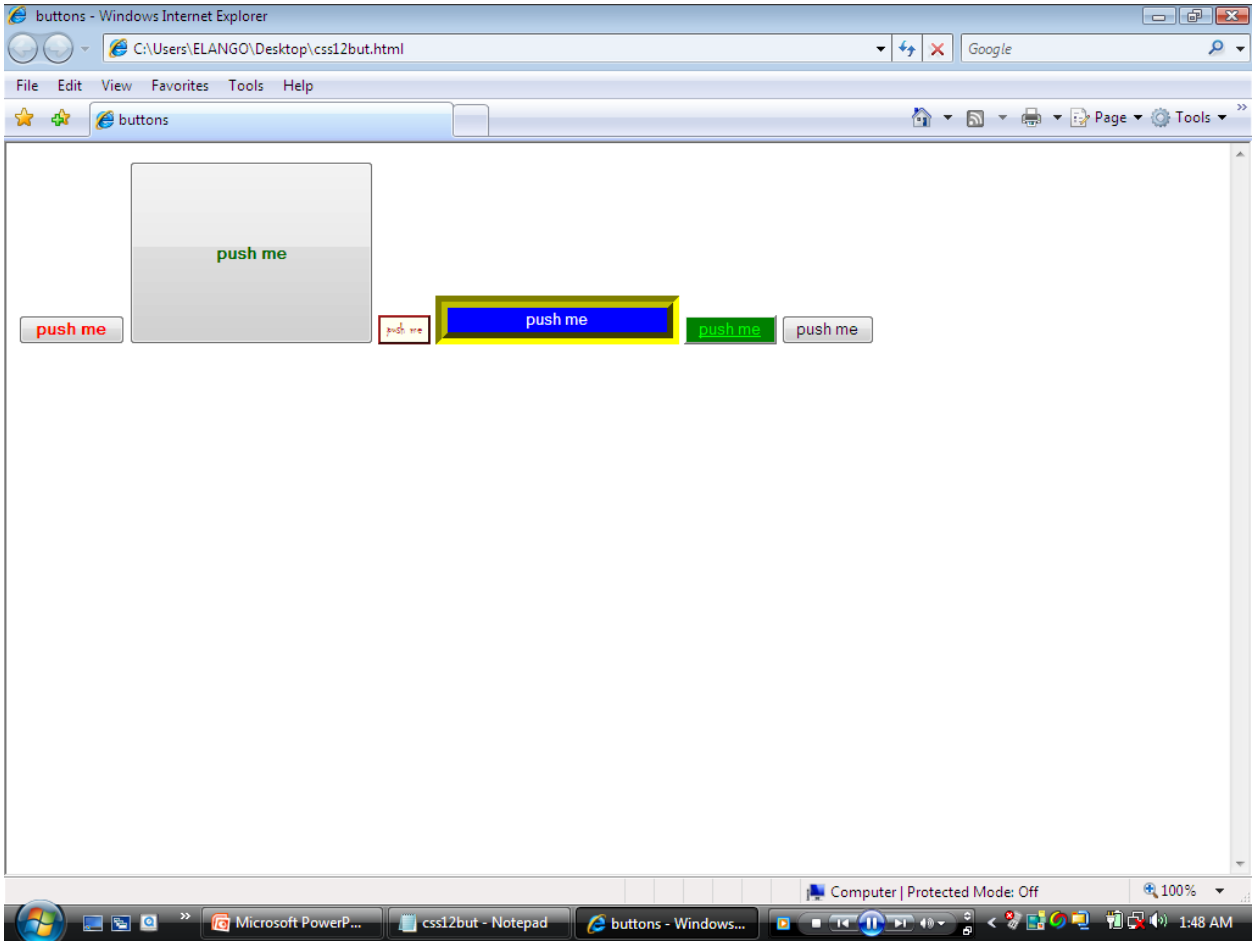
A simple form , button with the words “push me”

```
<html>
<head>
 <title>but</title>
</head>
<body>
 <input type="button" name="somebutton" value="push me">
</body>
</html>
```

Applying CSS1 style sheets to several buttons

```
<html>
<head>
<title> buttons</title>
<style type="text/css">
 #one{color: red; font-weight: bold}
 #two{color: darkgreen; font-weight: bold; background-image: url(C:\Users\Public\Pictures\Sample Pictures\forest.jpeg); width:200; height:150}
 #three{color: firebrick; background-color: ivory; border-color: firebrick; font-family: chiller}
 #four{color: white; background-color: blue; border-color: yellow; border-style: groove; border-width:10; width:200}
 #five{text-decoration: underline; background-color: green; color: lime}
</style>
</head>
<body>
<!--form 1-->
<form name="form1">
 <!--button 1-->
 <input id="One" type=button name="button1" value="push me">
 <!--button 2-->
 <input id="Two" type=button name="button1" value="push me">
```

```
<!--button 3-->
<input id="Three" type=button name="button1" value="push me">
<!--button 4-->
<input id="Four" type=button name="button1" value="push me">
<!--button 5-->
<input id="Five" type=button name="button1" value="push me">
<!--button 6-->
<input type=button name="button" value="push me">
</form> </body> </html>
```



Creating a SELECTION list

- A selection list, or drop-down list box is really a couple of different element.
- The first is the select element, which is the box, and the second is one or more option elements, which contain the box entries.

- Creating a selection list generates a text-field-sized element with an arrow.
- Clicking the down arrow in the box next to the list exposes the list elements in a drop-down box big enough to hold all the elements.

Example:

```
<html>
<head>
 <title>but</title>
</head>
<body>
 <select name="selection">
   <option> ONE
   <option> TWO
   <option selected> THREE
   <option> FOUR
</body>
</html>
```

- There are three attributes for the SELECT object:

name – element name.

size – number of options visible when page opens, set to one by default.

multiple – specifies that more than one option can be selected.

Applying CSS1 attributes to a select and option elements

```
<html>
<head>
 <title>list</title>
<style type="text/css">
 option {background-color: lime; color: red}
 option.two{background-color: blue; color: yellow}
 select{background-color: red; margin: 1.0in; font-family: algerian; font-weight: bold;font-size: 18pt}
</style>
</head>
```

```
<body>

<!--form 1-->
<form name="form1">
 <select name="selection">
   <option value=1> First selection
   <option class=two value=2> Second selection
   <option value=3 selected> Third selection
   <option class=two value=4> Fourth selection
</select>
</form>
</body>
</html>
```

Adding Radio Buttons and Checkboxes to a webpage

- The radio button is a way to provide a set of mutually exclusive choices.
- Only one button can be checked with the radio button, and clicking one of the buttons deselects any previous selection.
- Checkboxes, on the other hand , provide a method of selecting several mutually inclusive choices.
- You can select one box or more of the options represented by the boxes.

```
<html>
<head>
 <title>but</title>
</head>
<body>
<!--radio buttons-->
 <input type="radio" name="thegroup" value="one" checked>one
  <input type="radio" name="thegroup" value="two" >two
  <input type="radio" name="thegroup" value="three">three
<p>
<!--checkboxes-->
```

```
 <input type="checkbox" name="ckhbox1" value="checkone" checked>check
one
  <input type="checkbox" name="ckhbox2" value="checktwo">Two
  <input type="checkbox" name="ckhbox3" value="checkthree">three
</body>
</htmL>
```

Uploading files with fileupload

- The fileupload element provides a means for the web page reader to specify a file for loading to the web server.
- When you create one of these controls, a textbox for the filename and a button labeled browse… are created within the webpage.
- The web page reader can type a filename and path into the box or click the browse button to select a file.
- When the form is submitted, the file is also appended to the form data being sent to the server.

Accessing Text with the Text Controls: text, text area and password

- The text, text area and password input elements are all methods of accessing text from the web page reader.
- The text element provides for single-line text values such as a name, and text area provides a control that can accept a block of text several words wide and several lines long.
- The password element hides the values being entered with the usual display of asterisks.

Three HTML input elements , a text control,a textarea control and a password control

```
<html>
<head>
 <title>text</title>
</head>
<body>
 <input type="text" value="Enter information here" name="text1" size=80>
```

```
<p>
 <textarea rows=20 cols=50></textarea>
<p>
 <input type="password" size=20>
</body>
</html>
```

Creating Interpage Persistence with the hidden Element

- There is a great trick to use when maintaining information persistently between the webserver application and the web pages showing on the client.
- This technique is accomplished using the hidden form elements.
- The hidden form element is an element that contains information stored within the web page but not displayed to the web page reader.

Submitting and Resetting the form with submit and reset

- These two INPUT elements submit the form to the form processing application , or reset the form values, respectively.
- To create either of these elements, the following code is used,

```
<INPUT TYPE="submit" VALUE="Send Form">
<INPUT TYPE="reset" VALUE="Reset Form Value">
```

- Using the submit element submits the form.
- An alternative approach could be to create some other element , and based on trapping an element event, issue the form submit method, as shown here:

```
document.forms[0].submit();
document.forms[0].reset();
```

Working with the New HTML 4.0 form Elements and Extensions

- The HTML 4.0 has several new form elements and form element attributes.
- Label, image and fieldset are the three new HTML 4.0 form elements.
  - >> The label element is used to provide a label for a control, such as the text or text area controls.
  - >> The image control creates an image-based control that can be clicked in a manner similar to a button.
  - >> The fieldset control is an element that provides a visual grouping of other form

elements,such as radio buttons.

- Accesskey, tabindex, disabled and read only are the new HTML 4.0 element attributes.
- The accesskey attribute can be used to associate an accelerator key with a specific form element.
- The disabled attribute disables the focus from an element and prevents that same element's value from being submitted with the form.
- The tabindex attribute associates a specific tab order with an element.

10. Frames and Framesets

- Html frames slip the web page window into separate window views, each capable of holding a different html document.

Creating and Working with Frames

- Frame windows are made from more than one html file.
- One file contains the Frameset definition, including which source files make up the frames and how much space each will occupy.

The Frameset Element

Main.html

```
<frameset rows="80,*">
        <frame src="top.html"
        <frame src="content.html">
</frameset>
```

- Instead of creating two rows, you can also create two columns and load the web page contecnts into each.

```
<frameset cols="80,*">
    <frame src="menu.html">
    <frame src="main.html">
</frameset>
```

Nested Frameset

- Framesets that contain other framesets.
- Example

```
<frameset rows="*,250">
    <frame src="main.html">
    <frameset cols="200,200,*">
        <frame src=rose.jpg>
        <frame src=lily.jpg>
        <frame src=tulip.jpg>
    </frameset>
</frameset>
```

Frame Element Attributes

Attribute	Description
Name	Frame name
Src	Frame source
Frameborder	Setting this value to 1, draws a border around the frame. 0 removes the border.
Noresize	Turns off frame resizing
Marginwidth	Frame horizontal margin
Marginheight	Frame vertical margin
Scrolling	Setting this value to auto provides scrolling only when frame content does not fit within the frame space. Setting this value to yes always provides a scroll bar. Setting this value to no always turns off the scroll bar.

Accessing External References from Frames

- One of the disadvantage to using frames is that including a hypertext link within a frame page to a web page at an external web site loads that page into the frame, rather than directly into the browser window.

- This problem can be resolved by the target attribute which can be used with link elements.
- There are several different reserved keywords that can serve as the target attribute value:
    - Blank - loads page to new unnamed window
    - self - loads page to current frame or window
    - Parent - loads page to parent window
    - Top - loads page to original window

Inline Frames with IFrame

- Html 4.0 created a new frame element called the inline frame, which uses the tags <iframe> and </iframe>.
- Inline frames have the same attributes regular frames have, except that each inline frame window is embedded within a web page and attributes are specified directly for the frame window.
- Example:

```
<html>
<body>
<h1> This demonstrate inline frames  </h2>
<iframe src="main.html" height=120 width=95%>  </iframe>
<iframe src="content.html" height=300 width=95%>  </iframe>
</body>
</html>
```

11. Using Images with HTML

Embedding images within an Html Document

- Images are embedded within a web page with the use of IMG tag.
- This tag contains the source of the image file and other information, and does have an end tag.

The image formats

- The Graphics Interface Format (GIF) is the most popular image format in use within html files.

- The GIF89a version allows for the specification of the number of colors to include within the image file.
- The Joint Photographic Experts Group (JPEG) format retains all the image's colors – but also supports a compression technique that discards data from the image nonessential to the display.
- Higher levels of compression lead to smaller file sizes, but also decrease the quality of the image.
- The Portable Network Graphics (PNG) format is considered a replacement for the GIF format.
- PNG supports a more efficient compression routine and two-dimensional interlacing, which controls how an image is displayed while it downloads.

The IMG Attributes

- Images are embedded within a web page using the IMG tag.
  <img src=flowes.gif width=100 height=100 alt="Flowers are very Beautiful" hspace=20>
- Attributes supported for IMG tag are:
  - src – URL of the image source file
  - alt - alternative text
  - Align – how image aligns, deprecated
  - Height – height to reserve for image, deprecated
  - Width - width to reserve for image, deprecated
  - Border - size of border surrounding image, deprecated
  - Hspace – horizontal white space on either side of image, deprecated
  - Vspace - vertical white space on either side of image, deprecated
  - Id - image name
  - Class - style sheet class
  - Style - style sheet information
  - Title - element title

The IMG Width and Height Attributes

- The width and height of the image define the area of the window the image occupies.

- The same image object with different widths and heights

```
<img src=logo.jpg width=100 height=100>
<img src=logo.jpg width=50 height=50>
<p>
<img src=logo.jpg width=300 height=300>
<img src=logo.jpg width=200 height=150>
```

The alt Attribute

- Alt attribute provides alternative text for the browsers that do not pick up images.
- <img src=flower.jpg alt=" Flower – Rose">

The image object and accessing images within script

- The image object is accessed by creating a new image object or by accessing the images embedded into the web page via the image array.
- Each img tag within the web page is associated with one entry in the images array.
- Example

```
<html>
<head>
<title> Images  </title>
<script language="javascript">
Function change_image(num)
{
Var img=" ";
if(num==1)
    img = "photo7.jpg";
Else if(num==2)
    img = "photo8.jpg";
Else if(num==3)
    img = "photo9.jpg";
Else
    img = "photo10.jpg";
Document.images[0].src=img;
```

```
}
</script>
</head>
<body>
<form name=f1>
<img src="photo7.jpg">
<input type=radio name=r1 value=one onclick="change_image(1)"> one
<input type=radio name=r1 value=one onclick="change_image(2)"> Two
<input type=radio name=r1 value=one onclick="change_image(3)"> Three
<input type=radio name=r1 value=one onclick="change_image(4)"> Four
</form>
</body> </html>
```

The image object properties, accessed via image array

- Src - image URL
- Width - image width
- Height - image height
- Lowsrc - lowsrc image's url, first displayed when page is opened.
- Hspace - horizontal space
- Vspace - vertical space
- Border - border width
- Name - image name
- Complete - whether image has completed loading.

Images and Caching

- When you access an image for the first time, the image must be downloaded from the server.
- After the image is downloaded, though, it is cached on the client machine and subsequent accesses to the image pull it from the cache rather than the server.
- Example

```
<html> <head> <title> images </title>
<script language=javascript>
image_array = new array(4);
```

```
for(i=0; i<image_array.length; i++)
  image_array[i] = new image(100,150);
image_array[0].src = “photo7.jpg”;
image_array[1].src = “photo8.jpg”;
image_array[2].src = “photo9.jpg”;
image_array[4].src = “photo10.jpg”;
function change_image(num)
{      document.images[0].src = image_array[num].src;    }
</script>
</head>
<body>
<form name=f1>
<img src=“photo7.jpg”>
<input type=radio name=r1 value=one onclick=“change_image(1)”> one
<input type=radio name=r1 value=one onclick=“change_image(2)”> Two
<input type=radio name=r1 value=one onclick=“change_image(3)”> Three
<input type=radio name=r1 value=one onclick=“change_image(4)”> Four
</form>
</body>
</html>
```

Creating Image Rollover Effects

- The most popular use of changing images is to create a rollover effect for images that represent active links.
- When the reader moves the mouse over the image, the image takes on a different appearance, adding a highlight that provides feedback to the reader.
- Roll over with mouse down and mouse up events

```
<html>
<head>  <title>  images  </title>
<script language=“javascript”>
Highimage = new image(106,157);
Lowimage  =  new image(106,157);
```

```
Lowimage.src = “optnrm.jpg”;
Highimage.src = “pothigh.jpg”;
Function change_high(num)
{   document.images[num].src = highimage.src.  }
Function change_low(num)
{   document.images[num].src = lowimage.src.  }
</script>  </head>
<body>
<a href=“” onmousedown=“change_high(0)”  onmouseup=“change_low(0)”>  <img src=“optnrm.jpg”width=106 height=157 hspace=10>  </a>
<a href=“” onmousedown=“change_high(1)”  onmouseup=“change_low(1)”>  <img src=“optnrm.jpg”width=106 height=157 hspace=10>  </a>
<a href=“” onmousedown=“change_high(2)”  onmouseup=“change_low(2)”>  <img src=“optnrm.jpg”width=106 height=157 hspace=10>  </a>
</body.
</html>
```

12. Merging Multimedia, Controls and Plug- Ins with Html

Accessing sound and video in web pages

- Multimedia was first used in web pages to include sound.
- Both Netscape Navigator and Internet Explorer support the use of sound in a web page.
- Beginning with Netscape’s Navigator version 3.x, the browser used the Live-audio plug-in for sound, and the LiveVideo plug-ins for video.

Using the <EMBED> Tag

- The EMBED element is a simple-to-use technique that includes multimedia in a web page.
- Both IE and Navigator show a visible control in a web page if the following embedded tag syntax is used:

  <EMBED src=“example.mid” controls=console width=144 height=60>

- Each browser determines what plug-ins are currently installed to support files with the extension type of .mid and the browser then uses the associated plug-in.
- Netscape will most likely use the Live-Audio plug-in, and microsoft use the DirectShow control.
- The EMBED element can also be used to support video files.

Netscape's LiveAudio Plug-In

- The LiveAudio plug-in play files with extensions of .wav, .aiff, .au and .mid.
- The LiveAudio plug-in attributes are:
    - SRC - Audio Source File URL
    - AUTOSTART - Whether file starts playing as soon as it is loaded.
    - LOOP - Whether to loop sound file
    - Starttime - specifying a time in the source code to begin playing, specified in Minutes:Seconds
    - EndTime - Specifying a time in the source code to end playing
    - Volume - A value between 0 and 100, representing percentage of sound volume.
    - Width - Width of display
    - Height - height of display

The methods that can be called from script for the LiveAudio plugin are the following:

- Play
- Stop
- Pause
- Start-time
- End-time
- Setvol
- Isready
- Isplaying
- Ispaused
- Getvolume

The <OBJECT> tag

- The object element is used for embedding generic content into a web page.

- The object element has gained favor because of its use of the PARAM element, embedded within the beginning and ending <object> tags, that provides name-value pairs as a method of passing attributes to the embedded element.

The attributes specific to the OBJECT element are:

- CLASSID - Location of an object's implementation
- CODEBASE - Path used to resolve URL references
- CODETYPE - Internet media type
- DATA - Location of data rendered by control
- TYPE - internet Media type of data
- STANDBY - message to display while control is downloading

The attributes of PARAM element are:

- Name
- Value
- Valuetype

Example

<object id="thename" classid="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A">
<param name="filename" value="f.avi">
</object>

Embedding Sound and Video into an Html Page

- With DirectShow, a multimedia file can be inserted into a page using a hypertext link, embedded into the page using the <embed> tag, or embedded as an ActiveX control using the <object> tag.
- Three different techniques of using DirectShow to play a .mid file

<html><body>
 Song
<object id="themusic" classid="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A">
<param name="filename" value="f.avi">
</object>
<EMBED SRC="EXAMPLE.MID">

The DirectShow object Properties, Events & Methods

Properties are:

- Filename - url of multimedia source file
- AllowHideControls - whether the web page reader can hide or display controls
- Appearance – whether the control has an inset border or flat border
- Autostart - starts the multimedia stream play as soon as the control loads
- Bordestyle - controls appearance of border
- Currentstate - whether control is stopped, playing or paused.
- Enabled - whether the control is enabled
- Fullscreenmode - to display video fullcreen
- Playcount - number of times to loop playback
- Readystate - controls readiness state
- Volume – controls sound volume

Methods:

- AboutBox - version and copyright information about the control
- IsSoundCardEnabled - whether sound card is installed and enabled
- Pause - pause playback
- Run - run multimedia stream
- Stop - stop stream playback

Events:

- Displaymodechange - when display mode property is changed
- Error - when an error occurs
- openComplete - when source code has finished loading
- PositionChange - When media position is changed
- StateChange - player state changes
- Timer - for timing events

Microsoft's Direct Animation Technology

- DirectAnimation is a set of objects, a set of built-in controls, and an API that are accessible from script, java, C++ or other code.
- With these sets you can integrate the use of scripting, java and controls.

The DirectAnimation controls are

- Structured Graphics - creates a 2D images
- Path – controls the movement of any object
- Sequencer - controls and synchronizes the actions of several different elements
- Sprite - controls the animation frames and playback

Internet Explorer Built-In Filters

- Microsoft created the css1 style visual filters to control the appearance of ordinary HTML elements.
- The visual filter effects can do such things as make an element semi-transparent, rotate an image, remove colors, or add specific lighting effects to a page.

Visual filter effects:

- Glow - adds a glow outlining the element
- Gray – drops the element's color palette
- Light - creates a light source on a page
- Shadow - creates solid shadow

When and when not to use Multimedia

- There are some limitations to using multimedia.

Restrict download sizes

- If you want to include the use of video or sound, such as company theme song, you might want to consider placing these on a separate page so these bandwidth-hogging files do not impede your reader's access to other information.

Copyright Information

- If you don't want your image, video or song saved and used by another site, you must make sure that all copyright information on the multimedia file is displayed prominently on the same page the file is accessed from.
- Speaking of copyright, do not ever copy and reuse images, videos, or sounds that are not public domain and that you do not have permission to use.

Cross- Browser Compatibility

- If you want to use one browser's specific technique, but your page is accessed by both of these popular browsers, use the browser-specific technique to enhance an effect, not supply the entire effect.

Cross- Platform Compatibility Issues

- Not all multimedia files play equally well on all platforms.
- The rule of thumb is to deliver content to the lowest common denominator – what works for all platforms – or to provide different links to different multimedia types, and let the reader choose which format to use.

UNIT II

Contents

13. Client – Side Scripting: Scripting Basics

Scripting Basics

- Html is a powerful mechanism for layering out pages.
- However, with html alone, you are limited to static pages.
- Client side scripting extends the static web pages by providing a mechanism for web page authors to create dynamic, interactive pages for their users.
- The script is written in one of the supported programming languages and then embedded in a web page's html code.

What is Client – Side Scripting

- Client – Side scripting refers to creating scripts that are executed in the user's web browser, the web client.
- A client side script is typically a small program embedded within an html document.
- Whenever a web browser that supports client side scripting encounters one of these scripts, it executes the program by interpreting the commands.

Scripts and Programs

- Client side scripts are intended to be cross –browser and cross-platform compatible.
- The intent with java is to have a single code base that, when compiled to byte code, runs properly on all platforms. A byte code file must be interpreted.
- Java, however, is a lot more challenging to learn than many web designers would like. Therefore client-side scripting languages were introduced.
- These languages are meant to have the same cross-platform compatibility, while being easy to learn. There is a subtle difference between scripts and programs.
- Scripts are typically code that is not compiled before being executed, whereas programs are compiled code that can be executed without the use of command interpreter.

Compiled Programs

- Compiling converts the source code into machine language executable code.

- Once the program has been compiled, that can be run on platform for which it has been compiled without any other program.
- Compiled programs have some advantages. First, because the source code is changed into machine-dependent executable code, it executes much faster than an interpreted program.
- Another advantage of compiled programs is that it can be distributed without compromising source code's integrity.
- Disadvantage of the compiled program is that the process of compiling a program can take some time, depending on the speed of machine. The largest disadvantage is the cross platform compatibility problem.

Interpreted Scripts

- Scripts are run through a command interpreter that interprets the commands at run time. A command interpreter is a compiled program.
- The advantage to scripts over programs is the ability to write and distribute a single code base across many platforms. Another advantage of scripts is their ease of maintenance.
- The biggest disadvantage is the speed at which scripts run. The big difference is that the translation is don't at run time.
- Scripts run significantly slower than their program counterparts.
- The other common disadvantage of scripts over programs is the integrity of source code.

Client –Side Scripting Languages

JavaScript

- Javascript was the first client-side scripting language developed by Netscape.
- The purpose of javascript is to provide a true programming language for use by web page authors to add more interactivity to web pages.
- The first version of javascript contained most of the core functionality for the scripting language.

Jscript

- In order for Microsoft to support javascript standard, it had to create its own implementation of javascript.

- The Microsoft implementation of javascript is jscript.

VBScript

- Around the time of Jscript's release, Microsoft introduced anotherr client-side scripting language called Visual Basic Scripting Edition, or VBScript.
- VBScript is based upon Visual Basic Product.

Placing Scripts in Web Pages

- The client – side scripts can be integrated within a web page.

The <script> tag

- In order to embed client – side scripts in to web pages, an html tag is needed to tell the browser that the following code is a script. This is done through the <script> tag.
- The <script> tag has the following syntax:
    - <script language=[scripting language]>
- For example,
    - <script language = "JavaScript">
- The <script> tag has a closing tag, </script>, to denote the end of the script.

Placing Scripts in HTML code

- Client – Side scripts can be placed anywhere in the html text. The most common place for scripts is head section.
- Using a script tag in the header section is a good place to define all the subroutines.
- JavaScript embedded in the <head> section

```
<head>
<title> Java Scripting </title>
<Script Language="javascript">
  function display_message(string_messgae)
{
   window.staus = string_message;
}</script>  </head>
```

- The script can be included in the body of the html document, when the script needs to write some dynamic code to the document as it's being parsed upon loading.
- JavaScript example for the html body section

```
<script language="javascript">
  if ( useragent == "Netscape")
{
        document.write("Thank you for using Netscape")
}
</script>
```

- Javascript example embedded in html tags

<a href="toc.html" onMouseOver='window.status="Return to table of contents";return true' onMouseOut='window.status=" "; return true'>

Running client – side scripts

- Client side scripts are run by loading the web page in which they are embedded into a web browser that supports that scripting language.
- Depending on how the script is to be run, the web browser either executes the code immediately or waits until the user performs some action.

What can Client – Side Scripts Do?

- Client – Side scripts provide a mechanism for building more interactivity into web pages.
- Client – side scripts are full featured programming languages with conditional statements, control structures, and data structures.

Tasks that can be performed by using client-side scripting are validation of form data, status bar messages, image rollovers and cookies manipulation.

Limitations of Scripting

- One of the biggest problems with client – side script is that the script embedded into html pages is viewable by everyone who downloads the web page.
- Another limitation is the scope in which they can run.
- Client - Side scripts are also somewhat limited in the actions they can perform.

- Client – side scripts must contain commands that are independent of the platform on which they are run.
- Client – Side scripts do not possess all of the power of CGI scripts. CGI script is needed to handle the form data after it has been validated. CGI script is also needed to access information in a database.

14. Client – Side Image Maps

Server – Side Imagemaps versus Client-Side Imagemaps

What are ImageMaps?

- Imagemaps are simple text descriptions of shapes and their related coordinates in an image file.
- The purpose of an imagemap is to define multiple hotspots or clickable points on a single image.

How do Server – Side imagemaps work?

- Server – side imagemaps are reside on and are executed by the web server.
- Separate CGI script called Imagemap is used to interpret an imagemap.
- On the client side, the web page author includes the ISMAP attribute in the IMG tag of the image to be used with an imagemap.
- Then, in the <A> tag, the href attribute points to a map file on the server, which has a .map filename extension.
- This .map file contains shapes and coordinates for the imagemap's clickable areas.
- When the user clicks an image that uses a server-side imagemap, the web browser calculates the mouse click's coordinates and sends it to the server, along with the name of the imagemap file.
- The web server opens the map file and finds the last area in which the coordinates lie.
- The corresponding URL is returned to the web browser.

Server – side imagemap file

imagemap file

Default /

Union TV

Poly /union-tv.html 35,27,46,69,44,94,71,23,45,23

Union File

Poly / union-film.html 5,83,23,83,29,56

- Every click on a server side imagemap requires extra load on the server in order to determine the area in which the coordinates lie.
- Another drawback to server- side imagemaps was their inaccessibility to users with text-only browsers.

How do Client-Side Imagemaps Work?

- Client – Side imagemaps operate much more efficiently than their server-side counterparts.
- The client machine performs the area matching for the coordinates of the mouse click.
- The coordinates for the area shapes and their related hrefs are listed in the body of the html document.
- The additional tags used to create client-side imagemaps are <Map> and <area>
- The UseMap attribute tells the web browser you are using a client – side imagemap.

Making Client – Side Imagemaps

- First start with image which is used to create a mapping.
- Then use the client-side imagemap tags, <map> and <area>, and the UseMap attribute for the <img> tag.

The <MAP> tag

- Used to define a client – side imagemap.
- There is only a single attribute for the <MAP> tag (NAME), which is used to specify the name of the imagemap.
- The syntax for the <MAP> tag is

<MAP name="map1">

…..

</MAP>

The <AREA> tag

- The <AREA> tag defines an area on the image and what action takes place when the user clicks in that area.
- All <AREA>tags must go between <MAP> </MAP> tags.
- The <AREA> tag has the following attributes:

  SHAPE

  COORDS

  ALT

  NAME

  TARGET

  HREF

The SHAPE attributes

- The shape attribute defines what type of shape the <AREA> tag is defining.
- There are 4 possible values for this attribute:

  Rect

  Circle

  Poly

  Default

Rect

- The rect shape is a rectangle and is defined by 2 sets of coordinates.

This first x and y coordinates marks the rectangle's upper-left corner.

- The second x and y coordinates specifies the lower-right corner.

Circle

- The circle shape is a standard circle.
- The first x and y coordinates marks the center point of the circle.
- The second x and y coordinates specifies a point on the edge of the circle.

Poly

- The poly shape refers to the polygon.
- To define a polygon area, simply supply the x and y coordinates for each point of the polygon.

Default

A shape attribute set to default is used to designate the action for any mouse clicks made whose coordinates are outside all other areas.

COORDS attribute

- The COORDS attribute is used to specify the numeric values for the x and y values of the area.

<AREA SHAPE=rect COORDS="19,20,49,40">

ALT attribute

- Used to specify ALT text to be displayed for that section of the imagemap.

The USEMAP attribute

- After the client-side imagemap has been created using the <MAP> and </MAP> tags, and have defined all the shapes using <AREA> tags, you need to tell the web browser that your image has an associated client-side imagemap.
- This is done by using the USEMAP attribute in the IMG tag.
- The USEMAP attribute must be assigned the name of the client-side imagemap.

USEMAP="#map1"

WYSIWYG Imagemap Editors

- There are three ways to determine the coordinates for the areas.
  - Take a guess
  - Determine the coordinates of each point
  - Use a WYSIWYG imagemap editor.
- WYSIWYG imagemap editors provide a graphical tool for creating imagemaps.
- Simply select the shape you want and click around in the image.
- The editor converts mouse clicks into the x and y coordinates and creates the <AREA> tag.
- 3 most popular WYSIWYG imagemap editors are:
  - Map Edit
  - Hotspots
  - LiveIamage

Enhancing Client-Side imagemaps with Client-Side Scripting

- One of the biggest benefits of using client-side imagemaps is that scripting can be added to the <AREA> tags.

<AREA SHAPE="polygon" COORDS="26,93,14,104,30,122" onMouseOver='window.status="Union Stage casting calls"; return true' onMouseOut='window.status=" ";return true'>

15. Introducing JavaScript

What is javascript?

- JavaScript is a client – side scripting language for the world wide web, that is similar to the syntax of the Java programming language.
- JavaScript is designed to provide limited programming functionality.

Why JavaScript?

- By executing more web functionality on the user's machine, webmasters can optimize their servers to serve more pages.
- The decrease in traffic from constant interaction with the server can also improve a server's performance.
- Because the local machine is doing the script processing, the user can view web pages much faster

Introducing JavaScript Syntax

- A simple JavaScript program:

```
<html>
<head>
<Title> Hello World  </Title>
</head>
<body>
<script language="JavaScript">
        document.write("Hello,World wide web");
</script>
</body>  </html>
```

Statements

- It is simply a line of code that contains some kind of instructions.
- Example

```
Document.write("<h2> Hello, WWW </h2>");
```

Blocks

- The grouping of statements is a block:

```
{
document.write("<h2> Each line is a statement </h2>");
document.write("<P> These statements, <BR>");
document.write("are part of a block");
}
```

Comments

- A single line comment is denoted with two slashes (//)

```
// this is comment, it will be ignored by the browser.
```

- Multi line comments

```
/*
  some text
*/
```

Data Types

- There are 5 basic data types in JavaScript: string, number, Boolean, object and function.

Variables

- Variables are "containers" for storing information.

Rules for JavaScript variable names:

- Variable names are case sensitive (y and Y are two different variables)
- Variable names must begin with a letter or the underscore character

Declaring (Creating) JavaScript Variables

- Creating variables in JavaScript is most often referred to as "declaring" variables.
- You can declare JavaScript variables with the **var statement**:

```
var x;
var carname;
```

- After the declaration shown above, the variables are empty (they have no values yet).
- However, you can also assign values to the variables when you declare them:

  Varx=5;

  var carname="Volvo";

Expressions

- The methods that can be employed to manipulate the data are called expressions.
- There are 2 types of expressions: numerical and logical
- Numerical expressions deal with the number data type.
- Logical expression might compare two data values, including strings, to see if they match.

Numerical expressions

- Addition, subtraction, multiplication and division are all types of numeric expressions.
- Numeric operators are: +, -, *, /, %

Logical Expressions

- These are expressions that, when evaluated, can return either a true or a false.
- Logical Operators

  && - And

  || - Or

  ! - Not

  == - Equal

  != - Not Equal

  > - Greater than

  >=

  <

  <=

Flow Control

- Conditional statements are used to perform different actions based on different conditions.
- In JavaScript we have the following conditional statements:

- **if statement** - use this statement to execute some code only if a specified condition is true
- **if...else statement** - use this statement to execute some code if the condition is true and another code if the condition is false
- **if...else if....else statement** - use this statement to select one of many blocks of code to be executed
- **switch statement** - use this statement to select one of many blocks of code to be executed

If Statement

- Use the if statement to execute some code only if a specified condition is true.

Syntax

```
if (condition)
{
code to be executed if condition is true
}
```

Example

```
<script  type="text/javascript">
//Write a "Good morning" greeting if
//the time is less than 10
var d=new Date();
var time=d.getHours();
if (time<10)
  {
  document.write("<b>Good morning</b>");
  }
</script>
```

If … Else Statement

- Use the if....else statement to execute some code if a condition is true and another code if the condition is not true.

Syntax

if (*condition*)

{

code to be executed if condition is true

}

else

{

code to be executed if condition is not true

}

Example

- <script type="text/javascript">

  //If the time is less than 10, you will get a "Good morning" greeting.

  //Otherwise you will get a "Good day" greeting.

  var d = new Date();

  var time = d.getHours();

  if (time < 10)

  {

  document.write("Good morning!");

  }

  else

  {

  document.write("Good day!");

  }

  </script>

If...else if...else Statement

- Use the if....else if...else statement to select one of several blocks of code to be executed.
- **Syntax**
- if (*condition1*)

  {

  code to be executed if condition1 is true

```
  }
else if (condition2)
  {
  code to be executed if condition2 is true
  }
else
  {
  code to be executed if condition1 and condition2 are not true
  }
```

JavaScript Switch Statement

- Conditional statements are used to perform different actions based on different conditions.
- Use the switch statement to select one of many blocks of code to be executed.
- **Syntax**

```
switch(n)
{
case 1:
  execute code block 1
  break;
case 2:
  execute code block 2
  break;
default:
  code to be executed if n is different from case 1 and 2
}
```

Example:

```
<script type="text/javascript">
//You will receive a different greeting based
//on what day it is. Note that Sunday=0,
//Monday=1, Tuesday=2, etc.
var d=new Date();
```

```
theDay=d.getDay();
switch (theDay)
{
case 5:
  document.write("Finally Friday");   break;
case 6:
  document.write("Super Saturday");    break;
case 0:
  document.write("Sleepy Sunday");    break;
default:
  document.write("I'm looking forward to this weekend!");
}
</script>
```

JavaScript For Loop

- Loops execute a block of code a specified number of times, or while a specified condition is true.
- **JavaScript Loops**
- Often when you write code, you want the same block of code to run over and over again in a row. Instead of adding several almost equal lines in a script we can use loops to perform a task like this.
- In JavaScript, there are two different kind of loops:
- **for** - loops through a block of code a specified number of times
- **while** - loops through a block of code while a specified condition is true

The for Loop

- The for loop is used when you know in advance how many times the script should run.

Syntax

```
for (var=startvalue;var<=endvalue;var=var+increment)
{
code to be executed
}
```

Example

```
<html>
<body>
<script type="text/javascript">
var i=0;
for (i=0;i<=5;i++)
{
document.write("The number is " + i);
document.write("<br />");
}
</script>
</body>
</html>
```

JavaScript While Loop

- The while loop loops through a block of code while a specified condition is true.

Syntax

- while (var<=endvalue)
  {
  code to be executed
  }

Example

```
<html>
<body>
<script type="text/javascript">
var i=0;
while (i<=5)
  {
  document.write("The number is " + i);
  document.write("<br />");
  i++;
```

```
 }
</script>
</body>
</html>
```

Arrays

- The array concept is used to store a set of values in a single variable name. This is a very important concept in any programming language

Array Definition:

- The array must first be defined before it is accessed or used. There are a variety of ways this can be accomplished in JavaScript. Below are some of the ways of defining arrays in JavaScript.

Defining an Array in JavaScript:

- Array is defined in JavaScript by making use of the keyword new.
- General format of defining array in JavaScript is as follows:

  var varaiblename = new **Array**( )
- For example, if a programmer wants to define an array exforsys, it is written as follows:

  var exforsys = new **Array**( )
- The format for adding elements in this structure is as follows:

  var varaiblename = **new Array**( )
  variablename[0]=”element1”
  variablename[1]=”element2”
  variablename[2]=”element3”
  variablename[3]=”element4”
  ……
- **var** Exforsys = **new Array**( )
  Exforsys[0]=”Training”
  Exforsys[1]=”Division”
  Exforsys[2]=”Institute”
  Exforsys[3]=”Company”

- It is also possible that if the programmer is certain of the array length, it can be defined while defining the array itself as:

  var Exforsys = **new Array**(4)

Shorter form:

- The elements can also be defined in a shorter form as:

  var variablename=**new Array**("element1","element2","elements3",….)

Accessing Arrays in JavaScript

- You can access an array element by referring to the name of the array and the element's index number.

Displaying Array Elements

document.write(Exforsys[1])

Functions

- A function is simply a block of code with a name, which allows the block of code to be called by other components in the scripts to perform certain tasks.
- Functions can also accept parameters that they use complete their task.
- JavaScript actually comes with a number of built-in functions to accomplish a variety of tasks.

Creating Custom Functions

- In addition to using the functions provided by javaScript, you can also create and use your own functions.
- General syntax for creating a function in JavaScript is as follows:

```
function name_of_function(argument1,argument2,…arguments)
{

……
//Block of Code
……

}
```

Calling functions

- There are two common ways to call a function: From an *event handler* and from another function.
- Calling a function is simple. You have to specify its name followed by the pair of parenthesis.

```
<SCRIPT  TYPE="TEXT/JAVASCRIPT">
      name_of_function(argument1,argument2,…arguments)
 </SCRIPT>
<html> <head> <title>Henley's Department Store</title>
<Script Language="JavaScript">
function welcomeMessage()
{ document.write("Welcome to Henley's Department Store!"); }
 </Script>
</head> <body>
<h1>Henley's Department Store</h1>
 <h3>Customers Order Processing</h3>
<Script Language="JavaScript">
welcomeMessage();
</Script>
 </body> </html>
```

16. Creating Simple JavaScripts

Formatting Scripts

- JavaScript scripts are set apart from the rest of the html in a web page by using the <script> tag.

```
<script>
     Here is some JavaScript
</script>
```

- The script accepts a parameter called Language, which is used to specify the type of scripting language that is being used.

```
<script Language="JavaScript"> …  </script>
```

- The html comment takes the form of <!-- To mark the beginning of a comment and -- > to mark the end of a comment.

```
<!-- This an Html Comment.
     It will be ignored by the browser -- >
```

- Browsers that understand the script tag ignore the Html comment and execute the script.
- If you need to make comments inside the script tag, you can do so using JavaScript comment.

Date and Time Entry

- The Date object is used to work with dates and times.
- Date objects are created with new Date().

```
<script Language="JavaScript">
    rightNow = new Date()
</script>
```

Methods of Date() object

- getDate() - Returns the day of the month (from 1-31)
- getDay() Returns the day of the week (from 0-6)
- getFullYear() Returns the year (four digits)
- getHours() Returns the hour (from 0-23)
- getMilliseconds() Returns the milliseconds (from 0-999)
- getMinutes() Returns the minutes (from 0-59)
- getMonth() Returns the month (from 0-11)
- getSeconds() Returns the seconds (from 0-59)
- setDate() Sets the day of the month (from 1-31)
- setFullYear() Sets the year (four digits)
- setHours() Sets the hour (from 0-23)
- setMilliseconds() Sets the milliseconds (from 0-999)
- setMinutes() Set the minutes (from 0-59)
- setMonth() Sets the month (from 0-11)
- setSeconds() Sets the seconds (from 0-59)

Example

```
<Html>
<head> <Title>  Date and Time Example   </Title>  </head>
<body>
<script language="JavaScript">
<!--
        rightNow = new Date();
        var status;
        hour = rightNow.getHours()
        minute = rightNow.gteMinutes()
        if (hour > = 13 )
                status = "PM";
        else
                status = "AM";
        if (hour>12)   hour = hour – 12;
Document.write (hour, ":", minute,status);
Month = rightNow.getMonth();
Month = month + 1;
Day = rightNow.getDate();
Year = rightNow.getYear();
Document.write(" ", month, "/", day, "/", year, " ");
-- >
</script>
</body>
</html>
```

Determining Browser Information

- The navigator object contains functions that are designed to return information about the browser, including the name and version of the browser and even the installed plug-ins.

Objects

- JavaScript is an object oriented programming language

- That means that elements within the language are treated as objects that can be used in different scripts.
- Another advantage of objects is that they have methods and properties associated with them.
- In javascript, the syntax for using objects is:
  Object.Method.Property

Using the Navigator Object

- JavaScript has an object called navigator, which contains properties and methods that can be used to find information about the version, configuration, and features of the current browser.
- Using javascript to determine information about the browser

```
<html>
<head> <title> Navigator installed plug-ins  </title>
<h2> Navigator </h2>
<script Language="JavaScript">
        var plug_count = navigator.plugins.length;
        for(var counter=0; counter < plug_count; counter++)
        {
                var plugin_num = counter + 1;
                document.write(" Plug In Number : " + plugin_num);
                document.write(navigator.plugins[counter].name);
                document.write(navigator.plugins[counter].filename);
        }
</script>
</head>
<body>  …  </body>
</html>
```

Linking Scripts to Windows Events

- In addition to Javascripts that are executed immediately when a page loads, it might also be useful to have a scripts that executes when a specific task occurs.

Example

```
<html>
<head> <Title> Alert on Unleaving  </title>
<body onUnload = “alert(‘Leave this page at your own peril!’);”>
…….  </body>  </html>
```

Alert Boxes and Confirmations

Alert Box

- An alert box is often used if you want to make sure information comes through to the user.

Example

- ```
  <html>
  <head>
  <script type="text/javascript">
  function show_alert()
  {
  alert("I am an alert box!");
  }
  </script>
  </head>
  <body>
  <input type="button" onclick="show_alert()" value="Show alert box" />
  </body>
  </html>
  ```
- A confirm box is often used if you want the user to verify or accept something.

Example

```
<html>
<head>
<script type="text/javascript">
function show_confirm()
{
var r=confirm("Press a button");
```
```

```
if (r==true)
 {
 document.write("You pressed OK!");
 }
else
 {
 document.write("You pressed Cancel!");
 }
}
</script>
</head>
<body>

<input type="button" onclick="show_confirm()" value="Show confirm box" />

</body>
</html>
```

Altering the Status Bar

- JavaScript allows to manipulate the text that appears in the status bar.
- The JavaScript window object contains properties and methods for manipulating the text in the staus bar.
- Example

```
<html>
<head> <title>  Changing the Status Bar  </title>
<script Language = “JavaScript”>
 function msg(text)
  {   window.status = text;   }
</script>
</head>
<body>
<h2> Manipulating Text in the Status Bar.  </h2>
```

```
<a href=http://www.yahoo.com
onMouseOver="msg('Click here for Yahoo!'); return true;"
onMouseOut = "msg('    '); return true;"
</a>
</body>  </html>
```

17. Using JavaScript for Forms

CGI versus JavaScript Forms

- Creating forms with JavaScript doesn't vary that much from creating forms with CGI.
- The difference is the way in which data is entered and processed.
- JavaScript performs some preprocessing before the form is submitted.
- JavaScript can be used to partially check integrity of the data entered into forms.

The Form Object

- Form object is used to access html form tags and properties through Javascript.
- The elements of a form are stored in the Forms[] array, which is property of the Document object.
- All the form elements are stored sequentially in the Forms array and can be accessed by their index number:

  Document.Forms[0]

- Consider a simple page:

```
<html>
<body>
<form>
        <input type="checkbox" Name="ElementOne">
</form>
<form>
        <input type="radio" Name="ElementOne">
        <input type="text"  Name = "ElementTwo">
</form>
```

```
</body>
</html>
```

- In the above example Document.Forms[0] refers to the first form and Document.Forms[1] refers to the second form.
- Document.Forms[1].Elements[0] is used to access the first element in the second form.

The properties of Form object are:

- ACTION - Action is the name of the CGI script or application on the server that handles the form input.
- ENCODING - refers to any special data encoding that might be passedto the server from script.
- METHOD - refers to method used to communicate form data to the server, it can be either get or post.
- TARGET - refers to the location where the form data is submitted.

Form Elements

- Form elements have a number of associated properties, methods, and event handlers that can be used to customize the forms or to manipulate the data that is contained on the form.

Button

- The button element represents a button on the current form. The button is created using <Input type="button"> tag, and can be used to represent any action the user can execute on a page.
- The onClick() event handler can be used to execute other functions or scripts when the button is pressed.

JavaScript Properties, methods, and events associated with the button element:

- Properties : form, name, type, value
- Methods : blur(), click(), focus()
- Event Handlerrs : onblur(), onclick(), onfocus()
- Html:

```
<form>
```

<input type="button" value="label" name="name" onclick="handler">
</form>

Checkbox Element

- Checkbox can be used to select a single value, and is created using <Input type="checkbox"> tag.
- The onclick() event handler provides a mechanism for linking functions to the selection of a checkbox.
- Properties : checked, defaultchecked, form, name, type, value
- Methods : blur(), click(), focus()
- Event Handlerrs : onblur(), onclick(), onfocus()
- Html:

  <form>
  <input type="checkbox" value="label" Name="name" Checked onclick="handler"> </form>

Radio

- Properties : checked, defaultchecked, form, name, type, value
- Methods : blur(), click(), focus()
- Event Handlerrs : onblur(), onclick(), onfocus()
- Html:

  <form>
  <input type="radio" value="label" Name="name" Checked onclick="handler">
  Label
  </form>

Select Element

- The select element can take 2 forms: a single selection box, or, by using Multiple keyword, a multiple selection box.
- Properties : form, length, name, options, selectedindex, type
- Methods : blur(), click(), focus()
- Event Handlerrs : onblur(), onclick(), onfocus()
- Html:

  <form>

<select name="name" size=integer Multiple onchange="handler"
onblur="handler" onfocus="handlerr">
<option value="value" Selected>option label
<option value="value"> option label
</select>

Text Element

- The text element is used to create a one-line text input field on the form.
- The onchange() event handler can be used to invoke methods when text is entered into the field.
- Properties : defaultValue, form, name, type, value
- Methods : blur(), focus(), select()
- Event Handlers : onblur(), onchange(), onfocus()
- Html:

  <form>
  <input type="text" name="name" value="default" size=integer>
  </form>

Submit Element

- This button is used to submit the form data to a server application or cgi script for processing.
- Properties : form, name, type, value
- Methods : blur(), focus(), click()
- Event Handlers : onblur(), onclick(), onfocus()
- Html:

  <form>
  <input type="submit" name="name" value="default" onclick="handler">
  </form>

Reset Element

- This element provides users with a means of clearing any data entered into a form.
- Properties : form, name, type, value
- Methods : blur(), focus(), click()

- Event Handlers : onblur(), onclick(), onfocus()
- Html:

  <form>

  <input type="reset" name="name" value="default" onclick="handler">

  </form>

Form Element Properties

- Form - The form property represents the name of the current form.
- Name - The name property represents the value of the name attribute specified in the html definition of the current form element.

  <input type=text name="address">
- Type - The type property represents the type of input element that has been defined in the current form.
- Value - for form elements that have a data value that is going to be submitted, the value property represents that data.
- Checked - the checked property is a Boolean that indicates the current status of a checkbox element.
- Defaultchecked - the defaultchecked will return a true if the checkbox has been set to automatically be checkedd by using the checked keyword.
- Options - The options property is an array that contains the values for all of the selection options that have been defined in the <select> tag.
- Length - The length property represents the number of elements that are contained in the options array.
- SelectedIndex - This property represents the array index number of the selected <option> element.
- DefaultValue - The DefaultValue can be used to access the default text in elements.

Form Element Methods

- There are 4 form element methods that can be invoked to change the status of form elements.
- Blur() - The blur() method can be used to remove focus from a current element.

- Click() - The click() method simulate a mouse click on a button or another element that generates an onclick event.
- Focus() - When called it passes the user interface focus to that element, allowing the element to accept user input.
- Select() - The select() method is used to select input elements that can be selected, such as text input fields, text areas, and so on.

Form Element Event Handlers

- The event handlers are the methods that get called when a user interface event occurs.
- For example, clicking a button generates an onclick event for that button.
- Onblur() - onblur is the event handler calledd when the focus on a particular element is lost. It could be used to validate a field before progressing to the next.
- Onclick() - The onclick event handler defines actions that should be performed when an element, such as a button, is clicked.
- Onfocus() - The onfocus event handler is called when a form element receives user interface focus, for example, when a user clicks inside a textarea.
- Onchange - The onchange event handler is used when the user enters data into a textfield or other element that contains data that can be changed.

Validating Form Fields Using JavaScript

```
<html>  <head>  <title>Form Validation Example</title>
 <script>
function ValidateContactForm()
{
   var name = document.ContactForm.Name;
   var email = document.ContactForm.Email;
   var phone = document.ContactForm.Telephone;
   var nocall = document.ContactForm.DoNotCall;
   var what = document.ContactForm.Subject;
   var comment = document.ContactForm.Comment;
    if (name.value == "")
   {
```

```
        window.alert("Please enter your name.");
        name.focus();
}
if (email.value == "")
    {
        window.alert("Please enter a valid e-mail address.");
        email.focus();
      }
 if (email.value.indexOf("@", 0) < 0)
    {
        window.alert("Please enter a valid e-mail address.");
        email.focus();

    }
    if (email.value.indexOf(".", 0) < 0)
    {
        window.alert("Please enter a valid e-mail address.");
        email.focus();
}
if ((nocall.checked == false) && (phone.value == ""))
    {
        window.alert("Please enter your telephone number.");
        phone.focus();
}

    if (what.selectedIndex < 1)
    {
        alert("Please tell us how we can help you.");
        what.focus();
      }
```

```
    if (comment.value == "")
    {
window.alert("Please provide a detailed description or comment.");
        comment.focus();
}
  }
 function EnableDisable(chkbx)
{
    if(chkbx.checked == true)
    {
        document.ContactForm.Telephone.disabled = true;
    }
    else
    {
        document.ContactForm.Telephone.disabled = false;
    }
}
</script>  </head>
<body>
 <form method="post“ name="ContactForm" onsubmit=“ValidateContactForm();">
    <p>Name: <input type="text" size="65" name="Name"></p>
    <p>E-mail Address:  <input type="text" size="65" name="Email"></p>
    <p>Telephone: <input type="text" size="65" name="Telephone"><br>
        <input type="checkbox" name="DoNotCall"
        onclick="EnableDisable(this);"> Please do not call me.</p>
    <p>What can we help you with?
        <select type="text" value="" name="Subject">
            <option>  </option>
            <option>Customer Service</option>
            <option>Question</option>
            <option>Comment</option>
```

```
        <option>Consultation</option>
        <option>Other</option>
</select></p>
   <p>Comments:  <textarea cols="55" name="Comment">  </textarea></p>
   <p><input type="submit" value="Send" name="submit">
   <input type="reset" value="Reset" name="reset"></p>
</form>

</body>
</html>
```

18. Using JavaScript with Style Sheets

Dynamic Style Sheets

- Just recently, two technologies have been released for precise control of the layout and positioning of elements on a page.

Cascading Style Sheets

- CSS gave designers the ability to fully layout, position and define pages, without use of tables, and any other hacks.
- Using CSS, a designer could define the fonts each tag would use, its color, its margin, and many other properties.

Using JavaScript to Control Style Sheets

- With the release of the 4.0 browsers from Microsoft and Netscape, finally comes the capability to script style sheets using scripting language such as JavaScript and VBScript.

What are the Advantages?

These new technologies enable us to do:

- Make the text on the page adjust its size to adapt to the user's browser size
- Create text rollover effects without the need for graphics
- Create an Html based menu that can expand and collapse to show its contents
- Change text properties on-the-fly without a full page refresh

- Fade in and fade out page elements
- Unlike CGI, the script handles everything on the client side, making the web pages load quickly, thus enhancing the interactive experience.
- Creating a simple MouseOver effect

```
<html>
<head>
<title>  Simple mouseover effect  </title>
<style type="text/css">
A { text-decoration : none;color : brown;   }
</style>
<script language="javascript">
function mover()
{
        window.event.srcElement.style.backgroundColor = "Orange";
}
function mout ()
{
        window.event.srcElement.style.backgroundColor = "White";
}
</script>
</head>
<body>
<A href="http://www.site1.com" onMouseOver = "mover(); " onMouseOut = "mout();" >  Site1  </A>
<A href="http://www.site2.com" onMouseOver = "mover(); " onMouseOut = "mout();" >  Site2  </A>
<A href="http://www.site3.com" onMouseOver = "mover(); " onMouseOut = "mout();" >  Site3  </A>
</body>
</html>
```

The all Collection

- The all collection can be used to access any of the valid html tags on the page.
- It returns an array of elements that has been specified.
- The code to access <B> tags :

  document.all.tags("B")
- Then follow it by the property that need to be accessed:

  document.all.tags("B").style.fontStyle="italic";

Using the item() method for More Control

- The all collection returns a collection of the tags that has been chosen, the item() method is used to select an individual element in the array.
- Example

```
<html>
<head>
<title>  Some title   </title>
<script language="JavaScript">
function mclick()
{
        var b1 = document.all.tags("B").item(0);
        b1.style.cursor = "hand";
        var b2 = document.all.tags("B").item(1);
        b1.style.color = "green";
}
</script>
</head>
<body onclick="mclick();">
<B> Here is some bold text  </B>  <br>
<B> Here is another line of bold text  </B>
</body>
</html>
```

The styleSheets Collection

- The styleSheets collection is used to work with style sheets contained in the document. The styleSheets collection can be used to:
  - Find out how many style sheets are in the document
  - Add a style sheet
  - Replace a style sheet
  - Delete a style sheet
  - Disable a style sheet
  - Add a rule to the style sheet

Finding the number of style sheets in the document

- The styleSheets collection has a property called length, which is used to find out how many style sheets are in the document.
- Using the length property to find the number of style sheets in the document:

```
<html>
<head>
<title>  To find the number of style sheets   </title>
<style type=”text/css”>
H1   { font-size: 30 }
</style>
<script language=”javascript”>
Function count()
{
        Alert( document.styleSheets.length );
}
</script>
</head>
<body onLoad=”count();”>
<h1> A very simple document   </h1>
</body>
</html>
```

Adding a Style Sheet

- The style sheet can be added to the document by using:
  document.styleSheets(index).addImport("filename");
- Index is the index of the style sheet to which a new style sheet is added.
- For embedded style sheet only, new style sheet can be added.
- Code to check this:

```
if(document.styleSheets(0).href == "NULL")
{        document.styleSheets(0).addImport("stylesheet.css");
}
```

Replacing one Style Sheet with Another

- Using the styleSheets collection, an existing style sheet can be replaced with another.
- The only limitation is that the style sheet being replaced must be either be linked or imported into the document.
- Example

```
<html>
<head>  <title>  Replacing a Style Sheet  </title>
<LINK rel="stylesheet" href="stylesheet.css">
<script language="javascript">
function replacesheet()
{
        If(document.stylesheets.href != NULL)
        {   document.styleSheets(0).href = "new.css";  }
}
</script>
</head>
<body>
<div onClick="replacesheet();"> Click me to replace my style sheet  </div>
</body>
</html>
```

Disabling and Enabling a Style Sheet

- Style Sheets can be disabled by setting the disabled property to true.
- Example

```
<html>
<head>  <title>  Enabling and Disabling a Style Sheet   </title>
<style>
Body   { background-color : white }
H1 { font-size: 30pt; font-weight: bold; color: blue;  }
H2  { font-size: 18 pt; color: green; background-color:  yellow;  }
P   { font-size: 12 pt; color: purple; text-indent:30px  }
</style>
<script language=”javascript”>
function on()
{
        if(document.styleSheets(0).disabled)
                document.styleSheets(0).disabled = false;
        else
                document.styleSheets(0).disabled = true;
}
</script>
</head>
<Body>
        <H1> Title  </H1>
        <H2>  first paragraph  </H2>
        <P>  This is the first paragraph  .  </P>
        <H2>  Second Paragraph  </H2>
        <P>  This is the second paragraph.  </P>
        <div id=”state”>
        <input type=”button” onclick=”on();”>
        </div>
</body>  </html>
```

Adding a new rule to a Style Sheet

- New rules can be added to the embedded style sheet by using the addRule() method.
- Syntax

  document.styleSheets.addRule(“H1”, “Font-Color: green”);